Grooming Diffusion

Advanced diffusion

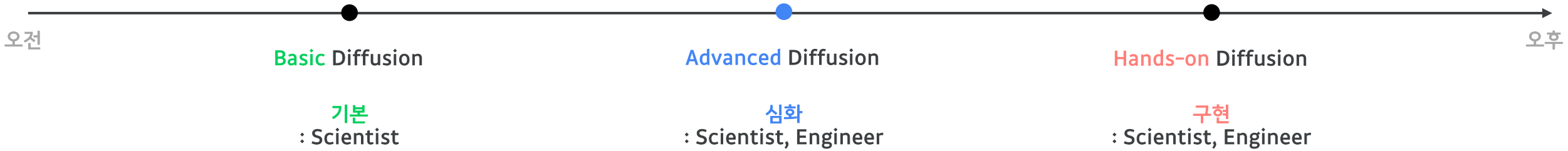
NAVER AI Lab

김준호

https://github.com/taki0112

NAVER AI LAB

NAVER AI LAB


오전
Basic Diffusion
기본
: Scientist
Advanced Diffusion
심화
: Scientist, Engineer
Hands-on Diffusion
구현
: Scientist, Engineer
오후

NAVER AI LAB

오전

Basic Diffusion

기본
: Scientist

Advanced Diffusion

심화
: Scientist, Engineer

Hands-on Diffusion

구현
: Scientist, Engineer

오후

2022

Stable diffusion, GALIP

Trending T2I

Evaluation

FID, Clip score

P2P, Dreambooth, Inversion

Editing

???

Code | HuggingFace

NAVER AI LAB

🤗 Diffusers provides pretrained diffusion models across multiple modalities, such as vision and audio, and serves as a modular toolbox for inference and training of diffusion models.

More precisely, 🤗 Diffusers offers:

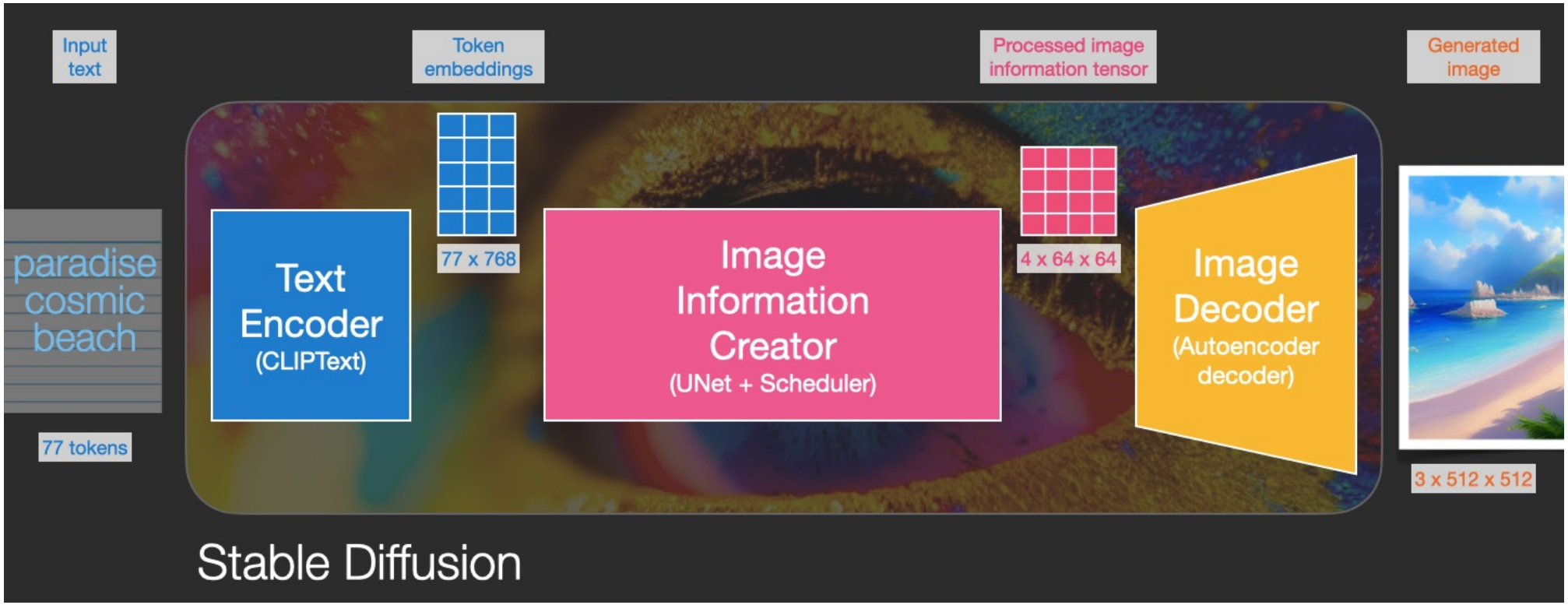
- State-of-the-art diffusion pipelines that can be run in inference with just a couple of lines of code (see src/diffusers/pipelines). Check this overview to see all supported pipelines and their corresponding official papers.
- Various noise schedulers that can be used interchangeably for the preferred speed vs. quality trade-off in inference (see src/diffusers/schedulers).
- Multiple types of models, such as UNet, can be used as building blocks in an end-to-end diffusion system (see src/diffusers/models).
- Training examples to show how to train the most popular diffusion model tasks (see examples, *e.g.* unconditional-image-generation).

github.com/huggingface/diffusers

NAVER AI LAB

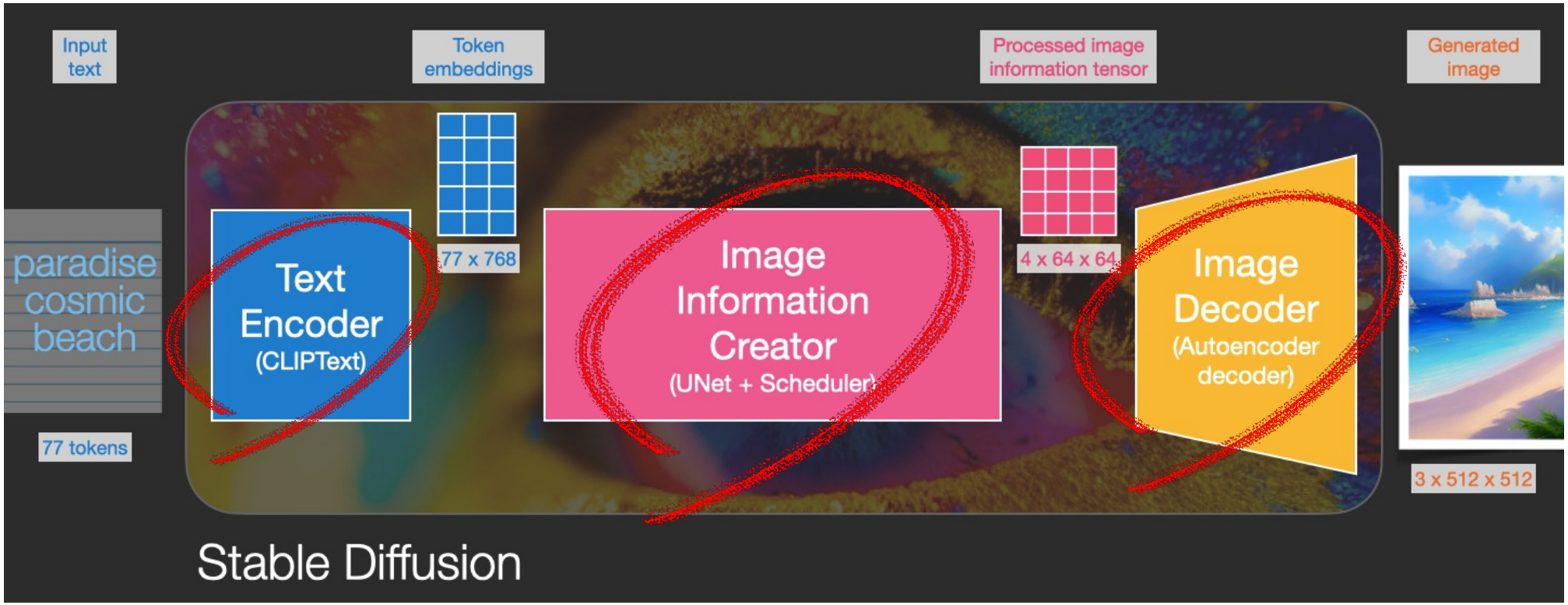
Stable Diffusion | Components

NAVER AI LAB



Stable Diffusion | Components

NAVER AI LAB



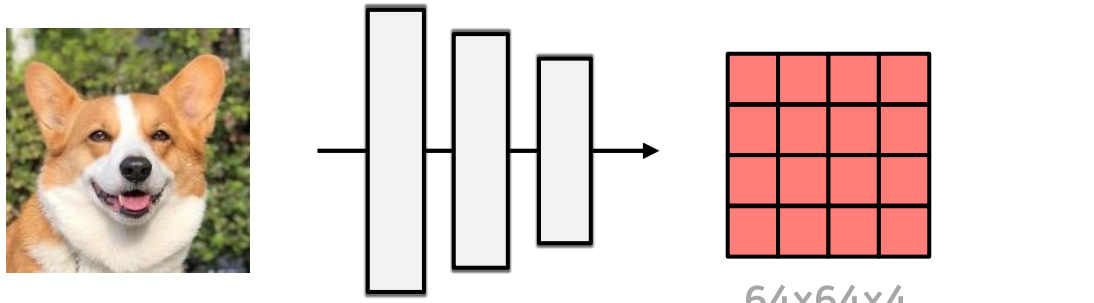
Stable Diffusion

Stable Diffusion | Autoencoder

NAVER AI LAB

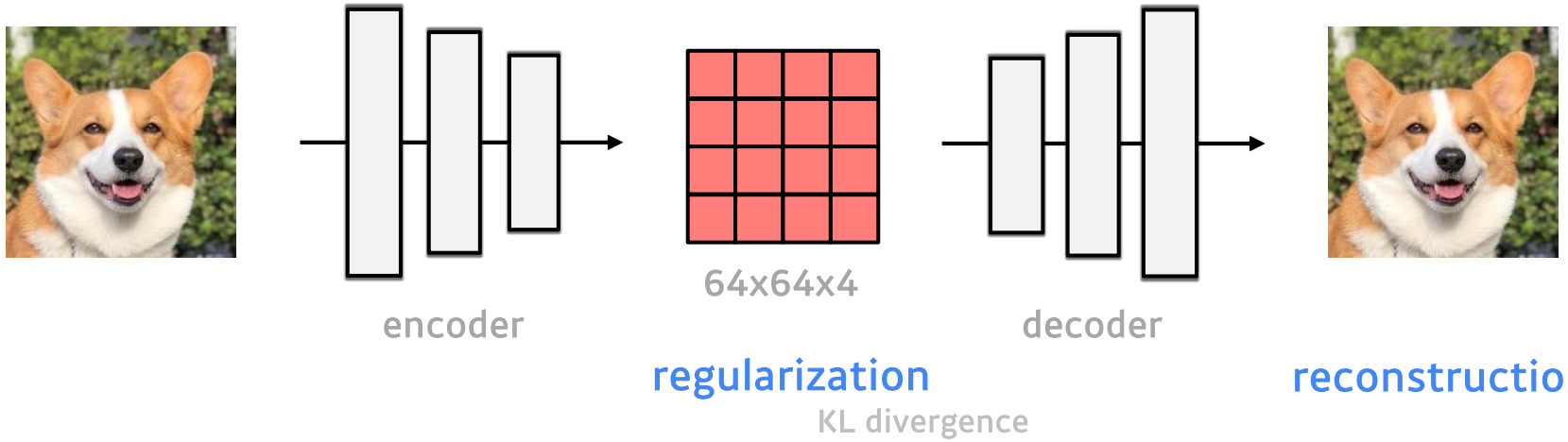
Stable Diffusion | Autoencoder

NAVER AI LAB



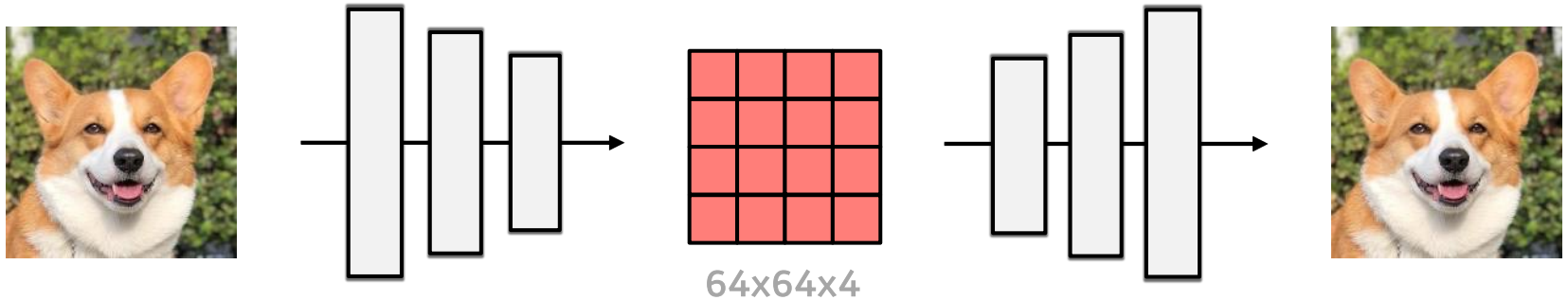
Stable Diffusion | Autoencoder

NAVER AI LAB



Stable Diffusion | Autoencoder

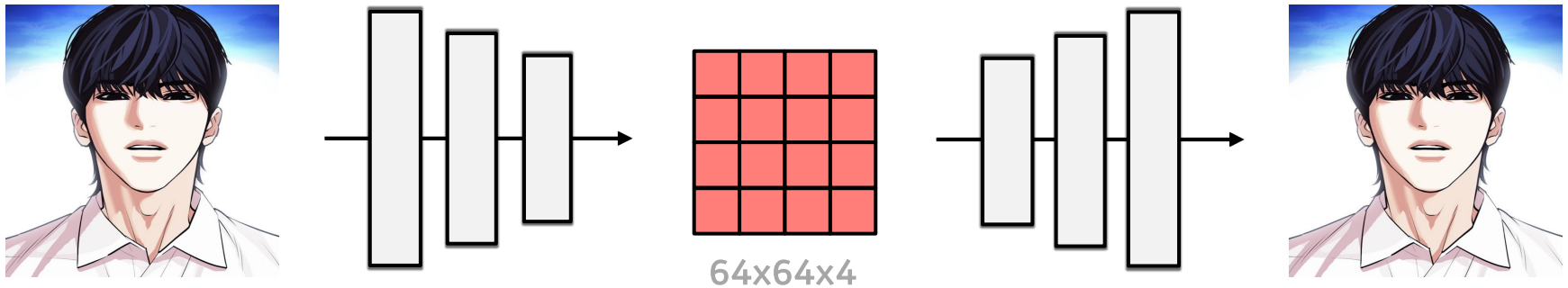
NAVER AI LAB



High-Resolution Image Synthesis with Latent Diffusion Models, CVPR 2022
github.com/rom1504/img2dataset

Stable Diffusion | Autoencoder

NAVER AI LAB



High-Resolution Image Synthesis with Latent Diffusion Models, CVPR 2022
github.com/rom1504/img2dataset

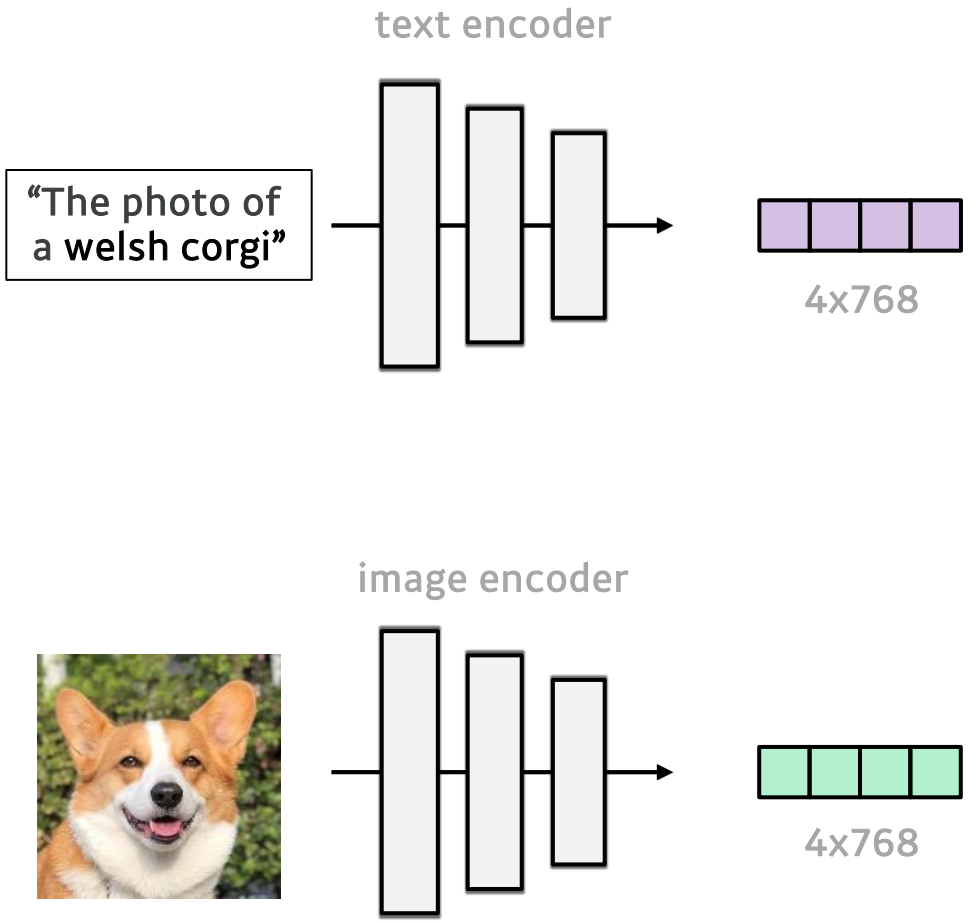
Stable Diffusion | CLIP

NAVER AI LAB

"The photo of a welsh corgi"

Stable Diffusion | CLIP

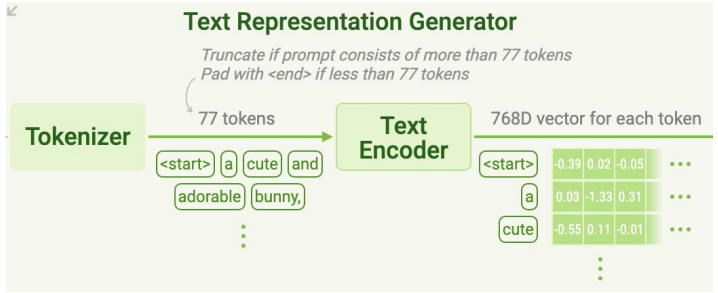
NAVER AI LAB

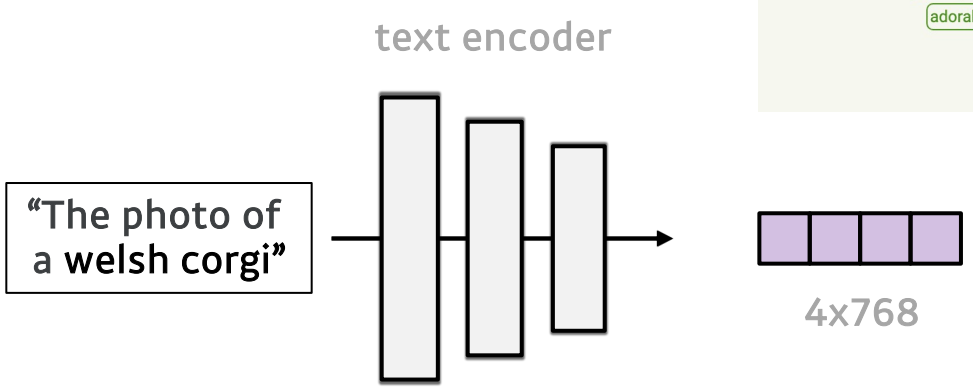


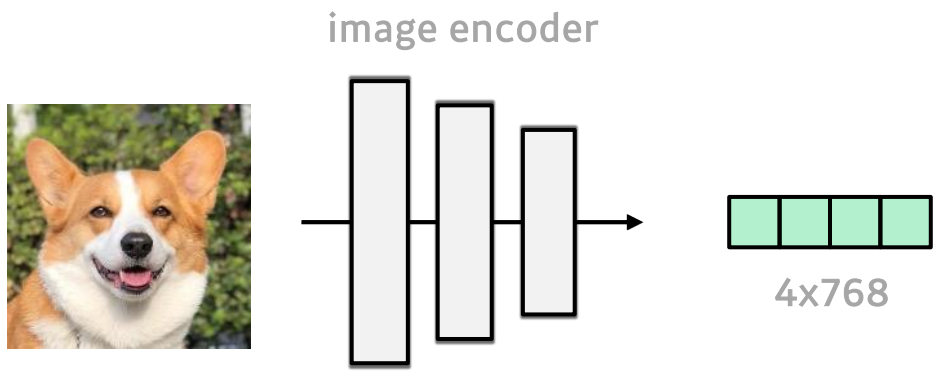
Learning Transferable Visual Models From Natural Language Supervision, ICML 2021

Stable Diffusion | CLIP

NAVER AI LAB

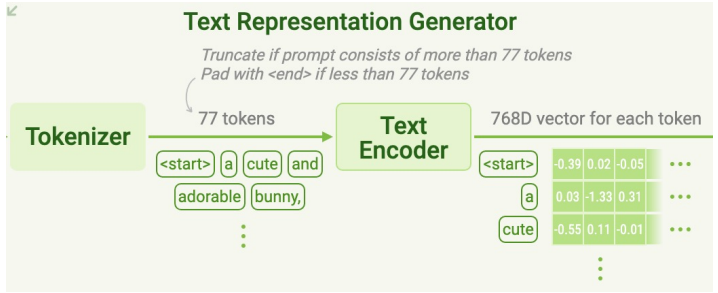


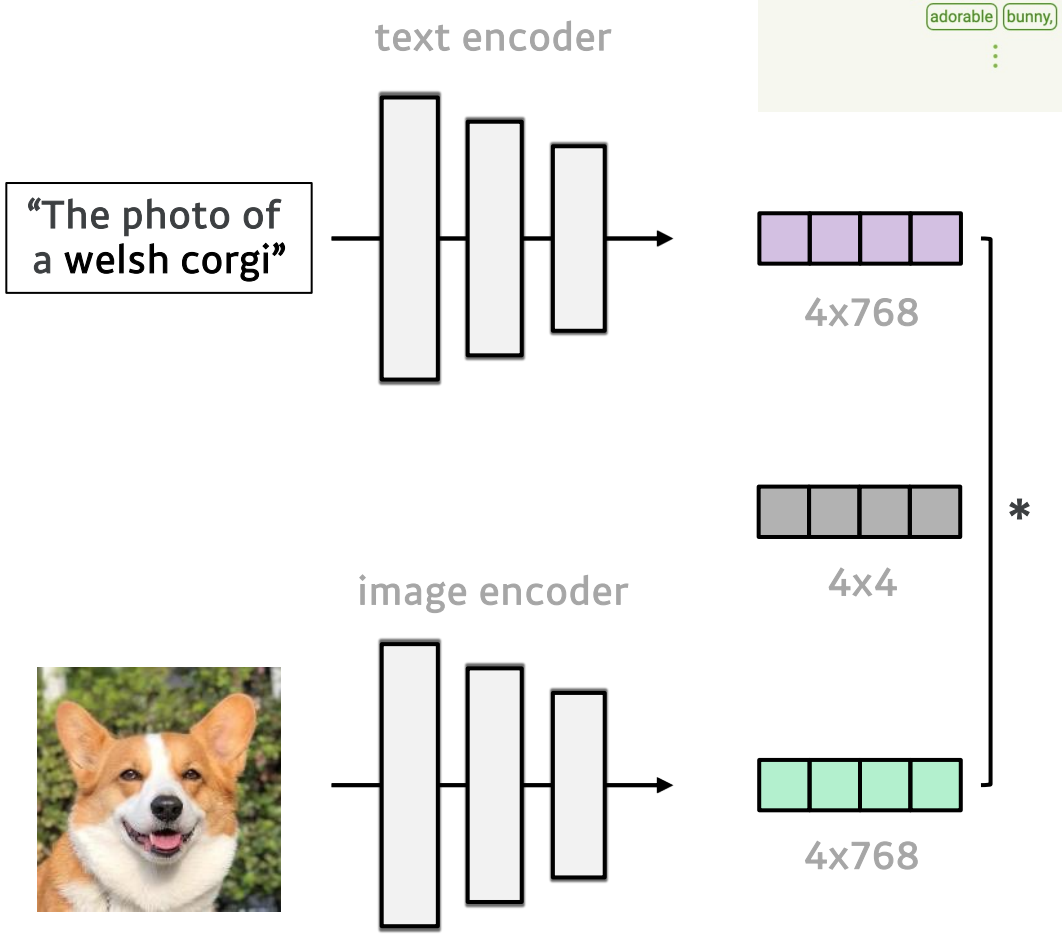




Learning Transferable Visual Models From Natural Language Supervision, ICML 2021

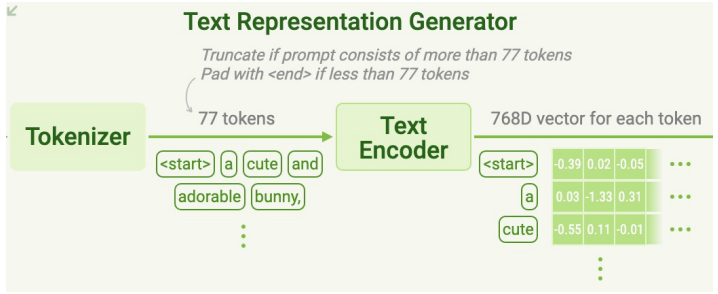
Stable Diffusion | CLIP





Learning Transferable Visual Models From Natural Language Supervision, ICML 2021

Stable Diffusion | CLIP



text encoder

"The photo of a welsh corgi"

