2023

ControlNet: Adding Conditional Control to Text-to-Image Diffusion Models

김대현

2023.06.09

목차

- **Abstract**
- Prior Approaches
- Proposed Solution
- Evaluation

01 Abstract

- Background
  - ✓ 어떻게 하면 Pre-trained large DM에 보다 다양한 종류의 input condition을 최적화하여 효율적인 transfer learning을 할 수 있을까?

- Contributions
  - ✓ 다양한 input condition(edge map, segmentation map, key points 등)으로 Pre-trained DM(Stable Diffusion)을 Control 가능
  - ✓ Task-specific condition을 학습할 수 있고, 50k 이하의 데이터셋에서도 robust하게 학습할 수 있음
  - ✓ 서버용 GPU가 아닌 개인 GPU로도 충분히 학습 가능

Source image
(for canny edge detection)

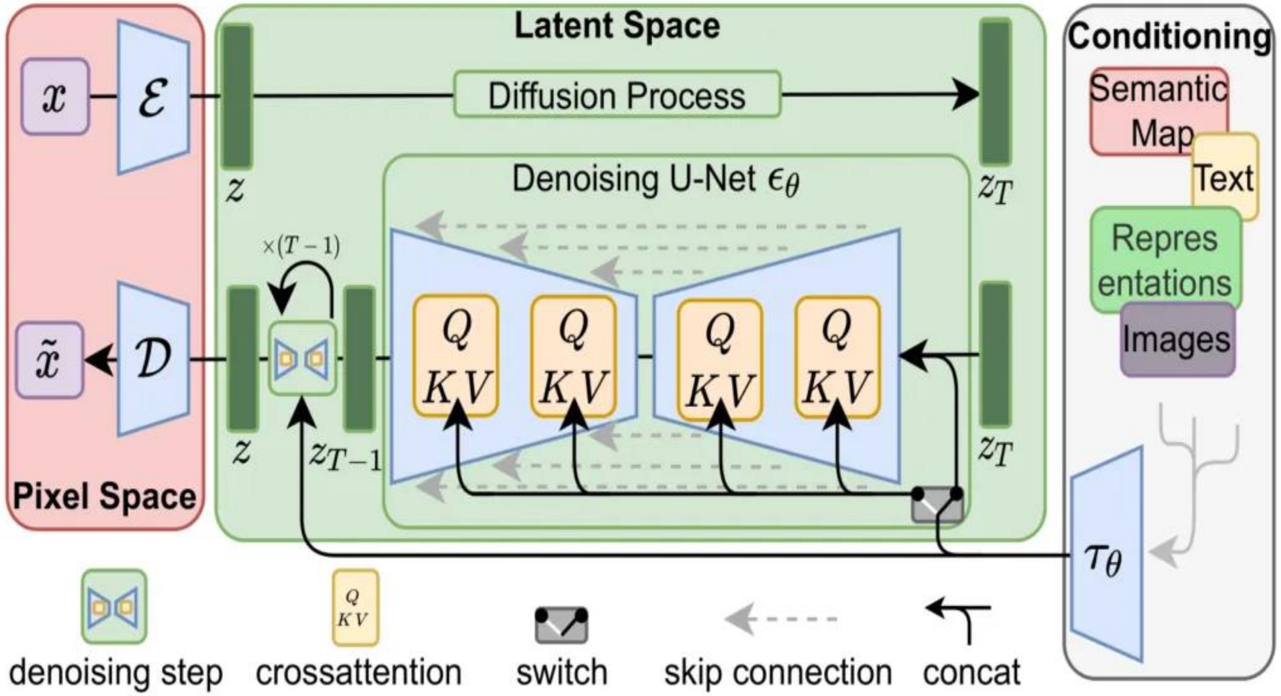
Canny edge (input)

Generated images (output)

목차

02 Prior Approaches

Stable Diffusion Model : Remind



- 특정 정보를 갖는 condition을 embedding 시킬 수 있는 task specific encoder τ_θ 활용 방안 제시
  - ✓ DM 학습 시에 τ_θ를 통해 추출된 condition vector를 attention layer를 통해 조건화할 수 있음

- 단순히 text만으로 SR 이미지를 만들 뿐만 아니라, 다양한 modality에 대한 학습된 encoder만 있다면 attention pooling를 활용하여 diffusion process를 학습시킬 수 있음

02 Prior Approaches

Stable Diffusion Model : Limitation

1. 특정 condition에 맞게 학습되려면 그만큼 network가 해당 condition을 이미지 생성에 잘 반영해야 함
   - ✓ 학습 데이터가 상당히 많이 필요함
   - ✓ Pose to image, semantic to image 등은 대량 확보가 어렵다고 함

2. 고차원의 이해가 필요한 작업들(depth, pose 등등)에는 최적화가 어려움
   - ✓ End-to-end learning* 할 수 있는 방법을 찾아야 함

End-to-end learning : Input에서 Output 까지의 pipeline network 없이 신경망으로 한번에 처리

목차

- Abstract
- Prior Approaches
- **Proposed Solution**
- Evaluation

03 Proposed Solution

What's your purpose?

