Zero-shot Image-to-Image Translation

2022314182 / 박 수 연

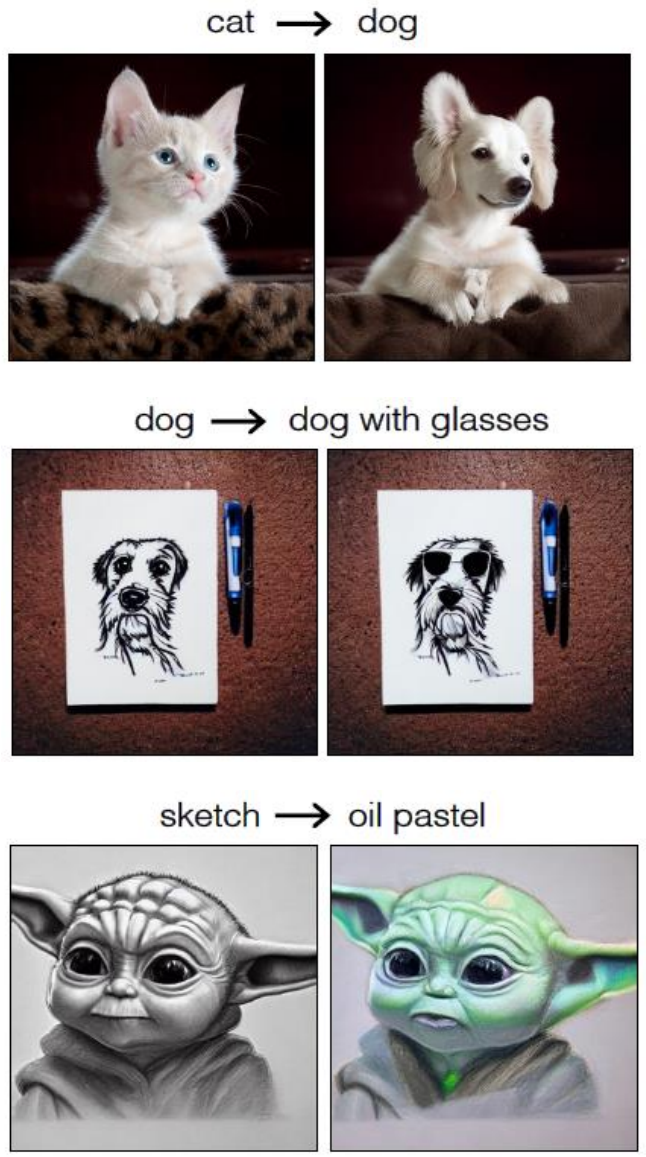
Contents

Introduction

1. Introduction
Pix2Pix-zero

이미지를 수정하기 위해서는
원래 이미지에서 수정하고 싶은 부분만 수정이 되고, 보존하고 싶은 부분은 유지를 할 수 있어야 한다.
이미지 생성 모델 Stable Diffusion 모델을 이용하여 이미지를 수정할 때,
수정하고 싶지 않은 부분까지 수정이 되어 버리는 문제가 발생된다.
이에, 본 논문에서는 특정 target 하는 부분만 수정이 되고, 유지하고 싶은 부분은 유지하면서,
수정된 부분이 기존 이미지와 잘 융합이 될 수 있는 방법을 제시하며,

그 방법은,
(1) 이미지의 수정이 잘 되기 위해 inversion 을 찾고
(2) 수정이 되며 안되는 부분은 기존 이미지의 cross attention map 을 이용하여 이를 유지하며
(3) 수정하고자 하는 부분은 edit direction 을 이용하여 수정
의 과정으로 이루어 진다.



