Adding Conditional Control to Text-to-Image Diffusion Models

Lvmin Zhang and Maneesh Agrawala

Stanford University

발표자 : 임지섭

Content

- **Abstract**
- **Introduction**
- **Related Work**
- **Method**
- **Experiment**
- **Limitation**

Abstract

- 사전 훈련된 Large diffusion model을 제어하여 추가적인 입력 조건을 지원하는 모델 ControlNet을 제안.
- ControlNet은 end-to-end 방식으로 과제별 조건을 학습하며, 작은 데이터셋(< 50k)에서도 학습이 robust하다. 또한, 학습 시간이 빠르고 개인 장치에서도 학습 가능하다.
- Large Diffusion model인 Stable Diffusion 같은 모델에 ControlNet을 추가하여 edge maps, segmentation maps, key point등의 condition의 입력을 가능하게 할 수 있다.
- 이러한 방법은 Large Diffusion model을 제어하는 방법을 풍부하게 하고 관련 응용 프로그램을 더욱 용이하게 할 수 있다.

Introduction

- 최근 large text-to-image model들이 등장하면서, 사용자가 입력한 간단한 prompt만 으로도 시각적으로 매력적인 이미지를 생성할 수 있게 되었다.
- 하지만 이러한 prompt 기반 방식으로 이미지를 생성하면서 image processing에서 생각해볼 수 있는 몇 가지가 있다.

  1. 특정 Task를 하는 데에 Large model들을 적용할 수 있을까?
  2. 범위가 넓은 문제 condition과 user control들을 처리하는데 어떤 프레임 워크를 만들어야 될까?
  3. 특정 Task를 할때 large model의 장점과 기능을 보존할 수 있을까?

Introduction

- 첫째, 특정 task에서 사용 가능한 데이터셋의 규모는 일반적인 이미지-텍스트 영역의 데이터 규모와 항상 같지 않다.
  많은 문제(object shape/normal, pose understanding, etc)들에서 최대 데이터셋 크기는 대부분 100k 이하이며, 이는 LAION 5B의 5×10^4배 이상 작다. 따라서 large model이 특정 문제를 해결할 때 **generalization 능력을 유지시키는 train 방법**이 필요.

- 둘째, 데이터 기반 솔루션으로 Image processing task들을 처리할 때 large computation cluster를 항상 사용할 수 없다.
  따라서 large model을 특정 task에 대해 optimizing 빠른 train 방법이 중요, 이를 위해 **fine-tuning, transfer learning** 등이 필요.

- 셋째, 다양한 Image processing 문제들은 problem definitions, user control, image annotation등에 대해 다양한 형태를 가지고 있어서 이런 문제들을 처리할 때 diffusion algorithm은 denoising process, multi-head attention activations 편집들을 제어해 "**procedural**(절차적)" 방식으로 조절할 수 있지만, 이러한 방식은 기본적으로 hand-craft 에 따라 규정된다.
  따라서 depth-to-image, pose-to-human등과 같은 task들은 raw input을 object level 이나 scene-level 로 해석하는 것을 요구하므로 procedural 방법은 덜 적합하다. 많은 task에서 솔루션을 얻기 위해서는 **end-to-end 학습이 필수**

Introduction

- Large diffusion model(Stable Diffusion 등)을 제어하여 추가적인 입력 조건을 지원하는 end-to-end 신경망 모델 ControlNet을 제안.
- ControlNet은 Large diffusion model 가중치를 **"trainable copy"**와 **"locked copy"**로 복제하여 사용
- "locked copy"는 수십억 개의 이미지에서 학습한 네트워크 능력을 보존하면서, "trainable copy"는 task-specific 한 데이터셋에서 조건부 제어를 학습한다. "trainable copy"와 "locked copy"는 "zero convolution"이라는 특수한 convolution layer로 연결
- "zero convolution"은 **production-ready weight**(기존 model의 weight) 가 보존 되기 때문에 훈련이 다양한 규모의 데이터셋에 대해 robust하며, 더 많은 **노이즈를 추가하지 않으므로** 새로운 레이어를 처음부터 훈련하는 것보다 빠르다.

