Context Encoders: Feature Learning by Inpainting

컨텍스트 인코더 기술: 인페이팅을 통한 특징 학습

캘리포니아 버클리 대학에서 네 명의 연구원에 의해 작성된 논문으로

2016년 CVPR 학회에서 채택된 논문

1. 이 논문이 무엇을 하고 싶은지?

: 영상에 나오는 상황을 기반으로 픽셀을 예측하는 것을 학습해서 결국 영상의 특징을 찾아내고 자연스러운 이미지를 만드는 알고리즘을 제시하겠다.

예를 들어 위 영상에서 흰 네모 박스 안의 픽셀을 예측하는 것을 학습해서 결국 이 영상이 건물인 것을 깨닫고 영상을 전체적으로 자연스럽게 만드는 것을 말한다.

(영상 인페인팅(image inpainting)은 영상에서 훼손된 부분을 복원하거나 영상 내의 불필요한 문자나 특정 물체를 제거한 후 삭제된 영역을 자연스럽게 채우기 위해 널리 사용되는 기법이다.)

이 알고리즘의 핵심은 '오토 인코더'와 유사한 '컨텍스트 인코더'이다.

'컨텍스트 인코더'가 컨텐츠(흰 부분)을 생성하도록 훈련된다. 그러기 위해 컨텍스트 인코더는 전체 이미지도 이해하고 사라진 부분을 그럴듯하게 예측해야만 한다.

(오토 인코더: 입력과 동일한 출력을 만드는 것을 목적으로 하는 신경망, 이미지를 더 낮은 차원으로 인코딩하고 다시 원래의 이미지로 디코딩함)

오토 인코더와 마찬가지로 컨텍스트 인코더는 인코더와 디코더로 구성된다.

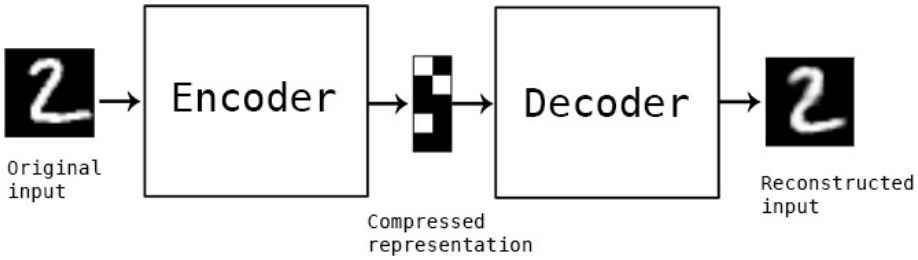
인코더는 영상의 컨텍스트(상황)을 소형의 잠재 표현으로 재구성한다.

디코더는 누락된 영상의 컨텐츠를 생성한다.

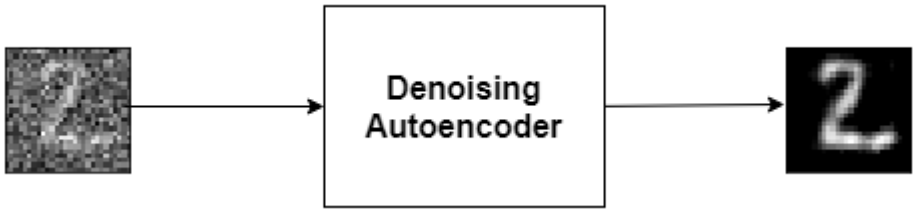
2. 이 논문이 기존의 논문과 어떻게 다른지?

이렇게 누락된 영상을 채우는 것을 '이미지 인페인팅'이라고 하는데 오토인코더를 응용해서도 구현할 수 있다.

오토인코더도 이미지를 입력되면 작은 차원인 '병목'레이어를 통과시키는 방식으로 이미지를 재구성한다. 하지만 오토인코더는 영상의 의미론적인 특징을 파악하지 못하고 단순히 영상을 압축시킬 수도 있다.



그래서 나온 것이 Donising Autoenoder라고 해서 입력할 이미지에 noise를 주고 네트워크 내에서 그 noise를 해결하는(원상태로 만드는) 방식으로 특징을 파악하는 시스템이 있다.



하지만 이미지에 noise를 주는 과정이 low-level에서 진행되서 번거롭고 원상태로 복구할 때도 생각보다 의미있는 정보를 요구하지 않아 특징을 잘 찾는다고 말하기 힘들다는 단점이 있다.

대조적으로 이 논문에 기재된 CE(Context Encoder)는 특징을 잘 파악해서 아주 힘든 일도 할 수 있다고 한다. 예를 들어 주변 픽셀로부터 힌트를 얻기 힘든 영상임에도 불구하고 누락된 영역을 채울 수 있다는 것이다.

오토인코더에서 나타나는 문제는 **reconstruction loss와 adversarial loss를 최소화함**으로써 제거하였다. L2라고 불리는 reconstruction loss는 누락된 영역의 전체 구조를 주변 상황과 관련해서 포착하는데 쓰이고 adversarial loss는 누락된 부분을 채우고 일관성을 유지시키는 여러 방법 중 특정 기법(모드)를 선택하는 것과 관련이 있다.