Method

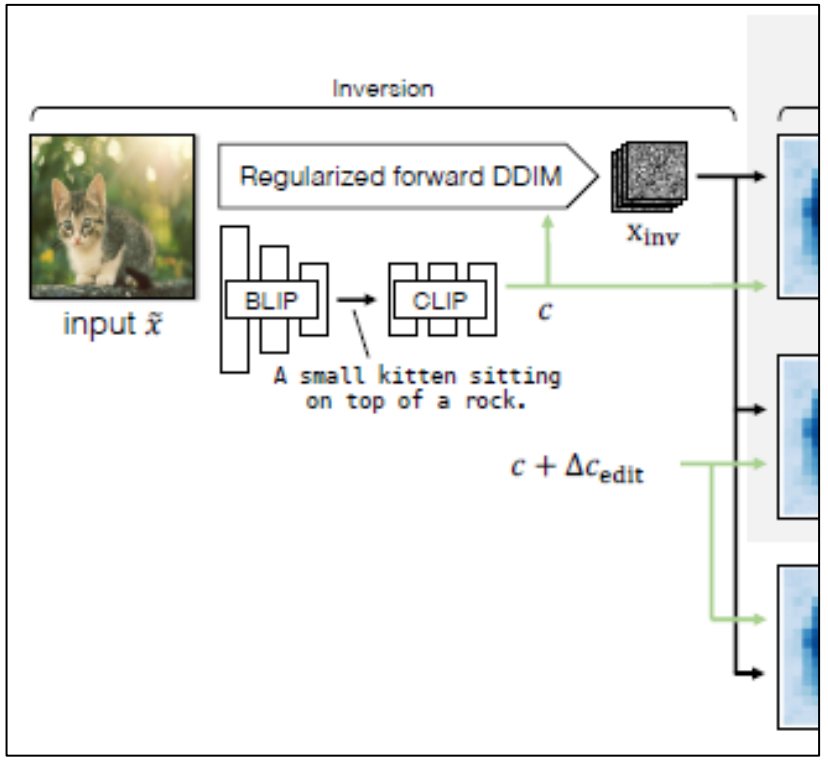
2. Method
(1) Inversion

1. inversion 이란?
   이미지를 구성하기 위해 필요한 조건을 찾는 것
   stable diffusion 을 통해서 이미지를 형성하기 위해서는 (1) random initial image (2) text condition 이 필요
   즉, random initial image 을 잘 형성하는 것이 중요함

2. uncontrolled inverted image 의 한계
   stable diffusion scheduler 을 통해서 얻어지는 inverted image 는 ideal 한 Gaussian white noise를 보이지 못함

3. ideal 조건 (Gaussian White noise)
   (1) no correlation between locations
   (2) zero mean, unit variance

4. controlling inverted image
   (1) L2 Loss
   (2) KL divergence Loss



2. Method
(1) Inversion

1. controlling inverted image

   (1) L2 Loss

$$\mathcal{L}_{\text{pair}} = \sum_{p} \frac{1}{S_p^2} \sum_{\delta=1}^{S_p-1} \sum_{x,y,c} \eta^p_{x,y,c} \left(\eta^p_{x-\delta,y,c} + \eta^p_{x,y-\delta,c} \right)$$

- standard Gaussian 분포를 따른다면, 픽셀 위치 rolled 되어도,특이하게 튀는 값이 없어야 함.
- 만약 standard Gaussian 분포를 따르지 않는다면 분포의 쏠림과 같은 현상이 있을 수 있음
- inverted image 의 rolled 전/후의 L2 loss 를 빼서, 보다 standard Gaussian 분포로 만듦

한 칸 rolling 한 이미지

(1) 가우시안 분포를 잘 따르는 경우

→ rolling 을 해도 그 분포가 유사

(2) 가우시안 분포를 잘 따르지 않는 경우

→ rolling 을 했을 때 분포가 상이

→ L2 loss 를 빼서 보다 정규 분포형으로 만들 수 있음

2. Method
(1) Inversion

1. controlling inverted image

(2) KL divergence loss

- 두 분포의 차이를 계산하는 방식
- 이상적인 standard Gaussian, N(0,1) 의 분포를 가지는 inverted image 를 가지기 위해서는 inverted image 의 분포가 N(0,1) 의 분포와 유사해야 함
- 즉, 두 분포의 차이를 계산하여 이를 줄임
- 두 분포, p(x)~N(μ_1, σ_1^2) 와 q(x)~N(μ_2, σ_2^2) 의 분포의 KL divergence 는,

