Image-to-Image Translation with Conditional Adversarial Networks

- 논문 : arXiv 버전, CVPR

Abstarct

- image-to-image 변환 문제에서 general-purpose solution 으로 conditional adversarial networks 을 제안
- 이 네트워크는 input image에서 output image로의 mapping을 학습하는 것 뿐 아니라, 이 mapping을 학습하는 loss function을 학습
- 이 방식은, pixel-to-pixel mapping이라는 기본 세팅은 같지만 특정한 알고리즘에 따라 다른 loss function을 적용했던 문제들과 달리 동일하고도 일반적인 적용이 가능

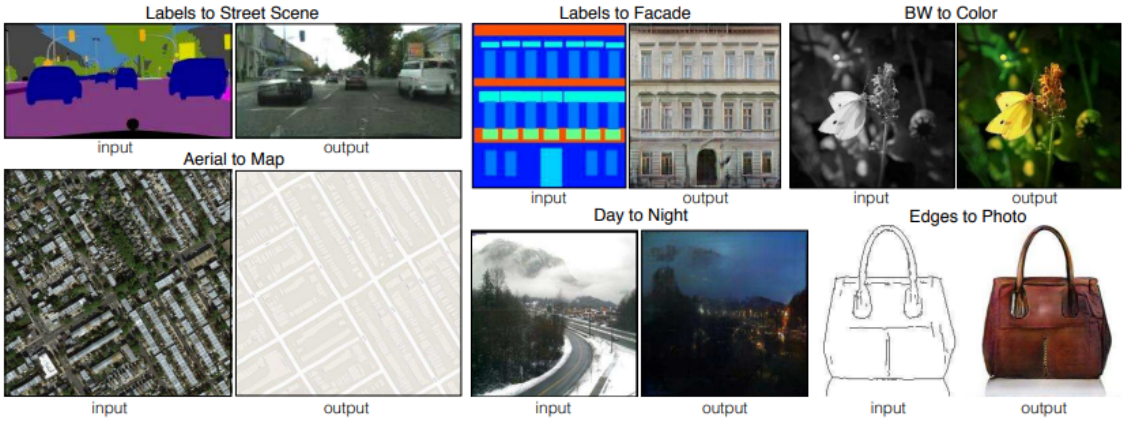


Figure 1: Many problems in image processing, graphics, and vision involve translating an input image into a corresponding output image. These problems are often treated with application-specific algorithms, even though the setting is always the same: map pixels to pixels. Conditional adversarial nets are a general-purpose solution that appears to work well on a wide variety of these problems. Here we show results of the method on several. In each case we use the same architecture and objective, and simply train on different data.

기존의 방법 및 시도

- 기존 image to image translation에서는 application-specific하게 loss를 정의해야 했음
- image to image translation에 GAN을 적용한 기존의 시도 :
  - non-connditional한 GAN이었기 때문에 L2 regression등의 방법을 써서 원하는 output을 얻어야 했음

- 즉, style transfer를 할지, super resolution을 할지에 따라 방법을 달리 조정

이 논문에서는?

- 이 논문에서는 GAN을 이용하여 사용자가 특정 사진과 비슷한 사진을 만들라는 high-level 요구 사항을 주면, discriminator와 generator가 동시에 학습하면서 그럴듯한 결과물을 내어주도록 함
- 각 요구 사항에 따라 방법을 조정하는 기존의 방식과 다르게, 조건부 생성모델인 cGAN을 바탕으로 포괄적인 작업을 할 수 있는 “common framework”인 pix2pix 소개