단순히 reconstruction loss만 사용하면 (c) 이미지처럼 흐릿한 결과를 반환하지만 두 loss를 모두 최소화하면 (d)와 같이 더 깔끔한 결과를 반환한다.

(c) Context Encoder ($L2$ loss)

(d) Context Encoder ($L2$ + Adversarial loss)

평가도 인코더와 디코더를 독립적으로 실시한다.

인코더는 이미지 패치(잘게 자른 이미지로 생각됨)의 컨텍스트(상황)를 인코딩하면 resulting feature(결과로 얻은 특징으로 생각됨)을 얻는데 이 feature와 인접한 컨텍스트(상황)을 검색하면 원래 패치와 의미적으로 유사한 패치가 생성된다고 한다. 이러한 작업이 곧 이미지 이해이기 때문에 이 이해도에 대해서 평가한다.

디코더는 CE가 누락된 영역을 채울 수 있다는 것을 보여준다. 큰 누락 영역에 대해서 합리적인 결과를 제공할 수 있는지 평가한다.

Related work

1. 기존의 방식들에 대한 개략적인 흐름.

- 비지도 학습

: 비지도 학습이란, 정답이 주어지지 않는 학습 전략이다. 어떤 입력에 대한 올바른 결과가 무인지 알 수 없다는 특징을 가지고 대표적인 예로는 군집화와 스케일링이 있다. 그리고 심층 비지도

학습의 연구 중 오토인코더가 있다. 그 중 노이즈를 제거할 수 있는 오토 인코더는

이미지 재구성 능력이 뛰어나다. 컨텍스트 인코더도 노이즈를 제거할 수 있는 오토 인코더라고 생각할 수 있지만 모델의 입력에 적용된 손상이 훨씬 커서 회복되기에 더 많은 정보가 필요하다고 합니다.

- 약한 지도학습과 자기 지도학습

: 지도 학습(supervised learning)이란, 정답이 주어지는 학습 전략이다. 어떤 인풋(input)에 대한 올바른 아웃풋(output)이 무엇인지 알 수 있다는 전제를 가진다. 따라서 지도 학습을 위해서는, 데이터 셋과 데이터 각각에 대한 정답을 제공받아야 한다.

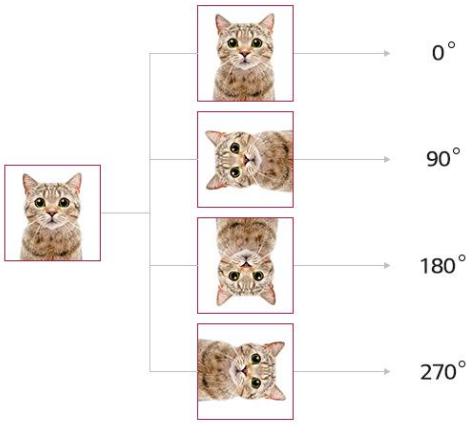
그러나, 약한 지도 학습(weakly supervised learning) 환경에서는, 주어지는 정답에 대한 정보가 제한된다. 비 지도 학습(unsupervised learning)과 같이 아무 정보도 주어지지 않는 경우와는 다르지만, 일부에 대한 정보만 제공받아 학습하고, 그러한 학습을 통해 제공받지 않은 정보를 예측해내야 한다. 영상 인식에서 객체에 대한 클래스 정보만을 제공받았지만, 영상 내의 객체 위치를 예측해내는 학습 모델을 예로 들 수 있다.
그리고 컴퓨터 비젼 분야에서 필요한 라벨링 작업은 주로 인적 자원을 이용하여 이루어지므로 많은 시간적, 경제적 비용이 소모되는데 CNN 기반 약한 지도 학습을 이용하면 비용 면에서 효과적이라고 한다.

'자기 지도 학습'은 비지도학습의 연구 주제 중 하나이다. 비지도학습이기 때문에 레이블이 존재하지 않는 데이터만 이용한다. 이때 사용되는 데이터는 image 가 될 수도 있고, text, speech, video 등 다양한 종류의 데이터가 될 수 있다.
그래서 결국 라벨이 없는 데이터를 이용해서 스스로 학습 후 분류하는 것을 의미한다.

이것이 가능하려면 '사전 학습'이 필요하다. 이 사전 학습에 대한 과제를 pretext태스트라고 한다.

