Google DeepMind

(A not very gentle)

Introduction to Diffusion Models

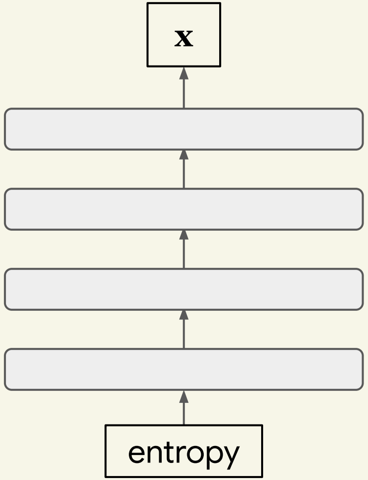
Yuge (Jimmy) Shi

Google DeepMind

1 Generative Models

Generative modelling: the probabilistic perspective

$$x \sim p(x)$$

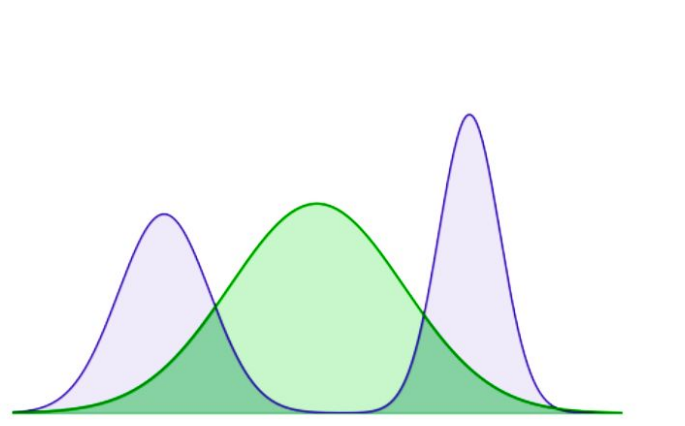


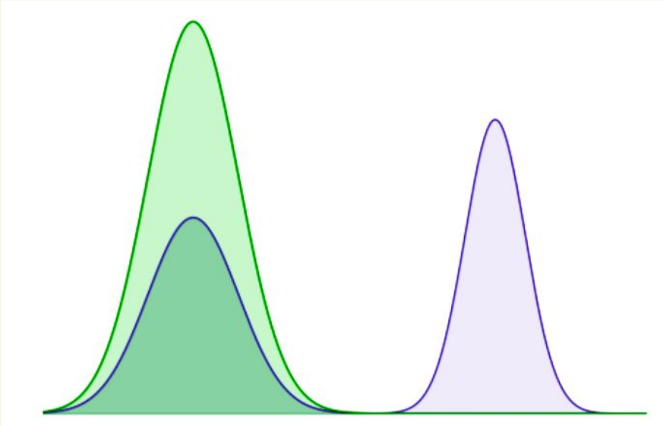
Explicit: autoregression, flows, VAEs, ...

Implicit: GANs, ...

Slide credit: Sander Dieleman

Mode-covering vs. mode-seeking behaviour





mode-covering
focus on **diversity**

e.g. likelihood-based models

mode-seeking
focus on **realism**

e.g. adversarial models

Slide credit: Sander Dieleman

2 Diffusion Models, a vibe-based overview

An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / …] from pure noise
- Uses **iterative refinement**

An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Training: forward process

Add various levels of noise (defined by `t`) to the image.

An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Training: forward process

Add various levels of noise (defined by `t`) to the image.

`t=0` `t=1` `t=2` `t=3` `t=1000`

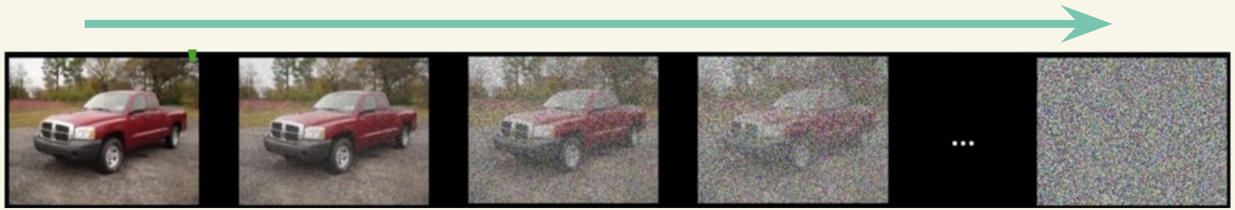
An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Training: forward process

Add various levels of noise (defined by `t`) to the image.



`t=0` `t=1` `t=2` `t=3` `t=1000`

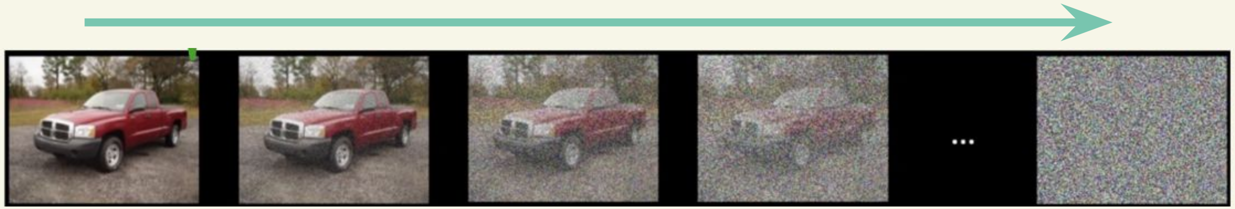
An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Training: forward process

Add various levels of noise (defined by `t`) to the image.
Objective: predict the noise added to the original image.



t=0 t=1 t=2 t=3 t=1000

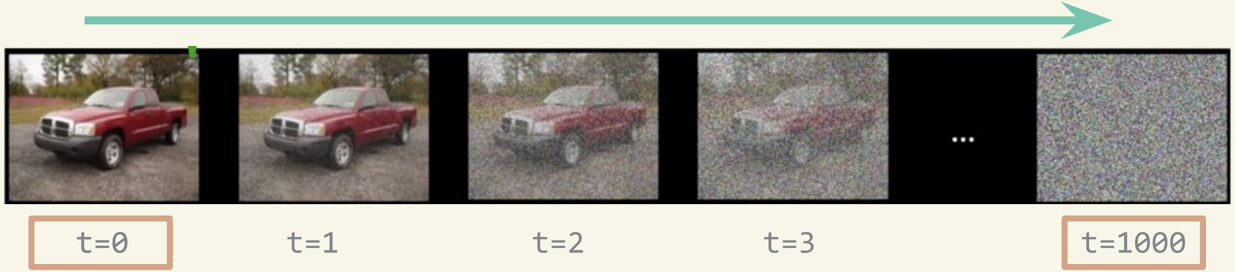
An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Training: forward process

Add various levels of noise (defined by `t`) to the image.
Objective: predict the noise added to the original image.



An Overview

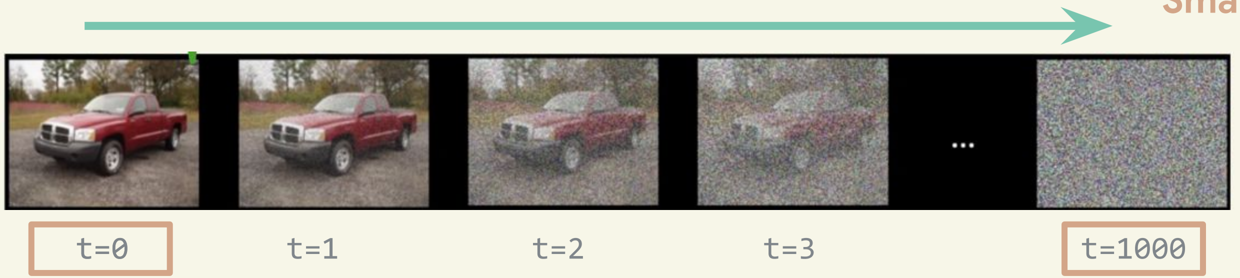
Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Training: forward process

Add various levels of noise (defined by `t`) to the image.
Objective: predict the noise added to the original image.

Big `t`: MORE noisy
Small `t`: LESS noisy



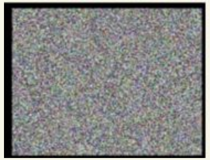
An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Inference: backward process

Start from pure noise



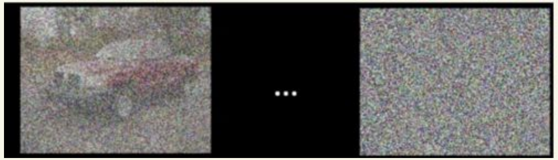
An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Inference: backward process

Start from pure noise, iteratively denoise until we get back to an un-noised image.



Google

An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Inference: backward process

Start from pure noise, iteratively denoise until we get back to an un-noised image.

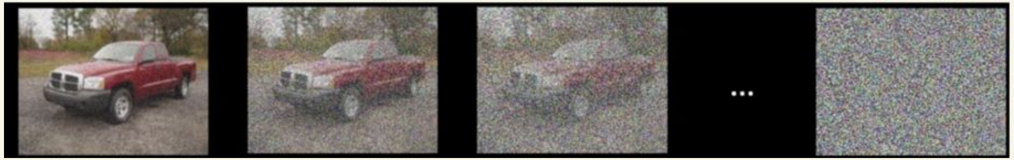
An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Inference: backward process

Start from pure noise, iteratively denoise until we get back to an un-noised image.