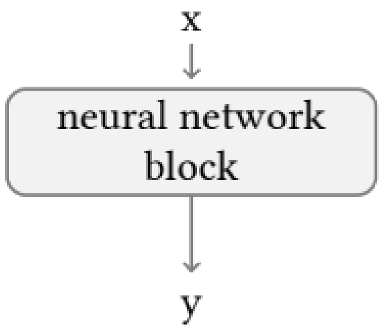
목표 : Depth map, sketch image 등 다양한 형태 (고차원의 이해가 요구되는) image에 최적화된 transfer learning architecture 제안하는 것

- 사전 훈련된 Stable Diffusion 모델 활용
- Zero Convolution을 활용

03 Proposed Solution

ControlNet : Before

- ControlNet은 전체 neural network의 행동을 통제하기 위해, neural network blocks*의 input conditions를 조작함
- x (input) : $\boldsymbol{x} \in \mathbb{R}^{h \times w \times c}$, feature map (2D feature 사용할 때)
- neural network block : $\mathcal{F}(\cdot\,;\Theta)$ Θ 는 전체 parameter들의 집합
- y (output) : 다른 feature map으로 transform
- Noised latent vector Z_t가 input으로 들어가서, 다음시점인 Z_{t-1}를 예측하는 것과 같음



(a) Before

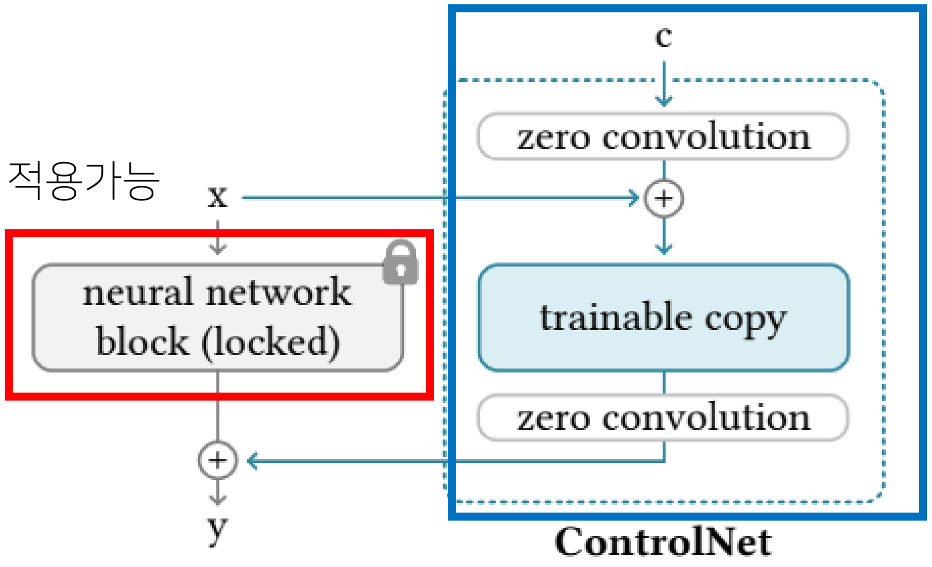
$$\boldsymbol{y} = \mathcal{F}(\boldsymbol{x};\Theta)$$

Network blocks : neural layer들을 지칭 (ex : “resnet” block , “multi-head attention” block, “transformer” block 등)

03 Proposed Solution

ControlNet : After

- DM의 parameter를 복사하여 새로운 학습 프레임워크를 병렬적으로 재구성
  - ✓ Locked copy : 학습 X, 기존 이미지 생성 network에 필요한 representation 유지
  - ✓ Trainable copy : 학습 O, conditional control을 위해 여러 task-specific dataset에 적용가능
- Θ 의 모든 parameter는 lock되어 trainable copy Θ_c 로 clone
  - ✓ Dataset이 작을 때의 overfitting을 피하고, 수억 장의 이미지로부터 학습된 large 모델의 퀄리티를 보존하기 위함
- nn block은 "zero conv"이라는 특별한 conv layer $\mathcal{Z}(\cdot;\cdot)$ 로 연결됨
  - ✓ Locked copy와 Trainable copy을 연결
  - ✓ 1x1 size conv layer는 weight와 bias 모두 0으로 초기화된 상태에서 학습
  - ✓ 두 개의 parameter instance $\{\Theta_{z1}, \Theta_{z2}\}$ 을 사용
- 학습된 ControlNet은 입력 조건의 Semantic Content를 인식함
  - ✓ 즉, Depth, Canny 등의 입력조건에 담긴 의미론적 내용을 output에 반영할 수 있다는 것을 의미함