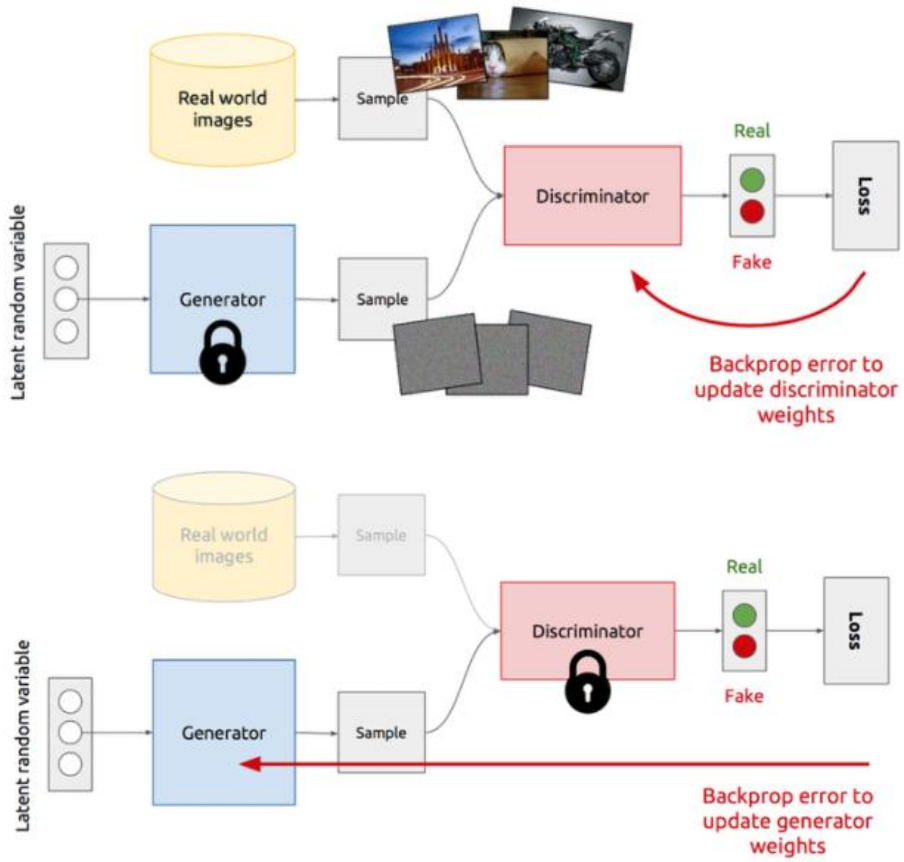
예를 들어 회전한 영상을 입력으로 주고 회전한 각도를 맞추게 하는 사전 학습 과제가 있다. 이 과정에서 기계는 상의 고품질 성 벡터를 추출하게 되고 이 추출한 것이 추후 분류에도 유리하게 작용할 것이라는 아이디어이다.

이러한 학습 또한 비전에서 라벨링하는 작업을 줄이기 때문에 비용 면에서 효과적이라고 한다.



- 이미지 생성

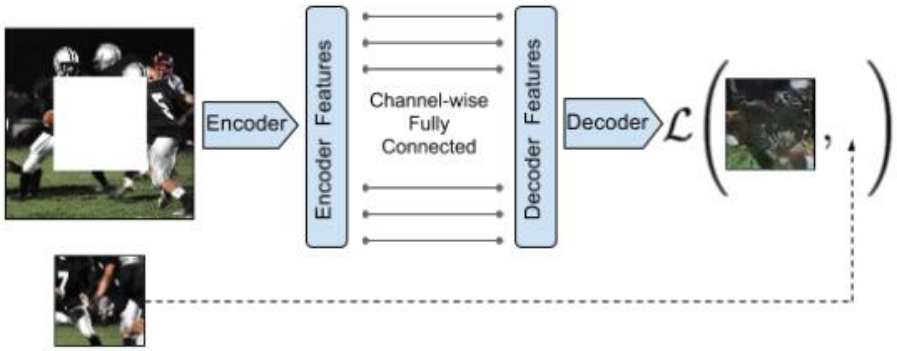
GAN(적대적 생성 모델)을 말한다. GAN은 생성자 신경망과 판별자 신경망이 서로 적대적으로 경쟁하면서, 훈련을 통하여 자신의 작업을 점점 정교하게 수행한다. 이 때 생성자는 가짜 이미지를 실제 이미지처럼 만드는 방법을 학습한다. (처음에는 랜덤 노이즈가 입력으로 들어간다) 그리고 판별자는 진짜 이미지와 가짜 이미지를 더 잘 구별하도록 모델을 훈련한다. 결국 생성자는 판별자를 속일 수 있을 만큼 교활해지며 판별자도 진짜 이미지에서 보다 복잡하고 미묘한 특징을 학습하여 더 판별을 잘하게 된다. 위 작업을 두 개의 적대적 모델이 교착 상태에 도달할 때까지 진행하는 것이다. 참고로 학습은 같이 되지 않기에 생성자 훈련 중에는 생성자 신경망만 판별자 훈련 중에는 판별자 신경망만 학습된다. 학습을 위한 역전파는 판별자가 추론한 예측값(y_pred)를 기준으로 진행되고, 진짜냐 가짜냐 둘 중 하나를 분류하는 이진 분류기이기 때문에 이진 교차 엔트로피의 손실값을 사용한다.