An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Inference: backward process

Start from pure noise, iteratively denoise until we get back to an un-noised image.

Google

An Overview

Diffusion model **is**

- An **implicit** generative model
- Generates data ~ [images / videos / audio / text (?!) / ...] from pure noise
- Uses **iterative refinement**

Inference: backward process

Start from pure noise, iteratively denoise until we get back to an un-noised image.

3 Diffusion Models, Forward Process

Forward process, q

$$q(x_{1:T} \mid x_0)$$

Forward process, q

$$q(x_{1:T} \mid x_0)$$

original image

Forward process, q

$$q(x_{1:T} \mid x_0)$$

Noisy images

original image

Forward process, q

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

Noisy images

original image

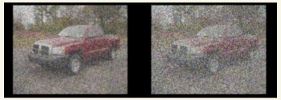
Forward process, q

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

Noisy images

original image

t=1 t=2



t=2 t=3

⋮

product of

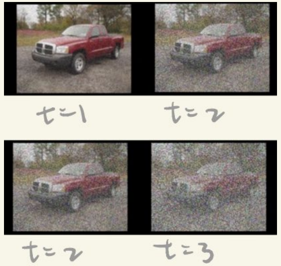
Forward process, q

the entire forward pass =

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

noisy images

original image



product of

Forward process, q

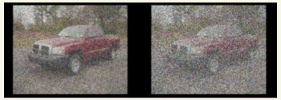
the entire forward pass = product over per-step forward pass

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

noisy images

original image

t=1 t=2



t=2 t=3

⋮

product of

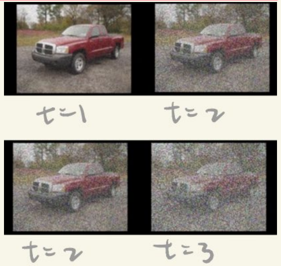
Forward process, q

the entire forward pass = product over per-step forward pass

$$q(x_{1:T} \mid x_0) = \boxed{\prod_{t=1}^{T} q(x_t \mid x_{t-1})}$$

noisy images

original image



How do we parameterise this?

product of

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

$q(x_t | x_{t-1})$ = Gaussian distribution with mean noisily at the previous time step & fixed variance, i.e.

Parametrisation of $q(x_t \mid x_{t-1})$

$q(x_t \mid x_{t-1})$ = Gaussian distribution with mean noisily at the previous time step & fixed variance, i.e.

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

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

$q(x_t | x_{t-1})$ = Gaussian distribution with mean noisily at the previous time step & fixed variance, i.e.

$$x_t = \mathcal{N}\left(\sqrt{1-\underbrace{\beta_t}_{\text{specifies noise level}}}\, x_{t-1}, \ \underbrace{\beta_t}_{\text{specifies noise level}} I\right)$$

Side quest: all about noise level

β_t controls the noise level through either **1)** shifting or **2)** widening the gaussian distribution parametrising $q(x_t \mid x_{t-1})$

Side quest: all about noise level

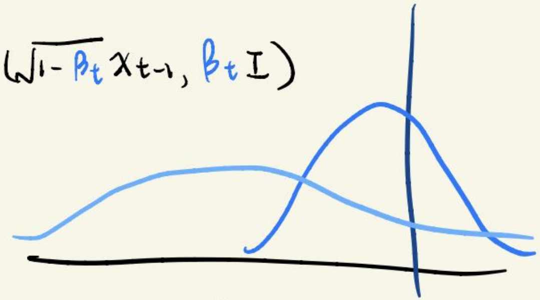
Side quest: all about noise level

β_t controls the noise level through either **1)** shifting or **2)** widening the gaussian distribution parametrising $q(x_t \mid x_{t-1})$

Side quest: all about noise level

β_t controls the noise level through either **1)** shifting or **2)** widening the gaussian distribution parametrising $q(x_t \mid x_{t-1})$

$$\mathcal{N}(\sqrt{1-\beta_t}\, x_{t-1}, \beta_t I)$$

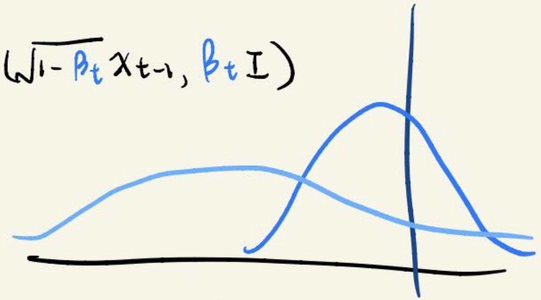


Side quest: all about noise level

β_t controls the noise level through either **1)** shifting or **2)** widening the gaussian distribution

parametrising $q(x_t \mid x_{t-1})$

$\mathcal{N}(\sqrt{1-\beta_t}\,x_{t-1}, \beta_t I)$

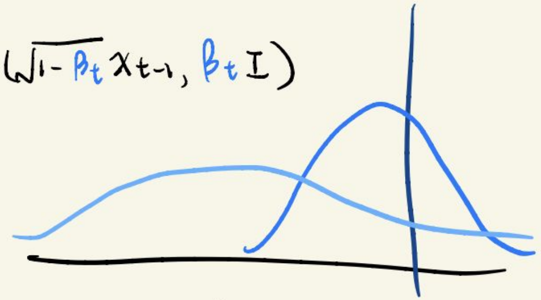


$\beta_t = 0$: x_t is exactly x_{t-1}

Side quest: all about noise level

β_t controls the noise level through either **1)** shifting or **2)** widening the gaussian distribution parametrising $q(x_t \mid x_{t-1})$

$\mathcal{N}(\sqrt{1-\beta_t}\,x_{t-1}, \beta_t I)$



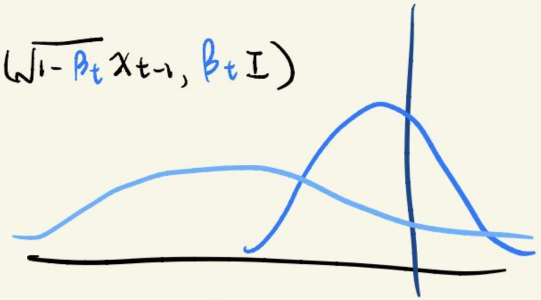
$\beta_t = 0$: x_t is exactly x_{t-1}

$\beta_t = 0.01$: slightly shifted & widened

Side quest: all about noise level

β_t controls the noise level through either **1)** shifting or **2)** widening the gaussian distribution parametrising $q(x_t \mid x_{t-1})$

$\mathcal{N}(\sqrt{1-\beta_t}\,x_{t-1}, \beta_t I)$



$\beta_t = 0$: x_t is exactly x_{t-1}

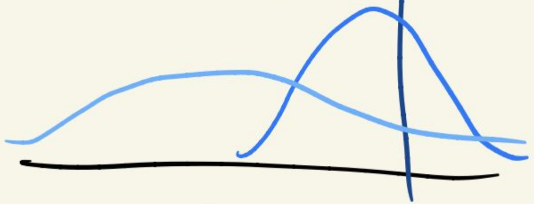
$\beta_t = 0.01$: slightly shifted & widened

$\beta_t = 0.1$: very shifted & widened

Side quest: all about noise level

β_t controls the noise level through either **1)** shifting or **2)** widening the gaussian distribution parametrising $q(x_t \mid x_{t-1})$

$\mathcal{N}(\sqrt{1-\beta_t}\,x_{t-1}, \beta_t I)$



$\beta_t = 0$: x_t is exactly x_{t-1}

$\beta_t = 0.01$: slightly shifted & widened

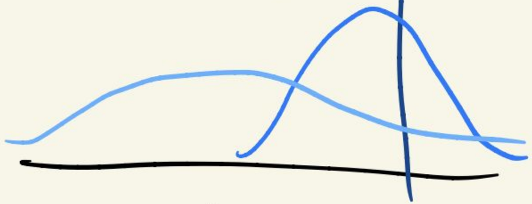
$\beta_t = 0.1$: very shifted & widened

So the larger β_t is, the noisier the next sample will be!

Side quest: all about noise level

β_t controls the noise level through either **1)** shifting or **2)** widening the gaussian distribution parametrising $q(x_t \mid x_{t-1})$

$\mathcal{N}(\sqrt{1-\beta_t}\, x_{t-1}, \beta_t I)$



$\beta_t = 0$: x_t is exactly x_{t-1}

$\beta_t = 0.01$: slightly shifted & widened

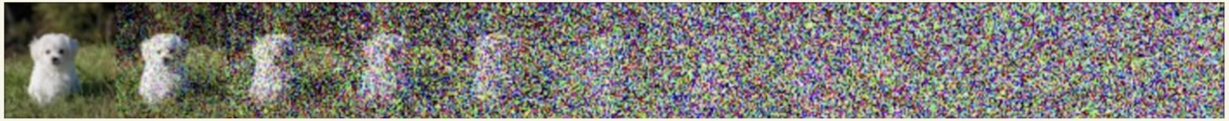
$\beta_t = 0.1$: very shifted & widened

So the larger β_t is, the noisier the next sample will be!