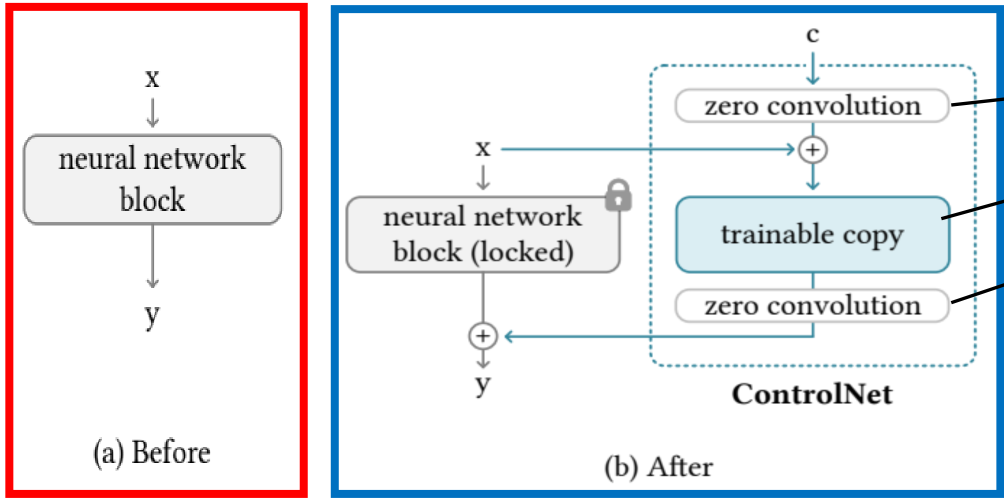


(b) After

$$\boldsymbol{y}_c = \boxed{\mathcal{F}(\boldsymbol{x};\Theta)} + \boxed{\mathcal{Z}(\mathcal{F}(\boldsymbol{x}+\mathcal{Z}(\boldsymbol{c};\Theta_{z1});\Theta_c);\Theta_{z2})}$$

03 Proposed Solution

ControlNet : Comparison with Before & After



(a) Before (b) After

$$y = \mathcal{F}(x; \Theta)$$

$$y_c = \mathcal{F}(x; \Theta) + \mathcal{Z}(\mathcal{F}(x + \mathcal{Z}(c; \Theta_{z1}); \Theta_c); \Theta_{z2})$$

$$\begin{cases} \boxed{\mathcal{Z}(c; \Theta_{z1})} = 0 \\ \boxed{\mathcal{F}(x + \boxed{\mathcal{Z}(c; \Theta_{z1})}; \Theta_c)} = \mathcal{F}(x; \Theta_c) = \mathcal{F}(x; \Theta) \\ \mathcal{Z}(\boxed{\mathcal{F}(x + \mathcal{Z}(c; \Theta_{z1}); \Theta_c)}; \Theta_{z2}) = \mathcal{Z}(\mathcal{F}(x; \Theta_c); \Theta_{z2}) = 0 \end{cases}$$

$$y_c = y$$

- ControlNet이 기존 neural network block에 적용되더라도, optimization 되기 전에는 아무런 영향을 미치지 않음
  - ✓ 기존 모델 neural block들의 capability, functionality, result quality는 완벽하게 보존됨
  - ✓ 이후의 optimization 과정은 fine-tuning만큼 빠름
  - ✓ Training이 시작되는 시점에서는, ControlNet 구조의 유무에 관계 없이 DM의 input/output과 ControlNet의 input/output은 전혀 차이 없음

03 Proposed Solution

ControlNet PoC : gradient is non-zero in zero conv

$\boldsymbol{I}$ 의 forward pass

$$\mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i} = \boldsymbol{B}_i + \sum_j^c \boldsymbol{I}_{p,i} \boldsymbol{W}_{i,j}$$

$\boldsymbol{W}$: Weight

$\boldsymbol{B}$: Bias

$\boldsymbol{p}$: 1x1 conv에 대한 spatial position (h x w)

$\boldsymbol{i}$: channel index

$\boldsymbol{I}$: 주어진 input map ($\boldsymbol{I} \in \mathbb{R}^{h \times w \times c}$)

Before Optimization : weight, bias 둘 다 0

$$\begin{cases} \dfrac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{B}_i} = 1 \\ \dfrac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{I}_{p,i}} = \sum_j^c \boldsymbol{W}_{i,j} = 0 \\ \dfrac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{W}_{i,j}} = \boldsymbol{I}_{p,i} \neq \boldsymbol{0} \end{cases}$$

After Optimization : weight는 0 아님

$$\boldsymbol{W}^* = \boldsymbol{W} - \beta_{\text{lr}} \cdot \frac{\partial \mathcal{L}}{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})} \odot \frac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})}{\partial \boldsymbol{W}} \neq \boldsymbol{0}$$

$$\frac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}^*, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{I}_{p,i}} = \sum_j^c \boldsymbol{W}^*_{i,j} \neq \boldsymbol{0}$$