Conditional GAN

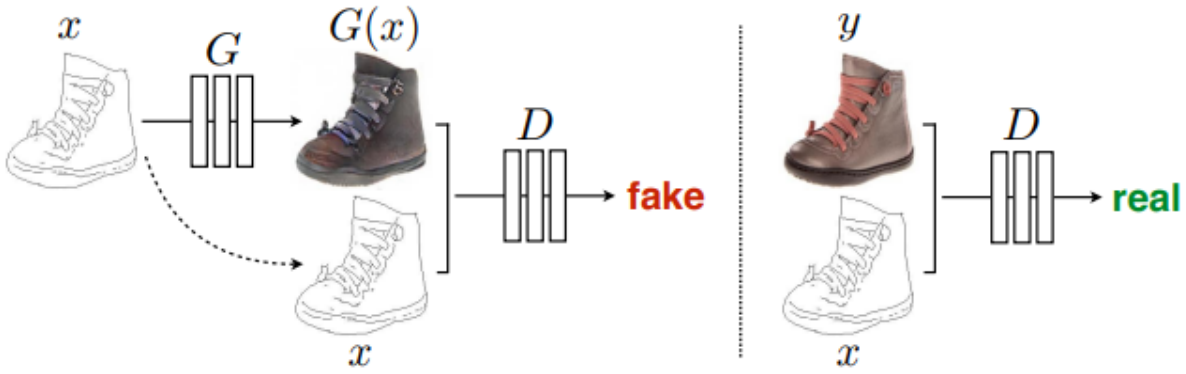


Figure 2: Training a conditional GAN to map edges→photo. The discriminator, D, learns to classify between fake (synthesized by the generator) and real {edge, photo} tuples. The generator, G, learns to fool the discriminator. Unlike an unconditional GAN, both the generator and discriminator observe the input edge map.

- GAN
  - G(generator)와 D(discriminator)의 상호 견제를 통하여, 기존의 Generator 모델과 다르게 확률 계산 없이도 좋은 성능을 보임
  - G(generator)의 input = random noise z
  - GAN을 사용하면 학습에 사용한 이미지와 흡사한 가짜 이미지를 만들 수 있는데, conditional GAN 이전에는 어떤 이미지를 만들어낼지 제어할 수 없었음
  - 그래서, 2014년 몬트리올 “Mehdi Mirza” 팀이 생성 이미지를 조종할 수 있는 “Conditional generative adversarial nets” 이라는 논문 발표 (논문 링크)
- Conditional GAN

- G(generator)의 input = **random noise** z(source image) + **condition** x (ground truth image)
  - **random noise** z : source image, **condition** x : ground truth image
- D(discriminator)의 input = $\{\text{fake}, x\}$ 또는 $\{y, x\}$
  - **fake** : generated image, y : output image
- condition x에 따라 원하는 output을 얻을 수 있음
- 예컨대, 기존의 GAN으로 MNIST를 학습시키면 input z에 대한 output이 0~9 중 어느 숫자로 나올지 사용자가 선택할 수 없었지만, cGAN에서는 이 점을 보완

pix2pix (이 논문에서 도입한 실험)

- pix2pix에서 시도한 것
  - noise vector z 이외에 조건 x를 추가로 input으로 하고 나니, z의 의미가 퇴색되더라!
    - z라는 noise는 output에 stochastic함을 주기 위해 필요했던 것
    - z가 없으면 output이 x에 deterministic해짐
    - 이런 의미에서, noise z는 Gaussian noise일 필요는 없음
- 그래서 이 논문에서 사용한 noise = 위의 결과에 의해, train과 test를 할 때 dropout을 섞어서 noise를 만들었고 이로부터 오는 stochastic함은 크지 않음을 확인
- **Figure 2** : 위와 같은 일련의 과정들을 거쳐, G의 input은 x만 취하고, z는 dropout 형태로 G안에 섞여있게 됨

Loss function of pix2pix

1. conditional GAN의 loss function

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))]$$

   - 기존의 vanilla GAN의 loss에 x 추가
   - generator G와 discriminator D의 minimax 게임
   - G는 최소화, D는 최대화

$$\text{i.e.} \quad G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D)$$

2. 실험적으로 사용한 loss (D의 input에서 x 제거)

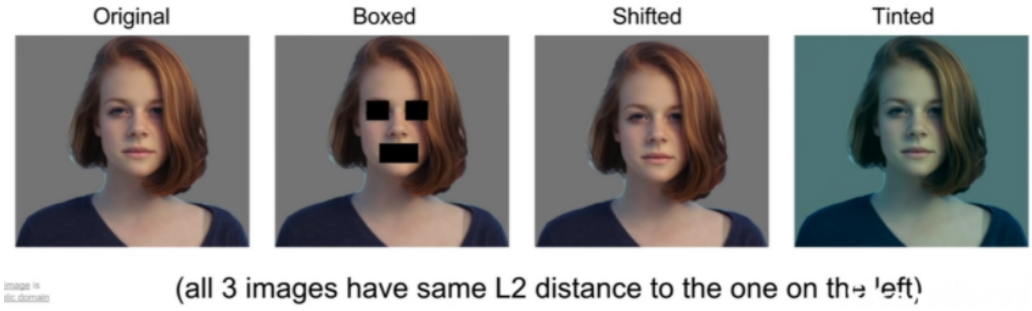
$$\mathcal{L}_{GAN}(G,D) = \mathbb{E}_y[\log D(y)] + \mathbb{E}_{x,z}[\log(1 - D(G(x,z)))]$$

