GAN

안남혁

Generative Adversarial Network

What are some recent and potentially upcoming breakthroughs in deep learning?

Answer | Request | Follow 131 | Comment | Share 11 | Downvote | ...

2 Answers

Yann LeCun, Director of AI Research at Facebook and Professor at NYU

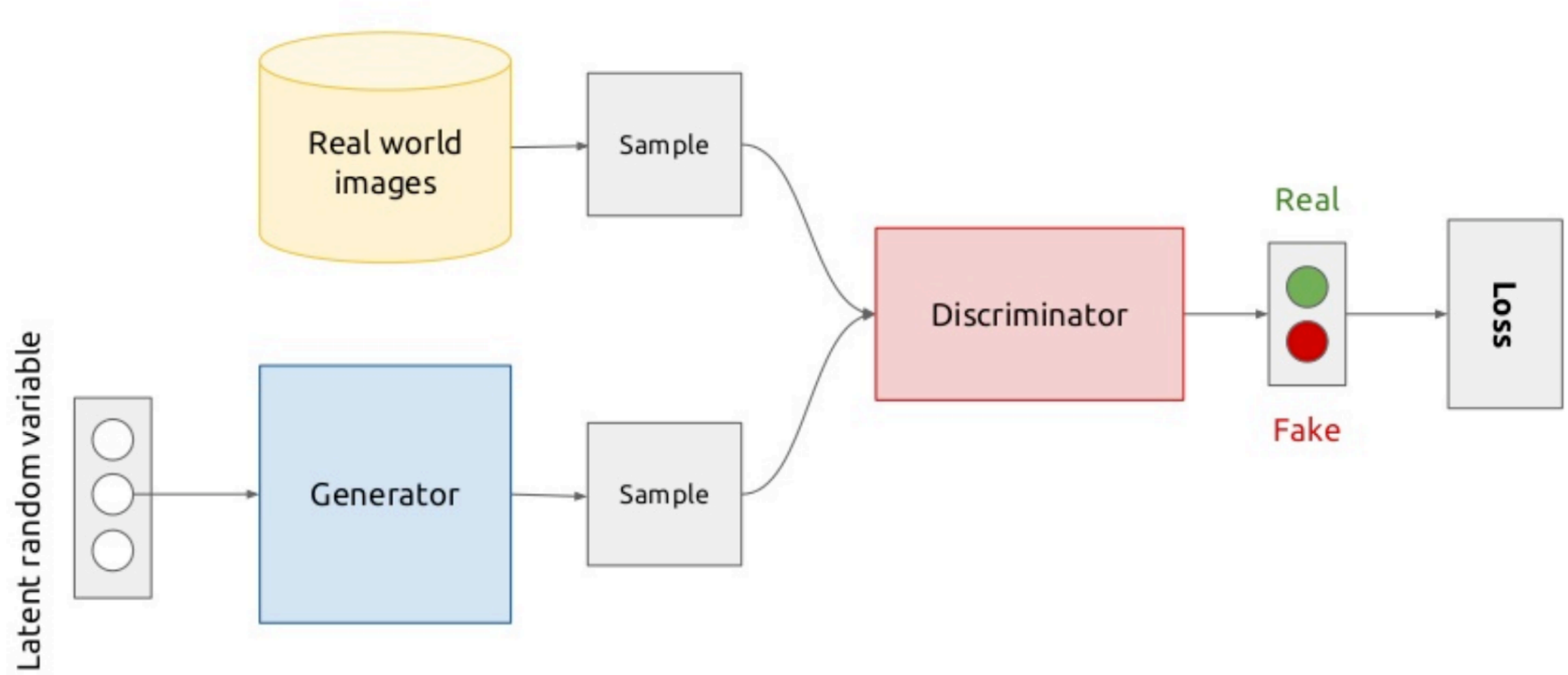
Written Jul 29 · Upvoted by Joaquin Quiñonero Candela, studied at Machine Learning and Nikhil Garg, I lead a team of Quora engineers working on ML/NLP problems

There are many interesting recent development in deep learning, probably too many for me to describe them all here. But there are a few ideas that caught my attention enough for me to get personally involved in research projects.