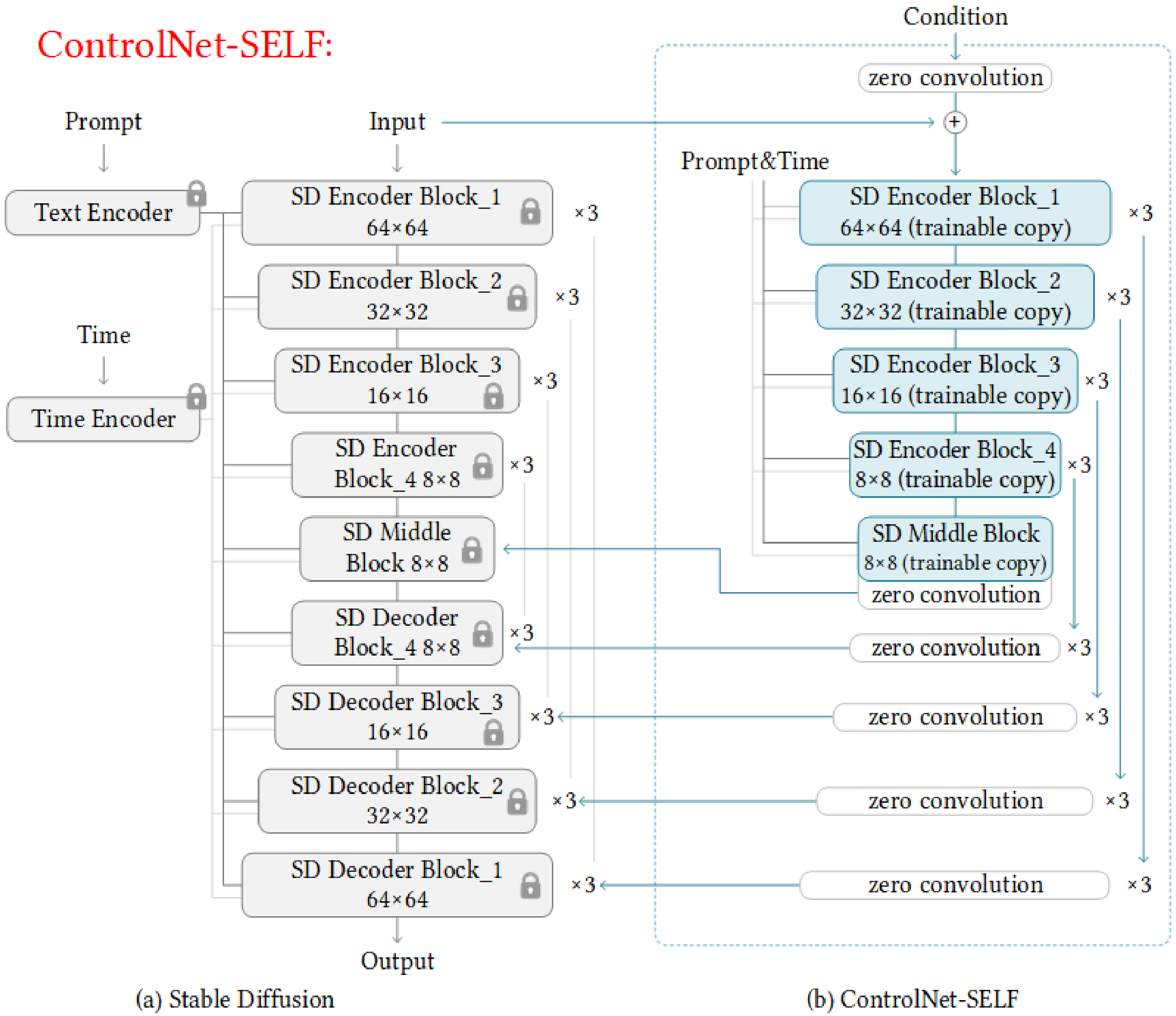
즉, feature $\boldsymbol{I}$가 0이 아닌 이상, $\boldsymbol{W}$ 은 non-zero matrix로 optimize될 것

-> $\boldsymbol{W}$, $\boldsymbol{B}$가 0으로 초기화 되더라도 $\boldsymbol{I}$ 만 0이 아니면, 학습 가능함을 의미

$\odot$: Element wise product (Hadamard product)

03 Proposed Solution

ControlNet in Image DM ：Architecture (ControlNet-SELF)