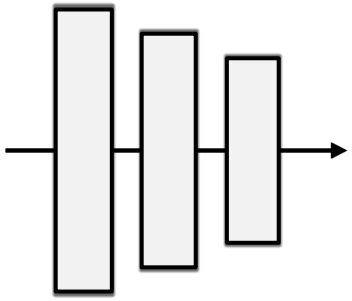
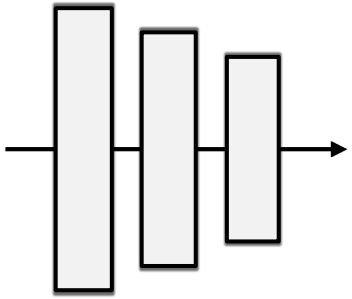
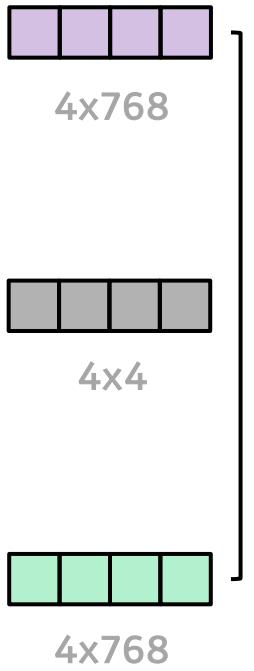


image encoder





text embed

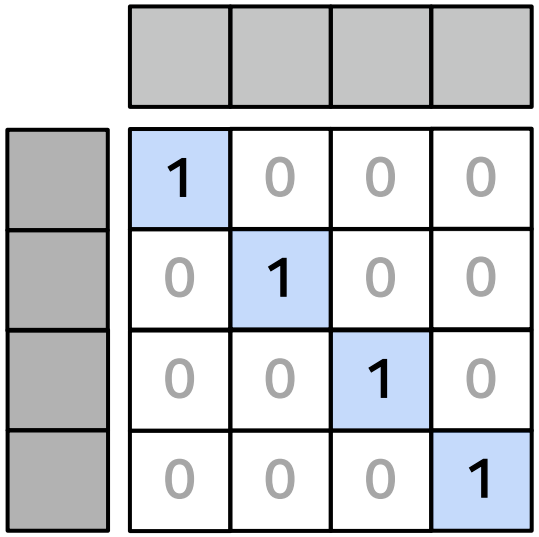
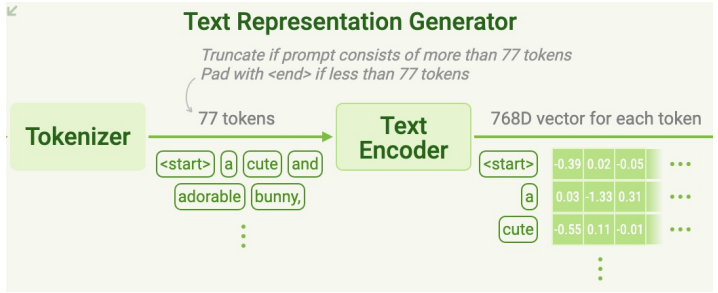
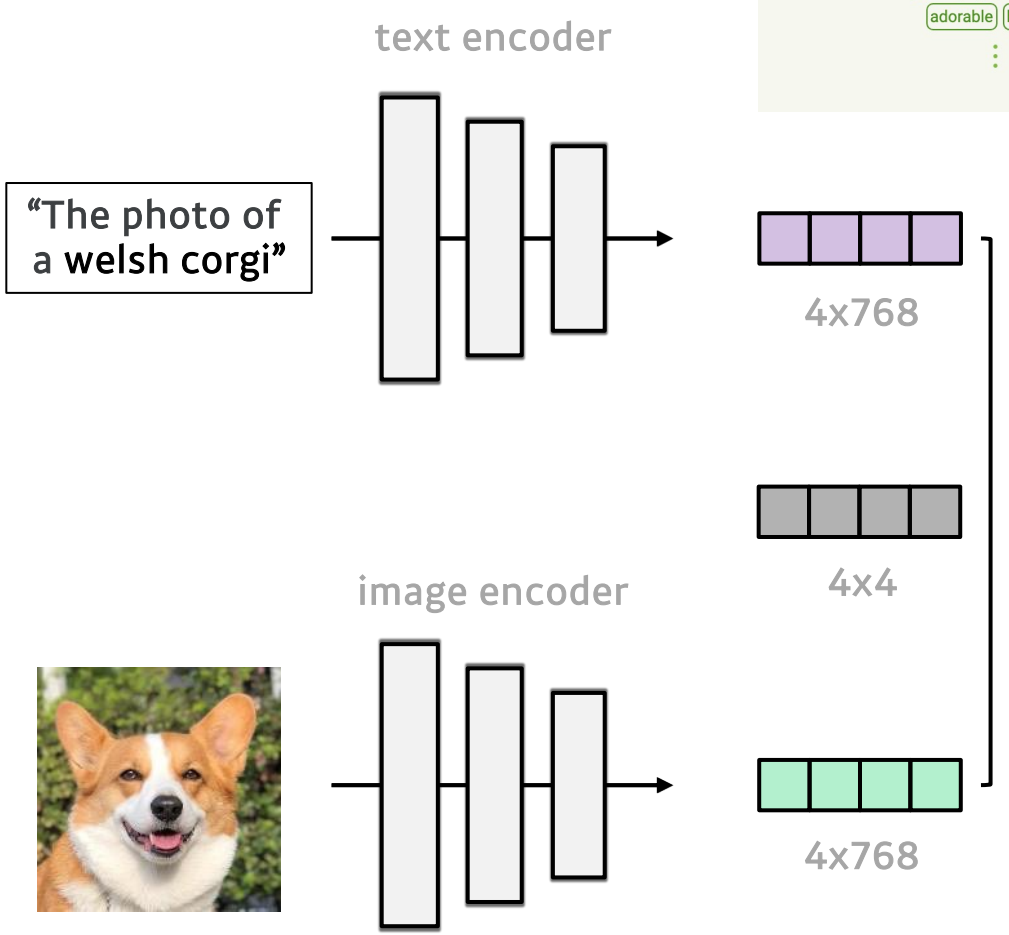


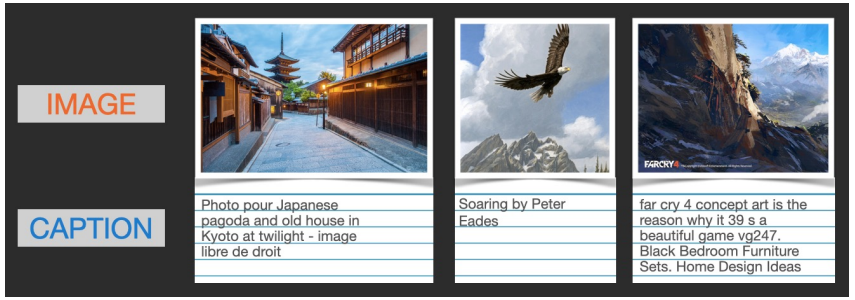
image embed

Stable Diffusion | CLIP

NAVER AI LAB







LAION-5B, 84.8TB

2B-en

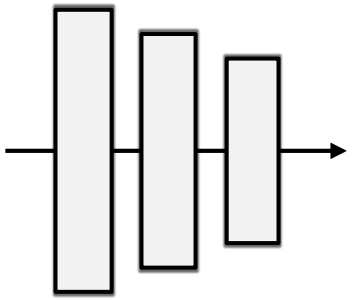
```python
import torch
from PIL import Image
import open_clip

model, _, preprocess = open_clip.create_model_and_transforms('ViT-B-32-quickgelu', pretrained='laio
tokenizer = open_clip.get_tokenizer('ViT-B-32-quickgelu')

image = preprocess(Image.open("CLIP.png")).unsqueeze(0)
text = tokenizer(["a diagram", "a dog", "a cat"])

with torch.no_grad(), torch.cuda.amp.autocast():
    image_features = model.encode_image(image)
    text_features = model.encode_text(text)
    image_features /= image_features.norm(dim=-1, keepdim=True)
    text_features /= text_features.norm(dim=-1, keepdim=True)

    text_probs = (100.0 * image_features @ text_features.T).softmax(dim=-1)

print("Label probs:", text_probs)  # prints: [[1., 0., 0.]]
```

github.com/mlfoundations/open_clip

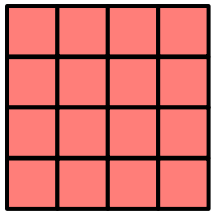
Learning Transferable Visual Models From Natural Language Supervision, ICML 2021