Introduction

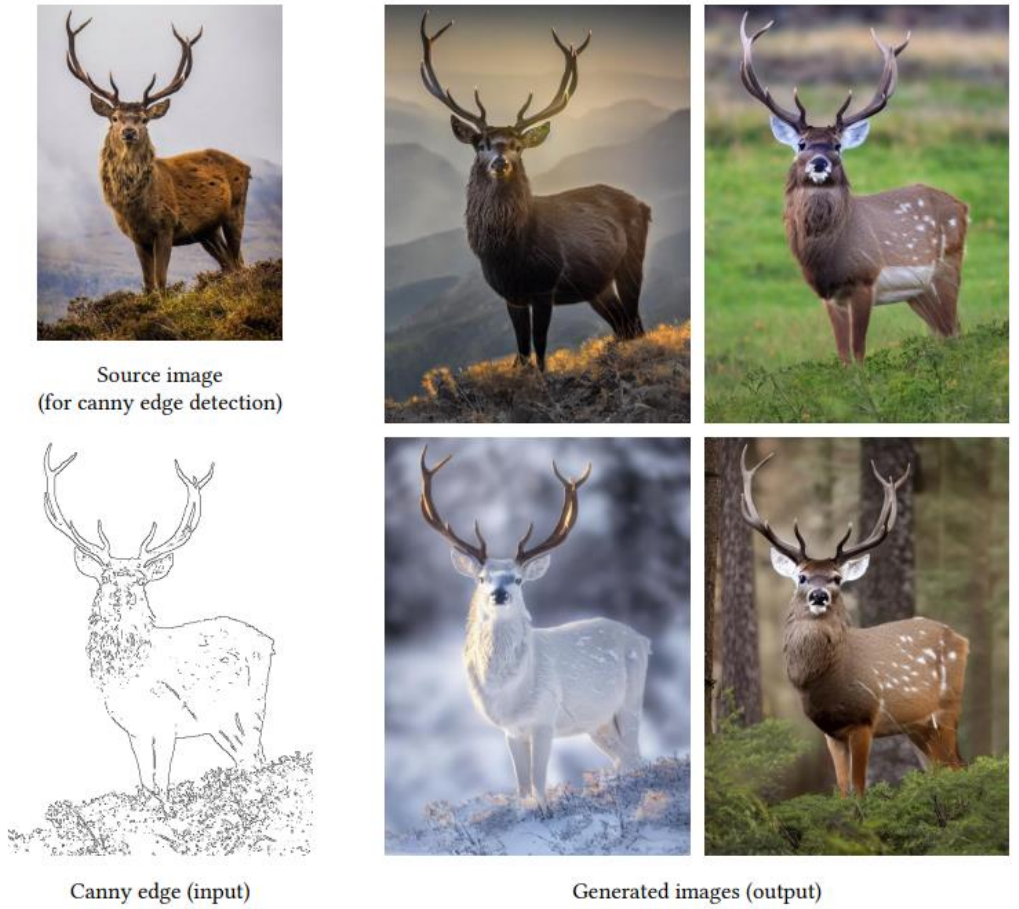


Figure 1: Control Stable Diffusion with Canny edge map. The canny edge map is input, and the source image is not used when we generate the images on the right. The outputs are achieved with a default prompt *"a high-quality, detailed, and professional image"*. This prompt is used in this paper as a default prompt that does not mention anything about the image contents and object names. Most of figures in this paper are high-resolution images and best viewed when zoomed in.

Related Work

- **HyperNetwork and Neural Network Structure**

  HyperNetwork은 큰 신경망의 가중치에 영향을 미치기 위해 작은 순환 신경망을 훈련시키는 방법(NLP에서 주로쓰였음)
  GAN 및 다른 machine learning task 에서도 성공적 이었다 이러한 아이디어에서 영감을 받아 Stable Diffusion에 작은 신경망을 연결하는 방법을 제공했다(NovelAI). ControlNet과 HyperNetwork는 신경망의 동작을 영향을 미치는 방식에서 유사점이 있다.