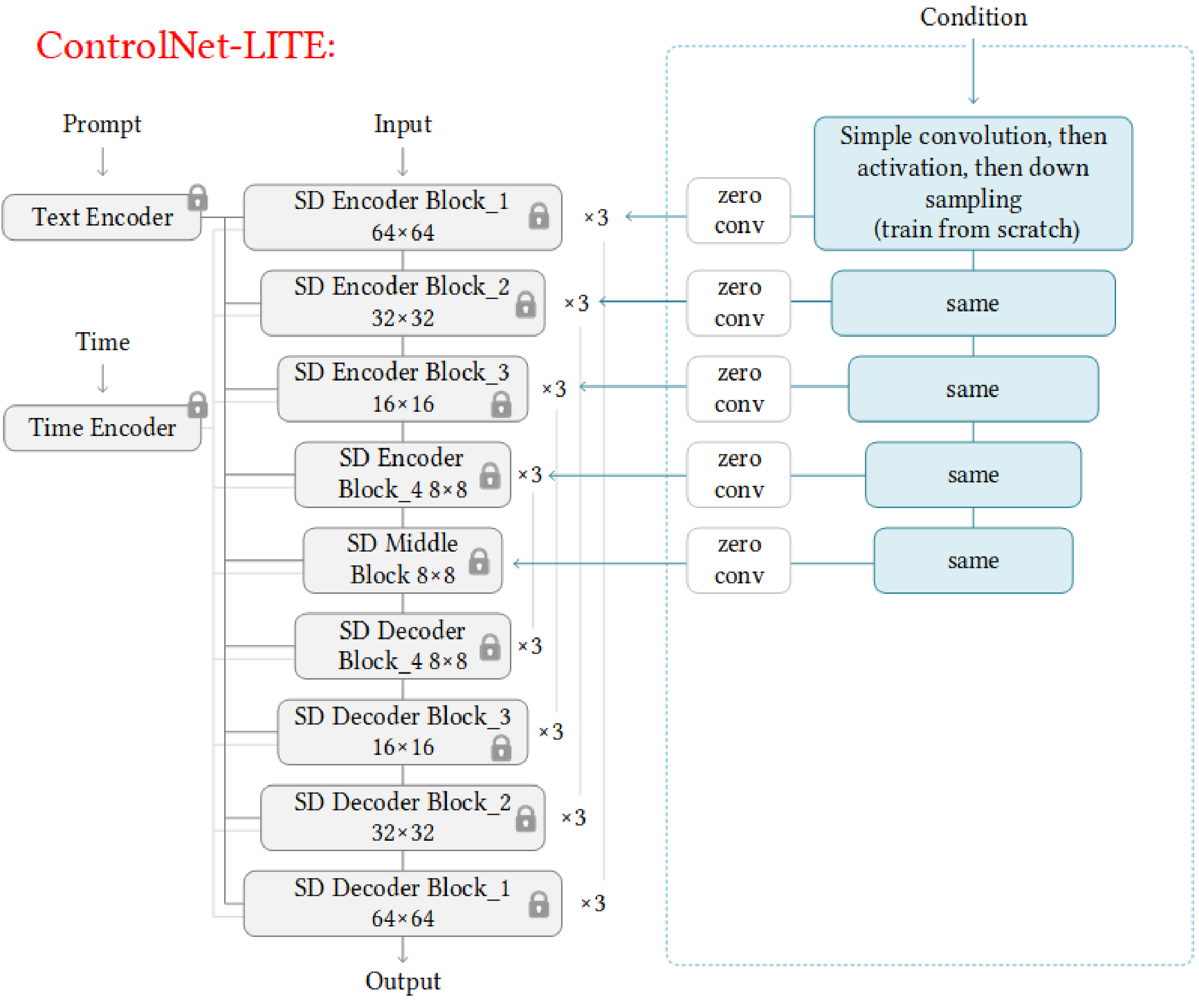
- Stable Diffusion uses a pre-processing method similar to VQ-GAN [11] to convert the entire dataset of 512 × 512 images into smaller 64 × 64 "latent images" for stabilized training. This requires ControlNets to convert image-based conditions to 64 × 64 feature space to match the convolution size.
  - ✓ 4x4 kernel, 2x2 stride로 구성된 Tiny network encoder를 활용하여 down-sampling

$$\mathcal{L} = \mathbb{E}_{z_0,t,\boldsymbol{c}_t,\boldsymbol{c}_\mathrm{f},\epsilon\sim\mathcal{N}(0,1)}\Big[\|\epsilon - \epsilon_\theta(z_t, t, \boldsymbol{c}_t, \boldsymbol{c}_\mathrm{f})\|_2^2\Big]$$

- t : Time Step
- $\boldsymbol{c}_t$: Text prompts
- $\boldsymbol{c}_\mathrm{f}$: task-specific condition(ControlNet의 input)
- z_t : noisy image
- ϵ_θ : diffusion 알고리즘