Stable Diffusion | Training

NAVER AI LAB



encoder



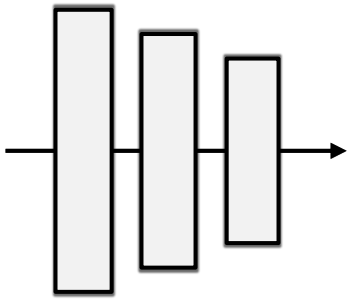
64x64x4

forward process

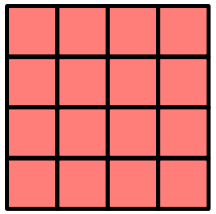
step 1

Stable Diffusion | Training

NAVER AI LAB



encoder

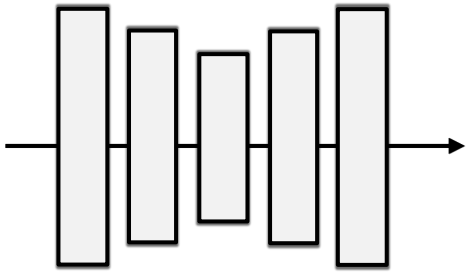


64x64x4

forward process

step 1

time embed



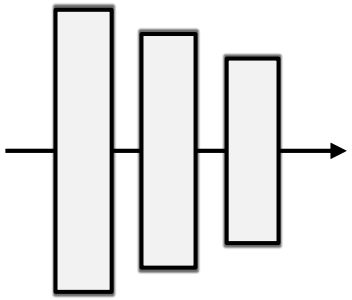
Unet

pred noise

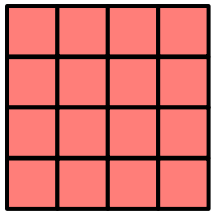
text embed
77x768

Stable Diffusion | Training

NAVER AI LAB



encoder



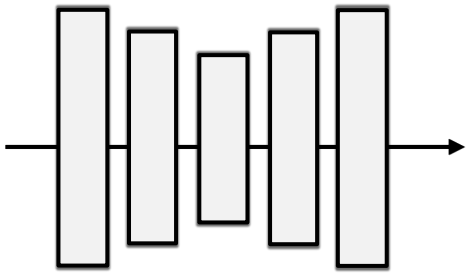
64x64x4

forward process

step 1

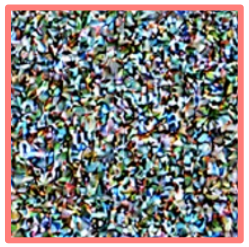
time embed

L2 loss



Unet

pred noise

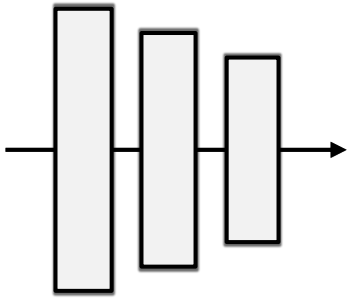


GT noise

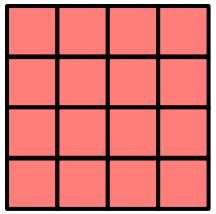
text embed
77x768

Stable Diffusion | Training

NAVER AI LAB



encoder

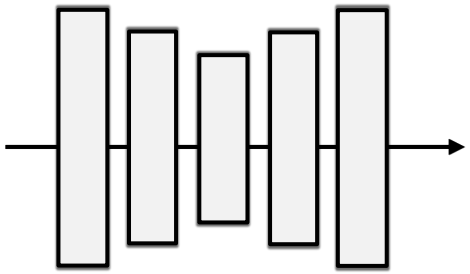


64x64x4

forward process

step 1

time embed



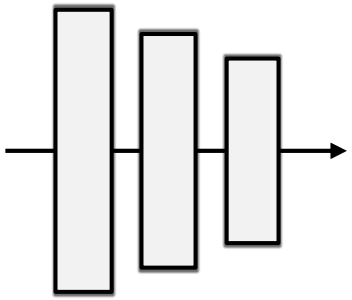
Unet

pred noise

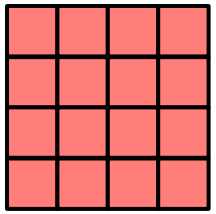
text embed
77x768

Stable Diffusion | Training

NAVER AI LAB



encoder

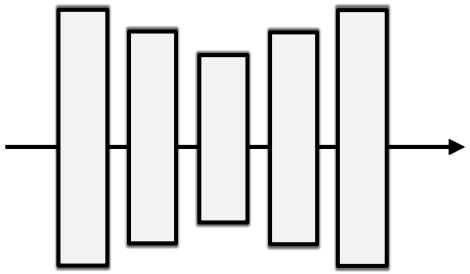


64x64x4

forward process

step 1

time embed



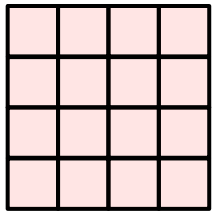
Unet

input latent

-

pred noise

=

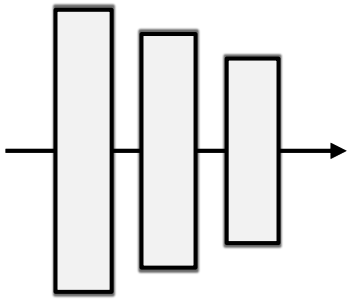


64x64x4

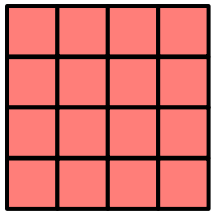
text embed
77x768

Stable Diffusion | Training

NAVER AI LAB



encoder

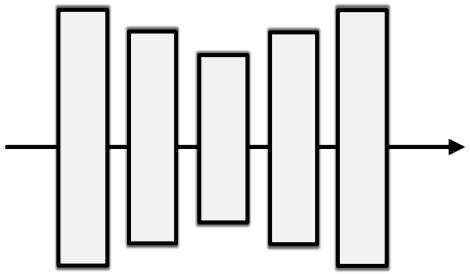


64x64x4

forward process

step 4

time embed



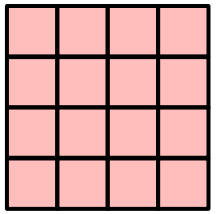
Unet

input latent

\-

pred noise

=

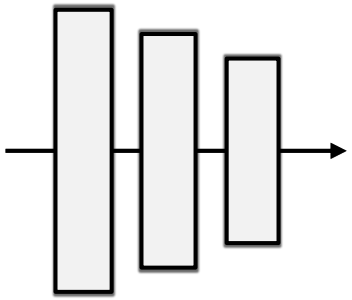


64x64x4

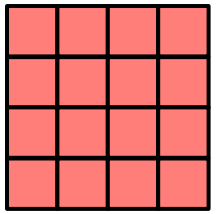
text embed
77x768

Stable Diffusion | Training

NAVER AI LAB



encoder

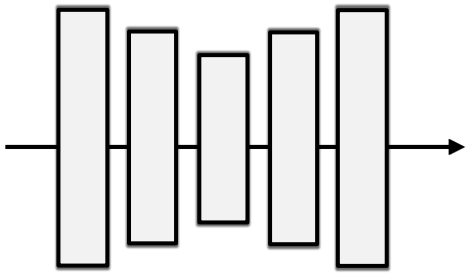


64x64x4

forward process

step 10

time embed



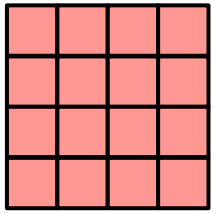
Unet

input latent

\-

pred noise

=

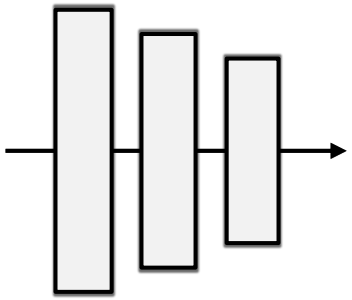


64x64x4

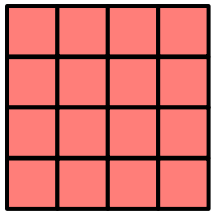
text embed
77x768

Stable Diffusion | Training

NAVER AI LAB



encoder

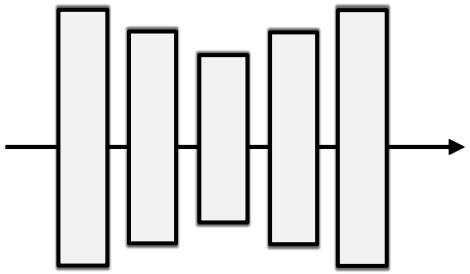


64x64x4

forward process

step 50

time embed



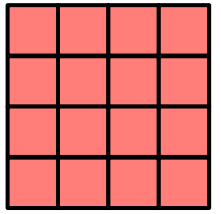
Unet

input latent

\-

pred noise

=

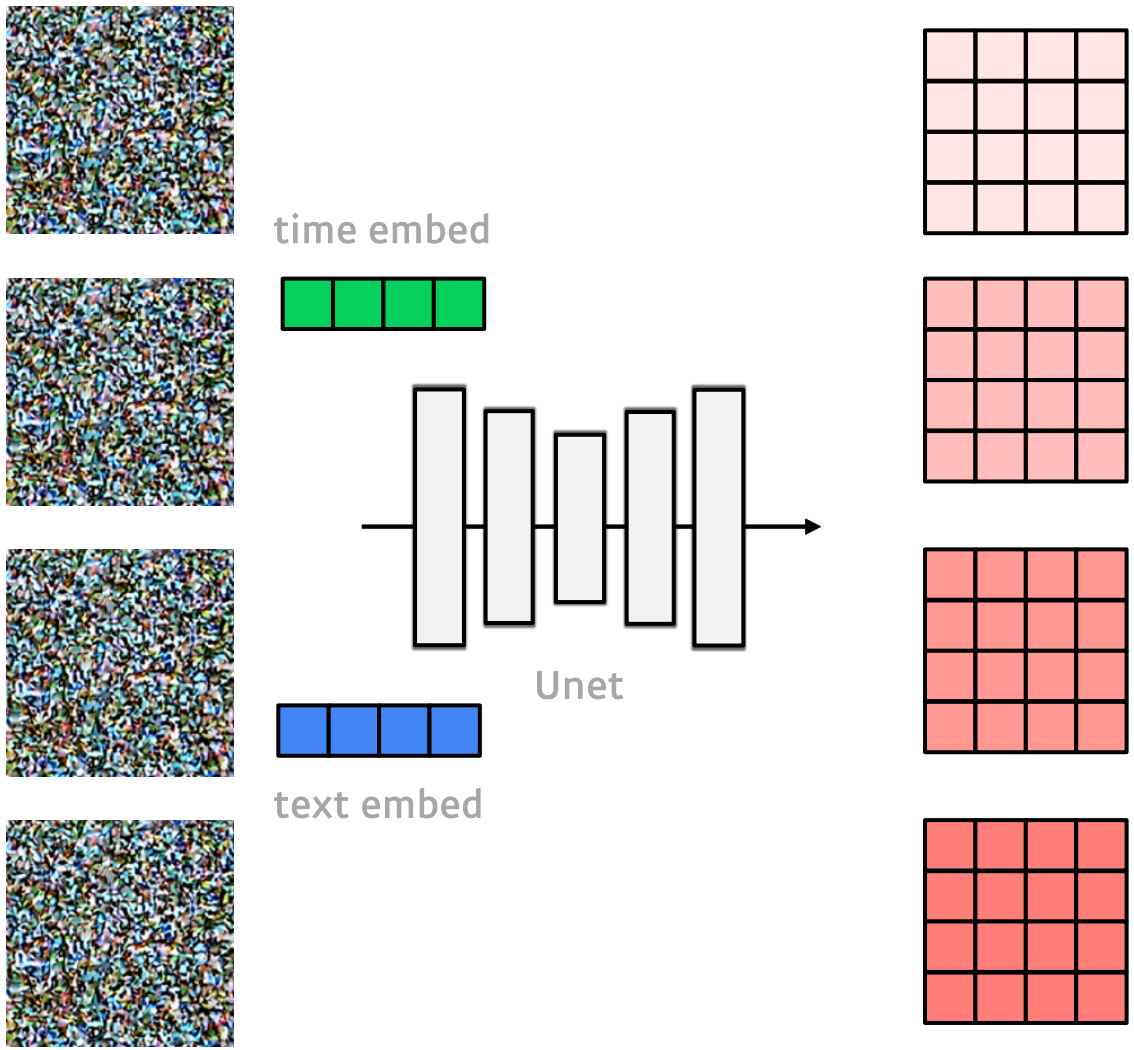


64x64x4

text embed
77x768

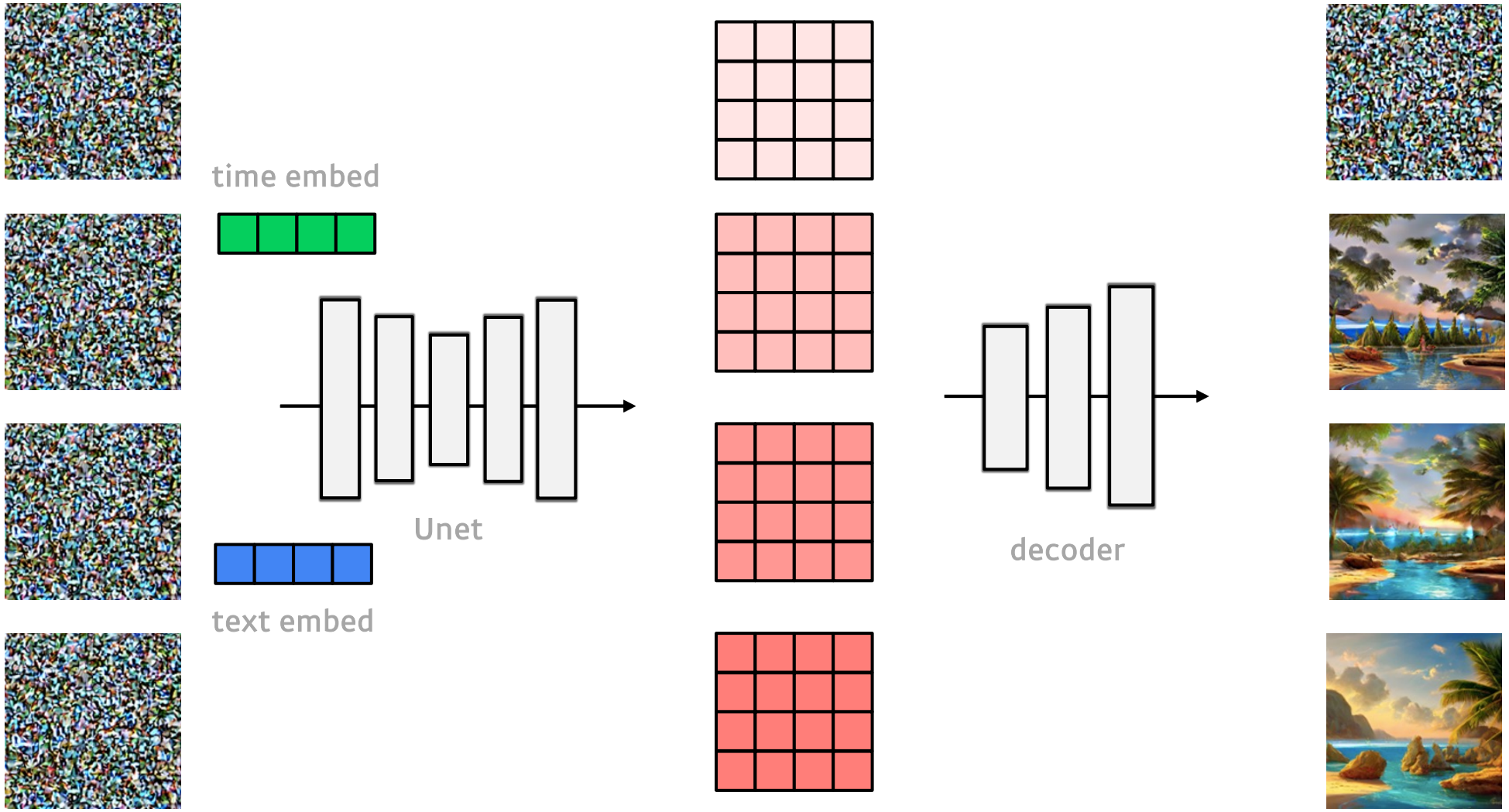
Stable Diffusion | Training results

NAVER AI LAB



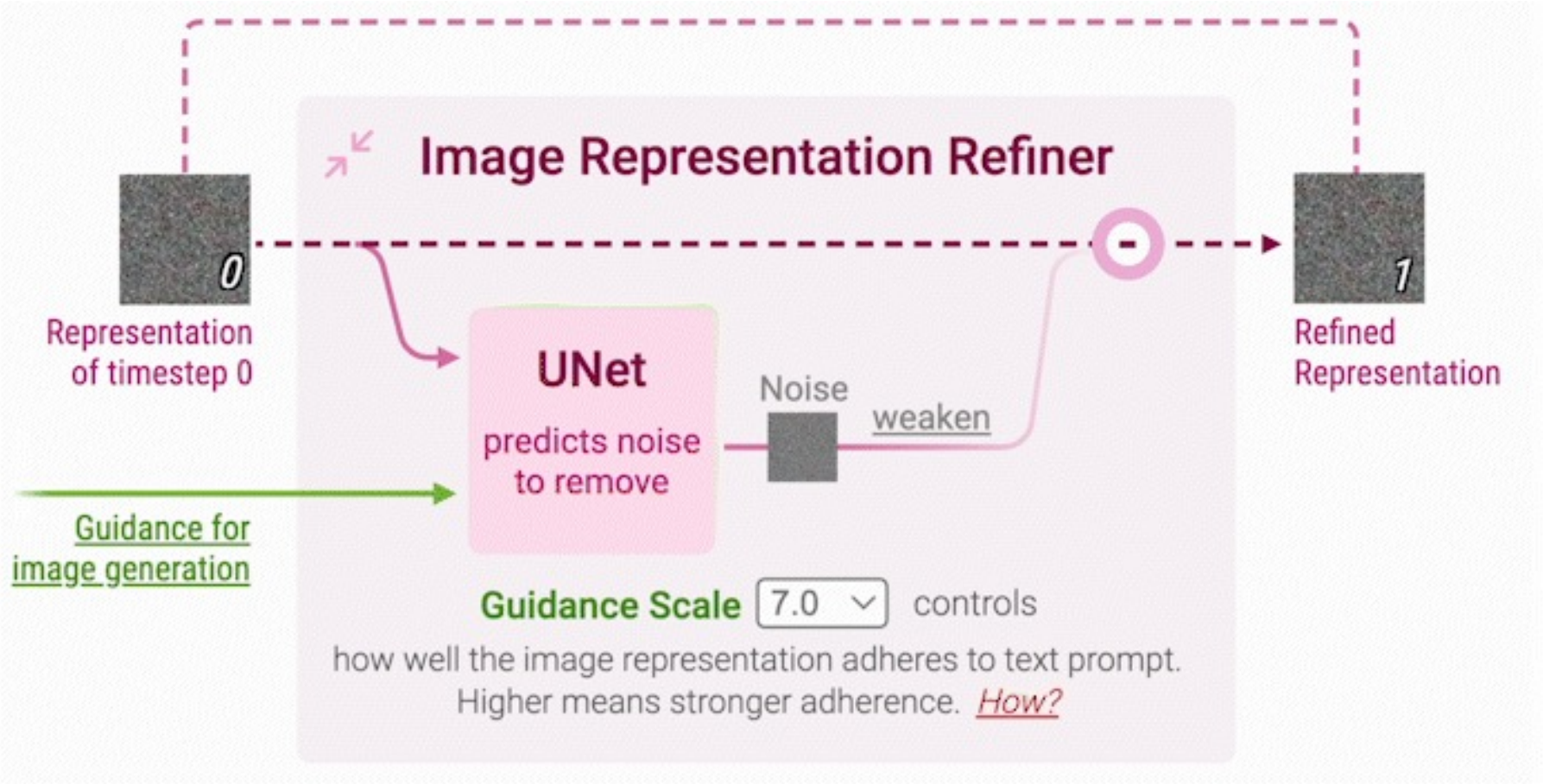
Stable Diffusion | Training results

NAVER AI LAB



Stable Diffusion | Training results

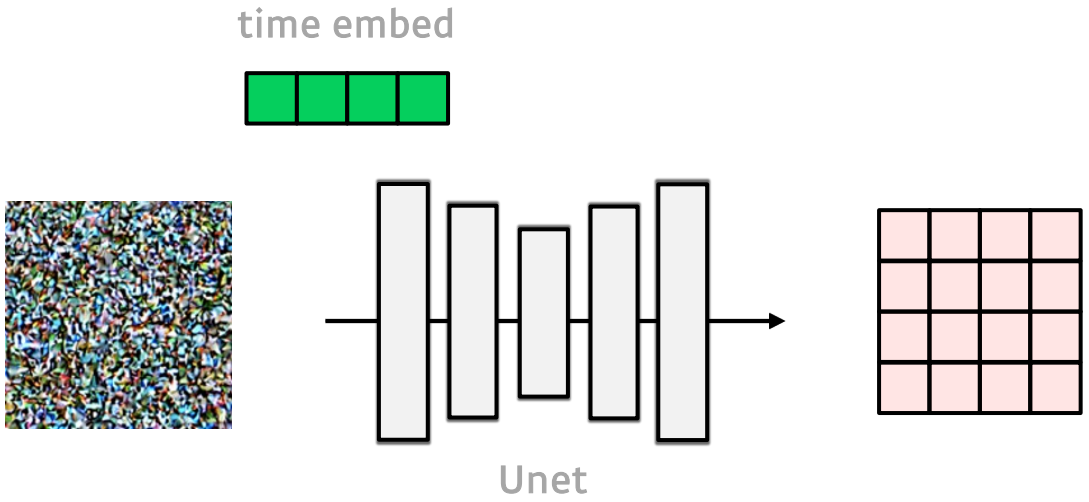
NAVER AI LAB



https://poloclub.github.io/diffusion-explainer/

Stable Diffusion | Conditioning

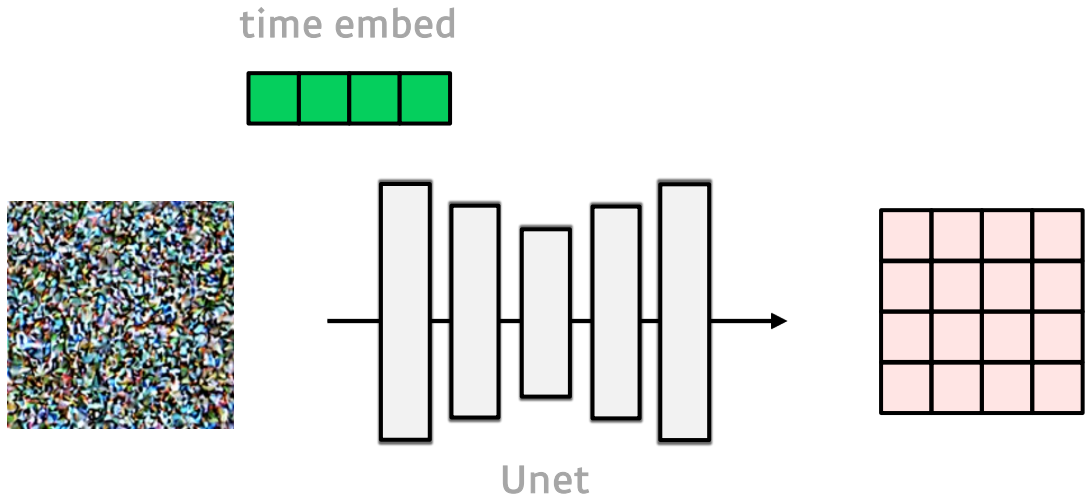
NAVER AI LAB



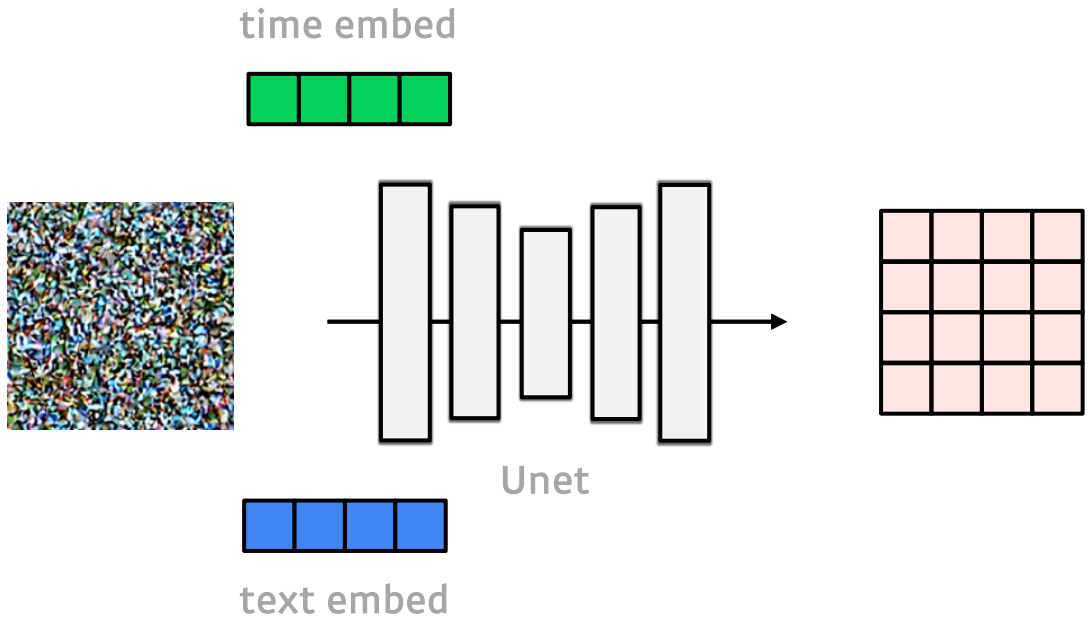
residual connection

Stable Diffusion | Conditioning

NAVER AI LAB



residual connection



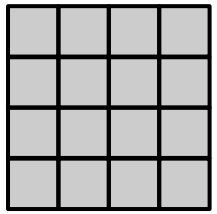
cross attention

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

$$Q = (N, d_k), K = (N, d_k), V = (N, d_v)$$

Stable Diffusion | Self-attention vs Cross-attention

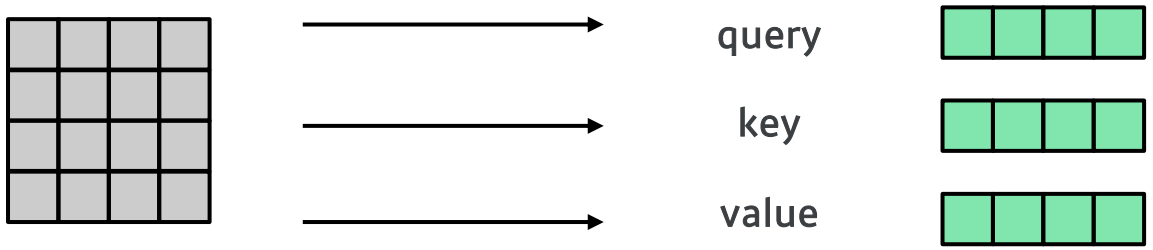
NAVER AI LAB



feature map

Stable Diffusion | Self-attention vs Cross-attention

NAVER AI LAB



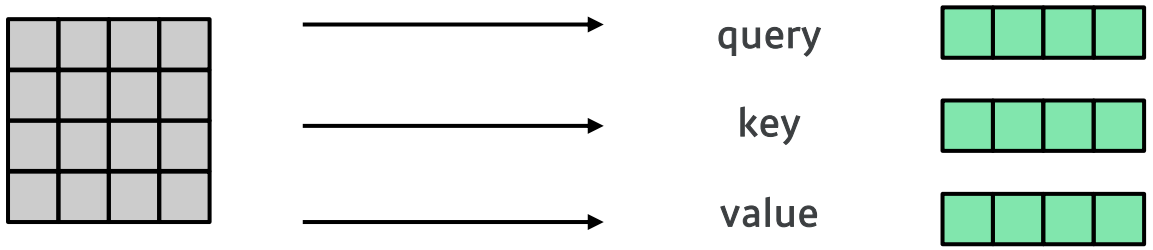
$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

$$Q = (N, d_k), K = (N, d_k), V = (N, d_v)$$

[N, seq_len, dim]

Stable Diffusion | Self-attention vs Cross-attention

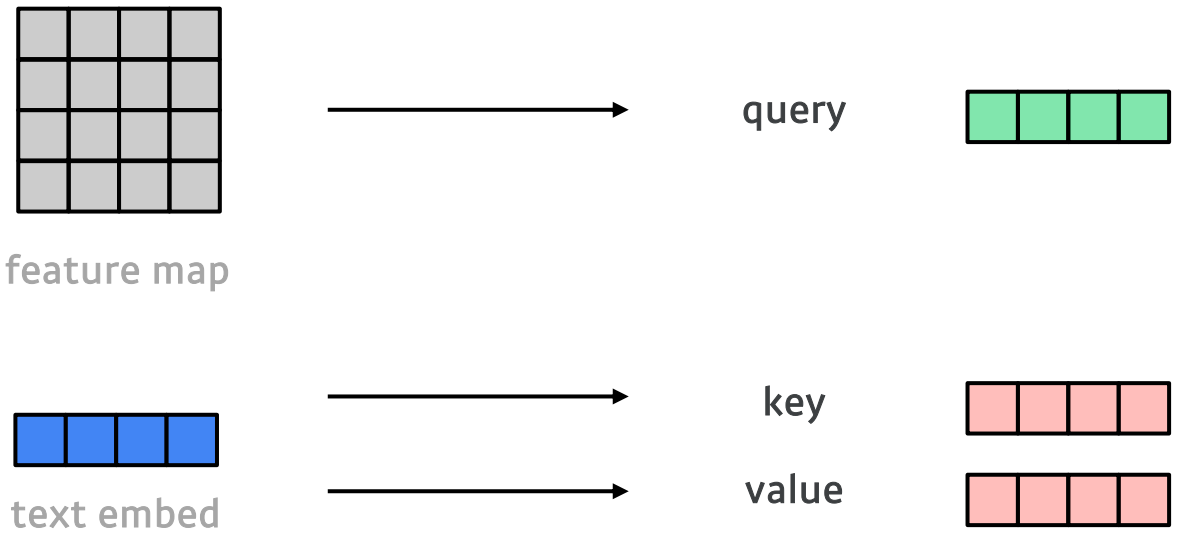
NAVER AI LAB



$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

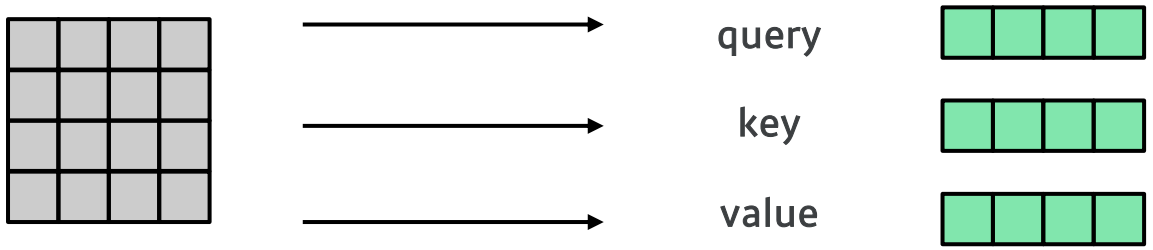
$$Q = (N, d_k), K = (N, d_k), V = (N, d_v)$$

[N, seq_len, dim]



Stable Diffusion | Self-attention vs Cross-attention

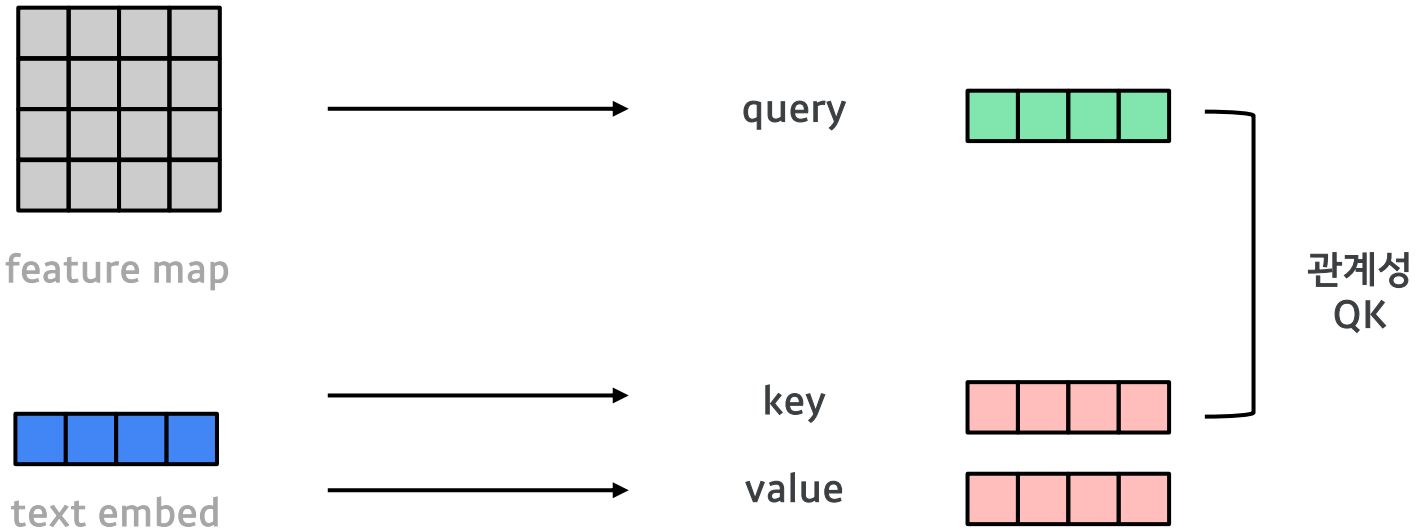
NAVER AI LAB



$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

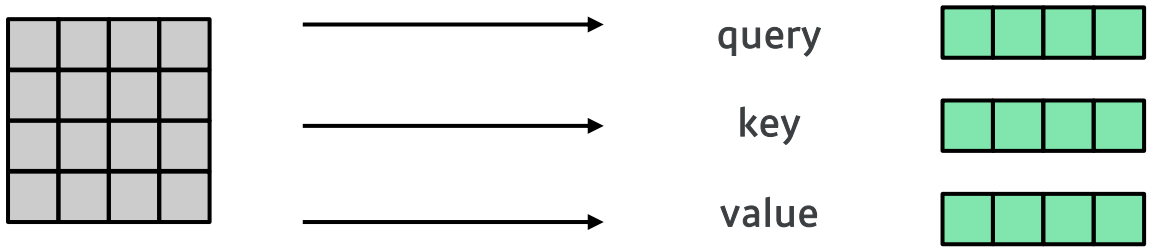
$$Q = (N, d_k), K = (N, d_k), V = (N, d_v)$$

[N, seq_len, dim]



Stable Diffusion | Self-attention vs Cross-attention

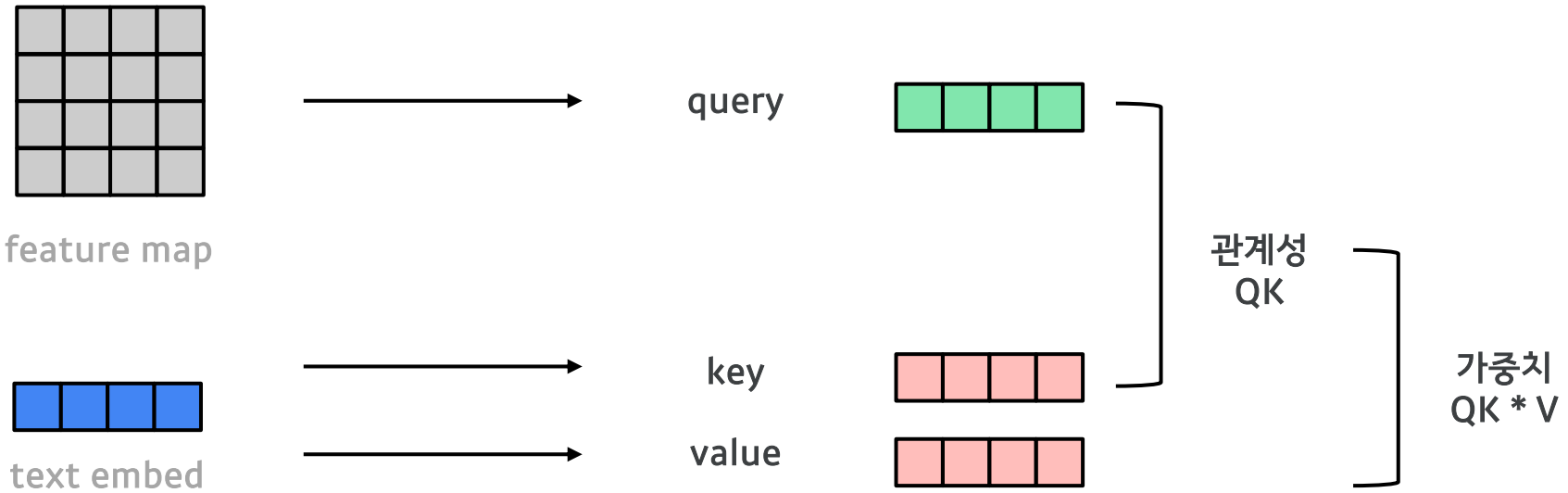
NAVER AI LAB



$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

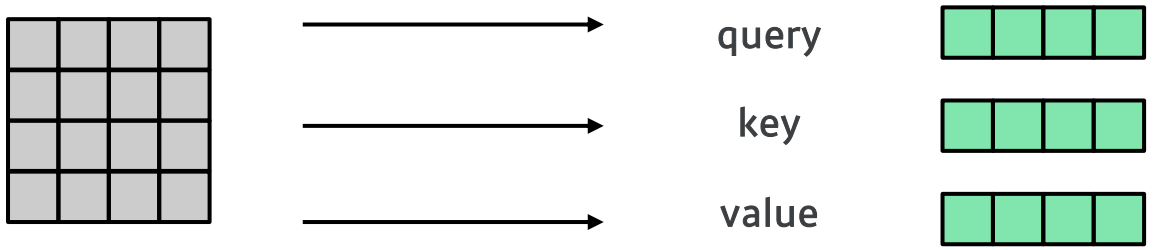
$$Q = (N, d_k), K = (N, d_k), V = (N, d_v)$$

[N, seq_len, dim]