A sensible noise scheduler is important!

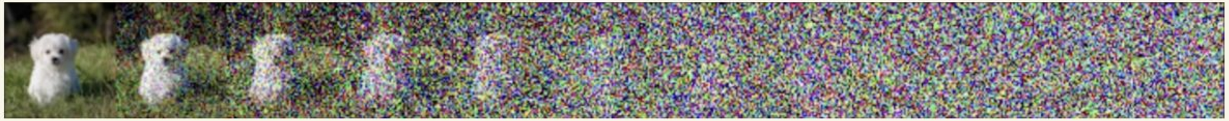
Side quest: all about noise level

DDPM (2020): linear schedule



Side quest: all about noise level

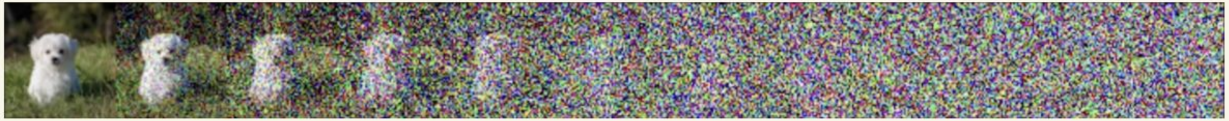
DDPM (2020): linear schedule



Problem: information gets destroyed too quickly

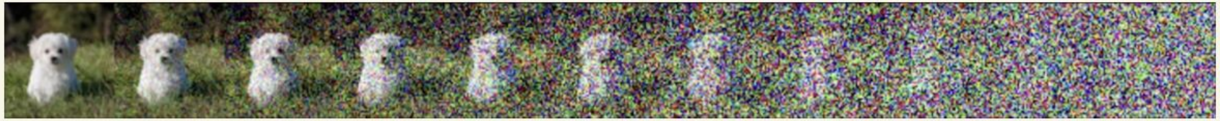
Side quest: all about noise level

DDPM (2020): linear schedule



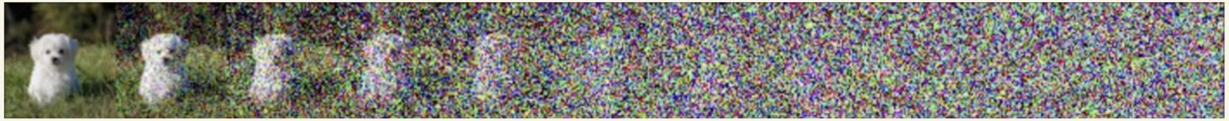
Problem: information gets destroyed too quickly

OAI (2021): cosine schedule



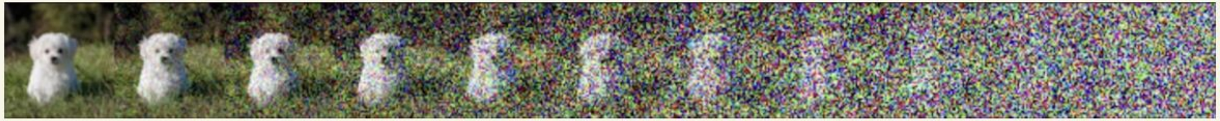
Side quest: all about noise level

DDPM (2020): linear schedule



Problem: information gets destroyed too quickly

OAI (2021): cosine schedule



More sensible as information gets destroyed more evenly as time grows.

Forward process, q

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

Forward process, q

Cumulative product

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

Forward process, q

Cumulative product

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

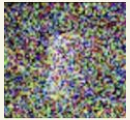
At training time, you want to be able to randomly sample `t~(1, T)`, and have the model predict the noise at that level `t`

Forward process, q

Cumulative product

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

x_0 

x_t =  , t=3

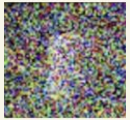
Forward process, q

Cumulative product

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

x_0

$q(x_1|x_0)$

x_t =  , t=3

Forward process, q

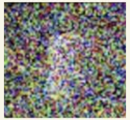
Cumulative product

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

x_0

$q(x_1|x_0)$

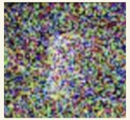
$q(x_2|x_1)$

$x_t =$  , t=3

Forward process, q

Cumulative product

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

$x_t =$  , t=3

x_0

$q(x_1|x_0)$

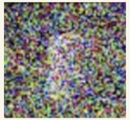
$q(x_2|x_1)$

$q(x_3|x_2)$

Forward process, q

Cumulative product

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

x_t =  , t=3

x_0

$q(x_1|x_0)$

$q(x_2|x_1)$

$q(x_3|x_2)$

Seems like a lot of work :/

Gaussian distribution to the rescue!

Fun fact: Product of Gaussian is still Gaussian

Gaussian distribution to the rescue!

Fun fact: Product of Gaussian is still Gaussian

Iterative product need not apply! We can generate any x_t from x_0 in just one step:

Gaussian distribution to the rescue!

Fun fact: Product of Gaussian is still Gaussian

Iterative product need not apply! We can generate any x_t from x_0 in just one step:

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

Gaussian distribution to the rescue!

Fun fact: Product of Gaussian is still Gaussian

Iterative product need not apply! We can generate any x_t from x_0 in just one step:

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

maths ✨

Gaussian distribution to the rescue!

Fun fact: Product of Gaussian is still Gaussian

Iterative product need not apply! We can generate any x_t from x_0 in just one step:

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

maths ✨

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

Gaussian distribution to the rescue!

Fun fact: Product of Gaussian is still Gaussian

Iterative product need not apply! We can generate any x_t from x_0 in just one step:

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

maths ✨

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

where $\bar{\alpha}_t$ denotes the cumulative $1-\beta_t$

Gaussian distribution to the rescue!

Fun fact: Product of Gaussian is still Gaussian

Iterative product need not apply! We can generate any x_t from x_0 in just one step:

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

maths ✨

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

where $\bar{\alpha}_t$ denotes the cumulative $1-\beta_t$

🚨 **Derivation incoming**

Google

Derivation:

Let us first define

Derivation:

Let us first define

β_t Noise added at t

Derivation:

Let us first define

β_t — Noise added at t

$\alpha_t = 1 - \beta_t$ — Information kept at t since (t-1)

Derivation:

Let us first define

β_t Noise added at t

$\alpha_t = 1 - \beta_t$ Information kept at t since (t-1)

$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ Information kept at t since 0

Derivation:

Given the original forward process

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

β_t	Noise added at t
$\alpha_t = 1 - \beta_t$	Information kept at t since `(t-1)`
$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$	Information kept at t since `0`

Derivation:

β_t Noise added at t

$\alpha_t = 1 - \beta_t$ Information kept at t since `(t-1)`

$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ Information kept at t since `0`

Given the original forward process, we can use **reparametrisation trick** to get

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

Google

Derivation:

Given the original forward process, we can use **reparametrisation trick** to get

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

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

Gaussian noise

$\epsilon \sim \mathcal{N}(0, 1)$

β_t	Noise added at t
$\alpha_t = 1 - \beta_t$	Information kept at t since `(t-1)`
$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$	Information kept at t since `0`

Google

Derivation:

Given the original forward process, we can use **reparametrisation trick** to get

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

$$= \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \varepsilon$$

Gaussian noise

$\epsilon \sim \mathcal{N}(0,1)$

Then we can sub in $\alpha_t = 1 - \beta_t$

$$= \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon$$

β_t — **Noise added at t**

$\alpha_t = 1 - \beta_t$ — **Information kept at t since** `(t-1)`

$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ — **Information kept at t since** `0`

Derivation:

Given the original forward process, we can use **reparametrisation trick** to get

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

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

Gaussian noise

$\epsilon \sim \mathcal{N}(0, 1)$

Then we can sub in $\alpha_t = 1 - \beta_t$

$$= \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \epsilon$$

Substitute x_{t-1} by $q(x_{t-1}|x_{t-2})$

$$= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_t \alpha_{t-1}}\, \epsilon$$

β_t Noise added at t

$\alpha_t = 1 - \beta_t$ Information kept at t since (t-1)

$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ Information kept at t since 0

Derivation:

Given the original forward process, we can use **reparametrisation trick** to get

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

$$= \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \varepsilon$$

Gaussian noise

$\epsilon \sim \mathcal{N}(0, 1)$

Then we can sub in $\alpha_t = 1 - \beta_t$

$$= \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon$$

Substitute x_{t-1} by $q(x_{t-1}|x_{t-2})$

$$= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{1-\underbrace{\boxed{\alpha_t \alpha_{t-1}}}_{\bar{\alpha}_t}}\, \varepsilon$$

β_t	**Noise added at t**
$\alpha_t = 1 - \beta_t$	**Information kept at t since (t-1)**
$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$	**Information kept at t since 0**

Derivation:

Given the original forward process, we can use **reparametrisation trick** to get

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

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

Gaussian noise

$\epsilon \sim \mathcal{N}(0, 1)$

Then we can sub in $\alpha_t = 1 - \beta_t$

$$= \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \epsilon$$

Substitute x_{t-1} by $q(x_{t-1} \mid x_{t-2})$

$$= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_t \alpha_{t-1}}\, \epsilon$$

$$\ldots$$

$$q(x_t \mid x_0) = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon$$

β_t	Noise added at t
$\alpha_t = 1 - \beta_t$	Information kept at t since `(t-1)`
$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$	Information kept at t since `0`