- 인페인팅 기술과 구멍 채우기

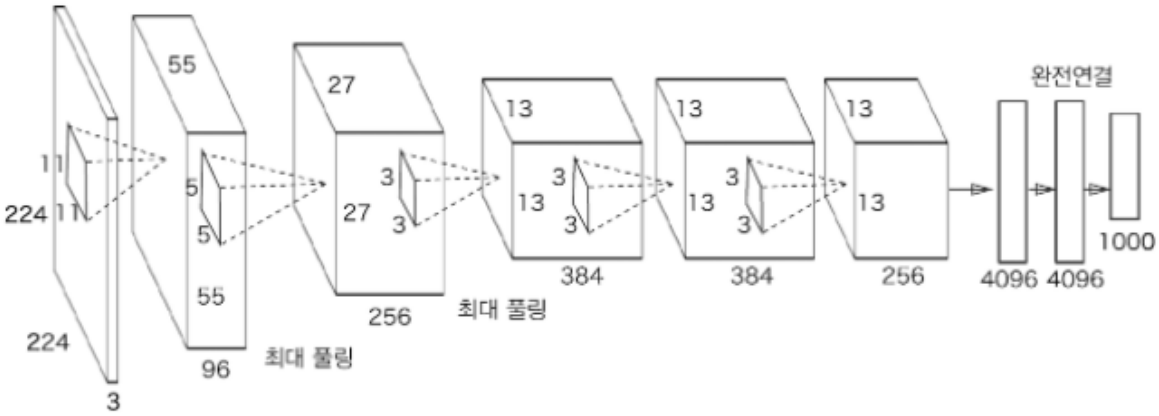
: 컴퓨터 그래픽에서 큰 구멍을 채우는 것은 일반적으로 수백만 개의 이미지 데이터 집합에서 가장 가까운 이웃을 사용하여 컷-페이스트 작업을 이용하는 장면 완성을 통해 이루어진다. 하지만 기존의 방식으로는 구멍이 완전하지 않다고 하기 때문에 위에서 언급했듯 reconstruction loss와 adversarial loss를 최소화하는 방법으로 해결한다.

3. Context encoders for image generation



CE의 구조는 간단한 Encoder-Decoder pipeline이다. Encoder는 누락된 영상으로부터 잠재 표현을 생성하고 Decoder는 누락된 이미지의 컨텐츠를 생성한다. 이 둘은 채널 별로 Fully Connected Layer로 돼있고 이렇게 하였을 때 디코더가 전체 이미지 내용을 추론할 수 있다. (인코더가 추론한 것을 가져올 수 있겠다고 이해)

Encoder는 없어진 부분을 갖고 있는 이미지를 latent vector(6x6x256)로 만든다. 구조는 AlexNet에서 파생되었다.



1. Alexnet구조로 5개의 conv + pooling layers로 이루어진다.

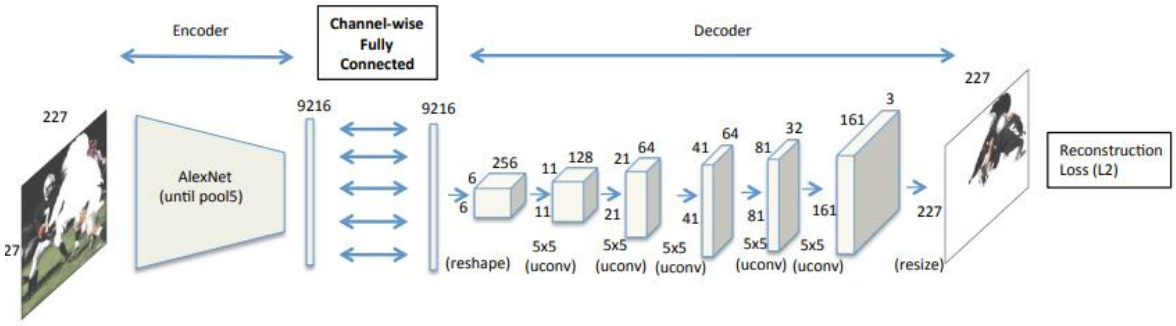
기존 Alexnet이 1000개의 카테고리로 분류하는 것을 학습했다면 여기서는 scatch로 표현될 수 있는 초기 random weights들로 context를 예측하는 것을 학습했다.

Convolution layers가 서로 fully conntected 되어 있으므로 가중치를 모두 공유하는 형태가 된다.

conv+pool연산을 통해 나온 특징맵(feature map)은 말 그대로 특징만 추출했고 모든 위치를 직접

연결하지는 않기에 정보가 특징맵의 한 모서리에서 다른 모서리로 전파할 수 있는 방법이 없다. 따라서 이 모든 정보 전파를 fully connected로 처리한다.