Stable Diffusion | Self-attention vs Cross-attention

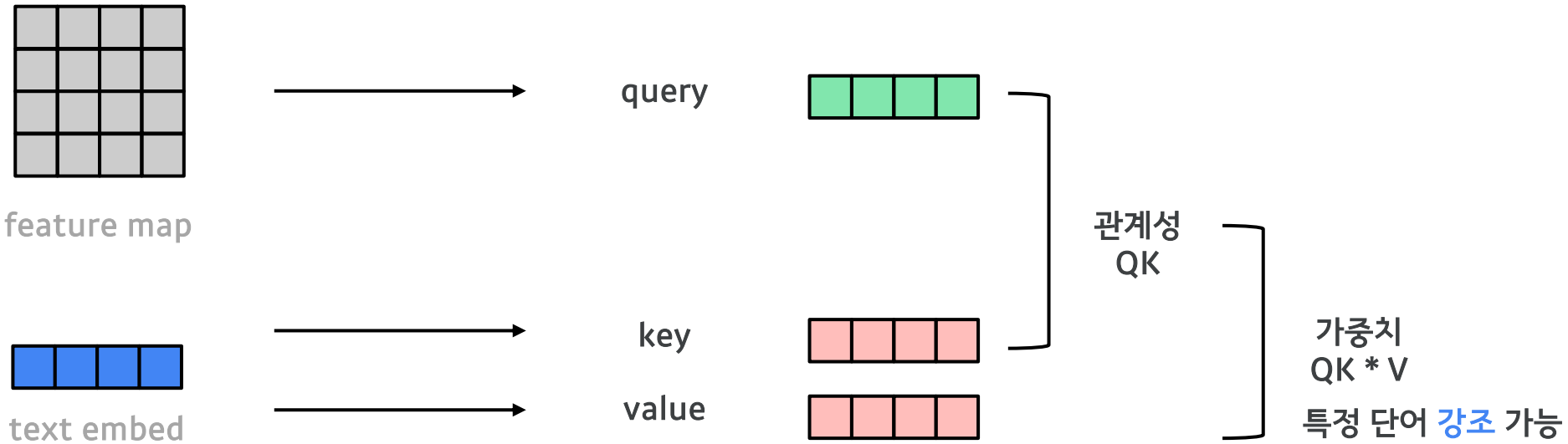
NAVER AI LAB



$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

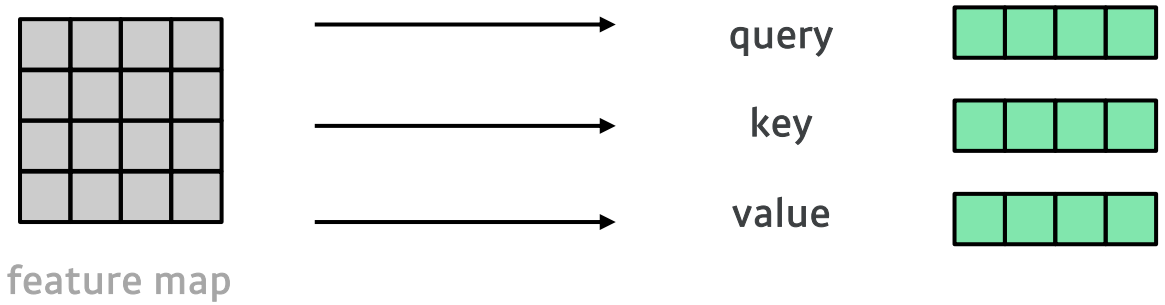
$$Q = (N, d_k), K = (N, d_k), V = (N, d_v)$$

[N, seq_len, dim]



Stable Diffusion | Self-attention vs Cross-attention

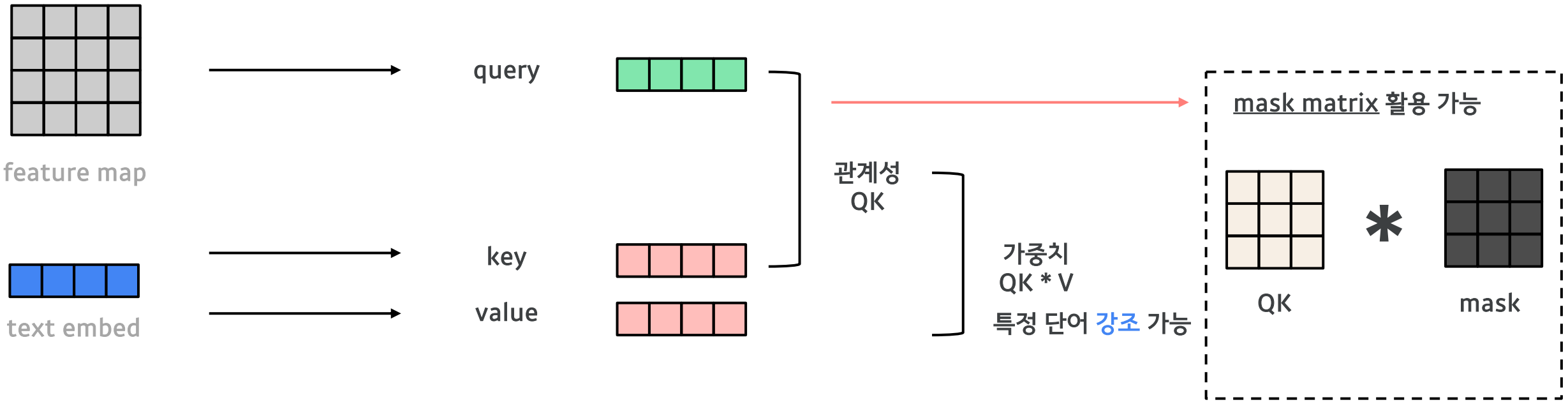
NAVER AI LAB



$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

$$Q = (N, d_k), K = (N, d_k), V = (N, d_v)$$

[N, seq_len, dim]



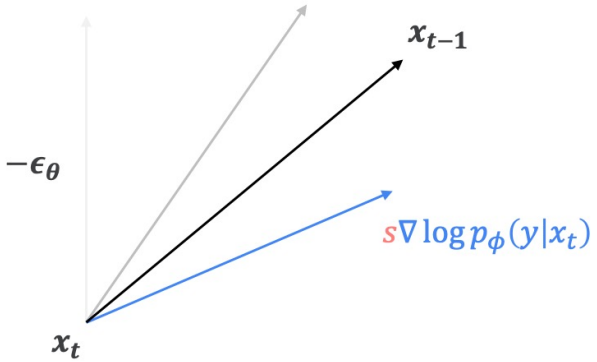
High-Resolution Image Synthesis with Latent Diffusion Models, CVPR 2022

Stable Diffusion | Classifier free guidance

NAVER AI LAB

$$x_{t-1} \leftarrow N\left(\mu + s * \Sigma \nabla \log p_\phi(y|x_t), \Sigma\right)$$

Figure 3: Samples from an unconditional diffusion model with classifier guidance to condition on the class "Pembroke Welsh corgi". Using classifier scale 1.0 (left; FID: 33.0) does not produce convincing samples in this class, whereas classifier scale 10.0 (right; FID: 12.0) produces much more class-consistent images.

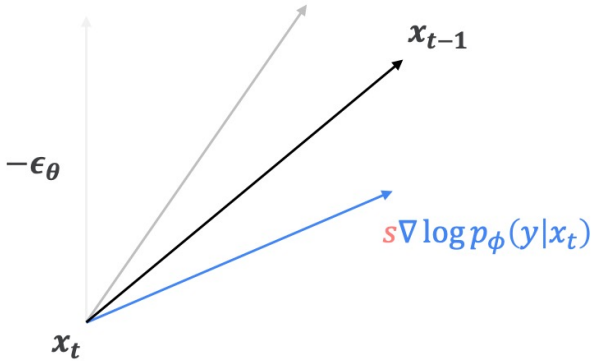


Stable Diffusion | Classifier free guidance

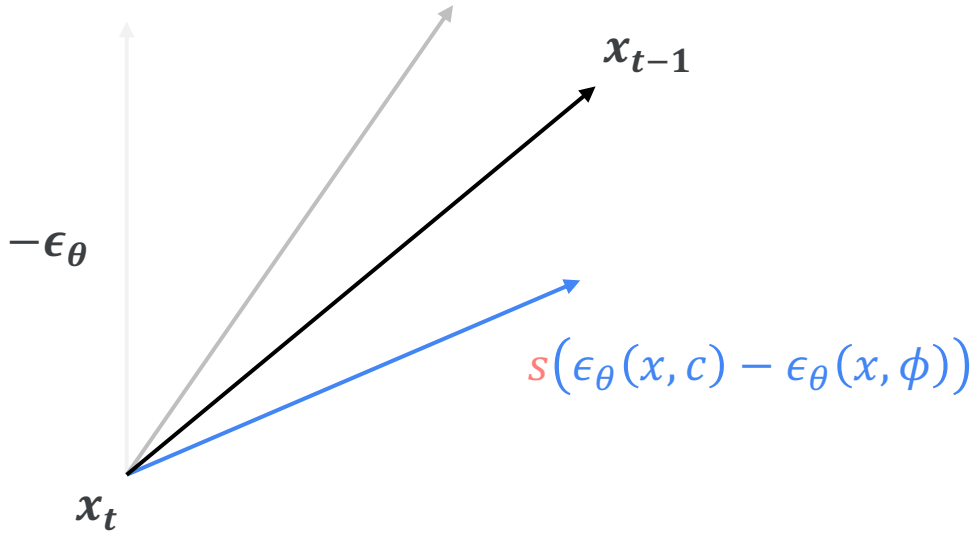
NAVER AI LAB

$$x_{t-1} \leftarrow N\left(\mu + s * \Sigma \nabla \log p_\phi(y|x_t), \Sigma\right)$$

Figure 3: Samples from an unconditional diffusion model with classifier guidance to condition on the class "Pembroke Welsh corgi". Using classifier scale 1.0 (left; FID: 33.0) does not produce convincing samples in this class, whereas classifier scale 10.0 (right; FID: 12.0) produces much more class-consistent images.



$$\tilde{\epsilon}_\theta(x, c) = \epsilon_\theta(x, \phi) + s * \left(\epsilon_\theta(x, c) - \epsilon_\theta(x, \phi)\right)$$



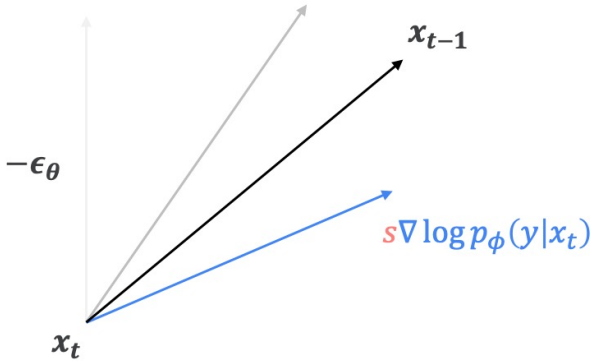
Classifier-Free Diffusion Guidance, NeurIPS 2021

Stable Diffusion | Classifier free guidance

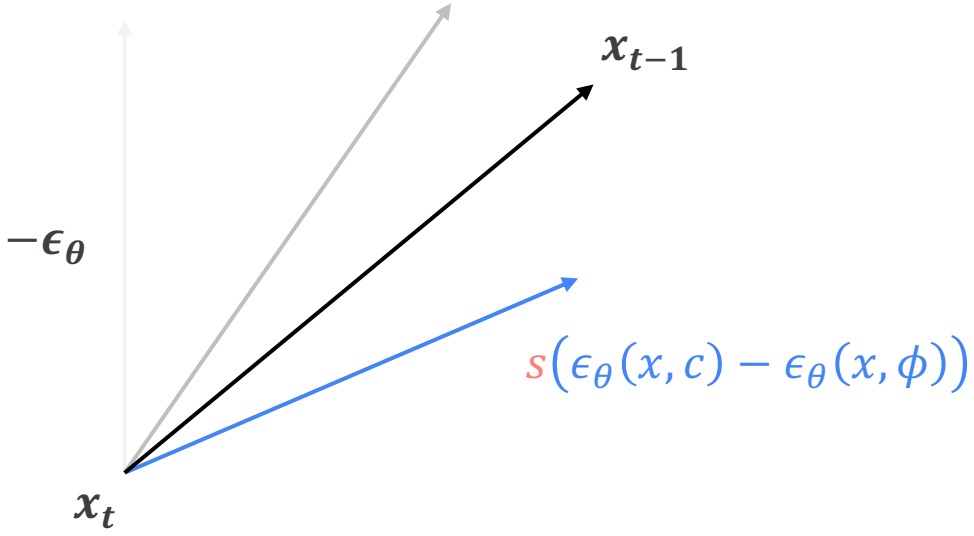
NAVER AI LAB

$$x_{t-1} \leftarrow N\left(\mu + s * \Sigma \nabla \log p_\phi(y|x_t), \Sigma\right)$$

Figure 3: Samples from an unconditional diffusion model with classifier guidance to condition on the class "Pembroke Welsh corgi". Using classifier scale 1.0 (left; FID: 33.0) does not produce convincing samples in this class, whereas classifier scale 10.0 (right; FID: 12.0) produces much more class-consistent images.



$$\tilde{\epsilon}_\theta(x, c) = \epsilon_\theta(x, \phi) + s * \left(\epsilon_\theta(x, c) - \epsilon_\theta(x, \phi)\right)$$



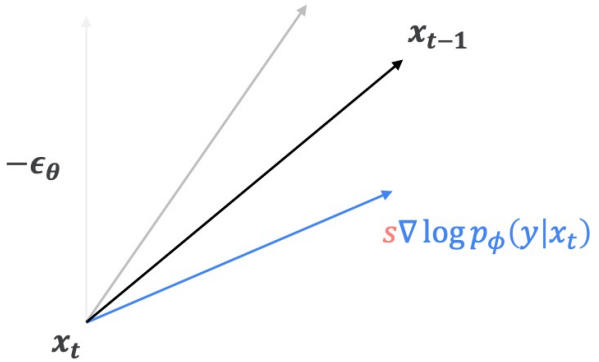
Classifier-Free Diffusion Guidance, NeurIPS 2021

Stable Diffusion | Classifier free guidance

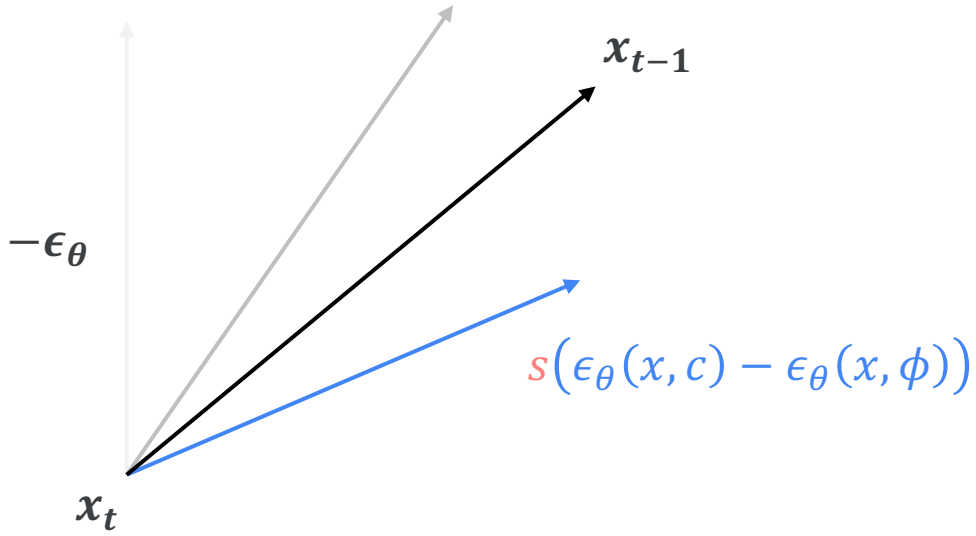
NAVER AI LAB

$$x_{t-1} \leftarrow N\left(\mu + s * \Sigma \nabla \log p_\phi(y|x_t), \Sigma\right)$$

Figure 3: Samples from an unconditional diffusion model with classifier guidance to condition on the class "Pembroke Welsh corgi". Using classifier scale 1.0 (left; FID: 33.0) does not produce convincing samples in this class, whereas classifier scale 10.0 (right; FID: 12.0) produces much more class-consistent images.



$$\tilde{\epsilon}_\theta(x, c) = \epsilon_\theta(x, \phi) + s * \left(\epsilon_\theta(x, c) - \epsilon_\theta(x, \phi)\right)$$




```
# perform guidance
if do_classifier_free_guidance:
    noise_pred_uncond, noise_pred_text = noise_pred.chunk(2)
    noise_pred = noise_pred_uncond + guidance_scale * (noise_pred_text - noise_pred_uncond)
```

Classifier-Free Diffusion Guidance, NeurIPS 2021

Stable Diffusion | Application

NAVER AI LAB

A collage of images created by Stable Diffusion. ***Image Credits:*** *Daniel Jeffries*

스페이스 오로라 극장, 제이스 앨런

Stable Diffusion | Application

NAVER AI LAB

스페이스 오로라 극장, 제이스 앨런

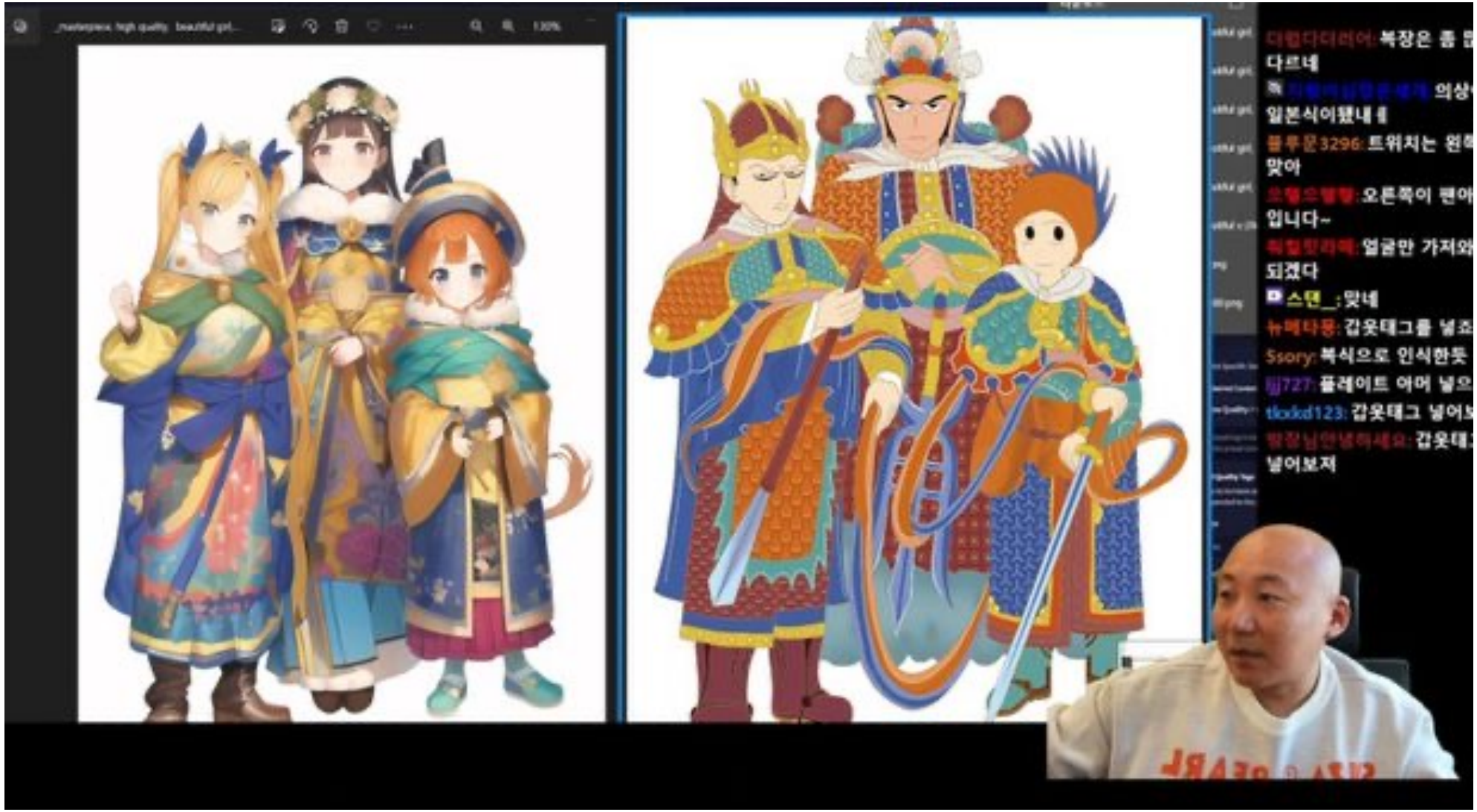
twitter.com/DiffusionPics/status/1588860369367453700?fbclid=IwAR2xJurCMp3rhTu9u39I3bn4osuVSSZs-OpL1ah2WaSC4ZZyPec4ANDjU-A

Stable Diffusion | Image inpainting

NAVER AI LAB

Stable Diffusion | Image inpainting

NAVER AI LAB



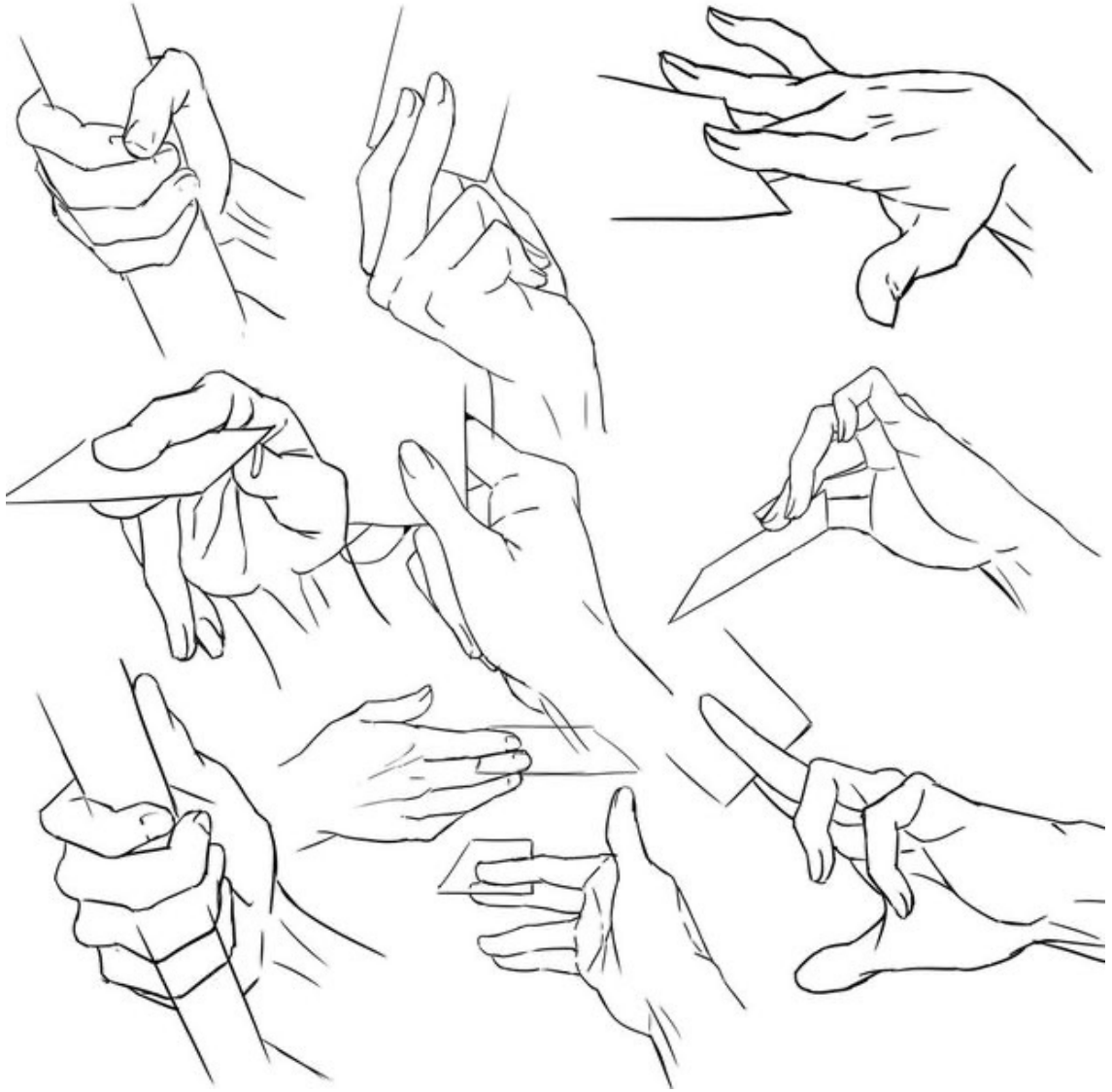
Stable Diffusion | Application

NAVER AI LAB

biz.chosun.com/science-chosun/science/2022/11/06/5362UB6VJZFWLBD25IENHNZKBY

Stable Diffusion | Application

NAVER AI LAB



biz.chosun.com/science-chosun/science/2022/11/06/5362UB6VJZFWLBD25IENHNZKBY

Stable Diffusion | Application

NAVER AI LAB

Stable Diffusion | Application

NAVER AI LAB

diablo3.inven.co.kr

Stable Diffusion | Application

NAVER AI LAB

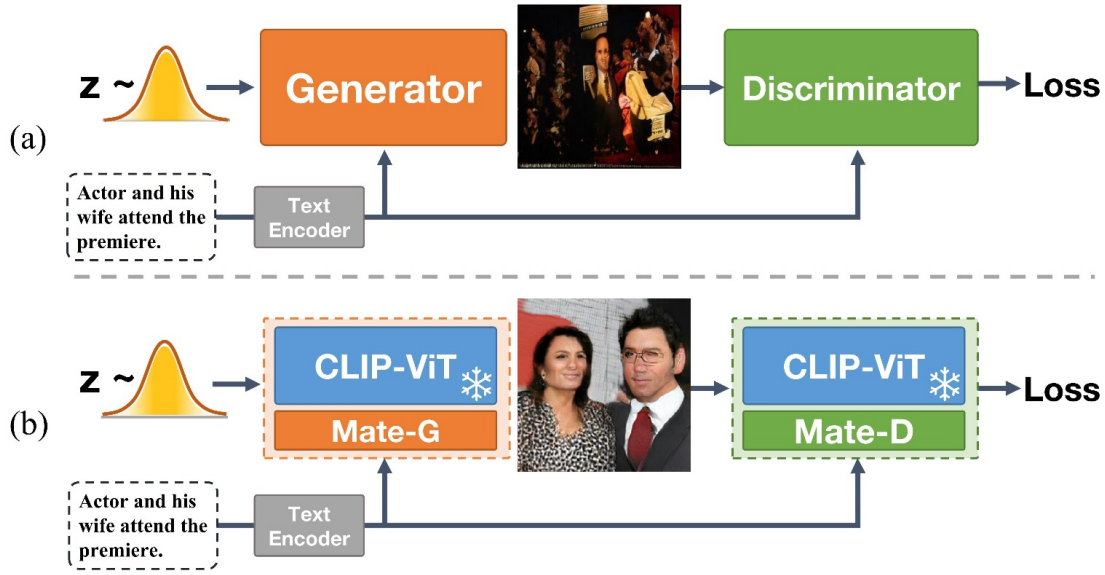
NAVER AI LAB

GALIP

GALIP | Summary

NAVER AI LAB

- Loss
  - Adversarial loss
    - w/ GP
- Image-Text align
  - Affine transformation
    - No cross-attention
- Architecture
  - Traditional GANs
- Contribution
  - 224x224, 8x3090 GPUs, 3days (CC-12M)



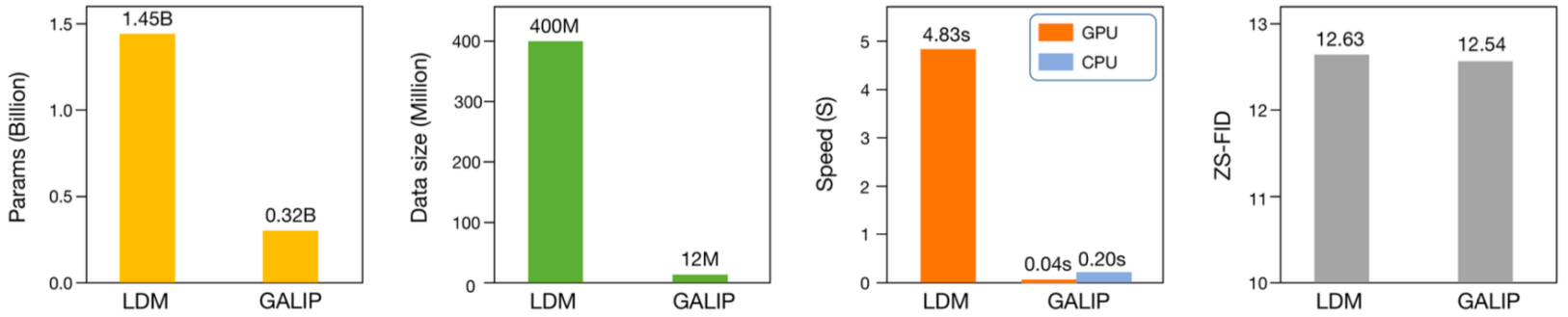


Table 1. The results of FID and CLIPSIM (CS) compared with the state-of-the-art methods on the test set of CUB and COCO.