The most important one, in my opinion, is adversarial training (also called GAN for Generative Adversarial Networks). This is an idea that was originally proposed by Ian Goodfellow when he was a student with Yoshua Bengio at the University of Montreal (he since moved to Google Brain and recently to OpenAI).

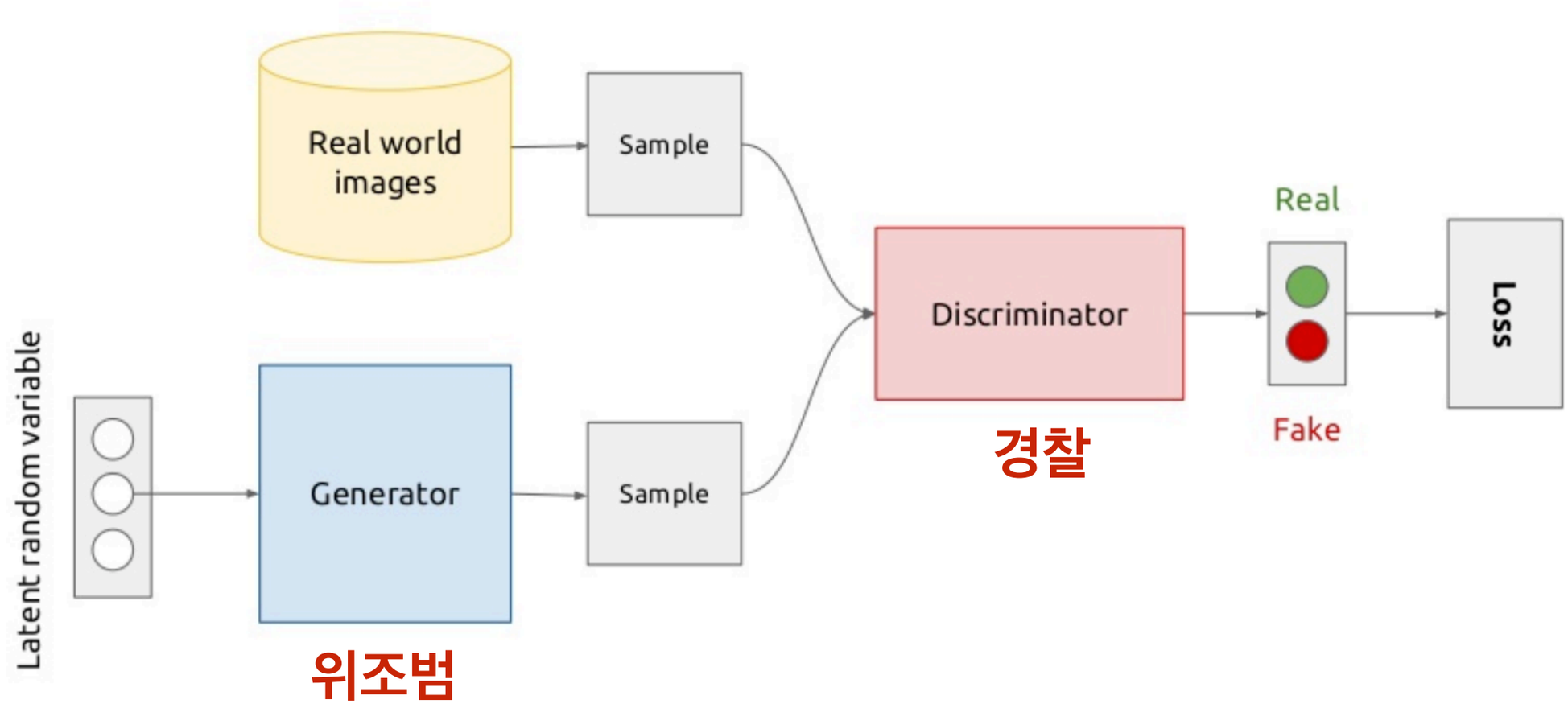
This, and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion.