- **Diffusion Probabilistic Model**

  Diffusion probabilistic model은 여러 model 들을 거쳐 DDPM, DDIM, 그리고 score-based diffusion 와 같은 중요한 훈련 및 샘플링방법으로 개선되었다. 이후 Diffusion 모델 훈련에 필요한 연산 비용을 줄이기 위해 latent image를 기반으로 LDM이 제안되었으며,이후 Stable Diffusion으로 확장

- **Text-to-Image Diffusion**

  Text-to-Image Diffusion은 CLIP 와 같은 사전 학습 언어 모델을 사용하여 텍스트 입력을 latent vector로 인코딩하고, 이미지 생성 작업에 적용하여 이미지를 생성 할 수 있다. Glide는 이미지 생성 및 편집을 모두 지원하는 text-guided diffusion model.
  Disco Diffusion은 텍스트 프롬프트를 처리하기 위한 clip-guided implementation. Stable Diffusion은 latent Diffusion의 대규모 implementation 텍스트에서 이미지 생성. Imagen은 latent image를 사용하지 않고 피라미드 구조를 사용하여 픽셀을 직접 확산시켜 텍스트에서 이미지를 생성하는 구조.

Related Work

- **Personalization,Customization, and Control of Pretrained Diffusion Mode**

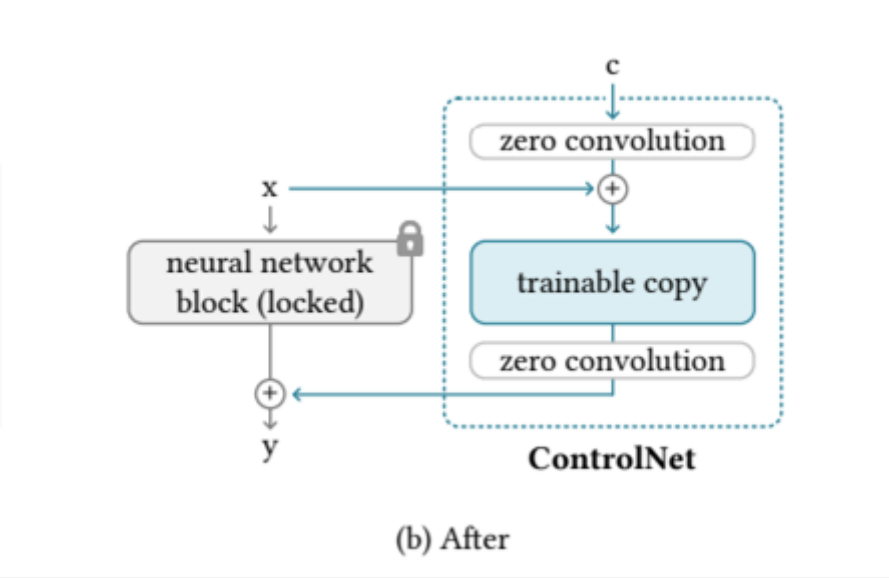
  SOTA image diffusion models은 텍스트-이미지 생성 방법으로 지배되고 있기 때문에,diffusion model을 control 하기 가장 직관적인 방법은 text guided model, 이러한 유형은 CLIP 기능을 조작하여 제어 할 수도 있다
  Textual Inversion과 DreamBooth는 같은 주제나 대상을 가진 작은 이미지 셋을 사용하여 생성된 결과물을 customize(personalize)하기 위해 제안된 방법.