Model	CUB		COCO	
	FID ↓	CS ↑	FID ↓	CS ↑
DM-GAN [57]	16.09	-	32.64	-
XMC-GAN [53]	-	-	9.30	-
DAE-GAN [37]	15.19	-	28.12	-
DF-GAN [42]	14.81	0.2920	19.32	0.2972
LAFITE [56]	14.58	0.3125	8.21	0.3335
VQ-Diffusion [11]	10.32	-	13.86	-
GALIP (Ours)	**10.08**	**0.3164**	**5.85**	**0.3338**

Table 2. We compare the performance of large pretrained autoregressive models (AR), diffusion models (DF), and GANs under zero-shot setting on the COCO test dataset.

Model	Type	Param [B]	Data size [M]	ZS-FID ↓
DALL-E [33]	AR	12	250	27.5
Cogview [6]	AR	4	30	27.1
Cogview2 [7]	AR	6	30	24.0
Parti-350M [51]	AR	0.35	>800	14.10
Parti-20B [51]	AR	20	>800	7.23
GLIDE [27]	DF	5	250	12.24
LDM [35]	DF	1.45	400	12.63
DALL·E 2 [32]	DF	6.5	250	10.39
Imagen [38]	DF	7.9	860	7.27
eDiff-I [1]	DF	9.1	1000	6.95
LAFITE [56]	GAN	0.15+0.08	3	26.94
GALIP (CC3M)	GAN	0.24+0.08	3	16.12
GALIP (CC12M)	GAN	0.24+0.08	12	12.54

GALIP | Qualitative results

NAVER AI LAB

Figure 7. Text-to-Image samples from GALIP (CC12M) and Latent Diffusion (LAION-400M) [35, 36]. We sample 16 images from each given text description, and randomly select one as the final generation result

Figure 9. Illustration of failure cases. It is still hard for current GALIP to synthesize some imaginary images. Enlarging the model size and training data may improve image quality.

Evaluation

Evaluation

Frechet Inception Distance

NAVER AI LAB

Univariate Normal Fréchet Distance =

$$(\mu_X - \mu_Y)^2 + (\sigma_X - \sigma_Y)^2$$

Multivariate Normal Fréchet Distance =

$$\|\mu_X - \mu_Y\|^2 + \mathrm{Tr}\left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y}\right)$$

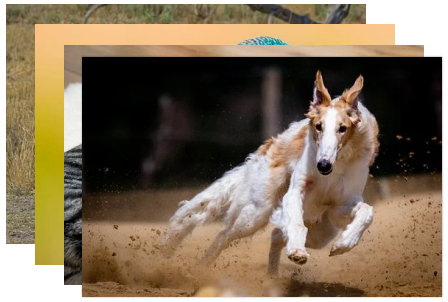
공분산 행렬의 대각성분은 분산 !

Evaluation

Frechet Inception Distance

NAVER AI LAB

Real samples

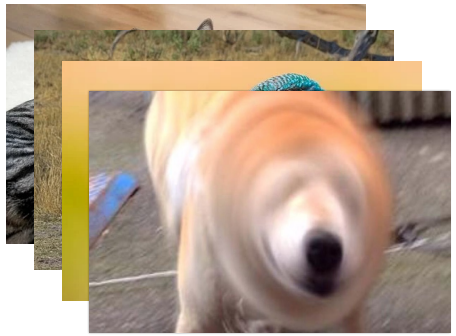


Inception v3

Y

(N x 2048)

Fake samples



Inception v3

X

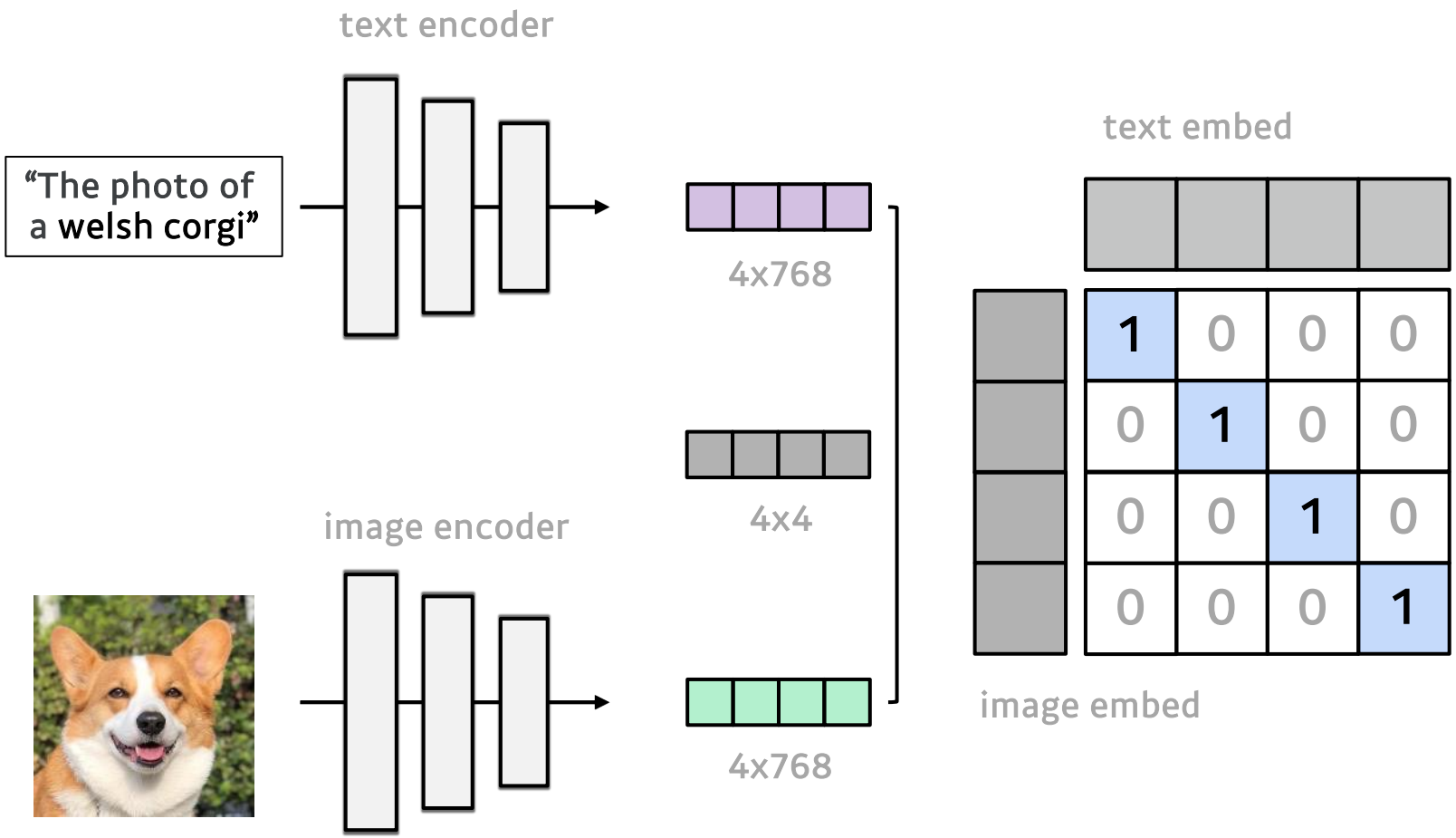
(N x 2048)

$$\text{FID} = \|\mu_X - \mu_Y\|^2 + \text{Tr}\left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y}\right)$$

Real and fake embeddings are two multivariate normal distributions

Evaluation | Clip Score

NAVER AI LAB

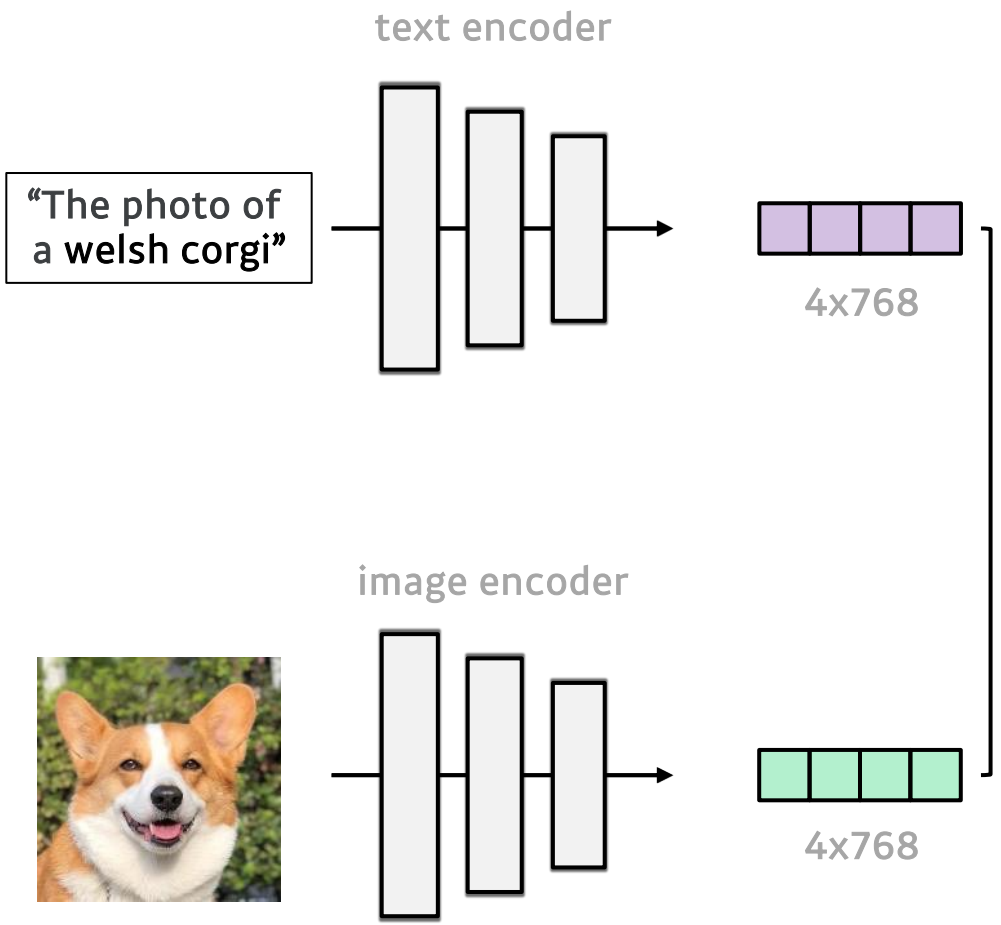


Contrastive Loss

https://huggingface.co/openai

Evaluation | Clip Score

NAVER AI LAB



Cosine Similarity

* 1: 관계가 깊다.

* 0: 관계가 없다.

* -1: 관계가 반대다.

openai/clip-vit-base-patch16
Zero-Shot Image Classification • Updated Oct 4, 2022 • 809k • ♡ 31

openai/clip-vit-base-patch32
Zero-Shot Image Classification • Updated Oct 4, 2022 • 4.18M • ♡ 204

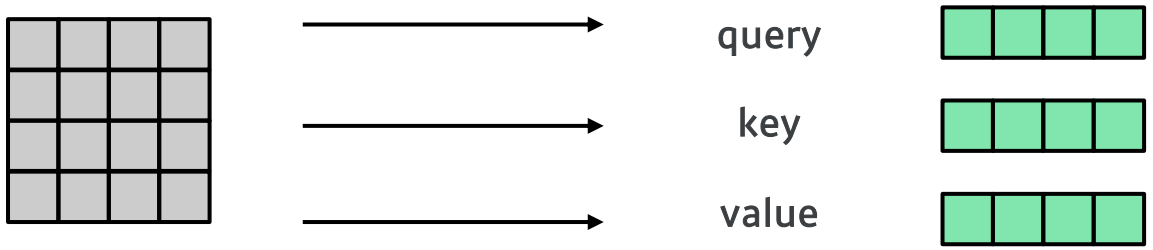
openai/clip-vit-large-patch14-336
Zero-Shot Image Classification • Updated Oct 4, 2022 • 381k • ♡ 39

openai/clip-vit-large-patch14
Zero-Shot Image Classification • Updated Oct 4, 2022 • 15.7M • ♡ 490

https://huggingface.co/openai

Editing with SD

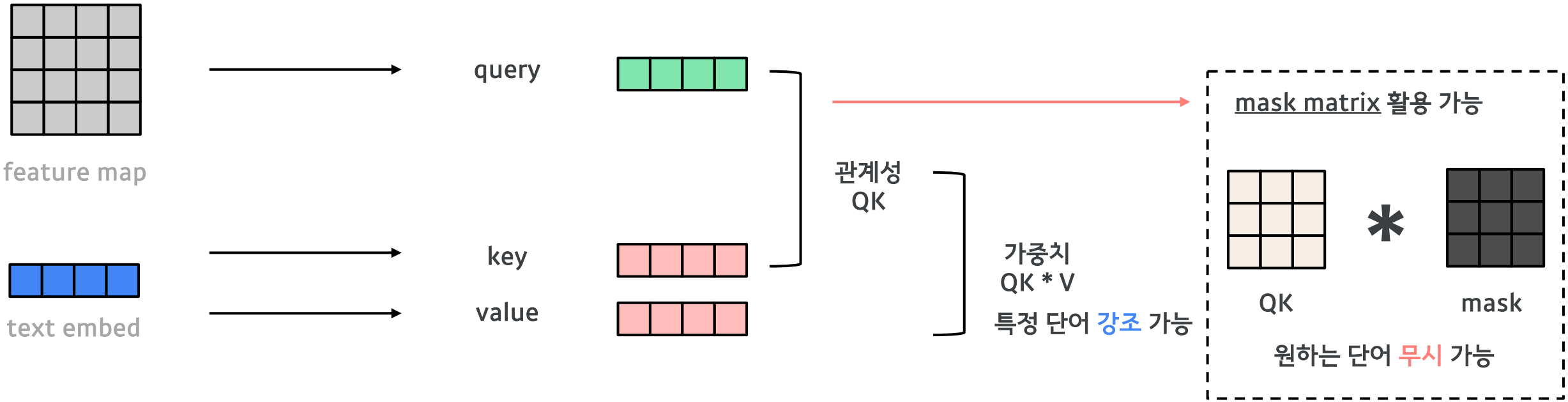
Editing method | Attention control



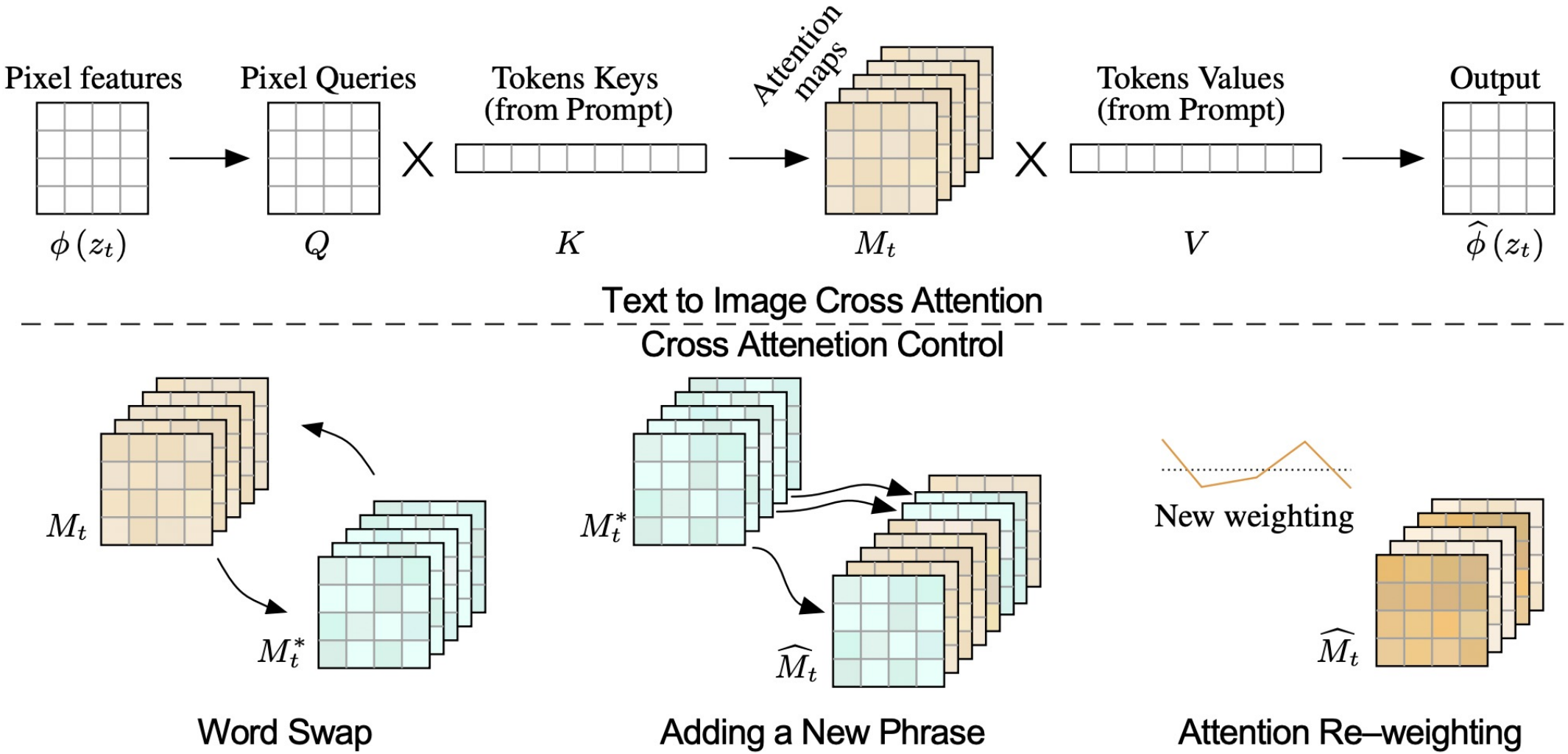
$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

$$Q = (N, d_k), K = (N, d_k), V = (N, d_v)$$

[N, seq_len, dim]