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

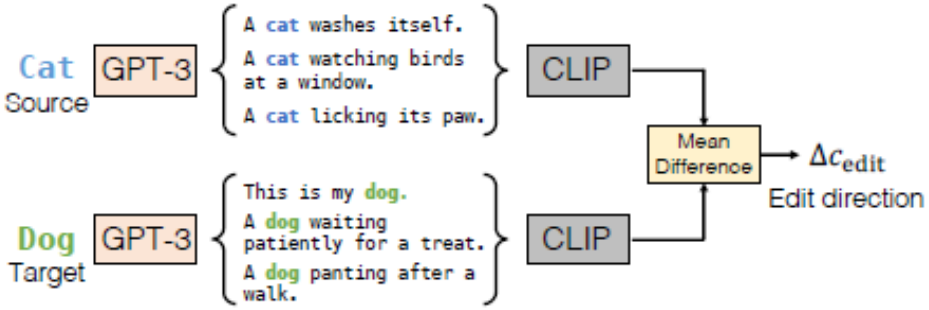
와 같이 구해지며, q 분포가 standard Gaussian 을 따른다고 할 때, $\mu_2 = 0$, $\sigma_2 = 1$ 이면, KL = -log(σ_1) + $\frac{1}{2}\sigma_1^2$ + $\frac{1}{2}\mu_1^2$ - $\frac{1}{2}$ = $\frac{1}{2}$ (variance + μ_1^2 - 1 -log(var)) 이며,

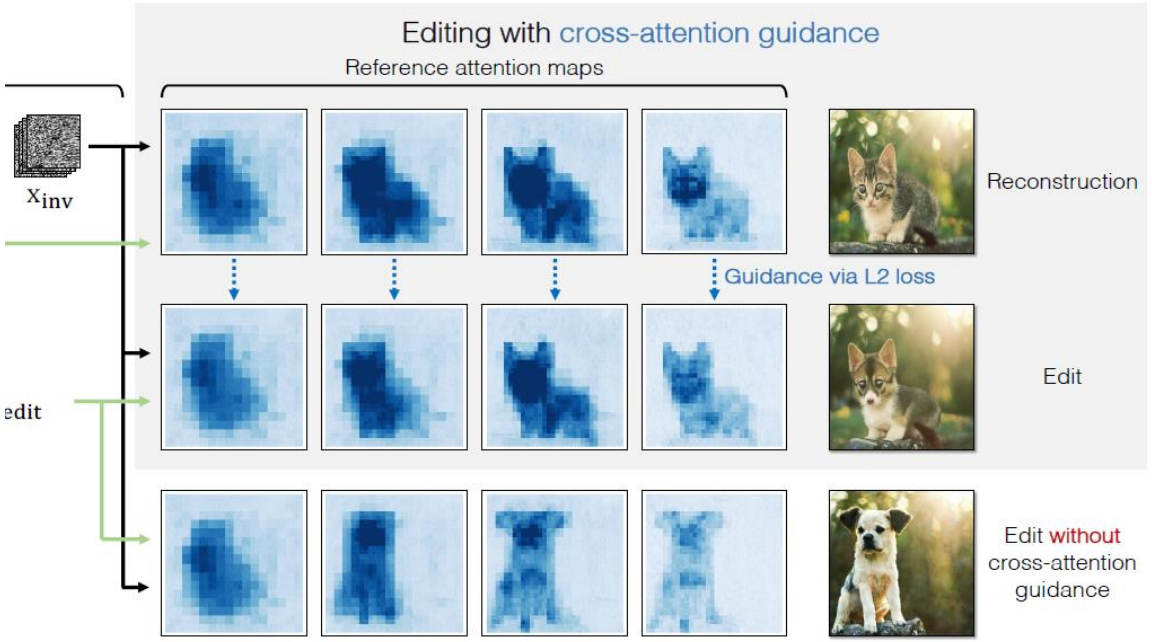
이를 코드적으로 아래와 같이 구현하고 있음

```
def kl_divergence(x):
    _mu = x.mean()
    _var = x.var()
    return _var + _mu ** 2 - 1 - torch.log(_var + 1e-7)
```

2. Method
(2) Discovering Edit Directions

1. source sentences & target sentences 생성
   - GPT-3 와 같이 target 으로 하는 object 가 들어가는 다양한 sentence 를 자동 생성
2. embedding difference between source & target
   - 각 text 의 clip embedding 을 구해 source / target 의 대표적인 embedding 을 구함
   - 이들의 차이 (ΔC_{edit})를 이용하여 edit direction 을 구함
3. 왜 다양한 texts 를 사용?
   - single text 보다 더 robust 함을 실험적으로 확인 (뒤에서 확인)
4. How to control edit
   - $C_{edit} = C + \Delta C_{edit}$ 을 하면, 원래 이미지의 structure 이 무너지는 현상이 있음
   - cross attention guidance 를 사용하여 기존의 이미지를 복원하는 상태와의 consistency 유지