Adversarial 학습



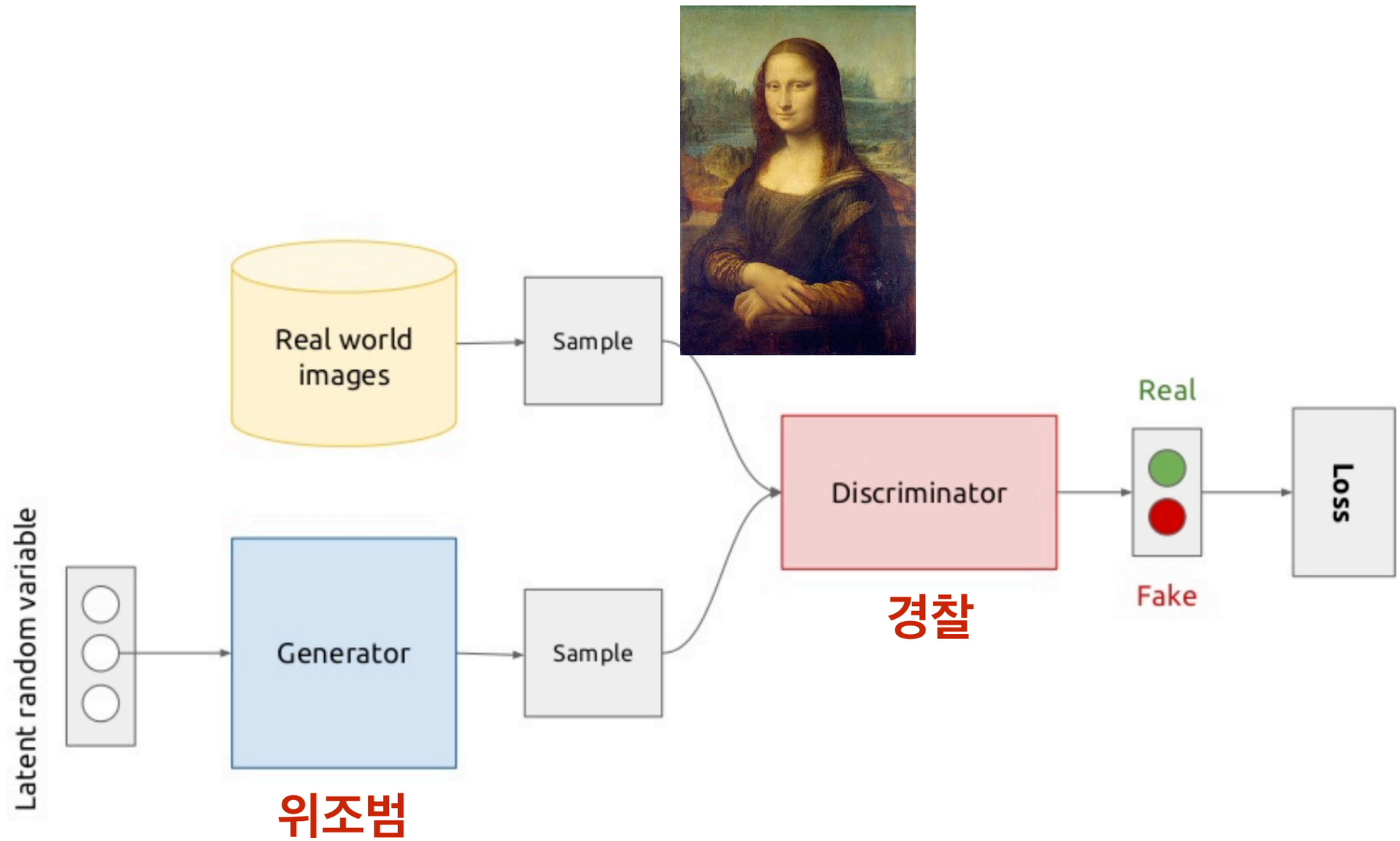
http://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-generative-models-and-adversarial-training-upc-2016