- 발상 : D에게 condition을 보여주지 않으면 어떨지에 대한 의문에서 출발
  - 이 결과 condition에 맞는 output을 주도록 G가 학습하지 않을 거라 생각
  - 만약 성능이 떨어지는 G가 condition으로 1을 받아 9를 만들었다고 할 때, D는 condition이 뭐였는지 모르기 때문에 G의 예측인 9가 보기에 그럴 듯 하다면 True라고 판단하고, G는 condition에 맞지 않는 예측을 했음에도 성능이 개선되지 않을 것임

3. 최종적으로 수정한 pix2pixGAN-loss (L1 distance 사용)

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x,z)\|_1]$$
$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda \mathcal{L}_{L1}(G)$$

- 이전의 접근법에서 GAN의 loss에 L2-distance와 같은 기존의 loss와 결합하는 것에 대한 효과성 발견
- 판별자 D의 역할은 변하지 않음
- G는 판별자를 속이는 것 외에도, L2 sense에서 ground truth와 output이 가깝도록 해야 함
- L1 거리를 사용할 때, 덜 blurring되기 때문에 이를 사용하여 실험
  - L1이나 L2 loss는 마지막에 averaging을 하기 때문에 이미지가 blurry해진다는 단점
  - cs231n 강의를 보면 L2 loss는 아래의 2,3,4번째 그림들을 구분하지 못함

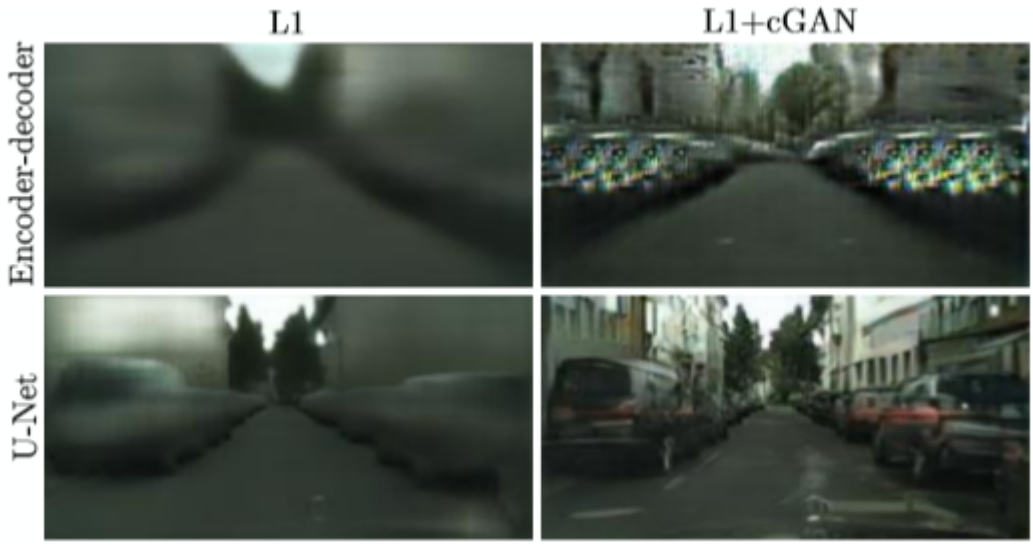


  - 하지만 이러한 L1/L2 loss는 GAN-loss와 상호 보완이 가능하며, GAN의 장점은 보기에 그럴듯한 이미지를 만들어낸다는 것

    - Blurry한 이미지들은 L1/L2 loss는 속일 수 있지만 D는 속이지 못함
    - 단점은 D를 속이기 위해 실제로는 없는 object들을 만들어낸다는 것이고, 논문에서는 여기에 L1 loss를 섞어 그 단점을 상쇄하고자 함

Generator (Unet)