2. Method
(2) Discovering Edit Directions

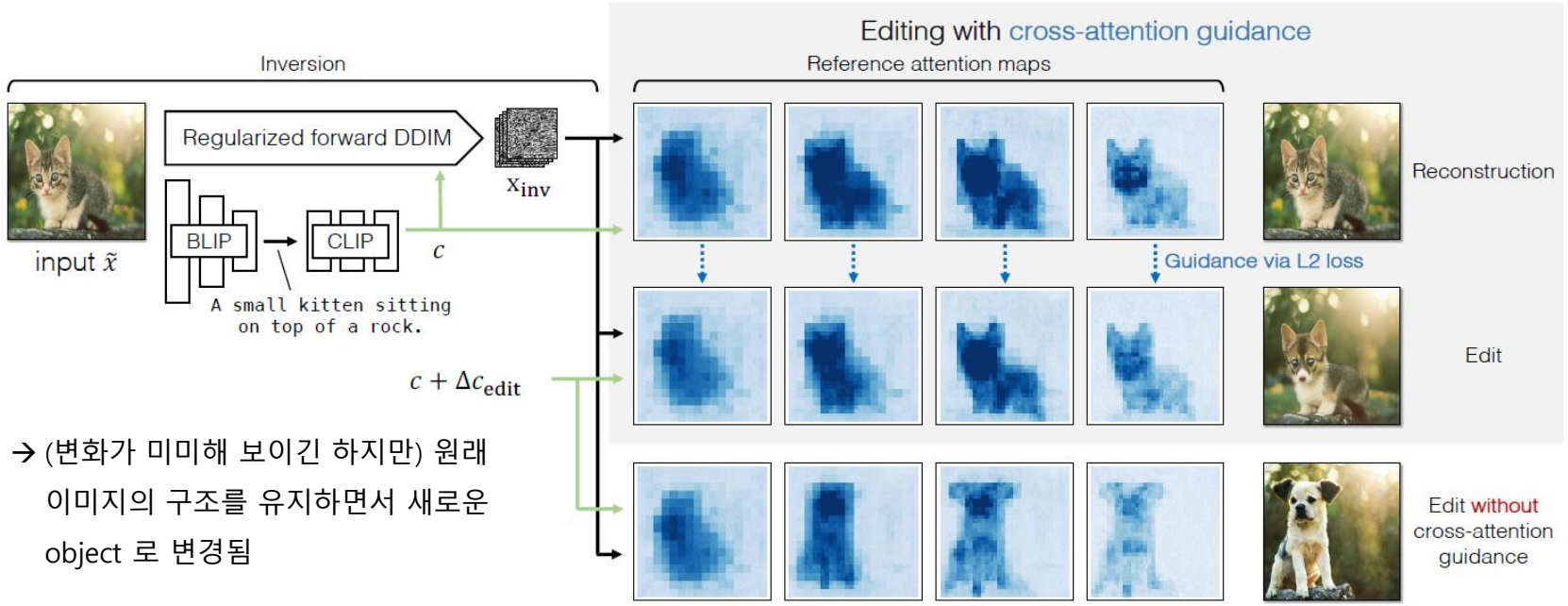
1. cross attention guidance

   (1) reconstruct original image : 매 timestep 에서의 cross attention map (원래 이미지의 structure 을 많이 반영) 구함

   (2) edit direction 을 통해서 edit 이미지에서의 cross attention map을 구함

   → original cross attention map 와 L2 loss 를 통해서 backprop 을 함으로써 원래 이미지의 structure 을 유지