Derivation:

Given the original forward process, we can use **reparametrisation trick** to get

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

$$= \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \varepsilon$$

Gaussian noise $\epsilon \sim \mathcal{N}(0, 1)$

Then we can sub in $\alpha_t = 1 - \beta_t$

$$= \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon$$

Substitute x_{t-1} by $q(x_{t-1}|x_{t-2})$

$$= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_t \alpha_{t-1}}\, \varepsilon$$

$$\cdots$$

$$q(x_t|x_0) = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon$$

β_t Noise added at t

$\alpha_t = 1 - \beta_t$ Information kept at t since `(t-1)`

$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ Information kept at t since `0`

This allows for batch training with different values of t:

t=0.76

t=0.92

t=0.12

$\Rightarrow$ Model

Derivation:

Given the original forward process, we can use **reparametrisation trick** to get

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

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

Gaussian noise

$\epsilon \sim \mathcal{N}(0, 1)$

Then we can sub in $\alpha_t = 1 - \beta_t$

$$= \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \epsilon$$

Substitute x_{t-1} by $q(x_{t-1}|x_{t-2})$

$$= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_t \alpha_{t-1}}\, \epsilon$$

$$\dots$$

$$q(x_t|x_0) = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon$$

β_t	Noise added at t
$\alpha_t = 1 - \beta_t$	Information kept at t since `(t-1)`
$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$	Information kept at t since `0`

This allows for batch training with different values of t:

t=0.76

t=0.92

t=0.12

⇨ Model

Also means that we need to tell the model the value of t so it doesn't get confused!

3 Diffusion Models, Objective

Objective

We mentioned before that the diffusion model is trained to predict the noise added to the input, given the timestamp `t` and the noised input x_t :

Objective

We mentioned before that the diffusion model is trained to predict the noise added to the input, given the timestamp `t` and the noised input x_t :

$$\mathcal{L} = \mathbb{E}_{t, x_0, \epsilon} \left\| \epsilon - \epsilon_\theta(x_t, t) \right\|^2$$

Objective

We mentioned before that the diffusion model is trained to predict the noise added to the input, given the timestamp t and the noised input x_t :

added noise

$$\mathcal{L} = \mathbb{E}_{t, x_0, \epsilon} \left\| \epsilon - \epsilon_\theta(x_t, t) \right\|^2$$

predicted noise

Objective

We mentioned before that the diffusion model is trained to predict the noise added to the input, given the timestamp t and the noised input x_t :

$$\mathcal{L} = \mathbb{E}_{t, x_0, \epsilon} \|\underbrace{\epsilon}_{\text{added noise}} - \underbrace{\epsilon_\theta(x_t, t)}_{\text{predicted noise}}\|^2$$

Note: *the actual objective is maximising the log likelihood of $p(x)$ through* ***variational lower bound****, but we skip these dark magic for now and show this simple objective instead :)*

Objective: different formulations

We are predicting the noise, but from the forward process we know

Objective: different formulations

We are predicting the noise, but from the forward process we know

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

Google

Objective: different formulations

We are predicting the noise, but from the forward process we know

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

x_t is a linear combination of ϵ and x_0.

All linear combinations are valid and used for different cases:

Objective: different formulations

We are predicting the noise, but from the forward process we know

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

x_t is a linear combination of ϵ and x_0.

All linear combinations are valid and used for different cases:

- ϵ epsilon-prediction, most common

Objective: different formulations

We are predicting the noise, but from the forward process we know

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

x_t is a linear combination of ϵ and x_0.

All linear combinations are valid and used for different cases:

- ϵ
- x_0 **x0-prediction, more direct/intuitive**

Objective: different formulations

We are predicting the noise, but from the forward process we know

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

x_t is a linear combination of ϵ and x_0.

All linear combinations are valid and used for different cases:

- ϵ
- x_0
- $\alpha_t \epsilon - \sigma_t x_0$ v-prediction, "direction" to move in to reach x0

Google

Objective: different formulations

We are predicting the noise, but from the forward process we know

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

x_t is a linear combination of ϵ and x_0.

All linear combinations are valid and used for different cases:

- ϵ
- x_0
- $\alpha_t \epsilon - \sigma_t x_0$
- $\epsilon - x_0$ **flow matching**

Objective: different formulations

We are predicting the noise, but from the forward process we know

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

x_t is a linear combination of ϵ and x_0.

All linear combinations are valid and used for different cases:

- ϵ
- x_0
- $\alpha_t \epsilon - \sigma_t x_0$
- $\epsilon - x_0$

All equivalent except for relative weighting of noise levels!

4 Diffusion Models, Backward Process

Backward process, p

Sampling iteratively from x_T (pure noise) to x_0 (original data).

$$p_\theta(x_{0:T}) = p_\theta(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)$$

Google

Backward process, p

Sampling iteratively from x_T (pure noise) to x_0 (original data).

$$p_\theta(x_{0:T}) = \underbrace{p_\theta(x_T)}_{\text{unit-variance Gaussian: } \mathcal{N}(0,1)} \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)$$

Backward process, p

Sampling iteratively from x_T (pure noise) to x_0 (original data).

$$p_\theta(x_{0:T}) = \underbrace{p_\theta(x_T)}_{\substack{\text{unit-variance} \\ \text{Gaussian}: \mathcal{N}(0,1)}} \prod_{t=1}^{T} \underbrace{p_\theta(x_{t-1} \mid x_t)}_{x_{t-1} = x_t - \text{predicted noise}}$$

Backward process, p

Sampling iteratively from x_T (pure noise) to x_0 (original data).

$$p_\theta(x_{0:T}) = \underbrace{p_\theta(x_T)}_{\substack{\text{unit-variance} \\ \text{Gaussian}: \mathcal{N}(0,1)}} \prod_{t=1}^{T} \underbrace{p_\theta(x_{t-1} \mid x_t)}_{x_{t-1} = x_t - \text{predicted noise}}$$

*** ish, let's look at this a little closer**

A closer look at $p_\theta(x_{t-1} | x_t)$

Google

A closer look at $p_\theta(x_{t-1} | x_t)$

Let's say our model is predicting noise. Given the original forward process formulation it is easy to get the mean prediction of x_{t-1} :

$$\mu_\theta(x_t, t) = \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon$$

A closer look at $p_\theta(x_{t-1}|x_t)$

Let's say our model is predicting noise. Given the original forward process formulation it is easy to get the mean prediction of x_{t-1} :

$$\mu_\theta(x_t, t) = \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon$$

We want to **sample** from the Gaussian distribution that parametrises $p_\theta(x_{t-1}|x_t)$, so we use reparametrisation trick to take variance into considerations:

A closer look at $p_\theta(x_{t-1}|x_t)$

Let’s say our model is predicting noise. Given the original forward process formulation it is easy to get the mean prediction of x_{t-1} :

$$\mu_\theta(x_t, t) = \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon$$

We want to **sample** from the Gaussian distribution that parametrises $p_\theta(x_{t-1}|x_t)$, so we use reparametrisation trick to take variance into considerations:

$$x_{t-1} = \underbrace{\frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_\theta(x_t, t)\right)}_{\mu_\theta(x_t, t)} + \boxed{\beta_t \varepsilon}$$

A closer look at $p_\theta(x_{t-1} | x_t)$

Let's say our model is predicting noise. Given the original forward process formulation it is easy to get the mean prediction of x_{t-1} :

$$\mu_\theta(x_t, t) = \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon$$

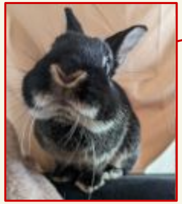
We want to **sample** from the Gaussian distribution that parametrises $p_\theta(x_{t-1} | x_t)$, so we use reparametrisation trick to take variance into considerations:

$$x_{t-1} = \underbrace{\frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_\theta(x_t, t)\right)}_{\mu_\theta(x_t, t)} + \boxed{\beta_t \varepsilon}$$

We do this iteratively until we get to the original image at t=0!