일반적인 방식으로 Encoder와 Decoder를 완전히 연결하면 parameter의 수가 훈련이 어려울 정도로 증가할 수 있다. 이에 채**널 방식의 Fully Conneted layer를 구성**하여 Decoder에 연결한다.



(b) Context encoder trained with reconstruction loss for feature learning by filling in *arbitrary region dropouts* in the input.

채널 방식의 Fully conneted layer는 입력 레이어에 크기가 **n*n인 feature map이 m개 있으면 n*n 차원의 m개 feature map을 출력**하여 Decoder에 전달한다.

이 때, 각 feature map 마다 활성함수를 적용하며 전달하게 된다.

fully connected Layer와의 차이점은 파라미터 개수가 mn^4이다. 그리고 특징맵을 연결하는 파라미터가 존재하지 않고 특징맵 내에서만 정보를 전파한다.

Decoder는 전달받은 특징맵을 사용하여 이미지의 픽셀을 생성한다. 디코더는 RELU활성화를 거친 5개의 up-convolution layer로 이루어져 있다. up-convolution layer는 고해상도 이미지를 생성하는 역할을 한다.

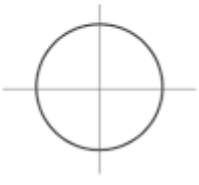
3.2. Loss function

L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$



[L1: 두 개의 벡터를 빼고, 절대값을 취한 뒤, 합한 것. 예를 들어, x=(1,2,3), y=(-1,2,4)라면 d(x,y)=|1-(-1)|+|2-2|+|3-4|=2+0+1=3

L2: 두 개의 벡터의 각 원소를 빼고, 제곱을 하고, 합치고, 루트를 씌운 것. 예를 들어, x=(1,2,3), y=(-1,2,4)라면 d(x,y)=root(4+0+1)=root(5)]

. L2 distance가 '두 개의 벡터의 각 원소를 빼고, 제곱을 하고, 합치고, 루트를 씌운 것'이라면 L2 loss는 두 개의 벡터가 들어가던 자리에 실제값과 예측값이 들어가고 루트를 취하지 않는다는 차이가 있다.

단 L2 loss는 루트를 취하지 않기 때문에 이상치가 들어오면 오차가 제곱이 되어 이상치에 영향을 더 많이 받는다. l1이 상대적으로 이상치에 영향이 없다. robust하다는 특징을 지닌다.

reconstruction loss는 누락된 영역의 전체 구조를 주변 상황과 관련해서 포착하는데 쓰이고 adversarial loss는 누락된 부분을 채우고 일관성을 유지시키는 여러 방법 중 특정 기법(모드)를 선택하는 것과 관련이 있다.

이미지 x에 대해 컨텍스트 인코더 F는 출력 F(x)를 생성합니다.

M^를 픽셀이 떨어진 곳이면 어디든 1이고 입력 픽셀이면 0인 삭제된 영상 영역에 해당하는 이진 마스크로 한다.

훈련 중에 이러한 마스크는 각 영상 및 훈련 반복에 대해 자동으로 생성된다.

$$L2LossFunction = \sum_{i=1}^{n}(y_{true} - y_{predicted})^2$$

$$\mathcal{L}_{rec}(x) = \|\hat{M} \odot (x - F((1 - \hat{M}) \odot x))\|_2^2,$$

reconstruction loss에는 L2 loss를 사용한다

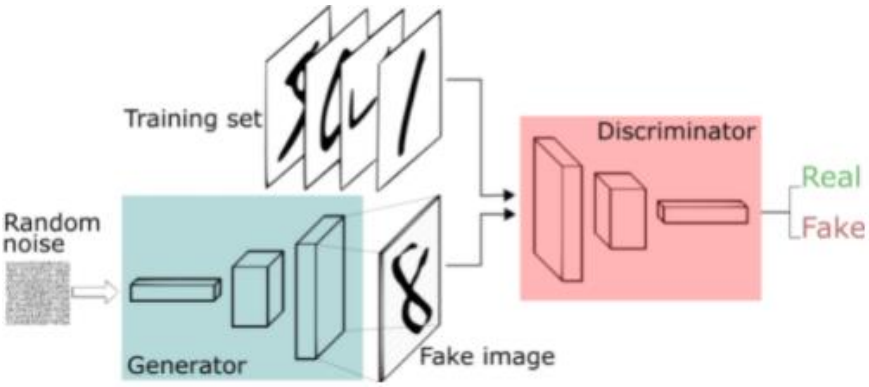
CE에서 정리한 수식은 위와 같다. 위에서 x는 이미지, M캐럿은 바이너리 마스크(픽셀이 떨어진 곳은 1을 두고 아니면 0으로 채운 영상), 그리고 중심에 점이 찍힌 원 기호는 하다마드 곱으로 같은 크기의 두 행렬의 각 성분을 곱하는 연산이다.