$$\mathcal{L}_{\text{xa}} = ||M_t^{\text{edit}} - M_t^{\text{ref}}||_2$$



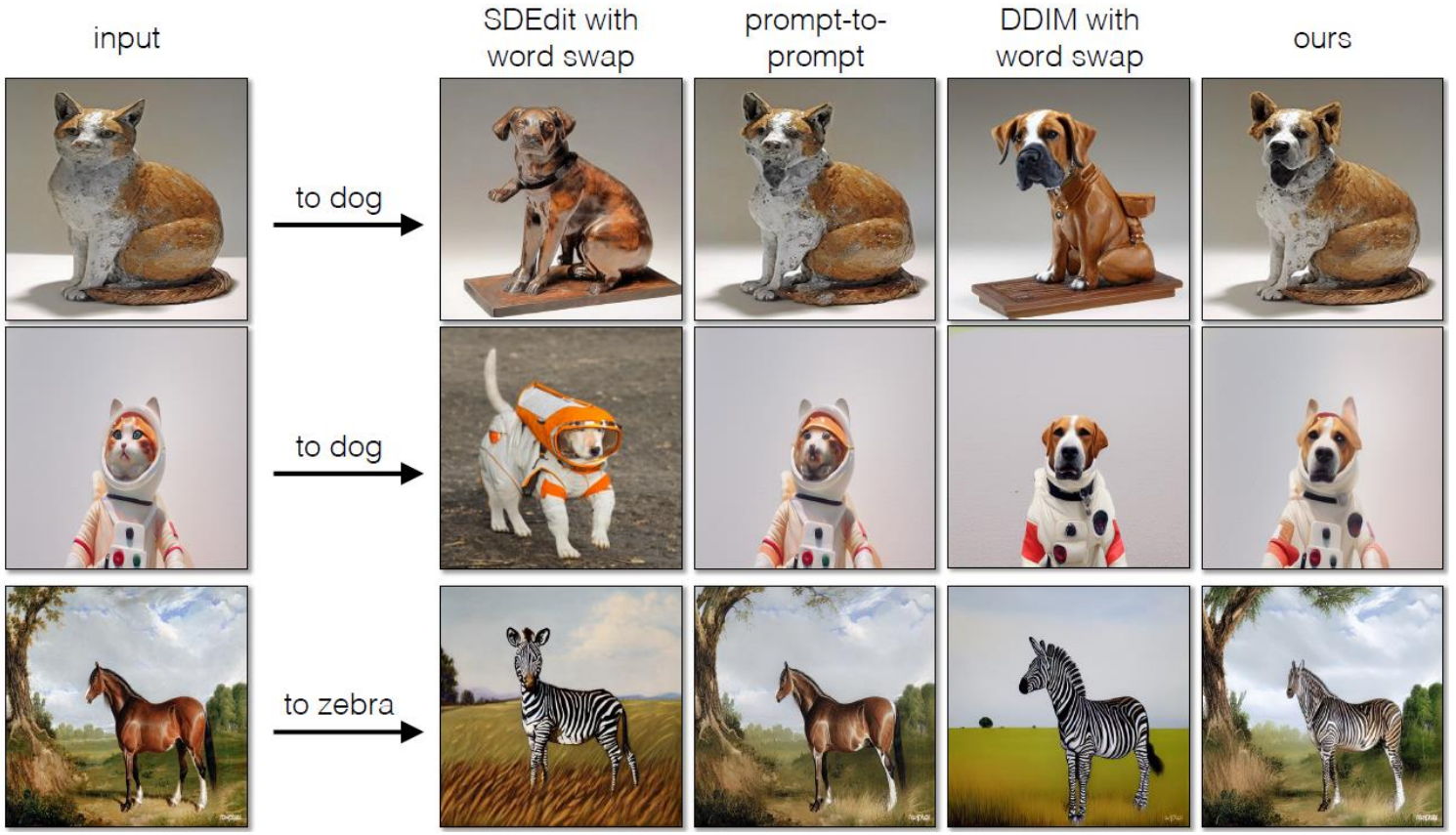
→ (변화가 미미해 보이긴 하지만) 원래 이미지의 구조를 유지하면서 새로운 object 로 변경됨

Experiment

3. Experiments
(2) Discovering Edit Directions

비교 방법론	How to edit image
SDEdit	원래 이미지를 특정 timestep 까지 noising 시키고, denoising 을 통해 이미지를 복원 시 target 으로 하는 새로운 word 를 이용함
Prompt-to-prompt	원래 이미지의 cross attention map 을 유지한다는 점에서 제안 논문과 동일하지만, 수정 방향의 경우 특정 단어에 한정해서 함 (본 제시안은 다양한 문장의 평균 벡터를 이용)
DDIM	원래의 이미지를 이용해 noise 를 구하고, noise를 이용해서 다시 이미지 복원 시 target 으로 하는 text 를 사용

1. SDEdit, DDIM 의 경우 수정되지 말아야 할 부분까지 수정이 됨
   → high BackGround LPIPS, High structure Dist
2. Prompr to prompt 의 경우 수정
   방향에 명확하게 반영되지 못하는 경우가 있음
   → low CLIP Accuracy