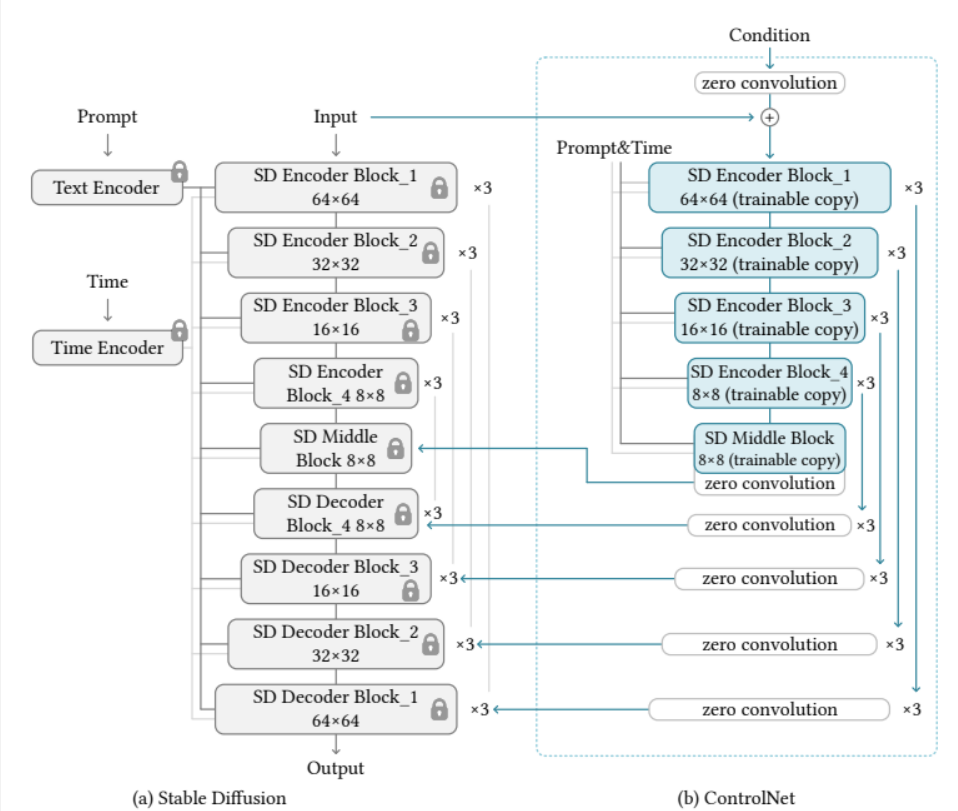
- **Image-to-Image Translation**

  Image to Image Translation과 ControlNet은 많은 공통 응용 프로그램이 있을 수 있지만, 그들의 동기는 본질적으로 다르다.
  Image to Image Translation은 서로 다른 도메인의 이미지 간의 매핑을 학습하는 것이 목표이며, ControlNet은 작업 특정 조건(condition)으로 diffusion model을 제어하는 것을 목표한다. sketch-guided diffusion과 같은 구체적인 분야에서는 은 diffusion프로세스를 조작하는 최적화 기반 방법입니다. 이러한 방법들은 experiments에서 테스트된다.