Paper에서는 L1과 L2 loss 모두 실험한 결과 유의한 차이는 못 느꼈다고 한다. 그리고 이러한 단순한 loss를 이용하면 고주파 세부 정보를 캡처하지 못한다. L2 loss를 사용하면 평균 픽셀 단위 오차는 최소화하는 장점이 있지만 평균 이미지는 흐릿하게 만드는 경향이 있다고 한다.

(c) Context Encoder
($L2$ loss)

이러한 단점은 Adversarial loss를 추가함으로써 완화시켰다.

Adversarial loss는 GAN을 기반으로 한다.



$$\min_{G}\max_{D} V(D,G) = \mathbb{E}_{x\sim Pdata(x)}[logD(x)] + \mathbb{E}_{z\sim Pz(z)}[\log(1 - D(G(z)))]$$

위는 GAN의 기본 수식(loss function)이다.

G는 Generator로 가짜 이미지를 진짜 이미지처럼 만들기 위한 비지도학습을 한다.

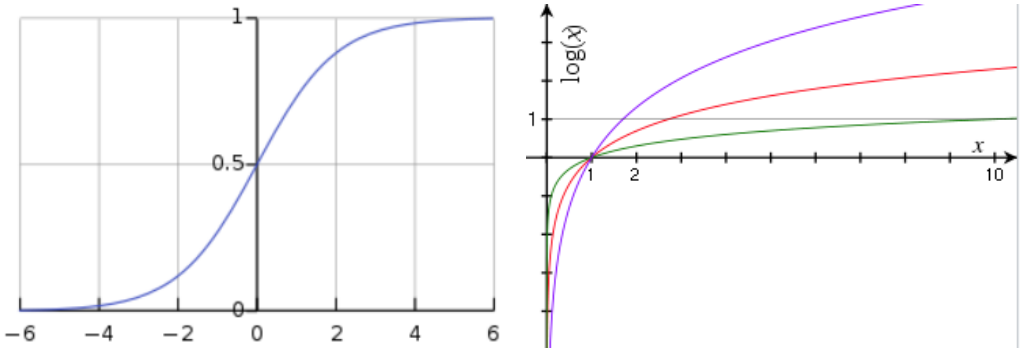
D는 Discriminator로 가짜 이미지와 진짜 이미지를 잘 분류하기 위한 지도학습을 한다.

z는 noise로 G의 첫 입력으로 들어간다. 따라서 만들어진 가짜 이미지는 G(z)로 불린다.

x는 진짜 이미지이며 Discrimitator의 입력으로 들어간다. 따라서 D(x)=1이 된다.

G(z)도 Discriminator의 입력으로 들어가는데 초기 D(G(z))=0이 된다.

Discriminator가 0과 1의 값으로 분류를 할 때는 sigmoid함수를 써서 0.5를 기준으로 구분한다.



D가 원하는 것은 D(x)가 항상 1이 되고 D(G(z))가 항상 0이 되도록 하는 것이다. 이 바람대로 된다는 것은 logD(x)값과 log(1-D(G(z)))값이 0이 되서 V(D,G)가 0, 즉 max값이 된다는 의미와 같다.

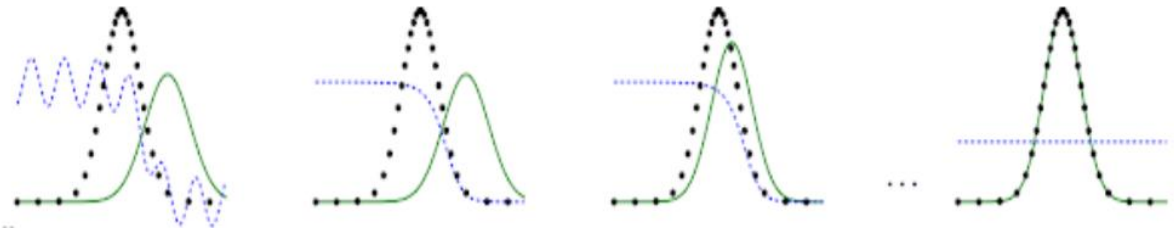
G는 D가 잘 분류하는지는 관심이 없다. 즉 D(x)에 대해서는 관심이 없다. G는 가짜 이미지를 진짜 이미지로 만들기 위해 D(G(z))가 1이 되도록 할 뿐이다.

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim Pdata(x)}[logD(x)] + \mathbb{E}_{z\sim Pz(z)}[\log(1 - D(G(z)))]$$

즉 위 수식에서 오른쪽 파트만 관심이 있다는 의미이고,

log(1-D(G(Z)))가 0이 되기만 바란다는 것이다. 이 바람대로 된다는 것은 V(D,G)가 –inf, 즉 min값이 된다는 의미와 같다.

정리하면 D의 목적은 GAN의 loss함수를 최대가 되도록 하는 것이고 G의 목적은 GAN의 loss를 최소가 되도록 하는 것이다.



그림으로 부가 설명을 하겠다.