Method	cat → dog			horse → zebra			cat → cat w/ glasses		sketch → oil pastel	
	CLIP-Acc (↑)	BG LPIPS (↓)	Structure Dist (↓)	CLIP-Acc (↑)	BG LPIPS (↓)	Structure Dist (↓)	CLIP-Acc (↑)	Structure Dist (↓)	CLIP-Acc (↑)	Structure Dist (↓)
SDEdit [35] + word swap	71.2%	0.327	0.081	92.2%	0.314	0.105	34.0%	0.082	21.2%	0.085
DDIM + word swap	72.0%	0.279	0.087	**94.0%**	0.283	0.123	37.6%	0.085	32.4%	0.082
prompt-to-prompt [22]	66.0%	0.269	0.080	18.4%	0.261	0.095	69.6%	0.081	10.8%	0.079
pix2pix-zero (ours)	**92.4%**	**0.182**	**0.044**	75.2%	**0.194**	**0.066**	**71.2%**	**0.028**	**75.2%**	**0.052**

3. Experiments
(2) Discovering Edit Directions

ablation study 를 통해

1. config A : 일반 DDPM 의 방식으로 했을 때, background 가 다 바뀌어 버림
2. config B,C 와 같이 DDIM 으로 바꿨을 때는 보다 이미지 edit 의 성능이 좋아짐
3. config E 와 같이 cross attention guide 가 있을 때 background 가 유지되면서 수정 방향이 잘 반영되고 있음을 알 수 있다.

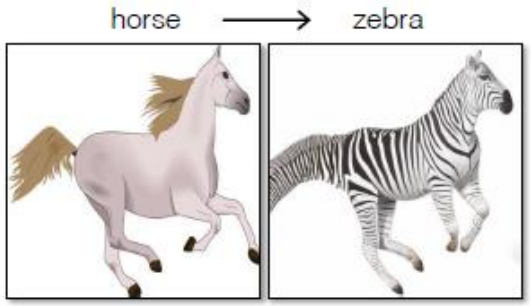
	Inversion	Edit	cat → dog			horse → zebra			cat → cat w/ glasses		sketch → oil pastel	
			CLIP-Acc (↑)	BG LPIPS (↓)	Structure Dist (↓)	CLIP-Acc (↑)	BG LPIPS (↓)	Structure Dist (↓)	CLIP-Acc (↑)	Structure Dist (↓)	CLIP-Acc (↑)	Structure Dist (↓)
A	DDPM	word swap	71.6%	0.392	0.126	93.2%	0.389	0.167	35.2%	0.122	55.2%	0.114
B	DDIM	word swap	72.0%	0.279	0.087	94.0%	0.283	0.123	37.6%	0.085	32.4%	0.082
C	DDIM w/ $\mathcal{L}_{\text{auto}}$	word swap	72.4%	0.283	0.089	94.0%	0.281	0.122	38.0%	0.087	35.2%	0.082
D	DDIM w/ $\mathcal{L}_{\text{auto}}$	sentence directions	**100.0%**	0.274	0.095	**97.6%**	0.290	0.130	20.8%	0.103	**88.4%**	0.087
E (ours)	DDIM w/ $\mathcal{L}_{\text{auto}}$	sentence directions w/ $\mathcal{L}_{\text{xa}}$	92.4%	**0.182**	**0.044**	75.2%	**0.194**	**0.066**	**71.2%**	**0.028**	75.2%	**0.052**

Conclusion

4. Conclusion

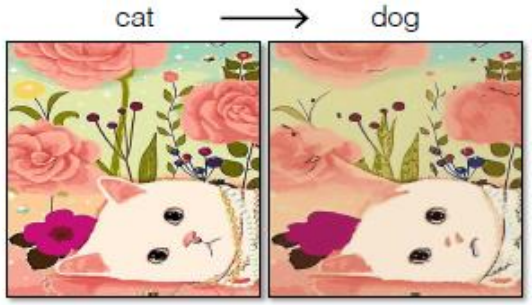
논문에서 밝히고 있는 한계점은,

1) pretrained diffusion model 의 cross attention map 의 크기가 64 resolution 으로 한정돼, detail 표현이 어려움



zebra 로 바뀌었을 때,
꼬리 부분이 잘 반영되지 않음
(+ 앞 다리가 변경됨)

2) object 의 구조가 특이한 경우 반영이 잘 되지 않음



사전 학습 모델의 일반적인 제약이라고 생각됨

THANK YOU