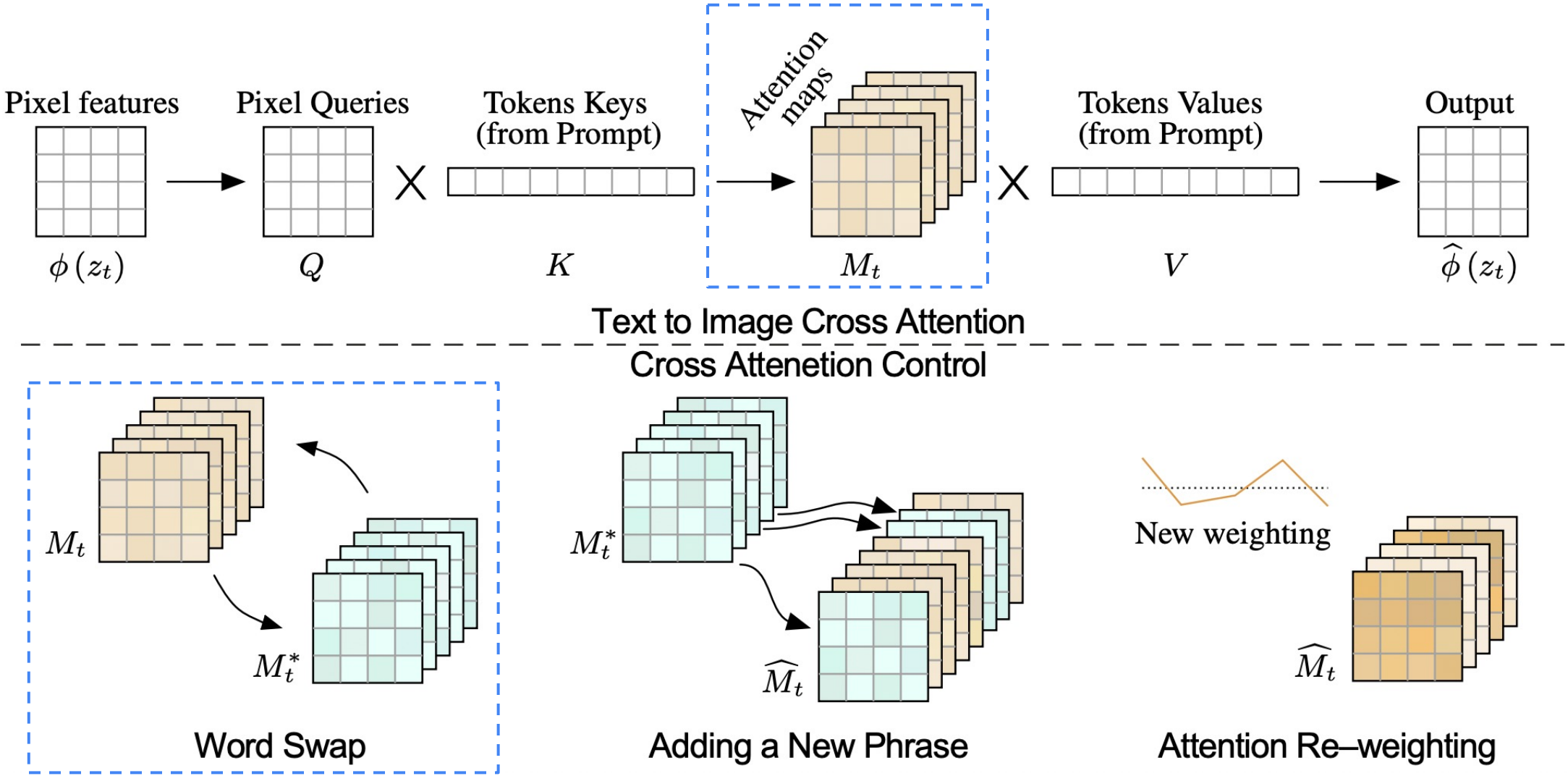


Editing method | Attention control



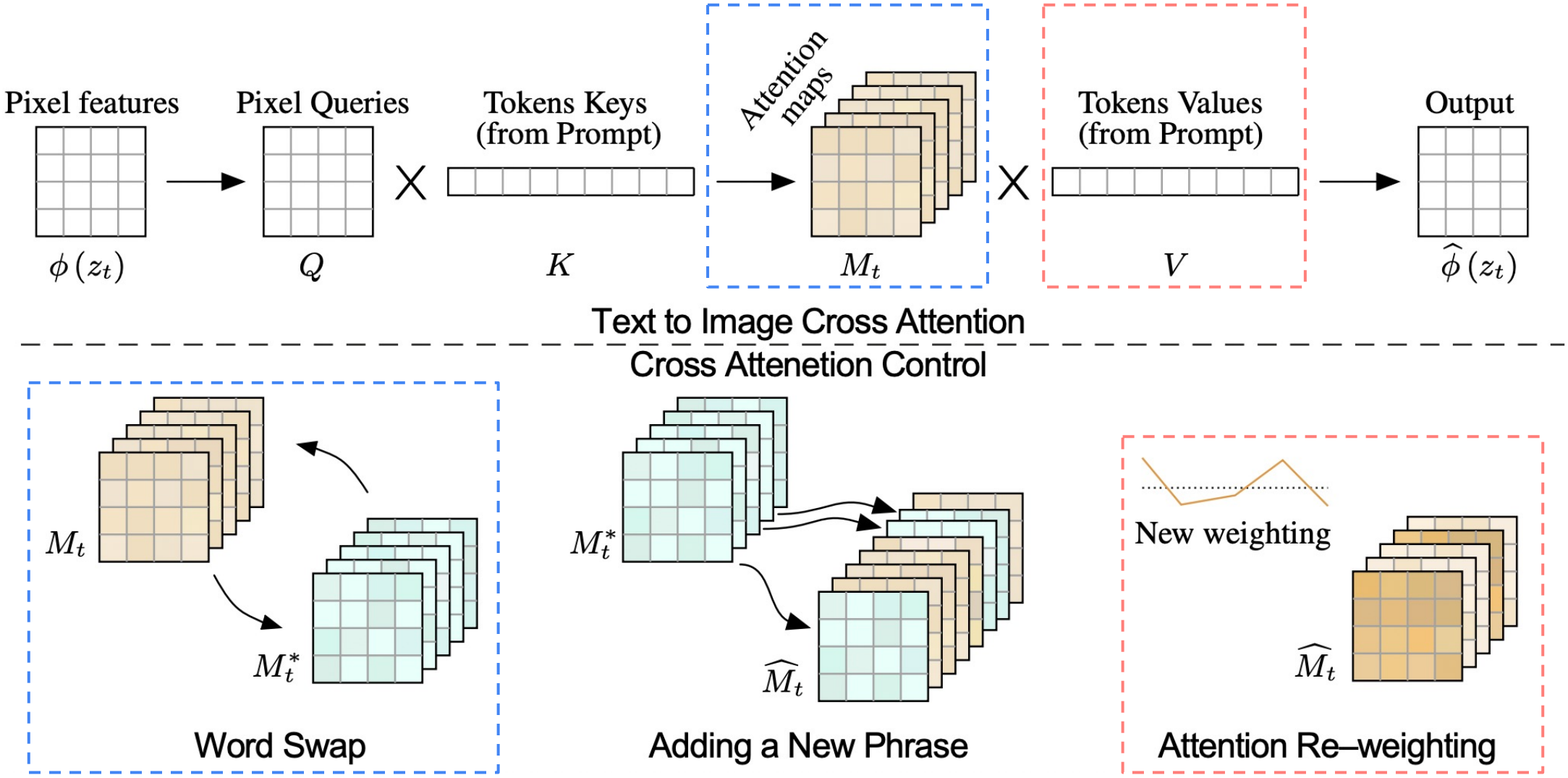
Prompt-to-Prompt Image Editing with Cross Attention Control, Arxiv

Editing method | Attention control



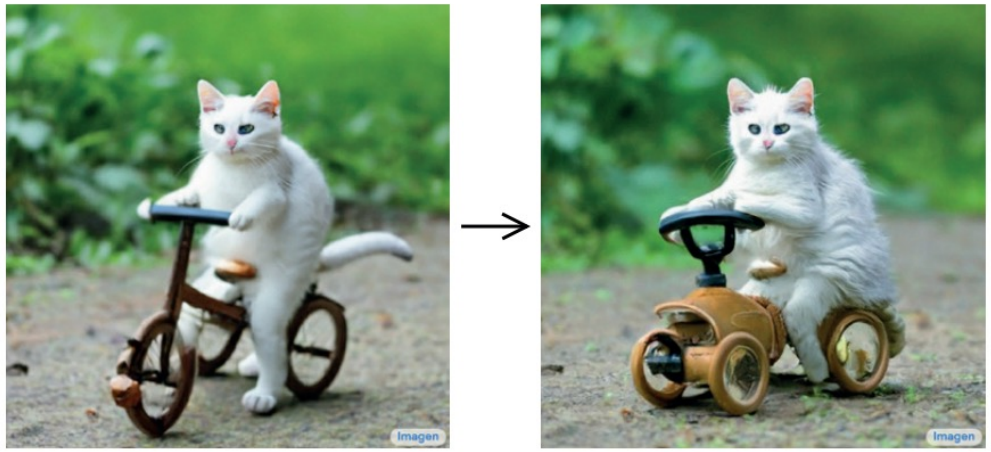
Editing method | Attention control

NAVER AI LAB

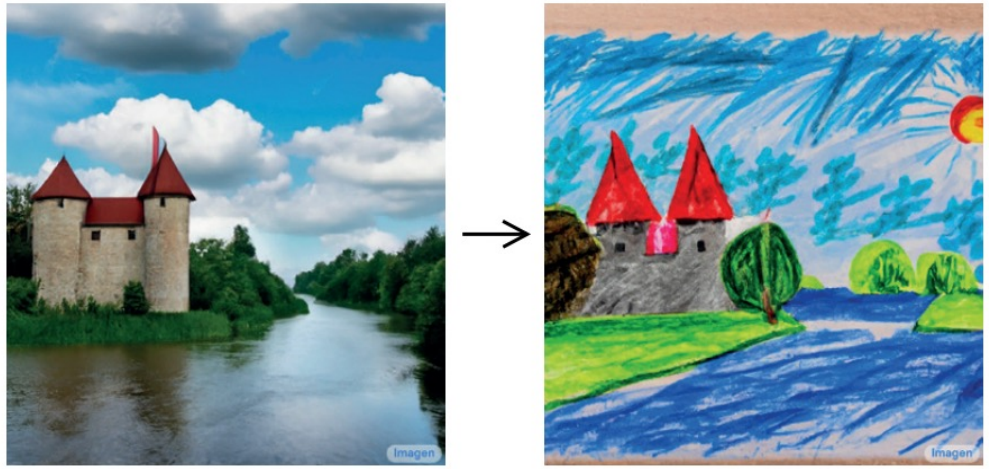


Prompt-to-Prompt Image Editing with Cross Attention Control, Arxiv

Prompt-to-Prompt | Results



"Photo of a cat riding on a ~~bicycle~~ car."

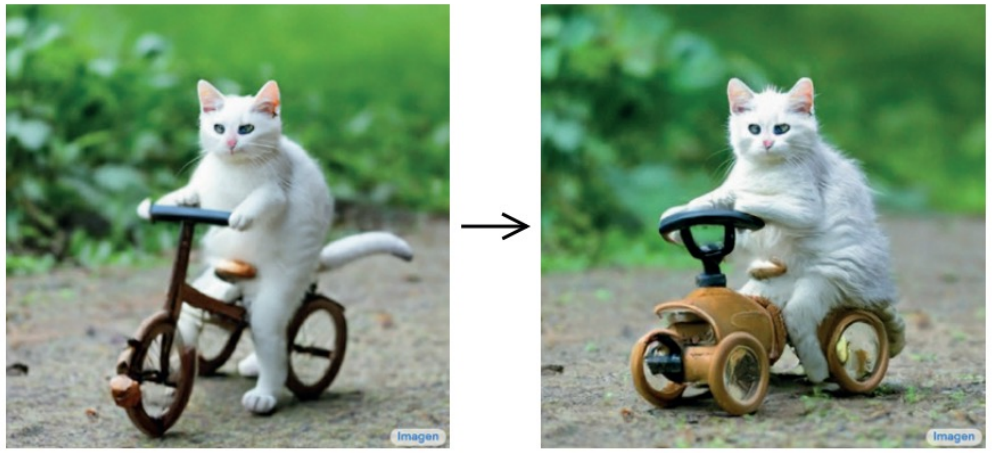


"Children drawing of a castle next to a river."

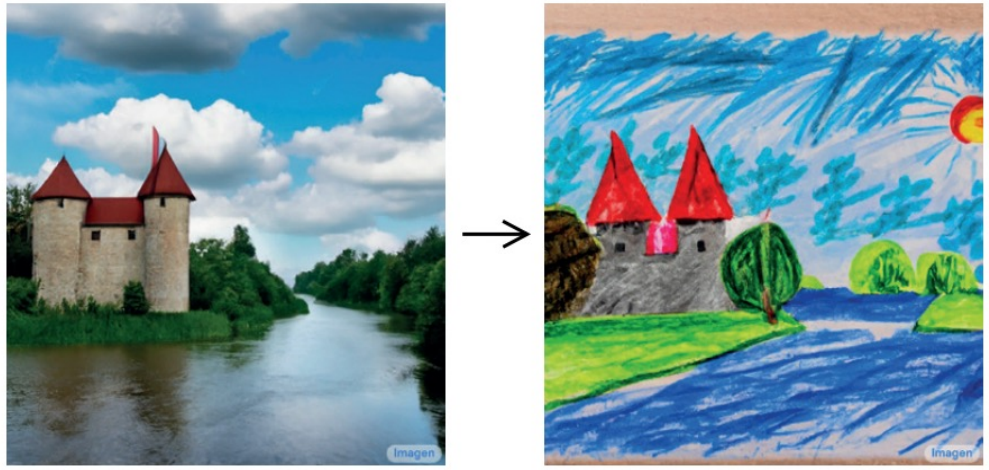
Prompt-to-Prompt Image Editing with Cross Attention Control, Arxiv

Prompt-to-Prompt | Results

NAVER AI LAB



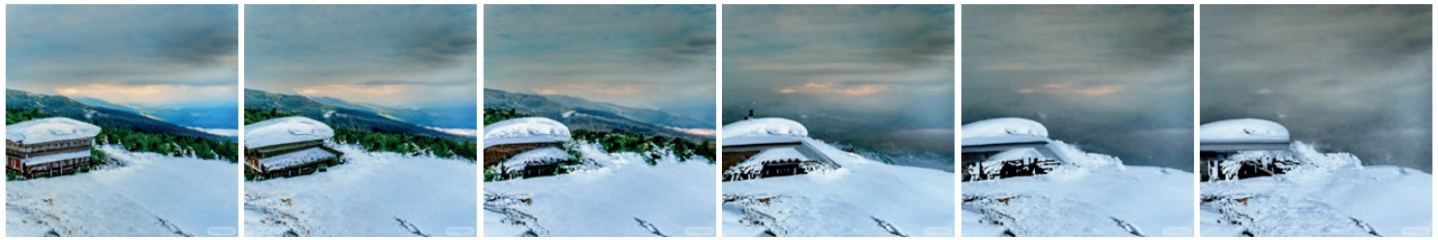
"Photo of a cat riding on a ~~bicycle~~ car."



"Children drawing of a castle next to a river."

"A photo of a birthday(↓) cake next to an apple."

"The picnic is ready under a blossom(↓) tree."



"A photo of a house on a snowy(↑) mountain."

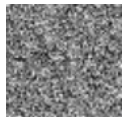
"My fluffy(↑) bunny doll.

Editing method | Dreambooth

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Arxiv

Editing method | Dreambooth

NAVER AI LAB

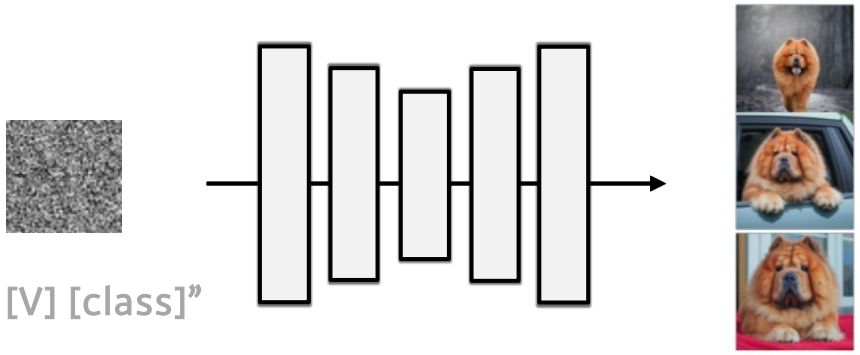


“A [V] [class]”

Input image

Editing method | Dreambooth

NAVER AI LAB



Input image

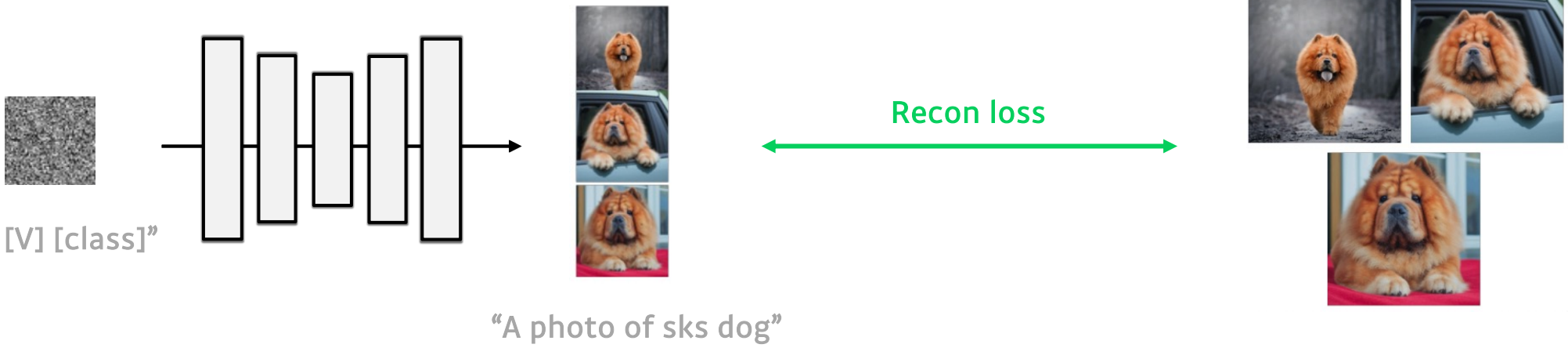
Editing method | Dreambooth

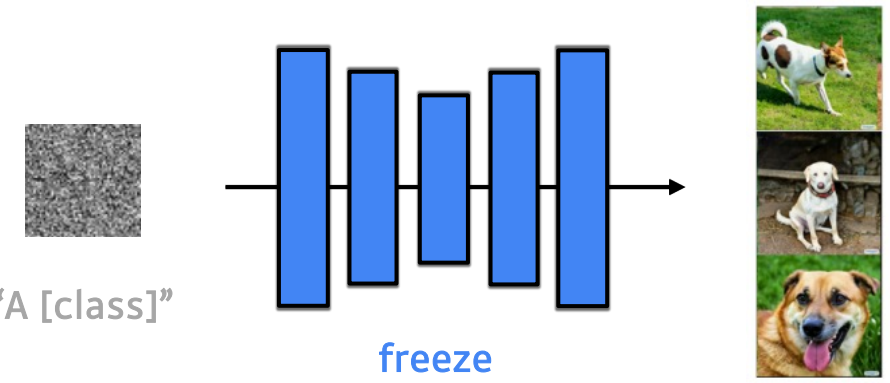
NAVER AI LAB

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Arxiv

Editing method | Dreambooth

NAVER AI LAB

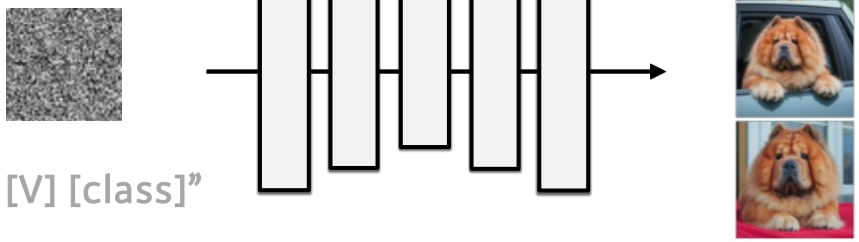




DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Arxiv

Editing method | Dreambooth

NAVER AI LAB

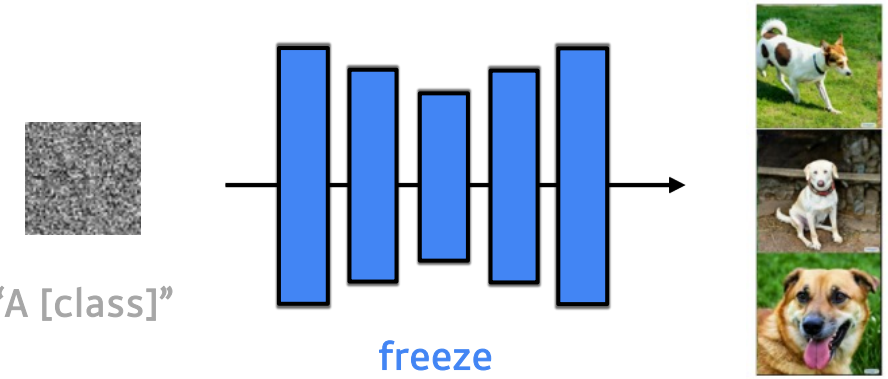


"A [V] [class]"

"A photo of sks dog"

Recon loss

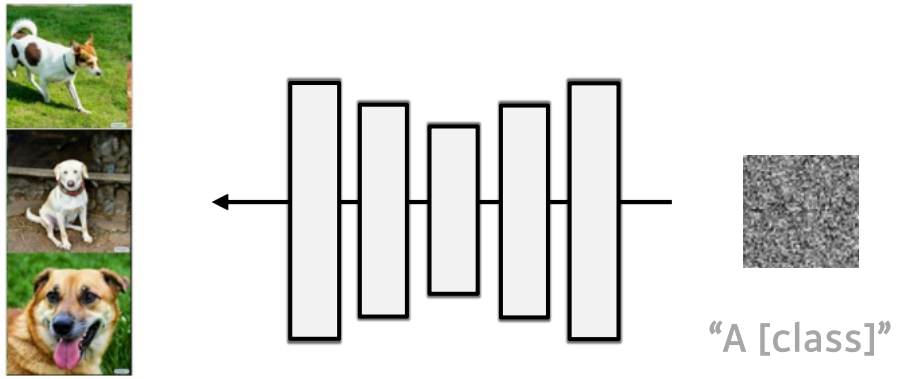
Input image



"A [class]"

freeze

"A photo of dog"

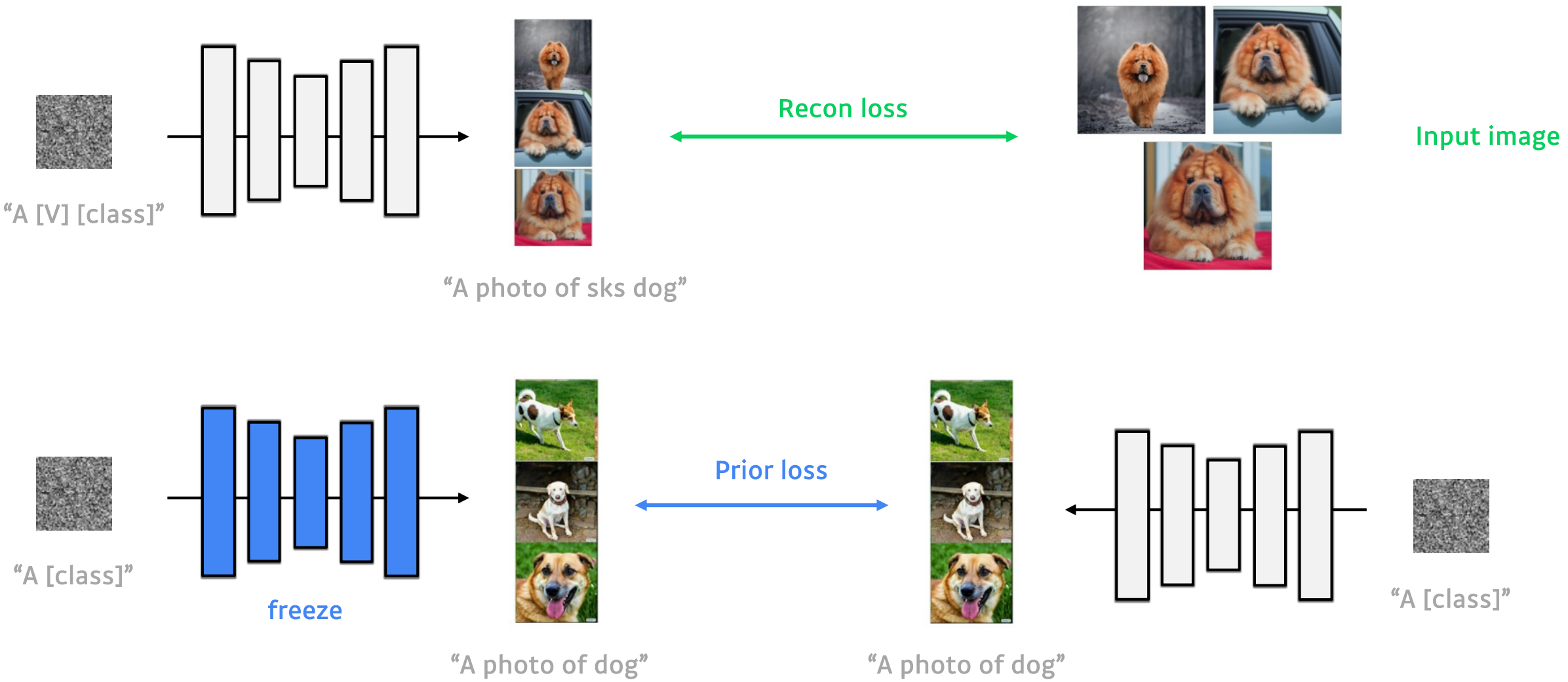


"A photo of dog"

"A [class]"

Editing method | Dreambooth

NAVER AI LAB



DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Arxiv

DreamBooth | Training

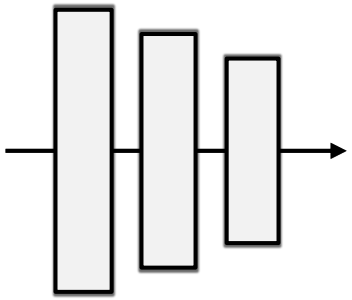
NAVER AI LAB

tuning image

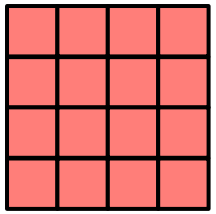
DreamBooth | Training

NAVER AI LAB

tuning image



encoder



64x64x4

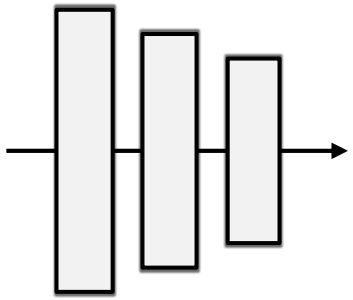
forward process

step T

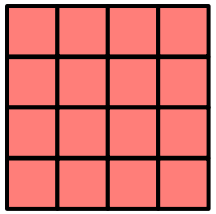
DreamBooth | Training

NAVER AI LAB

tuning image



encoder

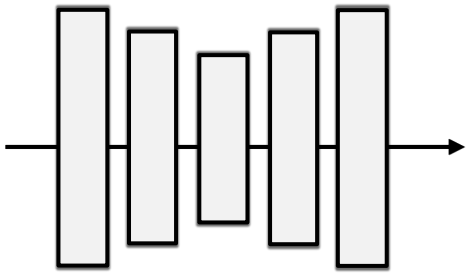


64x64x4

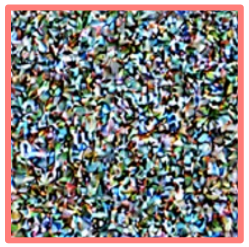
forward process

step T

time embed (T)



Unet



pred noise

“A photo of sks dog”

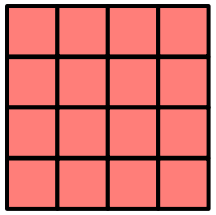
DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Arxiv

DreamBooth | Training

NAVER AI LAB

tuning image

encoder

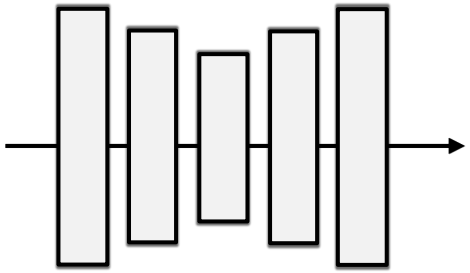


64x64x4

forward process

step T

time embed (T)



Unet

L2 loss

pred noise

-

GT noise

“A photo of sks dog”

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Arxiv

DreamBooth | Training

NAVER AI LAB

generated image

encoder

64x64x4

forward process

step T

time embed (T)

L2 loss

Unet

pred noise

GT noise

“A photo of dog”

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Arxiv

DreamBooth | Inference

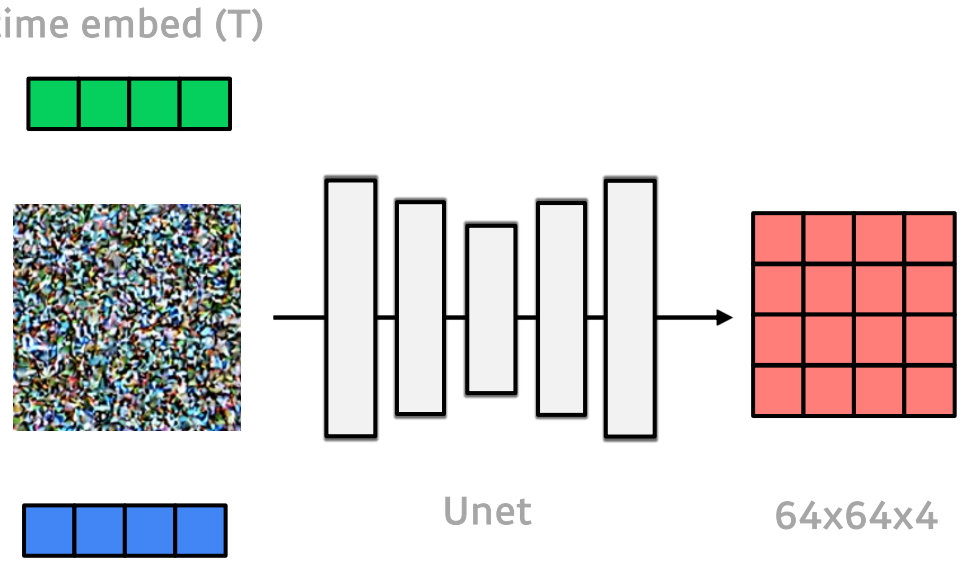
NAVER AI LAB

time embed (T)

“A sks dog wearing a [?]”