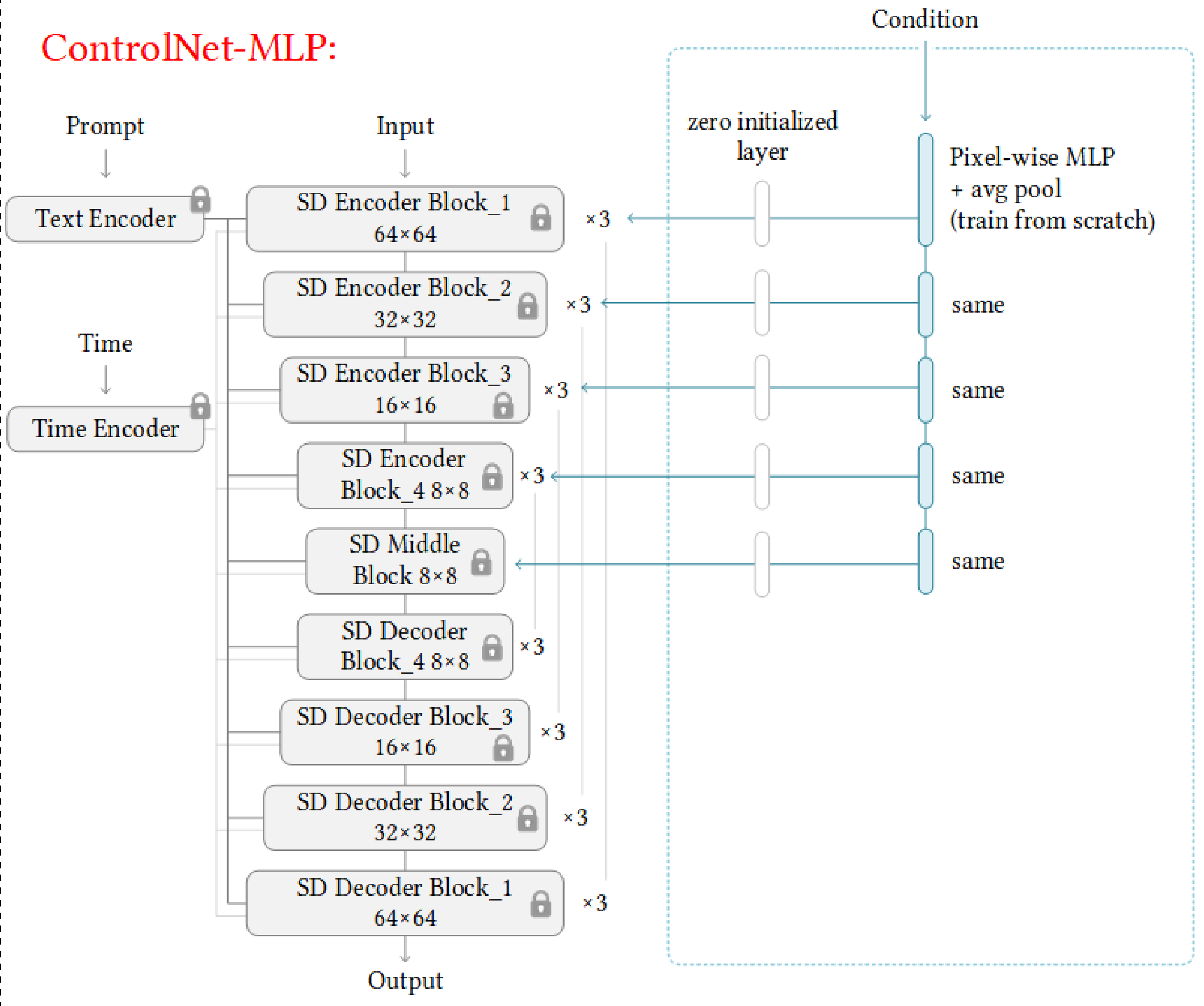
03 Proposed Solution

ControlNet in Image DM : Architecture (ControlNet-LITE, MLP)



(a) Stable Diffusion

(b) ControlNet-LITE



(a) Stable Diffusion

(b) ControlNet-MLP

03 Proposed Solution

Dataset

- 기존 dataset에 captioner를 붙여 condition-image-caption pair 제작
- Condition 마다 알맞은 pre-trained model library를 직접 활용하여 만들기도 함
- Dataset 규모 : 25,452(Normal Maps) ~ 3M(Depth-image-caption)
- Training time : 100~600 GPU-hours with Nvidia A 100 80G

Canny Edge We use Canny edge detector [5] (with random thresholds) to obtain 3M edge-image-caption pairs from the internet. The model is trained with 600 GPU-hours with Nvidia A100 80G. The base model is Stable Diffusion 1.5. (See also Fig. 4.)

Canny Edge (Alter) We rank the image resolutions of the above Canny edge dataset and sampled several sub-set with 1k, 10k, 50k, 500k samples. We use the same experimental setting to test the effect of dataset scale. (See also Fig. 22.)

Hough Line We use a learning-based deep Hough transform [13] to detect straight lines from Places2 [66], and then use BLIP [34] to generate captions. We obtain 600k edge-image-caption pairs. We use the above Canny model as a starting checkpoint and train the model with 150 GPU-hours with Nvidia A100 80G. (See also Fig. 5.)

HED Boundary We use HED boundary detection [62] to obtain 3M edge-image-caption pairs from internet. The model is trained with 300 GPU-hours with Nvidia A100 80G. The base model is Stable Diffusion 1.5. (See also Fig. 7.)

User Sketching We synthesize human scribbles from images using a combination of HED boundary detection [62] and a set of strong data augmentations (random thresholds, randomly masking out a random percentage of scribbles, random morphological transformations, and random non-maximum suppression). We obtain 500k scribble-image-caption pairs from internet. We use the above Canny model as a starting checkpoint and train the model with 150 GPU-hours with Nvidia A100 80G. Note that we also tried a more "human-like" synthesizing method [57] but the method is much slower than a simple HED and we do not notice visible improvements. (See also Fig. 6.)

Human Pose (Openpifpaf) We use learning-based pose estimation method [27] to "find" humans from internet using a simple rule: an image with human must have at least 30% of the key points of the whole body detected. We obtain 80k pose-image-caption pairs. Note that we directly use visualized pose images with human skeletons as training condition. The model is trained with 400 GPU-hours on Nvidia RTX 3090TI. The base model is Stable Diffusion 2.1. (See also Fig. 8.)