Backward Process: a visualisation

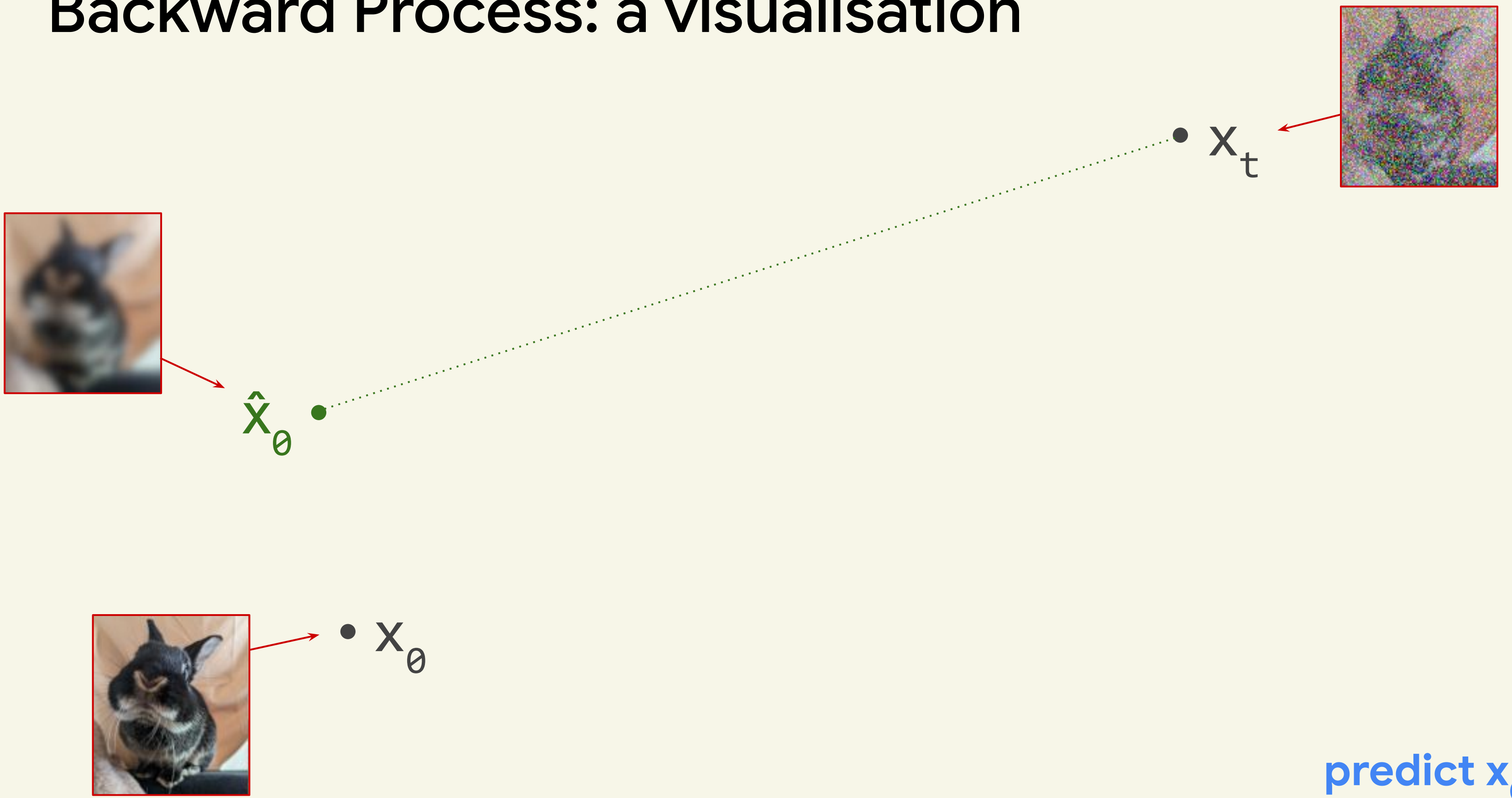
- x_t



- x_0

Slide credit: Sander Dieleman

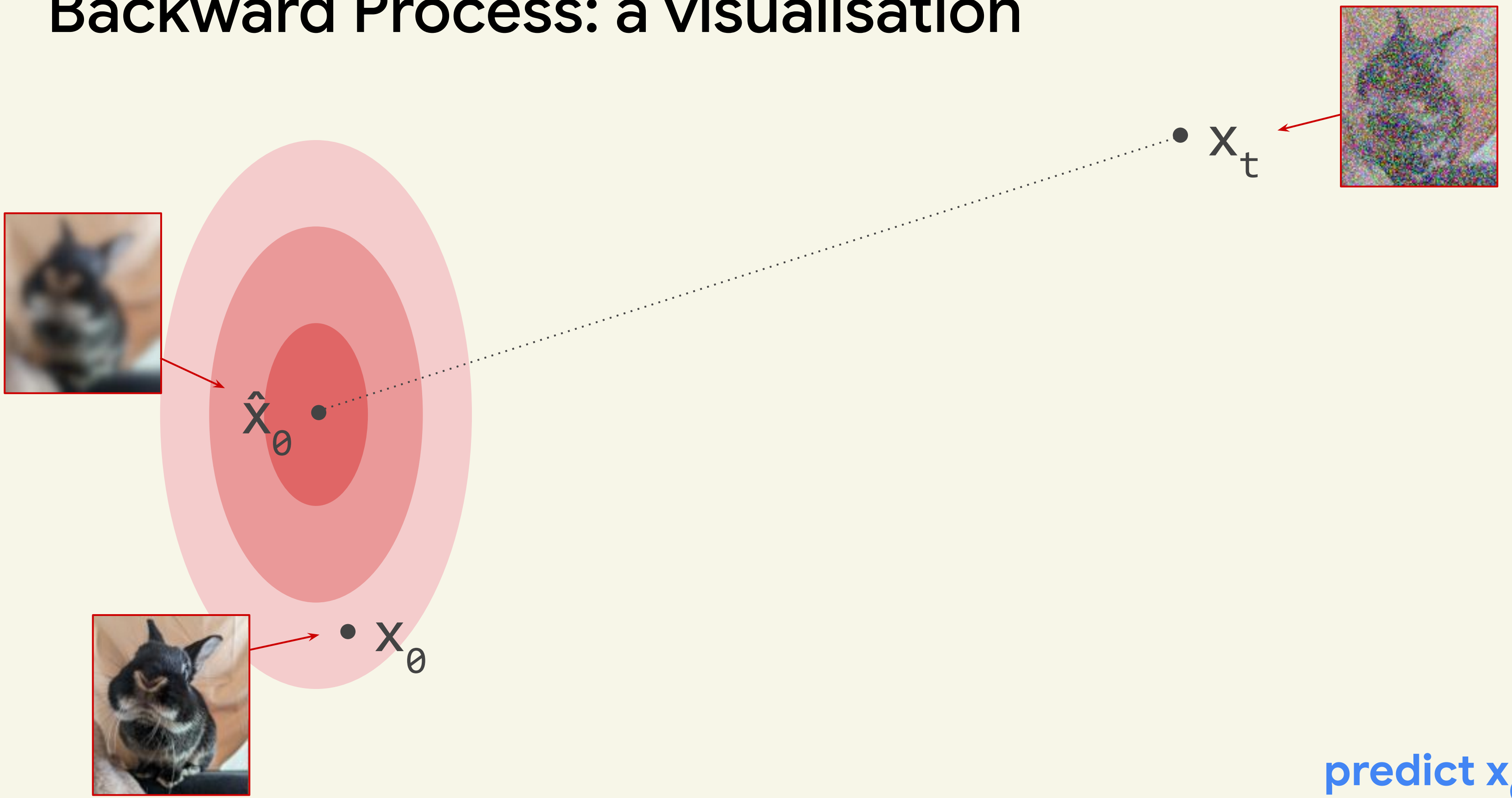
Backward Process: a visualisation



predict x_0

Slide credit: Sander Dieleman

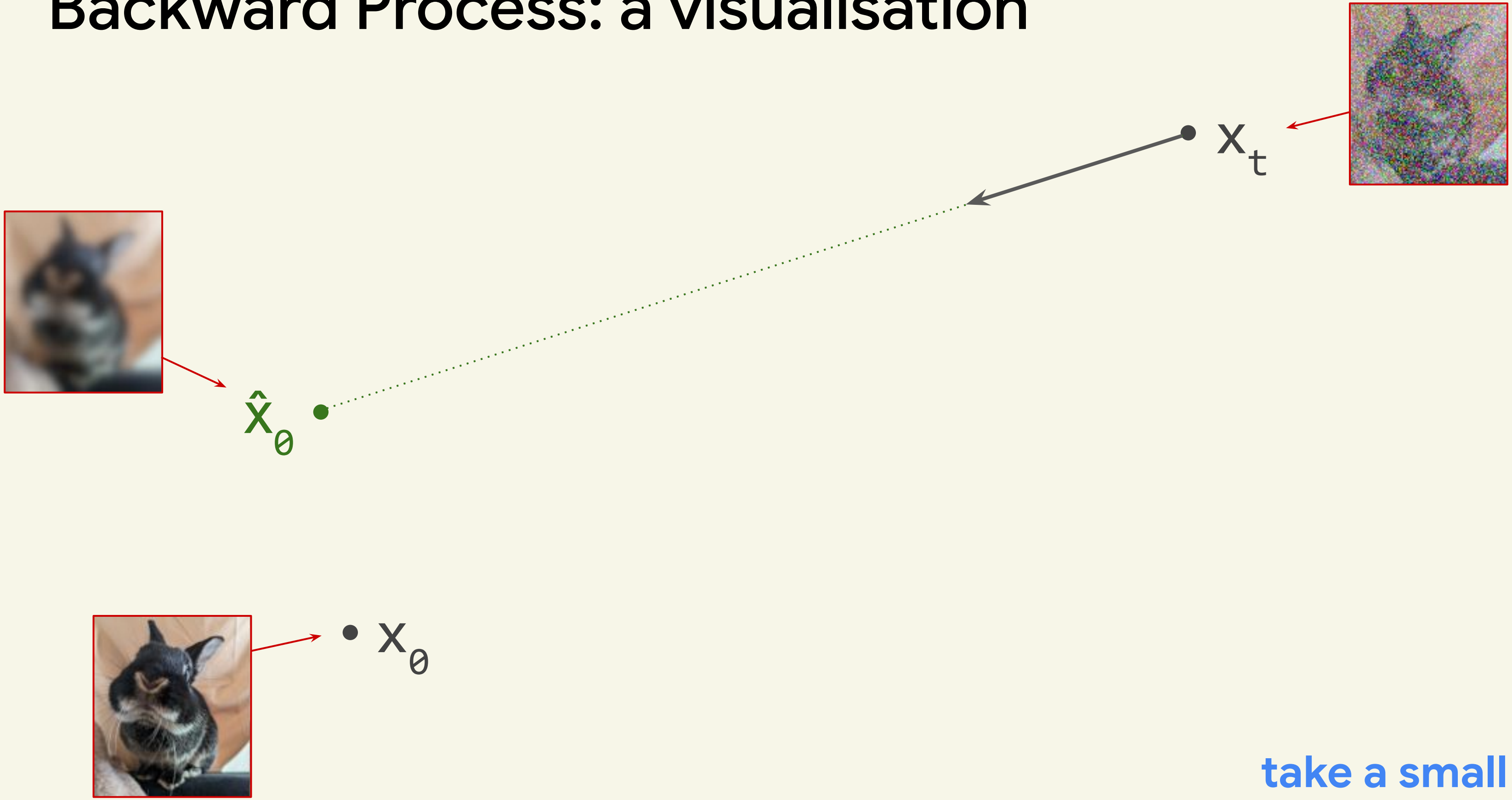
Backward Process: a visualisation



predict x_0

Slide credit: Sander Dieleman

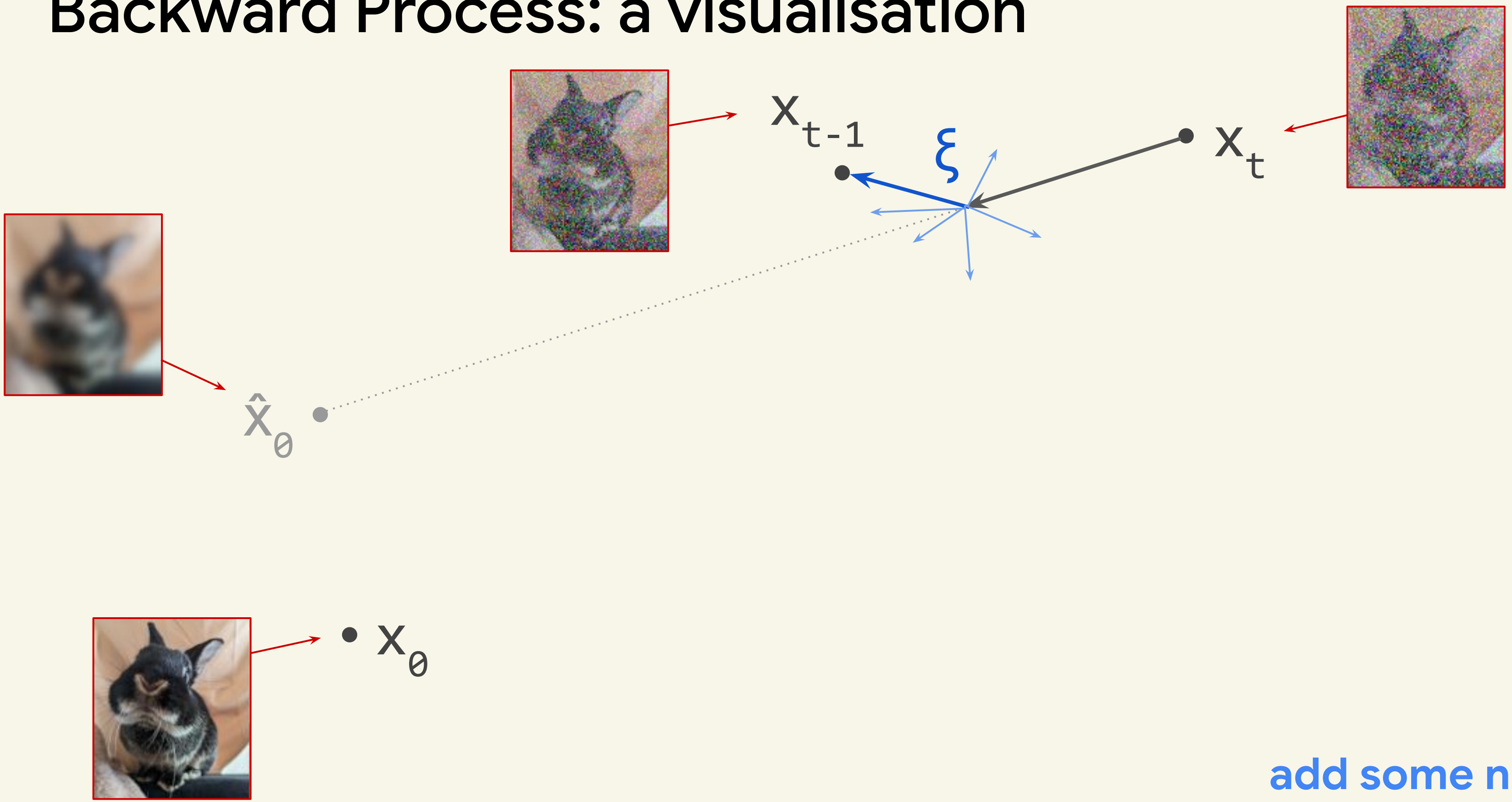
Backward Process: a visualisation



take a small step

Slide credit: Sander Dieleman

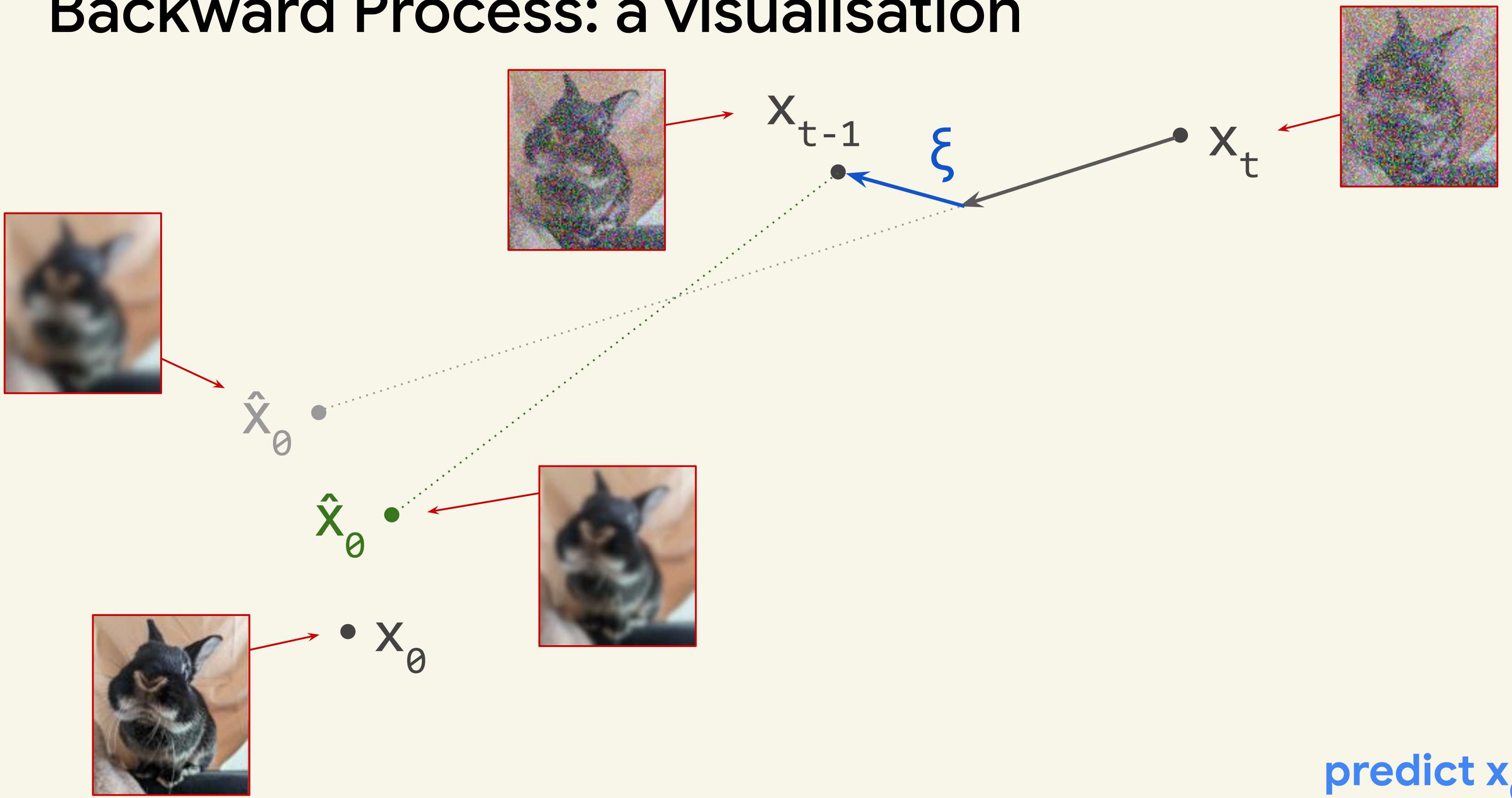
Backward Process: a visualisation



add some noise

Slide credit: Sander Dieleman