Method

- **ControlNet**

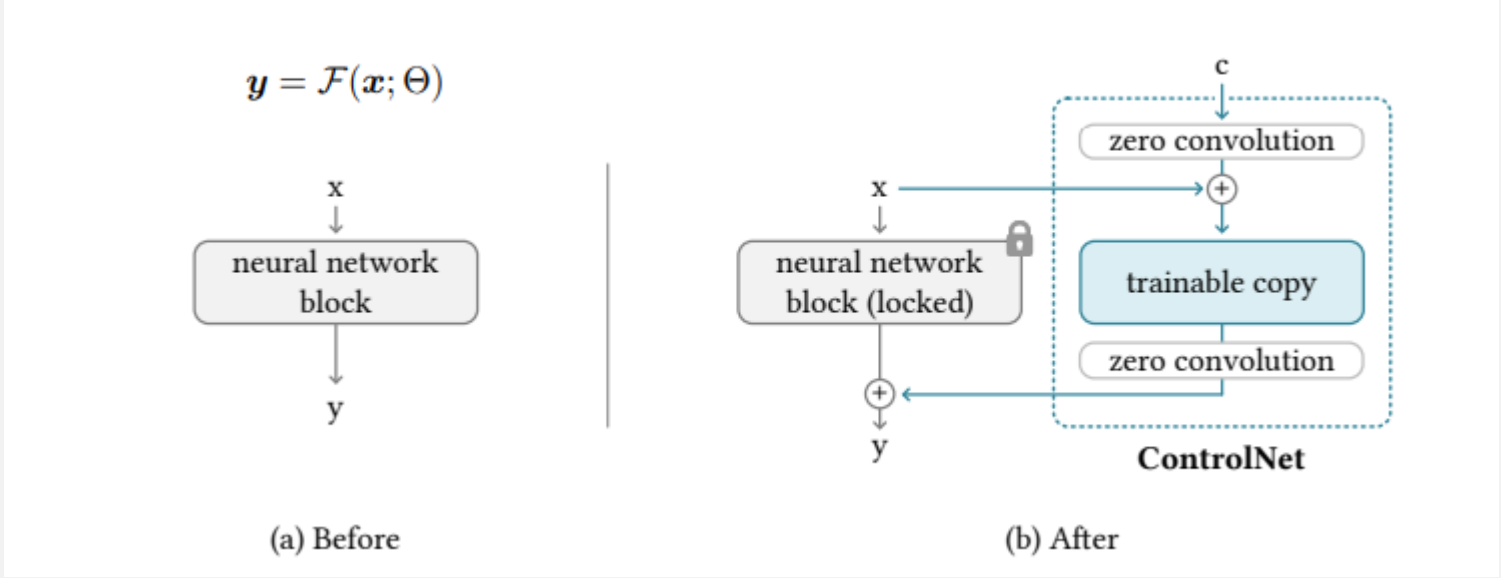




Method

- **ControlNet**

  ControlNet은 전체 신경망의 전반적인 동작을 조종하기 위해 "network block"(예를 들어 “resnet” block, “conv-bn-relu” block, multi-head attention block, transformer block등) 의 input condition을 조작한다.