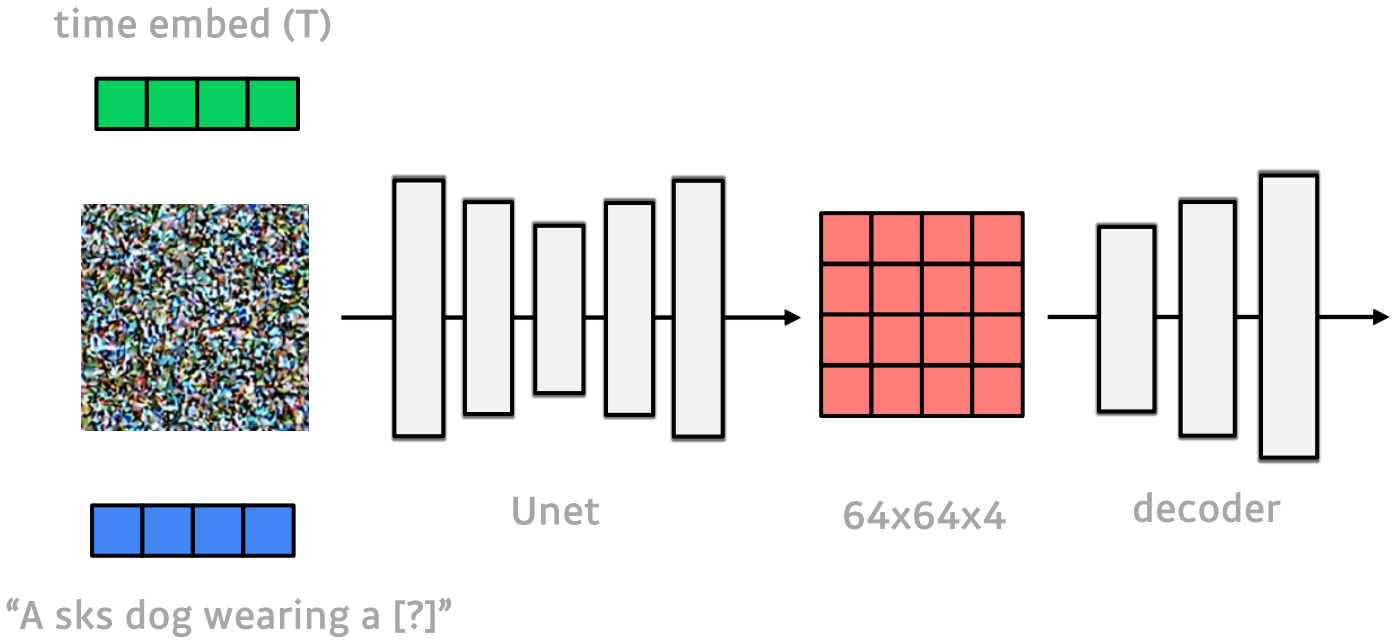
DreamBooth | Inference

NAVER AI LAB



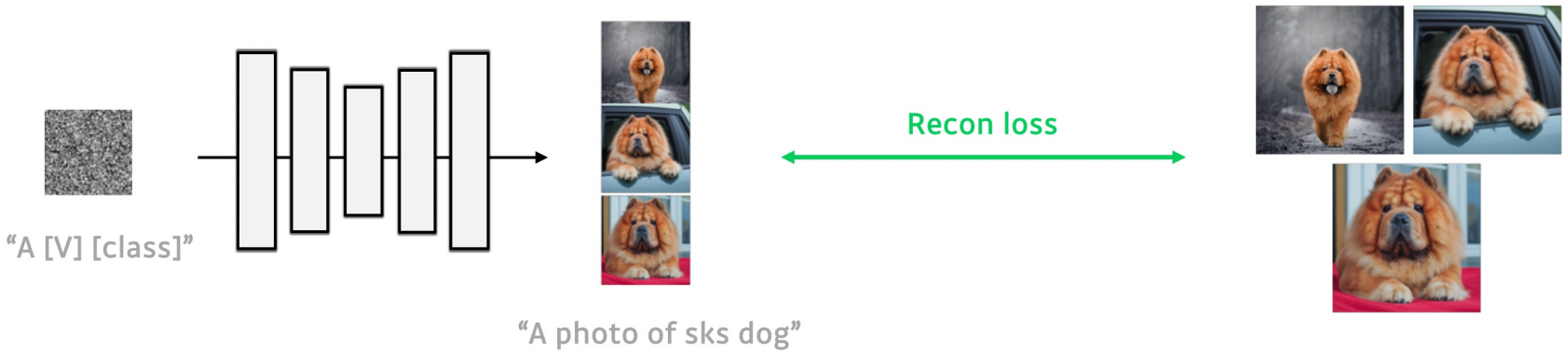
DreamBooth | Inference

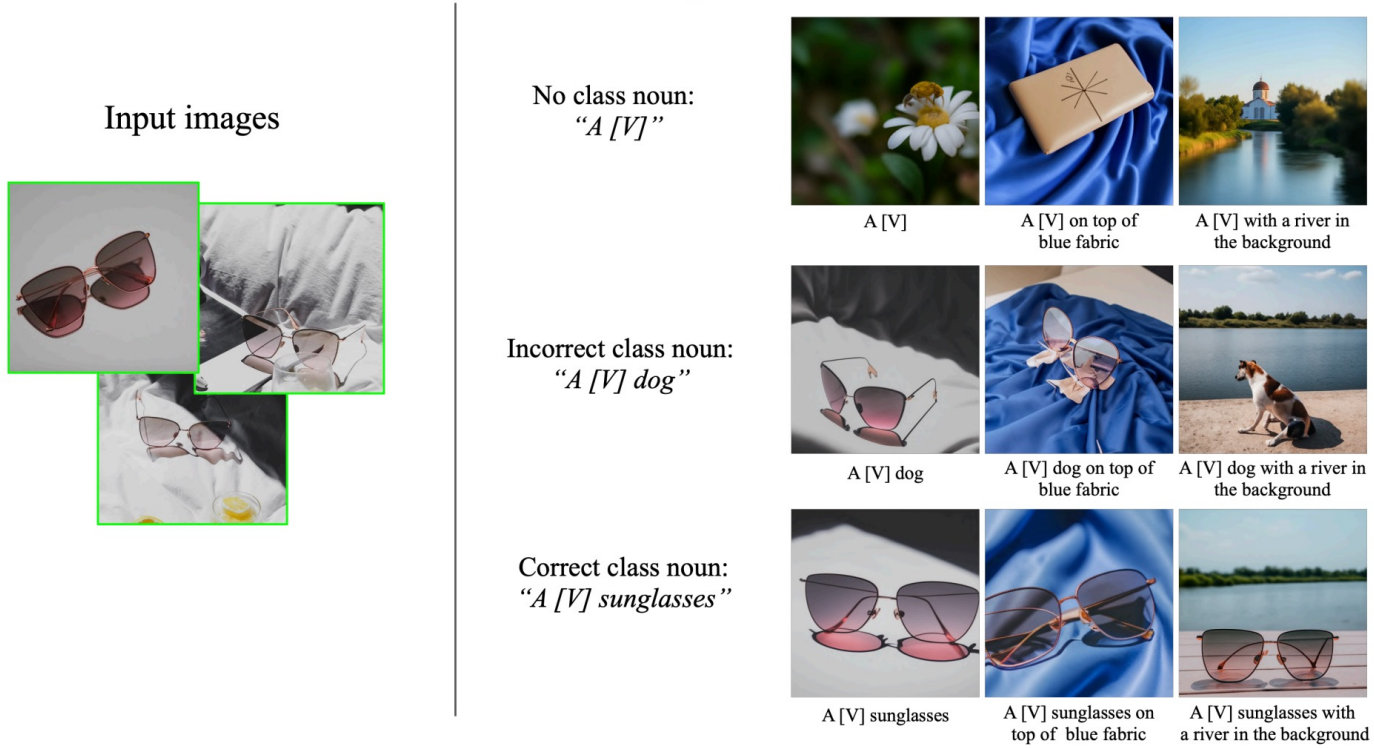
NAVER AI LAB



DreamBooth | Ablation study

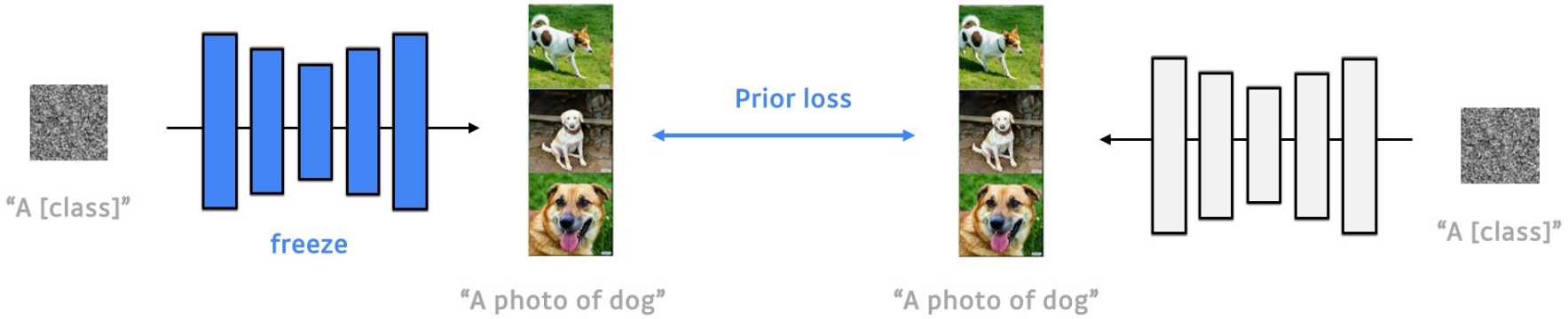
NAVER AI LAB





DreamBooth | Ablation study

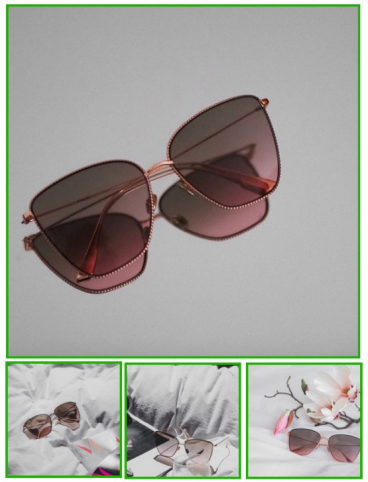
NAVER AI LAB



DreamBooth | Results

NAVER AI LAB

Input images



A [V] sunglasses in the jungle

A [V] sunglasses worn by a bear

A [V] sunglasses at Mt. Fuji

A [V] sunglasses on top of snow

A [V] sunglasses with Eiffel Tower in the background

Input images

Vincent Van Gogh

Michelangelo

Rembrandt

Johannes Vermeer

Pierre-Auguste Renoir

Leonardo da Vinci

Diffusion | Summary

NAVER AI LAB

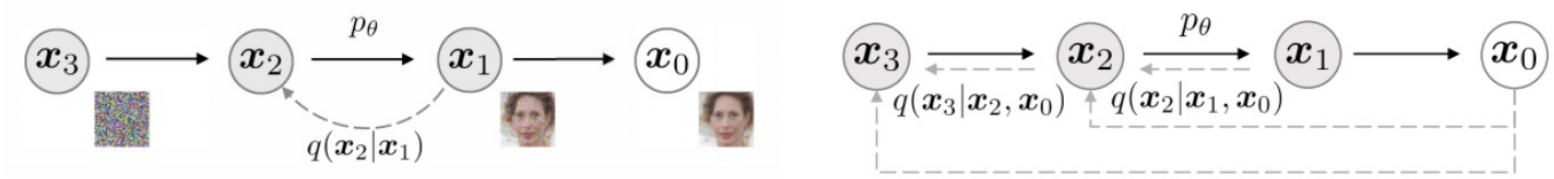


Figure 1: Graphical models for diffusion (left) and non-Markovian (right) inference models.

Denoising Diffusion Implicit Models

최근 Trend

$$x_{t-1} = \hat{\mu}_t(x_t) + \sigma_t \epsilon_t$$

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} P(f_t(x_t)) + D(f_t(x_t)) + \sigma_t \epsilon_t$$

$$* \; P(f_t(x_t)) = \frac{x_t - \sqrt{1-\bar{\alpha}_t} f_t(x_t)}{\sqrt{\bar{\alpha}_t}}$$

$$* \; D(f_t(x_t)) = \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} f_t(x_t)$$

DDPM

- $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon,\ \epsilon \sim \mathcal{N}(0, \mathbf{I})$ `(Forward)`
  - $\beta_1 = 10^{-4}, \beta_T = 0.02$
  - $\alpha_t := 1 - \beta_t,\ \bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$
- $\epsilon - \epsilon_\theta(\mathbf{x}_t) = \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon)$ `(Loss)`
  - ϵ_θ = prediction network
- $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \tilde{\beta}_t\epsilon,\ \epsilon \sim \mathcal{N}(0, \mathbf{I})$ `(Reverse)`
  - $\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$
    - $\tilde{\beta}_t = \beta_t$로 해도 성능차이 없음

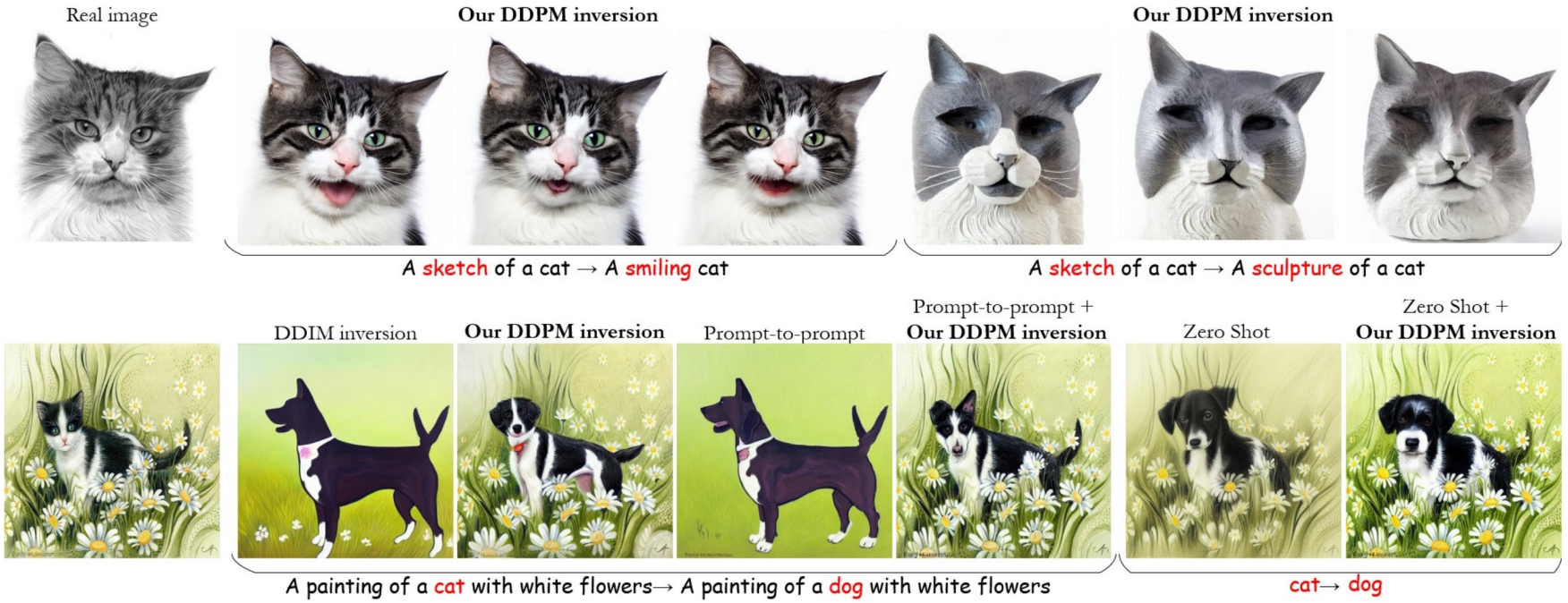
DDIM

- `In paper,` DDIM α = DDPM $\bar{\alpha}$
- $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ `(Forward)`
- $\epsilon - \epsilon_\theta(\mathbf{x}_t) = \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon)$ `(Loss)`
  - ϵ_θ = prediction network
- $\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\underbrace{\left(\frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(\mathbf{x}_t)}{\sqrt{\bar{\alpha}_t}}\right)}_{\text{predicted } \mathbf{x}_0 = f_\theta(\mathbf{x}_t)} + \underbrace{\sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\epsilon_\theta(\mathbf{x}_t)}_{\text{direction pointing to } \mathbf{x}_t} + \underbrace{\sigma_t\epsilon}_{\text{noise}}$

  `(Reverse)`
  - deterministic when $\sigma_t = 0$ → consistency (DDIM)
  - stochastic when $\sigma_t = 1$ → inconsistency (DDPM)

Editing method | DDPM Inversion

NAVER AI LAB



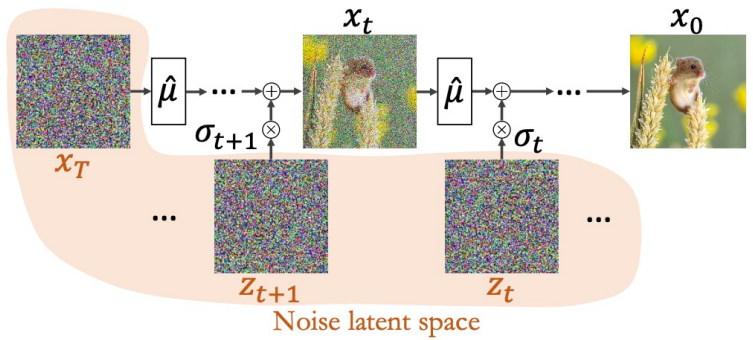
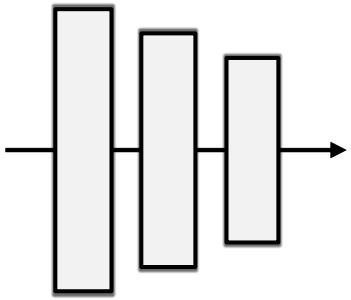


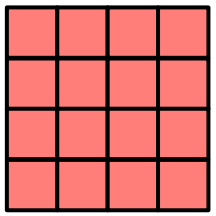
Figure 3: **The DDPM latent noise space.** In DDPM, the generative (reverse) diffusion process synthesizes an image x_0 in T steps, by utilizing $T + 1$ noise maps, $\{x_T, z_T, \ldots, z_1\}$. We regard those noise maps as the latent code associated with the generated image.