검은 점선은 실제 데이터 분포, 파란 점선은 discriminator distribution, 초록선은 generative discribution이다. 처음 discriminator가 학습을 하면 왼쪽에서 두 번째 그림처럼 좀 더 부드러운 잘 구별하는 분포가 만들어진다. (실제 데이터의 분포에서 값이 0이랑 가까울수록, 가짜 데이터의 분포 값이 0이상일수록 '이건 가짜(0)야'라고 말하는 discriminator 분포가 그려진다)

이후 G가 현재 discriminator가 구별하기 어려운 방향으로 학습을 하면 세 번째 그림처럼 되고 결국 discriminator가 둘을 전혀 구별하지 못하는 D(x)=1/2인 상태가 된다.

Paper에서는 이러한 GAN을 context예측을 위해 사용하려고 컨텍스트 정보를 조건화할 수도 있었다고 한다. 하지만 이렇게 GAN을 사용하였을 때 예측 작업에 대해 쉽게 훈련되지 않았다고 한다.

그 이유는 예측 샘플 대 실제 샘플을 쉽게 분류하기 위해, 적대적 판별자 D는 생성된 영역의 지각 불연속성과 원래 컨텍스트를 쉽게 이용하기 때문이라고 한다. (?)

(However, conditional GANs don't train easily for context prediction task as the adversarial discriminator D easily exploits the perceptual discontinuity in generated regions and the original context to easily classify predicted versus real samples.)

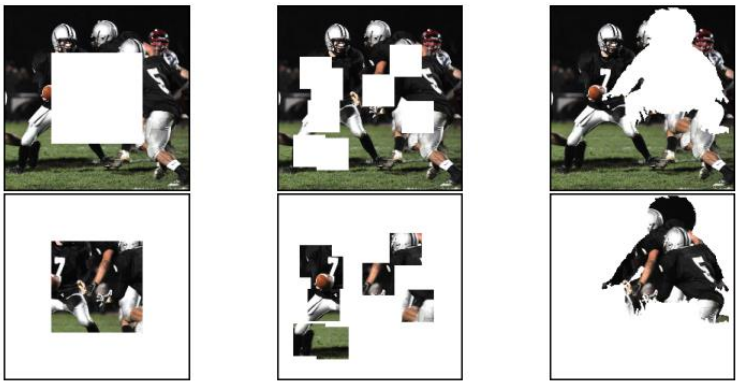
그래도 다행히 Generator가 노이즈 벡터에 의해 조절되지 않았을 때 더 나은 결과를 보였다는 것을 찾게 됐다고 한다. 따라서 최종 Adversarial loss는 아래와 같다.

$$\mathcal{L}_{adv} = \max_{D} \quad \mathbb{E}_{x \in \mathcal{X}}[\log(D(x)) + \log(1 - D(F((1 - \hat{M}) \odot x)))],$$

마지막으로 loss를 joint하고 수식을 나타내면 아래와 같다고 한다.

$$\mathcal{L} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{adv}\mathcal{L}_{adv}.$$

입력 영상의 제거된 영역은 다음 중 하나일 수 있다. paper는 아래의 세 가지에 대한 전략을 제시한다.



(a) Central region (b) Random block (c) Random region

Central Region

- 가장 간단한 모양은 '중앙 사각형 패치'
- 인페인팅에는 잘 작동한다.
- central mask 의 경계에 고정된 저수준의 특징을 학습한다.

  ➔ 제거된 영역에 대응되는 저수준의 이미지 특징을 찾는다.
- 이러한 낮은 수준의 feature 는 마스크가 없는 이미지에는 잘 일반화되지 않는 경향이 있다.

Random Block

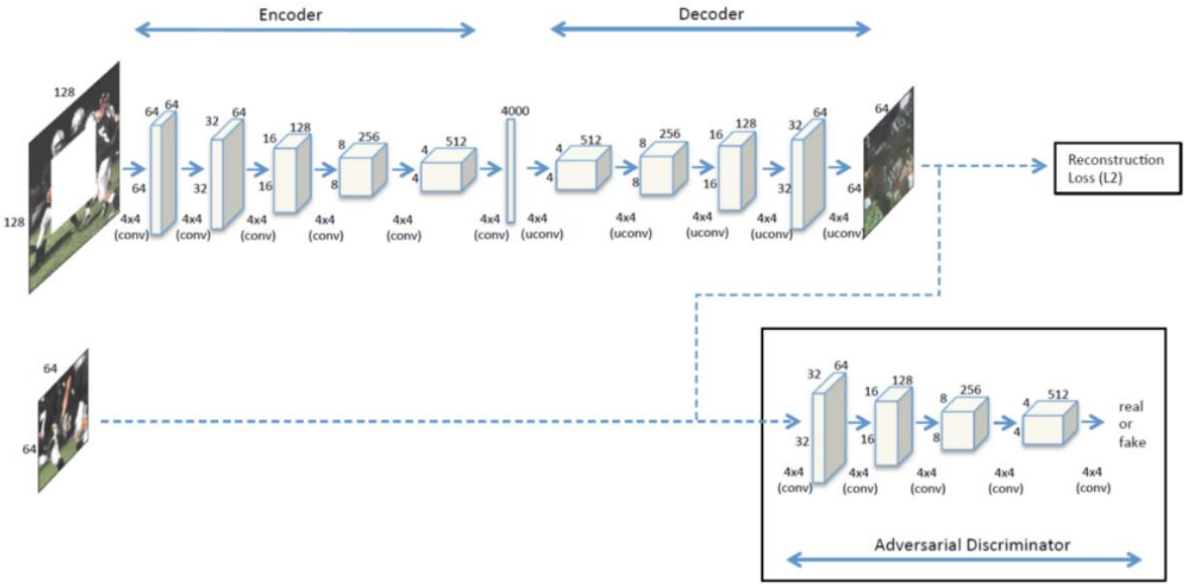
- 영상의 최대 1/4 을 차지하는 여러 겹치는 사각 마스크가 설정됨
- random block masking 은 sharp 한 경계의 특징에 의해서 저수준의 특징밖에 찾지 못한다.