- Image to image translation에서는 input과 output이 모두 high resolution
- 또한, input과 output이 표면적으로는 달라 보여도 같은 concept을 공유하고 있으므로 underlying structure도 같을 것임
- 이러한 가정 위에서 기존에 많이 사용되던 것이 encoder-decoder 구조
- CNN은 층이 깊어질수록 high level의 feature들을 뽑아냄
- 그렇게 뽑아낸 high-level feature는 그 이미지의 concept을 나타냄
- 그걸 바탕으로 upsampling하여 같은 concept이지만 달라보이는 output을 만들겠다는 아이디어라고 이해함
- 문제는 image to image translation에서 low level feature가 필요한 상황도 많다는 것
- 단순 encoder-decoder 구조에서는 이미지가 CNN을 거치면서 그런 low level feature들이 누락됨
    - 예를 들어, 흑백 이미지를 컬러화하고 싶다고 할 때, 우리가 원하는 건 edge 정보들은 그대로 유지한 채 색만 칠해주는 것인데, 그런 정보가 누락되어서 output이 전혀 다른 모양이면 곤란할 것임
- 따라서, U-Net 기반 구조를 사용
    - n개의 layer가 있다고 할 때 i번째 layer와 n-i번째 layer를 이어주는 것
    - 이렇게 하면 low level feature도 뒤쪽의 layer에 잘 전달
- U-Net 구조를 사용하여 low level에서의 디테일이 더 살아나는 예 확인 : L1 loss만 사용했을 때 blurry 하던 이미지가 cGAN을 섞어서 그럴듯해지고, 거기에 U-Net을 섞어서 low level feature들을 이용해 더 정확한 표현을 가능하게 함



Discriminator (patchGAN)

- 본 실험에서는 512*512 이미지를 62*62 패치 크기로 추출

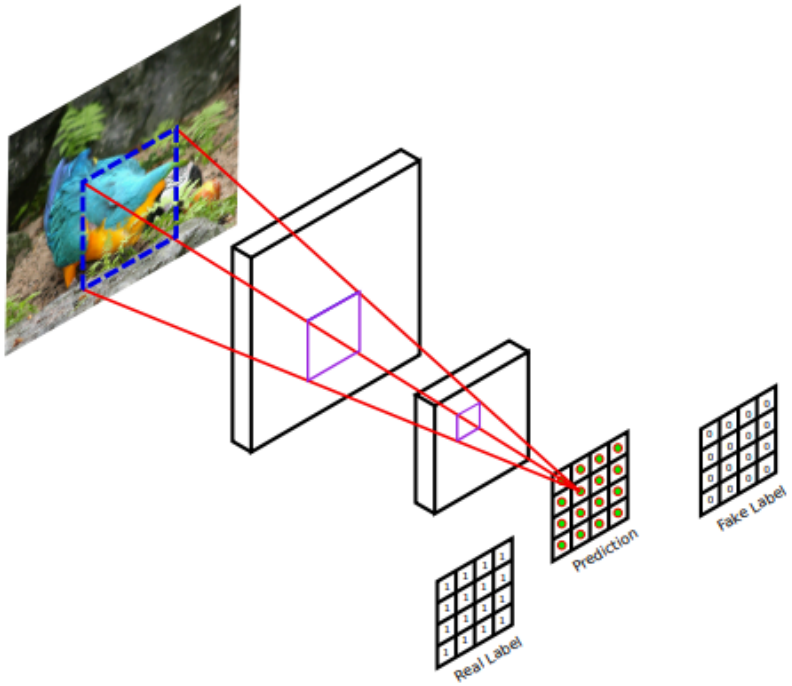


Figure 1: PatchGAN discriminator. Each value of the output matrix represents the probability of whether the corresponding image patch is real or it is artificially generated.

- 즉, 전체 이미지가 아닌 작은 patch에 대해 슬라이딩 방식으로 연산 수행

- 이 방식의 장점은 파라미터 수가 훨씬 적어 연산 속도 빠르고 전체 이미지의 크기에 영향을 받지 않기 때문에 구조적인 관점에서도 유연
- 이 방식은 pix2pix 이후에 발표된 것이며, 이것의 발표 이후에 GAN을 사용하여 영상을 변화하는 여러 논문들에서 대부분 patchGAN이나 이것의 변형을 사용하게 됨