Human Pose (Openpose) We use learning-based pose estimation method [6] to find humans from internet using the same rule in the above Openpifpaf setting. We obtain 200k pose-image-caption pairs. Note that we directly use visualized pose images with human skeletons as training condition. The model is trained with 300 GPU-hours with Nvidia A100 80G. Other settings are same with the above Openpifpaf. (See also Fig. 9.)

Semantic Segmentation (COCO) The COCO-Stuff dataset [4] captioned by BLIP [34]. We obtain 164K segmentation-image-caption pairs. The model is trained with 400 GPU-hours on Nvidia RTX 3090TI. The base model is Stable Diffusion 1.5. (See also Fig. 12.)

Semantic Segmentation (ADE20K) The ADE20K dataset [67] captioned by BLIP [34]. We obtain 164K segmentation-image-caption pairs. The model is trained with 200 GPU-hours on Nvidia A100 80G. The base model is Stable Diffusion 1.5. (See also Fig. 11.)

Depth (large-scale) We use the Midas [30] to obtain 3M depth-image-caption pairs from internet. The model is trained with 500 GPU-hours with Nvidia A100 80G. The base model is Stable Diffusion 1.5. (See also Fig. 23,24,25.)

Depth (small-scale) We rank the image resolutions of the above depth dataset to sample a subset of 200k pairs. This set is used in experimenting the minimal required dataset size to train the model. (See also Fig. 14.)

목차

- Abstract
- Prior Approaches
- Proposed Solution
- **Evaluation**

04 Evaluation

Experimental Settings

- CFG*-scale at 9.0
- Sampler : DDIM (20 steps)
- Prompts to test the models:
  - ✓ No prompt : “”
  - ✓ Default prompt : “a professional, detailed, high-quality image”
  - ✓ Automatic prompt : image captioning model (BLIP)
  - ✓ User prompt

CFG (classifier-free guidance) : text prompt에 대한 weight 값 (Stable Diffusion Web UI의 parameter)

04 Evaluation

Experiment : with Prompt

Input image

Scribble map

Prompt:
Professional high-quality wide-angle digital art of a house designed by frank lloyd wright. A delightful winter scene. photorealistic, epic fantasy, dramatic lighting, cinematic, extremely high detail, cinematic lighting, trending on artstation, cgsociety, realistic rendering of Unreal Engine 5, 8k, 4k, HQ, wallpaper

04

Evaluation

Experiment Result : with Prompt

SELF (Default)

LITE

MLP

04 Evaluation

Experiment : without Prompt

Input image

Scribble map

- 실험 의도 : input 만으로 content를 예측하고자 함
  - ✓ Prompt의 영향에서 벗어나 ControlNet encoder만의 영향력을 측정

04 Evaluation

Experiment Result : without Prompt

SELF (Default)