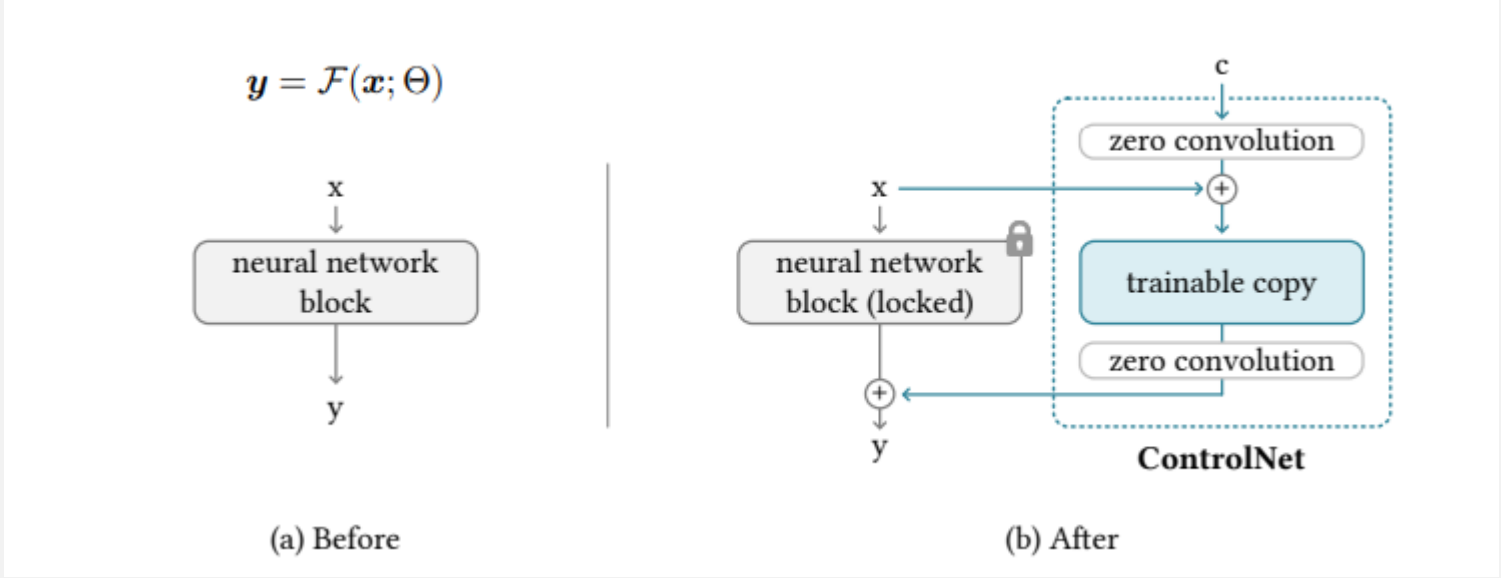
Method

- **ControlNet**

  Θ의 모든 parameter를 freeze 시키고이를 **trainable copy**인 Θc로 복사한다. 이 복사된 Θc는 외부 조건 벡터 c와 함께 학습된다.

  원래의 가중치를 직접 학습하는 것보다는 새로운 parameter를 학습 시키는 것이 데이터셋이 작을 때 overfitting을 방지하고 수십억 개의 이미지에서 학습한 large model의 production-ready quality 를 유지시키기 용이

  원래의 parameter = "locked copy" (이미지의 회색)
  새로운 parameter = "trainable copy"(이미지의 하늘색)



Method

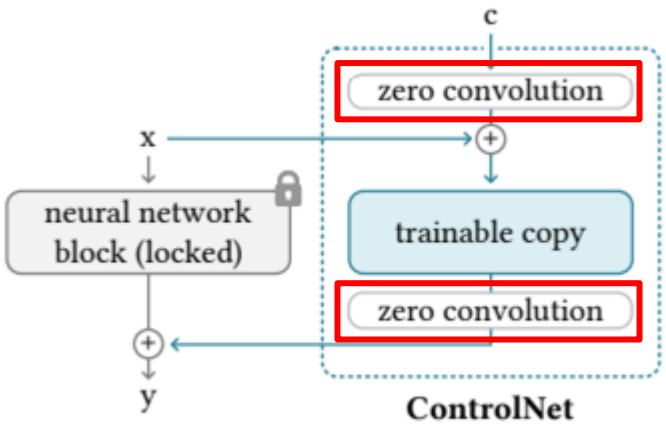
- **ControlNet**

  "network block"들은 "zero convolution"이라는 독특한 유형의 컨볼루션 레이어에 의해 연결

  "zero convolution"은 가중치와 바이어스가 모두 0으로 초기화된 1x1 컨볼루션 레이어.

  "zero convolution"의 연산을 Z(·; ·)로 표시하며, 두 개의 parameter {Θz1, Θz2} 인스턴스를 사용하여 ControlNet 구조를 구성

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



(b) After

Method

- **ControlNet**

  여기서 yc는 이 "network block" 의 output이 되며, "zero convolution"레이어의 weight와 bias가 모두 0으로 초기화되기 때문에 첫 번째 학습 단계에서 밑에 수식이 성립해서 결국 y = yc 가 된다.

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

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

$$\boldsymbol{y}_c=\boldsymbol{y}$$

가중치 W, 편향 B, input I에 대한 zero convolution의 gradient :

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

W, B = 0 인 초기 상태에서의 gradient :

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

Method

- **ControlNet**

Input에 대한 gradient는 0 으로 만들지만 Input이 0이 아닌 한 zero convolution의 W, B의 gradient는 0이 되지 않고 gradient descent가 가능하다

loss function L과 learning rate 인 βlr이 0 이 아니면 outside gradient (∂L/∂Z(I; {W, B}))는 0이 아니다.
그래서 밑에 식처럼 W*(gradient descent이후 가중치)를 얻을 수 있다

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

first step만 지나게 되면 0이 아닌 weight를 만들기 때문에 바로 다음 step에서는 0이 아닌 W를 얻게 되서 학습이 시작

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

※ ControlNet이 "network block"을 적용하더라도, optimization되기 전에는 아무런 영향을 미치지 않는다.
기존 모델 neural block들의 capability, functionality, result quality는 완벽하게 보존되며, 이후의 optimization 과정은 fine-tuning만큼 빠르다. (scratch부터 해당 layer들을 학습시키는 것보다 훨씬 빠르다)

Method

- **ControlNet**

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

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

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

$$y = wx + b$$

and we have

$$\partial y/\partial w = x, \partial y/\partial x = w, \partial y/\partial b = 1$$

and if $w = 0$ and $x \neq 0$, then

$$\partial y/\partial w \neq 0, \partial y/\partial x = 0, \partial y/\partial b \neq 0$$

which means as long as $x \neq 0$, one gradient descent iteration will make w non-zero. Then

$$\partial y/\partial x \neq 0$$

so that the zero convolutions will progressively become a common conv layer with non-zero weights.