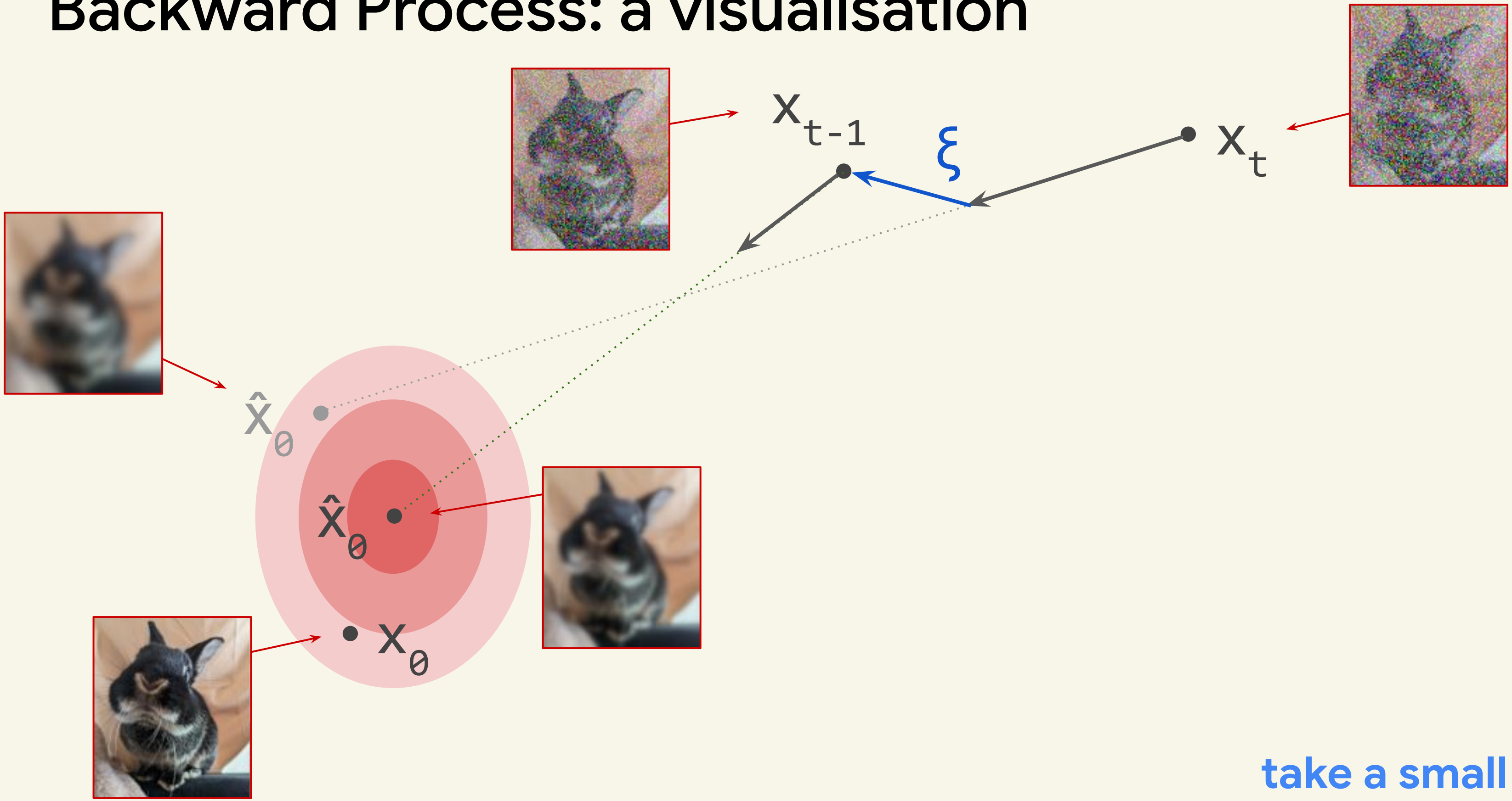
Backward Process: a visualisation



predict x_0

Slide credit: Sander Dieleman

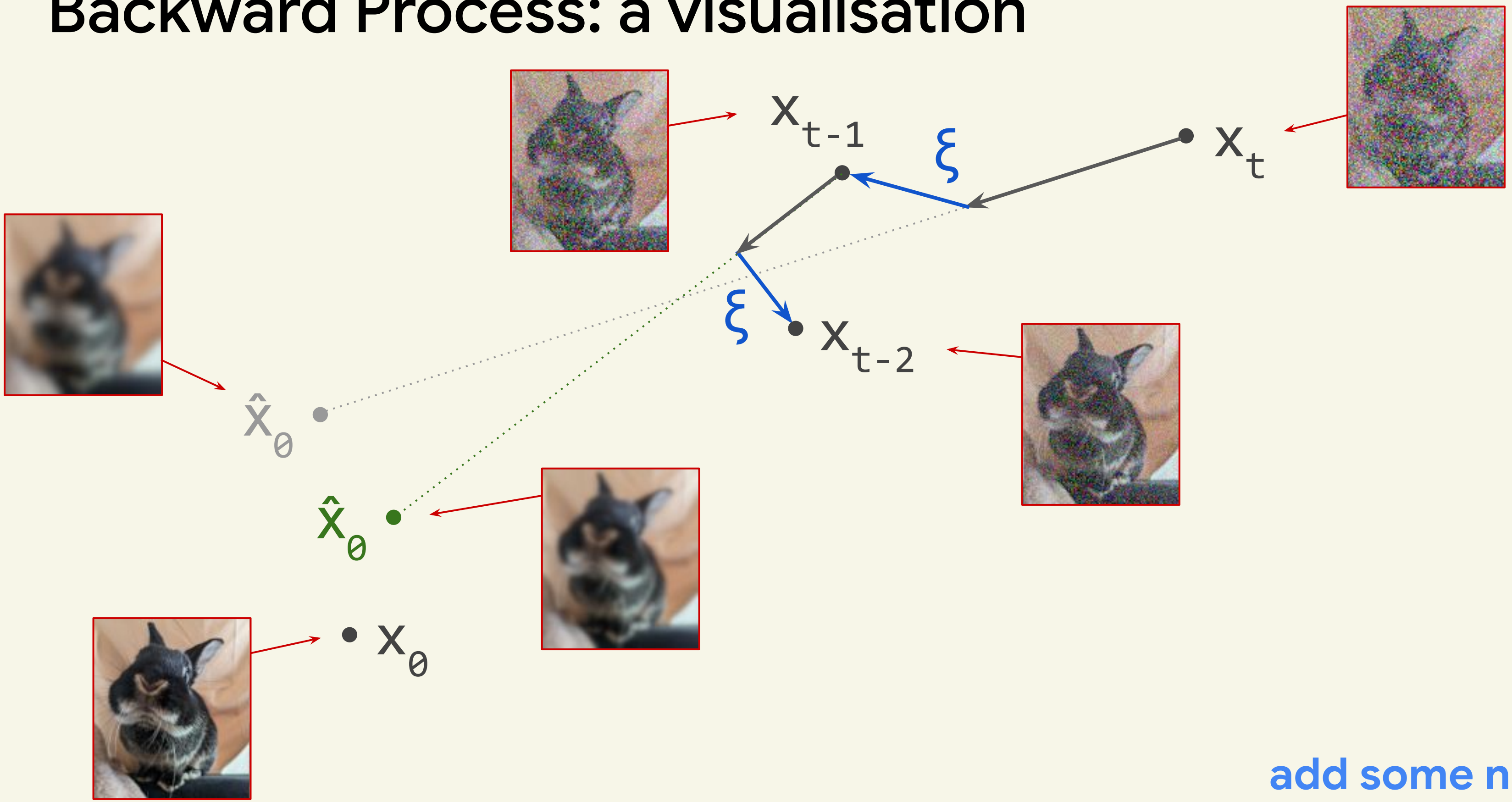
Backward Process: a visualisation



take a small step

Slide credit: Sander Dieleman

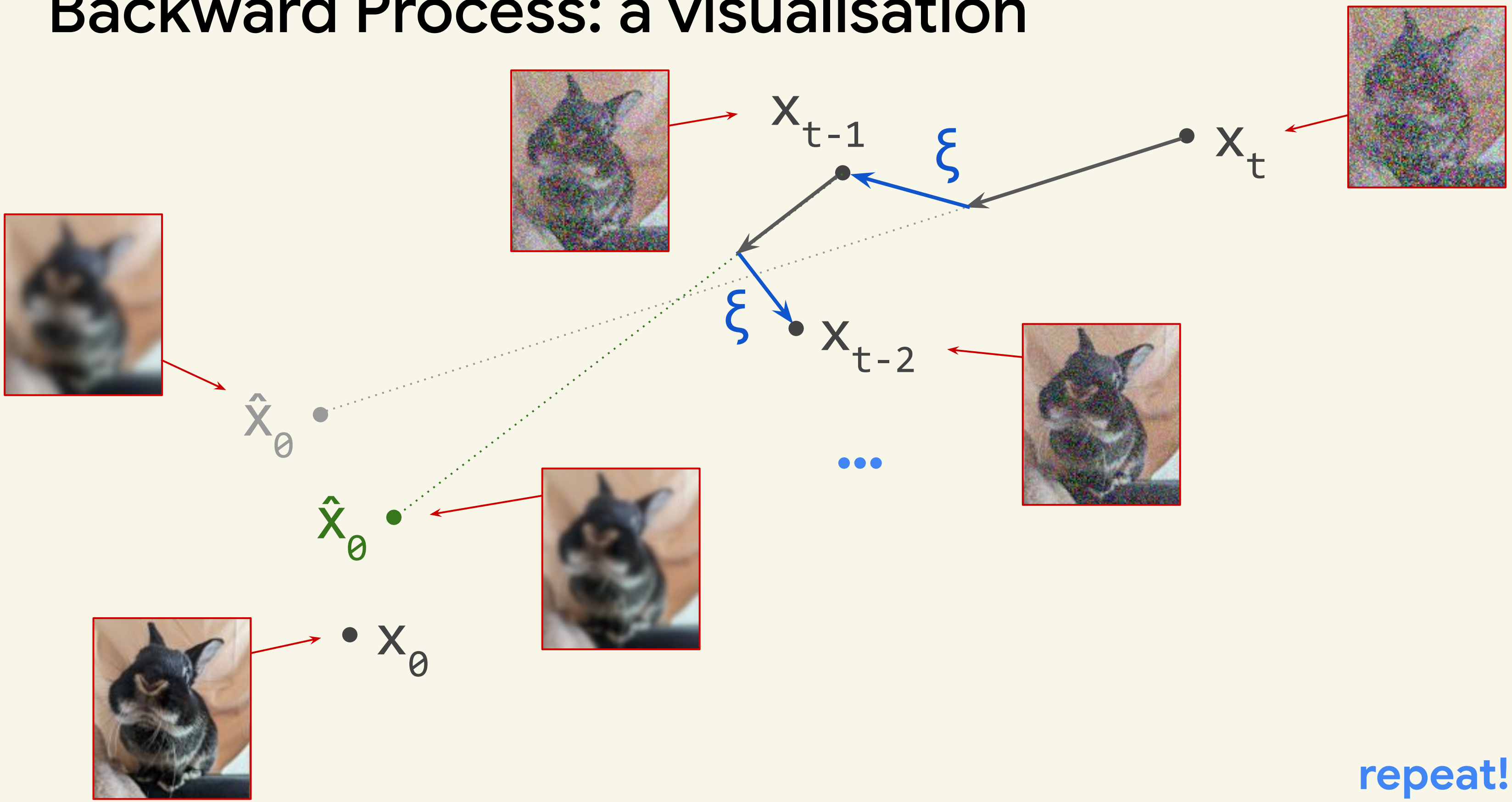
Backward Process: a visualisation



add some noise

Slide credit: Sander Dieleman

Backward Process: a visualisation



Slide credit: Sander Dieleman

5 Diffusion Models at Scale!

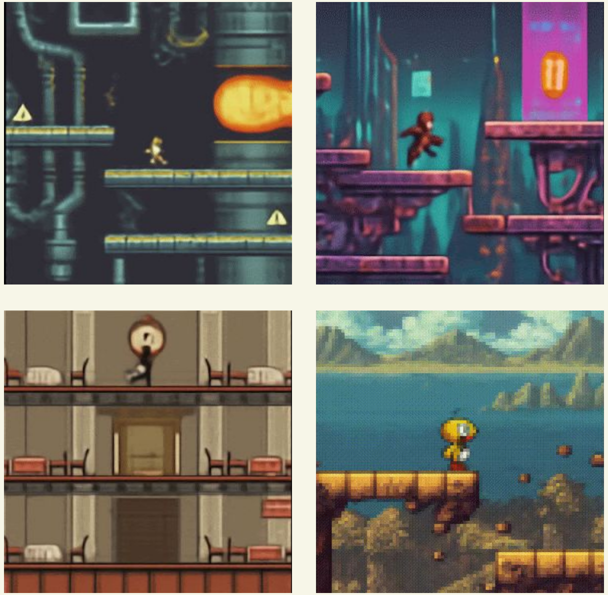
Genie: action => video!

Genie is an interactive generative model that is real-time playable.