LITE

MLP

Evaluation

Qualitative results

Canny Edge

Hough Line

Scribble

04 Evaluation

Qualitative results

HED edge

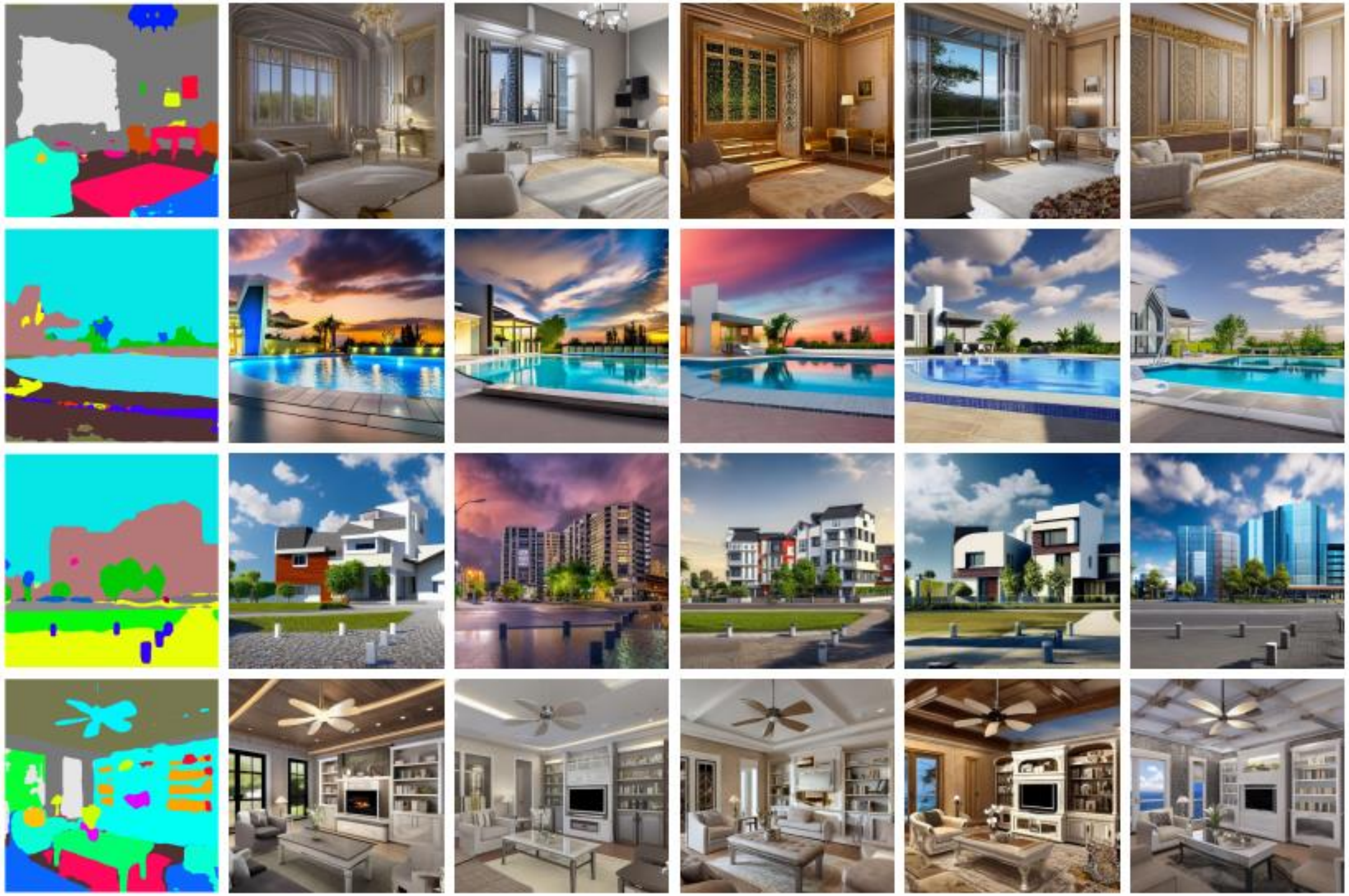
Openpifpal

Openpose

04

Evaluation

Qualitative results : Segmentation Image



Evaluation

Example Test : Animation to real image

https://www.philschmid.de/stable-diffusion-controlnet-endpoint

04

Evaluation

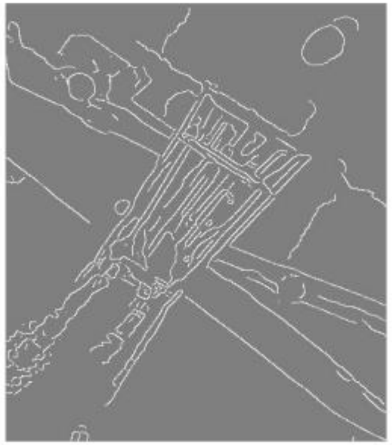
Example Test : Real to animation video

https://www.youtube.com/watch?v=xH3cBs4oV5c

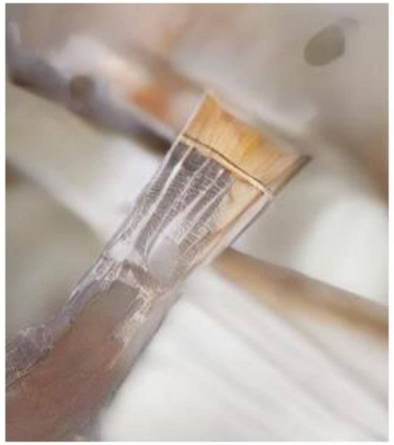
04

Evaluation

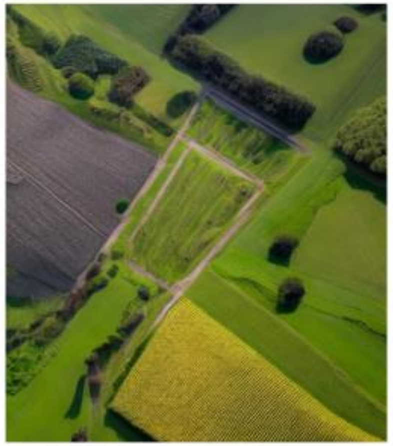
Limitation



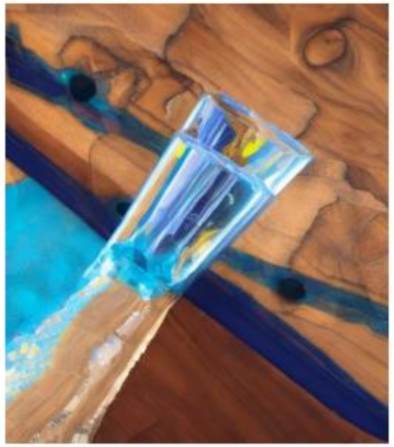
Input



Taming Transformer, Esser *et.al.*



Ours default
(Seems to be interpreted as a bird's eye view of an agricultural field)



Ours "a glass of water"
(Seems unable to eliminate the effects of mistaken recognitions)

Fig. 28 shows that when the semantic interpretation is wrong, the model may have difficulty to generate correct contents.

Figure 28: Limitation. When the semantic of input image is mistakenly recognized, the negative effects seem difficult to be eliminated, even if a strong prompt is provided.

Thank you!