Method

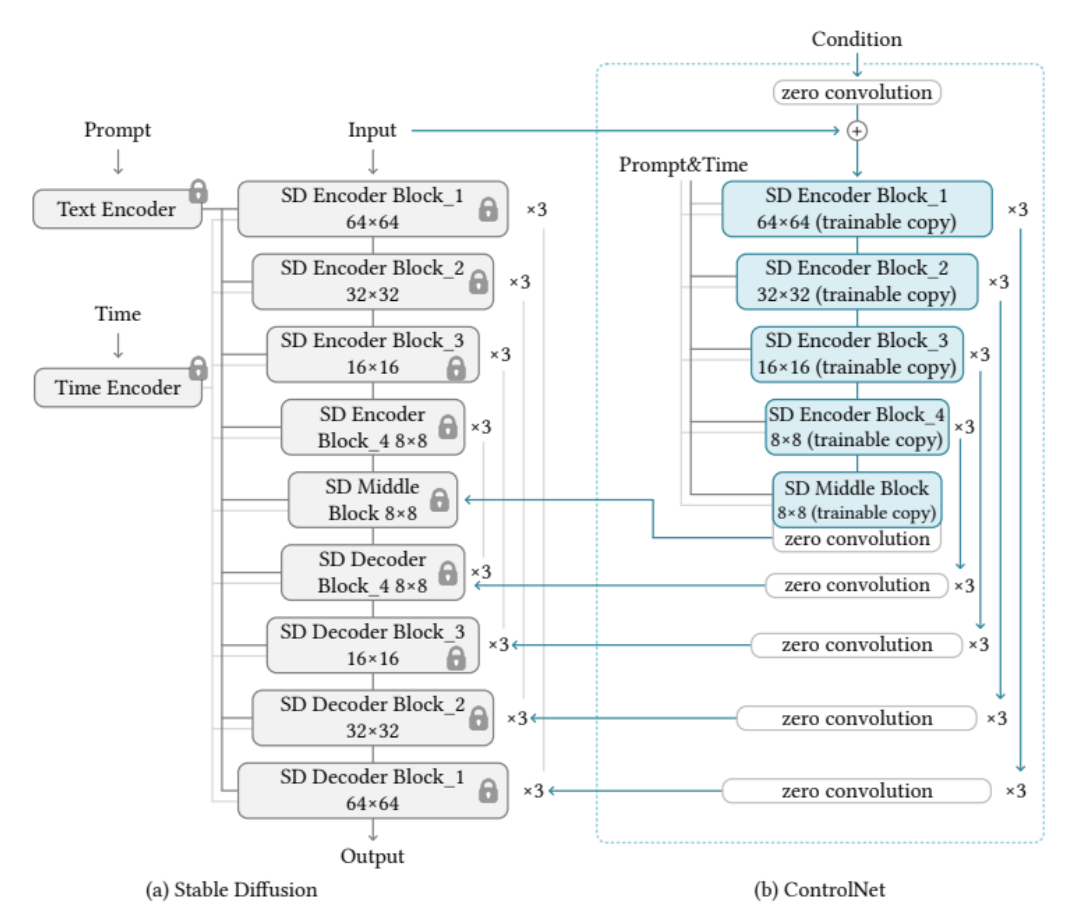
- **ControlNet in Image Diffusion Model**

  large diffusion model로는 Stable Diffusion을 사용

  ControlNet는 convolution size를 맞추기 위해 512x512의 image-based condition을 4 × 4 kernels ,2 × 2 strides (ReLU, channels 은 Gaussian weights로 initialized 된 16, 32, 64, 128) 사용해 **64x64 feature space로 encoding**

  tiny network E(·) 를 이용 ci를 cf 로 만든다

$$c_f = \mathcal{E}(c_i)$$



Method

- **Training**

  stable diffusion의 loss에 condition cf만 추가된 loss
  (noisy image : zt, time step : t, text prompts :ct, task-specific condition : cf)