Genie: action => video!

Genie is an interactive generative model that is real-time playable.

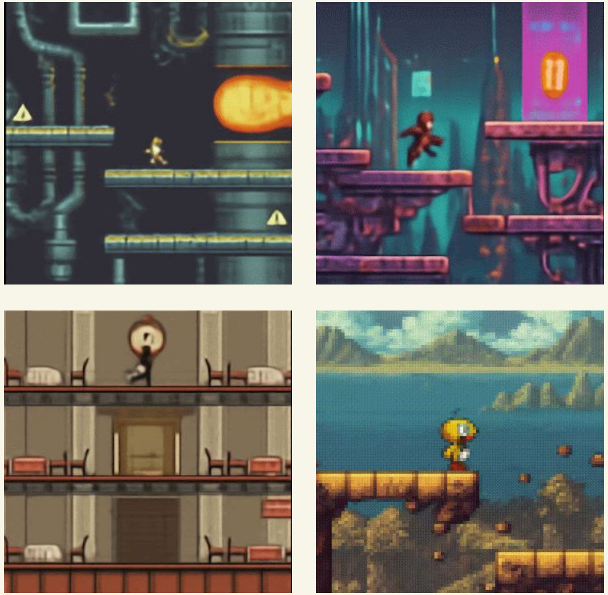
Genie1, Pre-diffusion (2024)

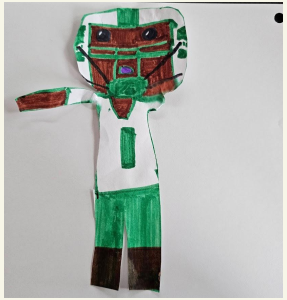


Genie: action => video!

Genie is an interactive generative model that is real-time playable.

Genie1, Pre-diffusion (2024)

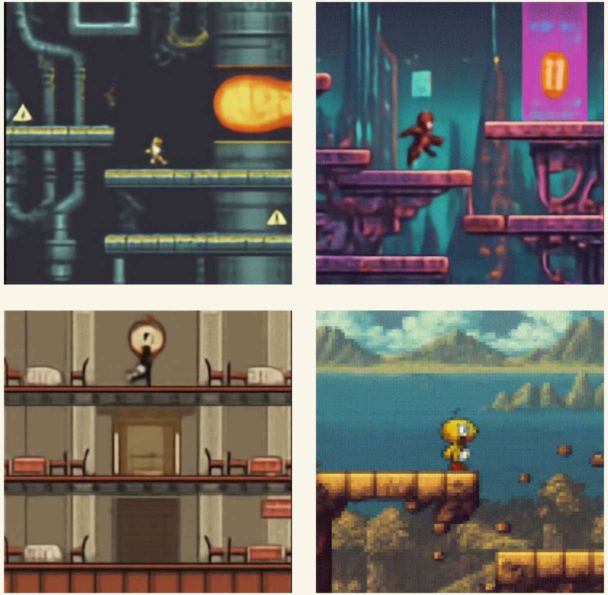




Genie: action => video!

Genie is an interactive generative model that is real-time playable.

Genie1, Pre-diffusion (2024)

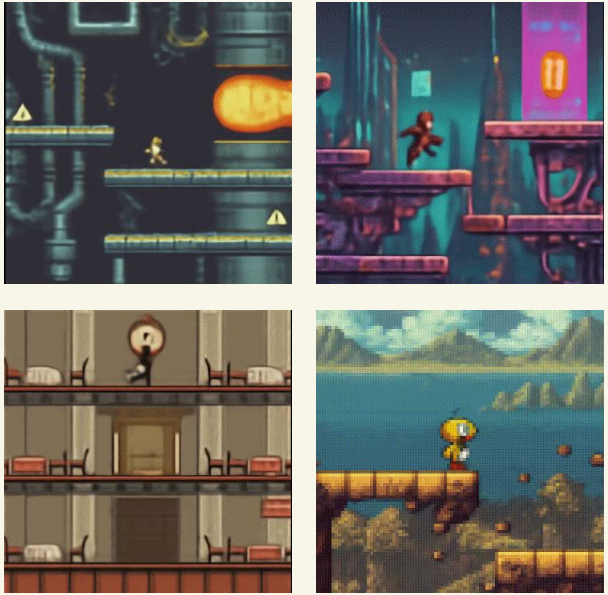


Google

Genie: action => video!

Genie is an interactive generative model that is real-time playable.

Genie1, Pre-diffusion (2024)



Genie2, Post-diffusion (2025)

Veo3

TikTok

Search

For You

Explore

Following

Upload

LIVE

Profile

More

Log in

Company

Program

Terms & Policies

© 2025 TikTok

...

2.5M

8520

200.9K

520.7K

00:00 / 00:32

yetivloglife

You may like

Bigfoot & Yeti keeping it cool in t...

bigfoottravels

2.1M · 6-5

@Brutus cant mes

yetivloglife

254.8K

What just happened??...

yetivloglife

41.5K · 6-6

feeling m #bigfoot

gurillaguy

660.1K

Pinned

The Veo team is hiring,
come join us :)

@YugeTen
jimmyshi@google.com