Random Region

- 이미지로부터 임의의 모양이 제거된 형태
- central region 과 random block 이 유사한 일반적인 특징을 찾아내는 것보다 훨씬 좋은 특징을 잘 찾는다.
- 따라서 **random region dropout** 은 특징을 찾는 용도로 많이 쓰인다.

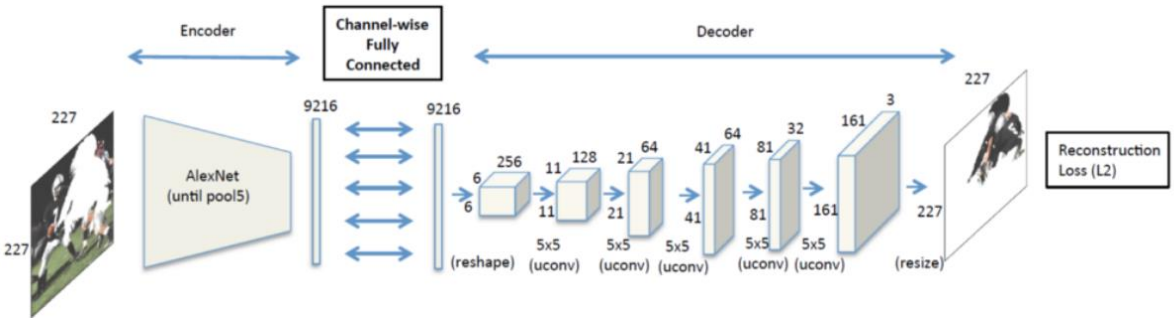
Two CNN Architectures

1. cnn for inpainting
2. cnn for feature learning



- Context encoder trained with **joint reconstruction and adversarial loss** for semantic **inpainting**, is as shown above.

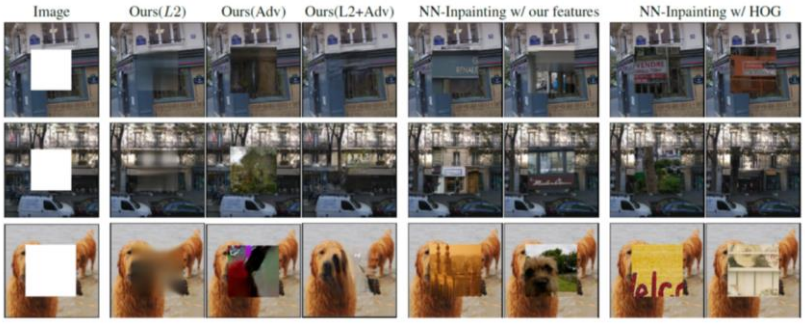
- **Center region dropout** is used.



- Context encoder trained with **reconstruction loss** for **feature learning**.

- **Arbitrary region dropout** is used.

Inpainting Results



- Nearest neighbor inpainting (NN) is compared.
  - The proposed reconstructions are well-aligned semantically.

Feature Learning Results

Pretraining Method	Supervision	Pretraining time	Classification	Detection	Segmentation
ImageNet [26]	1000 class labels	3 days	78.2%	56.8%	48.0%
Random Gaussian	initialization	< 1 minute	53.3%	43.4%	19.8%
Autoencoder	-	14 hours	53.8%	41.9%	25.2%
Agrawal *et al.* [1]	egomotion	10 hours	52.9%	41.8%	-
Wang *et al.* [39]	motion	1 week	58.7%	47.4%	-
Doersch *et al.* [7]	relative context	4 weeks	55.3%	46.6%	-
Ours	context	14 hours	56.5%	44.5%	30.0%

For semantic segmentation, ***using proposed context encoders for pretraining (30.0%) outperform a randomly initialized network (19.8%)*** *as well as a plain autoencoder (25.2%) which is trained simply to reconstruct its full input.*

Context encoders are competitive with concurrent self-supervised feature learning methods (Classification)