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{z}_0,t,\boldsymbol{c}_t,\boldsymbol{c}_\mathrm{f},\epsilon\sim\mathcal{N}(0,1)}\left[\|\epsilon - \epsilon_\theta(z_t, t, \boldsymbol{c}_t, \boldsymbol{c}_\mathrm{f})\|_2^2\right]$$

  stable diffusion의 loss

$$L_{LDM} := \mathbb{E}_{\mathcal{E}(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2\right]$$

Method

- **Improved Training**

  Small-Scale Training:
  RTX 3070TI 노트북 GPU에서 테스트함 1,2,3,4에 대한 연결을 끊고 중간 블록에만 연결, 학습 속도를 약 1.6배 높일 수 있다

  Large-Scale Training:
  적어도 8개의 Nvidia A100 80G 또는 동등한 장비에서 large dataset으로 훈련, 이 경우, 과적합의 위험이 비교적 낮기 때문에 50k steps 이상 반복 이러면 problem-specific model을 얻을 수 있다

Experiment

- **Experimental Setting**

  DDIM 샘플러를 사용 기본적으로 20 steps
  모델을 테스트하기 위해 세 가지 유형의 프롬프트를 사용

  (1) No prompt: 빈 문자열 ""을 prompt로 사용

  (2) Default prompt: Stable diffusion은 본질적으로 prompt로 훈련됨, 더 나은 설정은 “an image”, “a nice image”, “a professional image”와 같은 의미 없는 프롬프트를 사용하는 것. 이 논문에서는 “a professional, detailed, high-quality image” 를 기본 프롬프트로 사용

  (3) Automatic prompt: 완전 자동화 파이프라인의 최신 기술 수준을 테스트하기 위해 auto image captioning 방법(예: BLIP)을 사용하여 "default prompt" 모드로 얻은 결과를 사용하여 prompt를 생성, 생성된 프롬프트 사용

  (4) User prompt: 사용자가 prompt를 제공

Experiment

- **Qualitative Results**

  We present qualitative results in Fig. 4, 5,6,7,8,9,10,11,12,13,14,15.

  https://arxiv.org/pdf/2302.05543.pdf#page=20&zoom=100,144,678

Experiment

- **Ablation Study**

  Fig. 20 shows a comparison to a model trained without using ControlNet. That model is trained with exactly same method with Stability's Depth-to-Image model (Adding a channel to the SD and continue the training).
  Fig. 21 shows the training process. We would like to point out a "sudden convergence phenomenon" where the model suddenly be able to follow the input conditions. This can happen during the training process from 5000 to 10000 steps when using 1e-5 as the learning rate.
  Fig. 22 shows Canny-edge-based ControlNets trained with different dataset scales.
  4

Experiment

- **Comparison to previous methods**

  Fig. 14 shows the comparison to Stability's Depth-to-Image model.
  Fig. 17 shows a comparison to PITI [59].
  Fig. 18 shows a comparison to sketch-guided diffusion [58].
  Fig. 19 shows a comparison to Taming transformer [11].

Experiment

- **Comparison of pre-trained models**

  We show comparisons of different pre-trained models in Fig. 23, 24, 25.

Experiment

- **More Applications**

  Fig. 16 show that if the diffusion process is masked, the models can be used in pen-based image editing.
  Fig. 26 show that when object is relatively simple, the model can achieve relatively accurate control of the details.
  Fig. 27 shows that when ControlNet is only applied to 50% diffusion iterations, users can get results that do not follow the input shapes.

Limitation

- Fig. 28
  의미 해석이 잘못된 경우 모델이 올바른 내용을 생성하는 데 어려움을 겪을 수 있음을 보여준다.