Editing method | DDPM Inversion

NAVER AI LAB



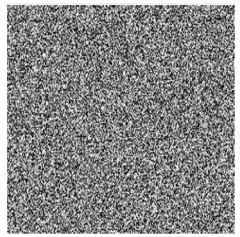
encoder



64x64x4

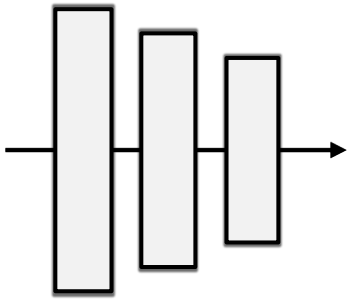
$$x_t = \bar{\alpha}_t x_0 + (1 - \bar{\alpha}_t)\epsilon_t$$

forward process

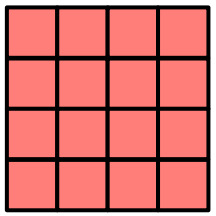


Editing method | DDPM Inversion

NAVER AI LAB



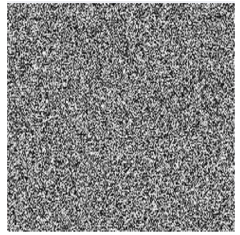
encoder

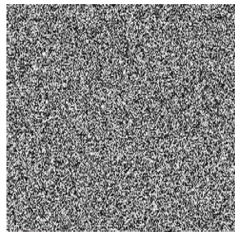


64x64x4

$$x_t = \bar{\alpha}_t x_0 + (1 - \bar{\alpha}_t)\varepsilon_t$$

forward process





x_T x_t x_0

Editing method | DDPM Inversion

NAVER AI LAB

x_T x_t x_0

Reverse process

$$x_{t-1} = \hat{\mu}_t(x_t) + \sigma_t z_t$$

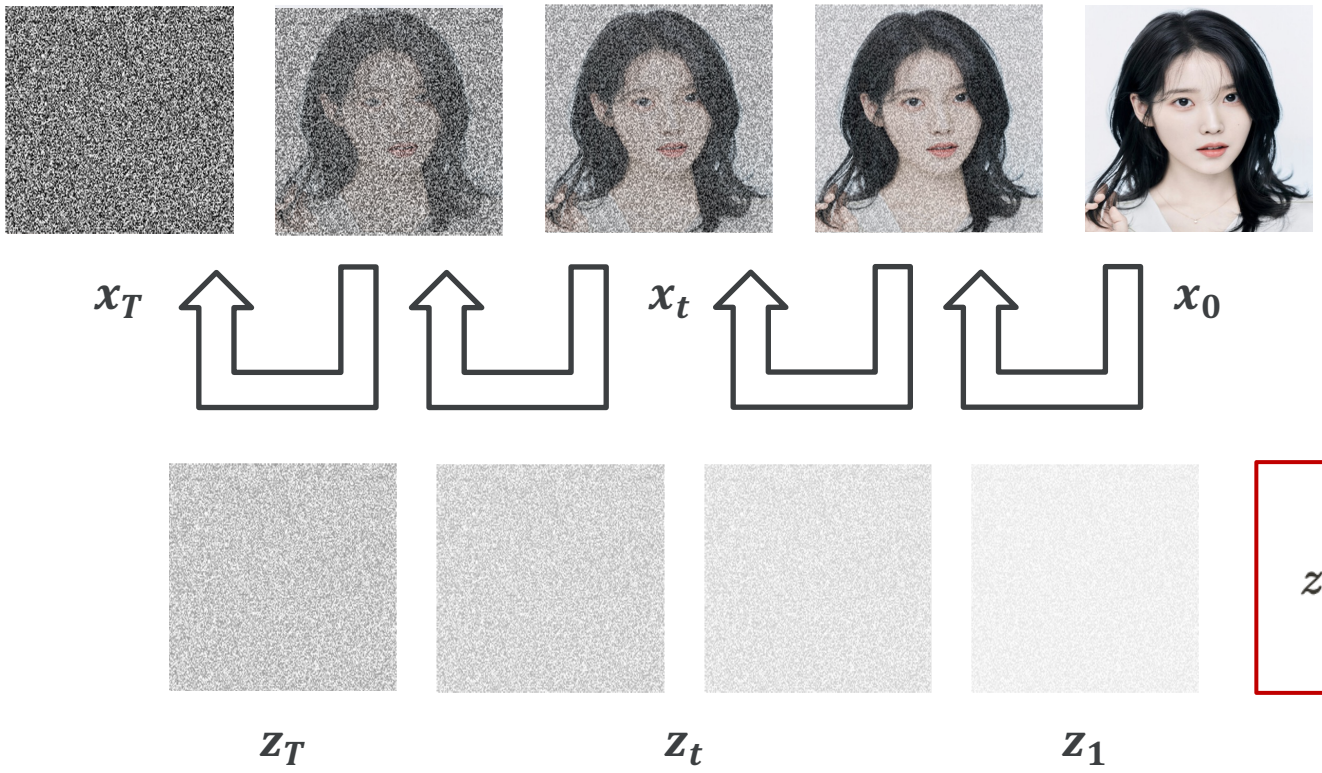
$$\hat{\mu}_t(x_t) = \sqrt{\bar{\alpha}_{t-1}} P(f_t(x_t)) + D(f_t(x_t))$$

$$P(f_t(x_t)) = \frac{x_t - \sqrt{1-\bar{\alpha}_t} f_t(x_t)}{\sqrt{\bar{\alpha}_t}}$$

$$D(f_t(x_t)) = \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} f_t(x_t)$$

Editing method | DDPM Inversion

NAVER AI LAB



$$z_t \leftarrow \frac{x_{t-1} - \hat{\mu}_t(x_t)}{\sigma_t}$$

Reverse process

$$x_{t-1} = \hat{\mu}_t(x_t) + \sigma_t z_t$$

$$\hat{\mu}_t(x_t) = \sqrt{\bar{\alpha}_{t-1}} P(f_t(x_t)) + D(f_t(x_t))$$

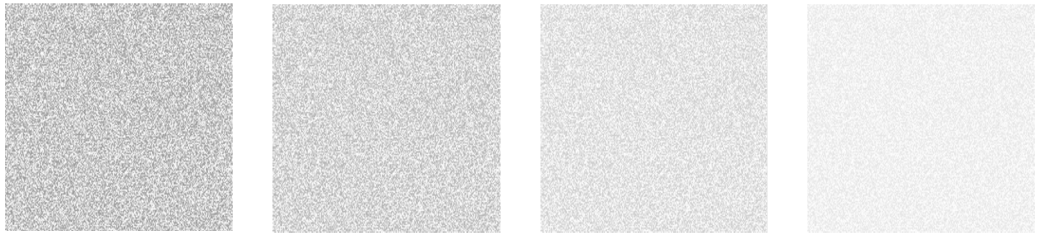
$$P(f_t(x_t)) = \frac{x_t - \sqrt{1-\bar{\alpha}_t} f_t(x_t)}{\sqrt{\bar{\alpha}_t}}$$

$$D(f_t(x_t)) = \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} f_t(x_t)$$

Editing method | DDPM Inversion

NAVER AI LAB

x_T x_t x_0



z_T z_t z_1

$$z_t \leftarrow \frac{x_{t-1} - \hat{\mu}_t(x_t)}{\sigma_t}$$

Reverse process

$$x_{t-1} = \hat{\mu}_t(x_t) + \sigma_t z_t$$

$$\hat{\mu}_t(x_t) = \sqrt{\bar{\alpha}_{t-1}} P(f_t(x_t)) + D(f_t(x_t))$$

$$P(f_t(x_t)) = \frac{x_t - \sqrt{1-\bar{\alpha}_t} f_t(x_t)}{\sqrt{\bar{\alpha}_t}}$$

$$D(f_t(x_t)) = \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} f_t(x_t)$$

x_T x_t x_0

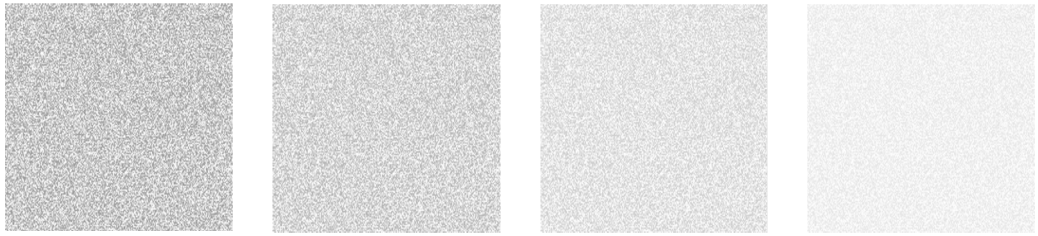
$$x_{t-1} \leftarrow \hat{\mu}_t(x_t) + \sigma_t z_t$$

x_t 는 x_0의 semantic, structure를 잘 담아내게되게 된다.

Editing method | DDPM Inversion

NAVER AI LAB

x_T x_t x_0



z_T z_t z_1

$$z_t \leftarrow \frac{x_{t-1} - \hat{\mu}_t(x_t)}{\sigma_t}$$

Reverse process

$$x_{t-1} = \hat{\mu}_t(x_t) + \sigma_t z_t$$

$$\hat{\mu}_t(x_t) = \sqrt{\bar{\alpha}_{t-1}} P(f_t(x_t)) + D(f_t(x_t))$$

$$P(f_t(x_t)) = \frac{x_t - \sqrt{1-\bar{\alpha}_t} f_t(x_t)}{\sqrt{\bar{\alpha}_t}}$$

$$D(f_t(x_t)) = \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} f_t(x_t)$$

x_T x_t x_0

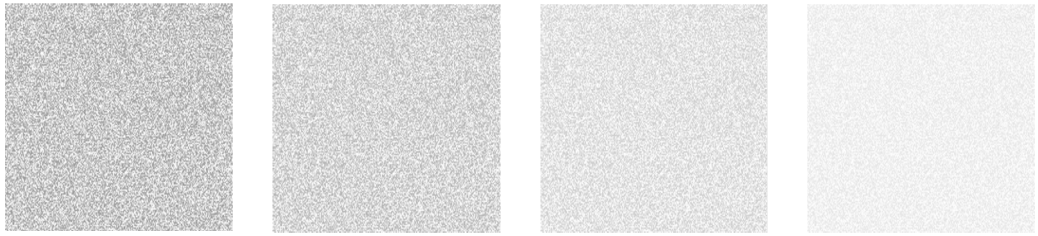
$$x_{t-1} \leftarrow \hat{\mu}_t(x_t) + \sigma_t z_t$$

x_t 는 x_0의 semantic, structure를 잘 담아내게되게 된다.

Editing method | DDPM Inversion

NAVER AI LAB

x_T x_t x_0



z_T z_t z_1

Reverse process

$$x_{t-1} = \hat{\mu}_t(x_t) + \sigma_t z_t$$

$$\hat{\mu}_t(x_t) = \sqrt{\bar{\alpha}_{t-1}} P(f_t(x_t)) + D(f_t(x_t))$$

$$P(f_t(x_t)) = \frac{x_t - \sqrt{1-\bar{\alpha}_t} f_t(x_t)}{\sqrt{\bar{\alpha}_t}}$$

$$D(f_t(x_t)) = \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} f_t(x_t)$$

$$z_t \leftarrow \frac{x_{t-1} - \hat{\mu}_t(x_t)}{\sigma_t}$$

$$x_{t+1} = \sqrt{\bar{\alpha}_{t-1}} \left(\frac{x_t - \sqrt{1-\bar{\alpha}_{t-1}} * f(x_t)}{\sqrt{\bar{\alpha}_{t-1}}} \right) + \sqrt{1-\bar{\alpha}_t} * f(x_t)$$

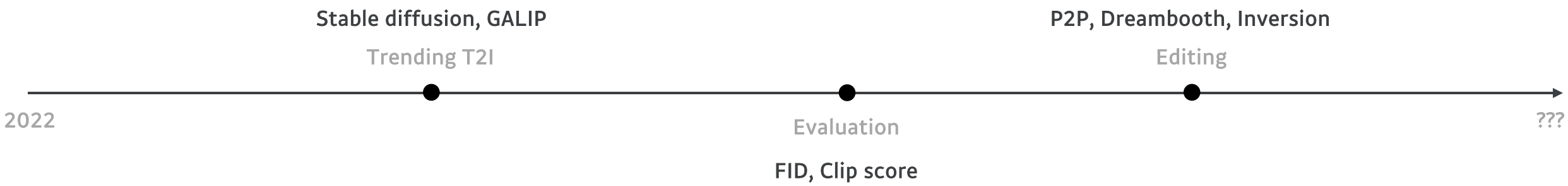
이건 ddim inversion

x_T x_t x_0

$$x_{t-1} \leftarrow \hat{\mu}_t(x_t) + \sigma_t z_t$$

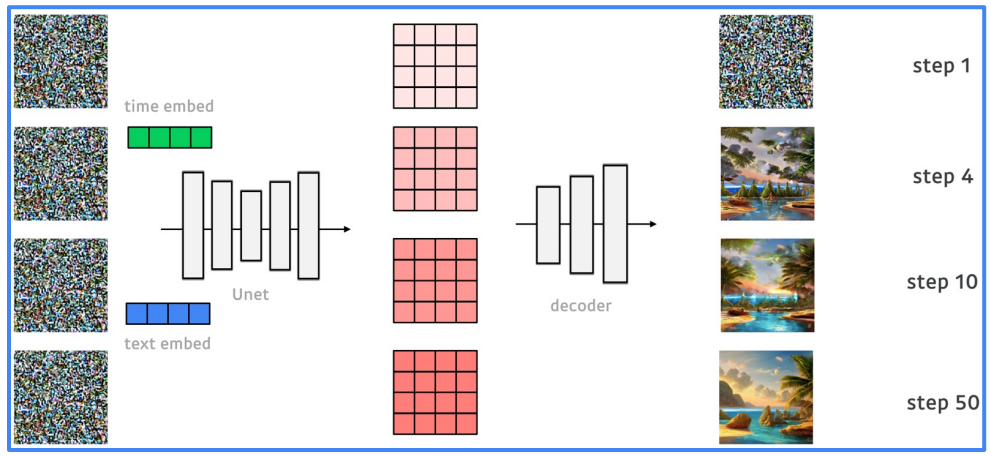
x_t 는 x_0의 semantic, structure를 잘 담아내게되게 된다.

Summary


Stable diffusion, GALIP
Trending T2I
P2P, Dreambooth, Inversion
Editing
2022
Evaluation
???
FID, Clip score

Summary

NAVER AI LAB



Stable diffusion, GALIP

Trending T2I

P2P, Dreambooth, Inversion

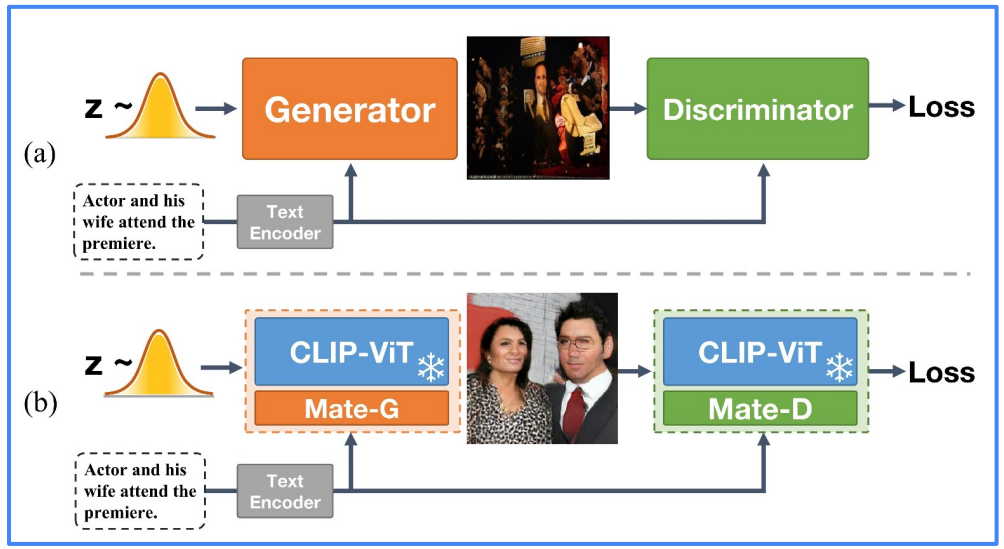
Editing

2022

Evaluation

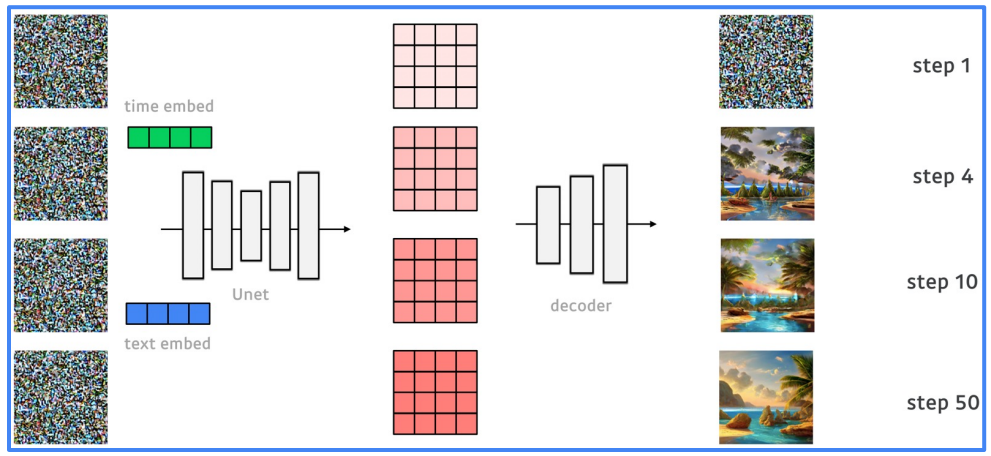
???

FID, Clip score



Summary

NAVER AI LAB



Stable diffusion, GALIP

Trending T2I

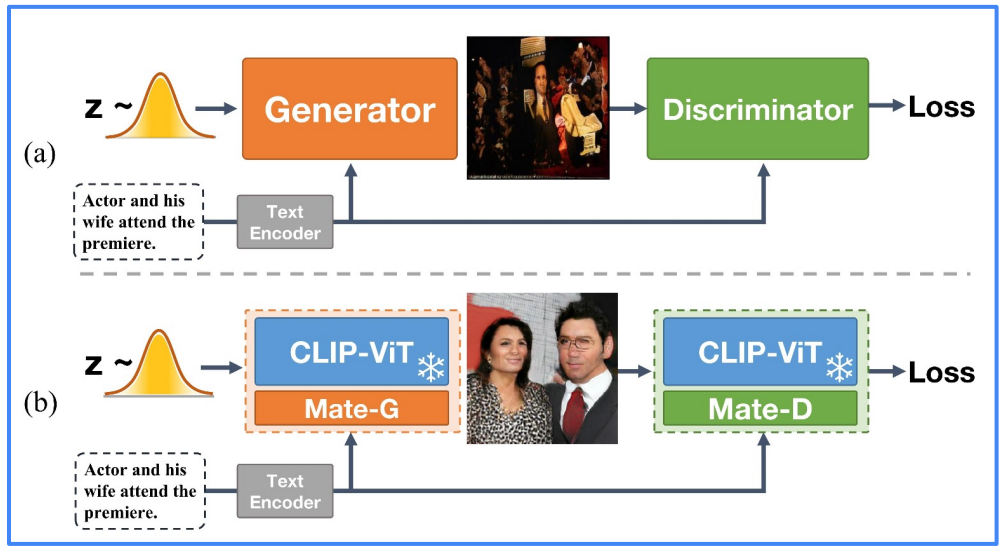
P2P, Dreambooth, Inversion

Editing

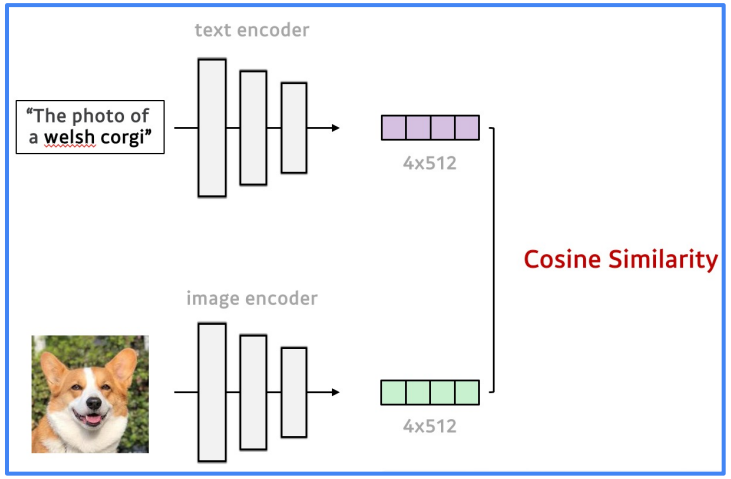
2022

Evaluation

???

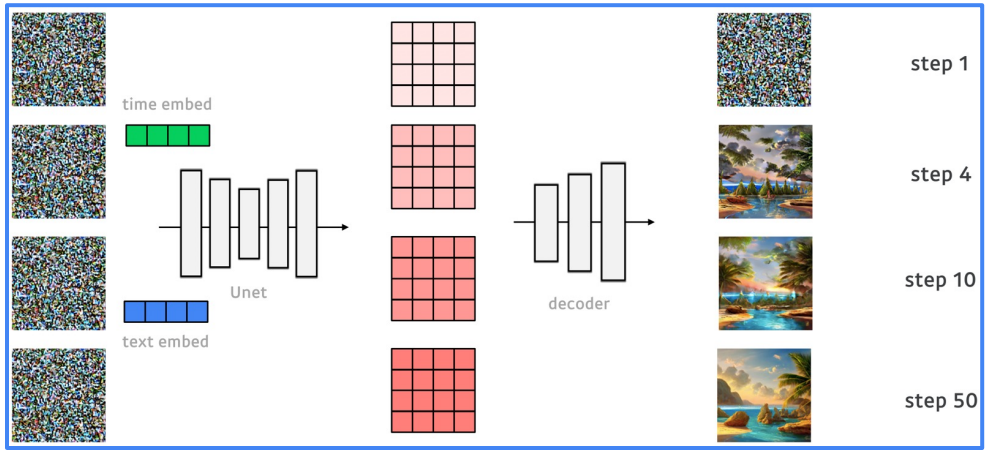


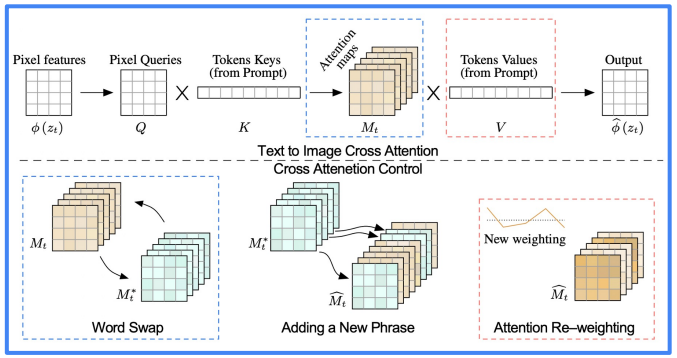
FID, Clip score

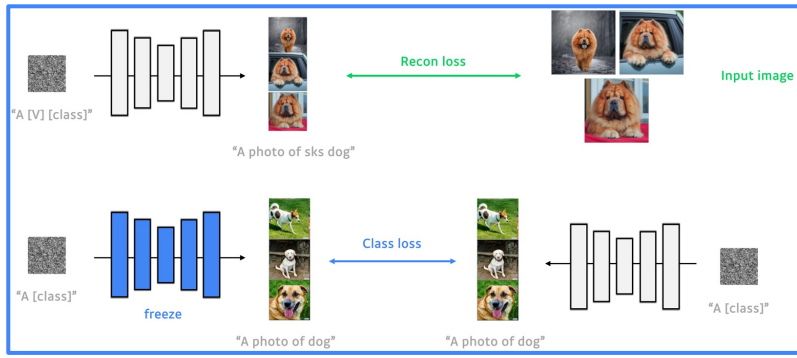


Summary

NAVER AI LAB







Stable diffusion, GALIP

P2P, Dreambooth, Inversion

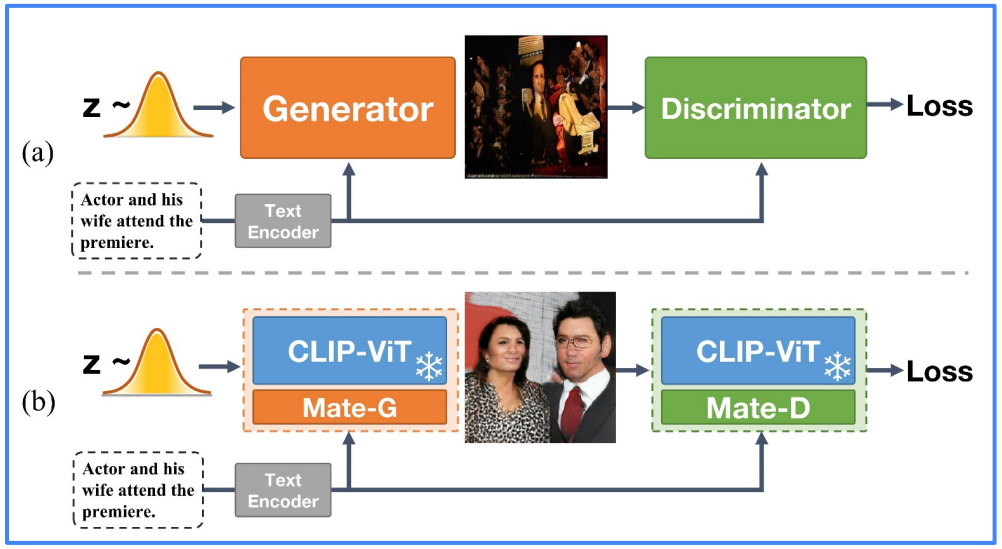
Trending T2I

Editing

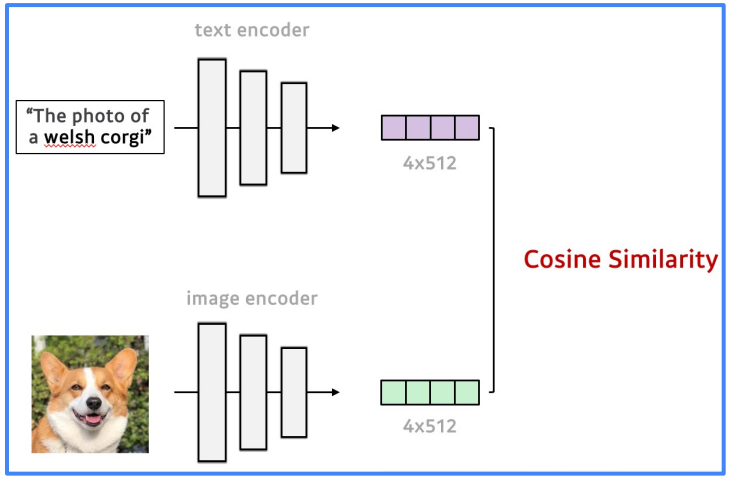
2022

Evaluation

???



FID, Clip score



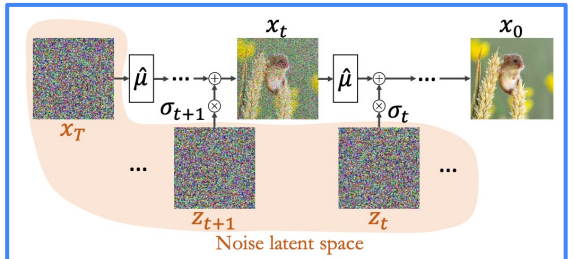


Figure 3: **The DDPM latent noise space.** In DDPM, the generative (reverse) diffusion process synthesizes an image x_0 in T steps, by utilizing $T + 1$ noise maps, $\{x_T, z_T, \ldots, z_1\}$. We regard those noise maps as the latent code associated with the generated image.

NAVER AI LAB

Thank you !

jhkim.ai@navercorp.com

Junho Kim

https://github.com/taki0112