Adversarial 학습



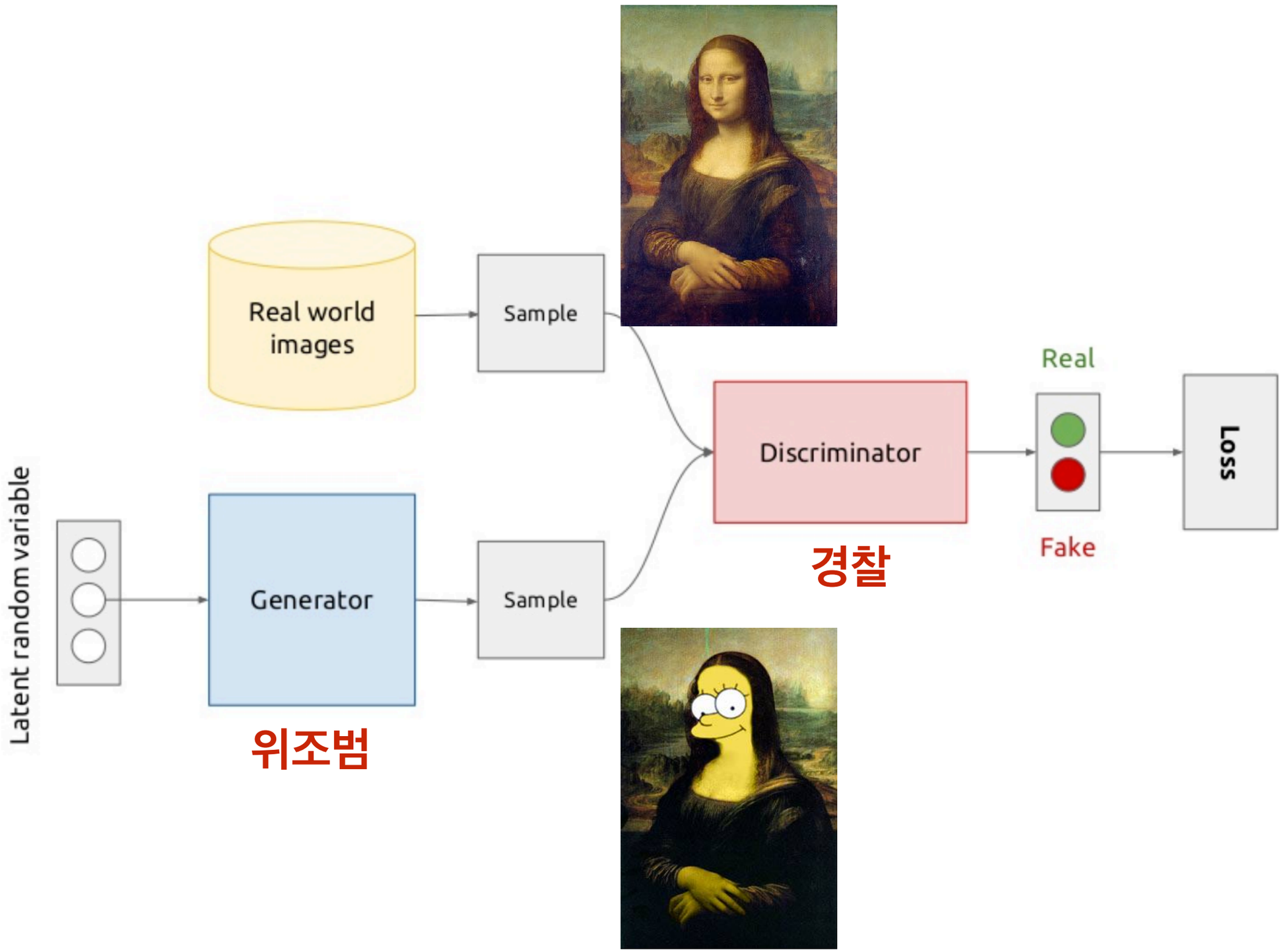
http://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-generative-models-and-adversarial-training-upc-2016

Adversarial 학습



http://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-generative-models-and-adversarial-training-upc-2016

Adversarial 학습



http://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-generative-models-and-adversarial-training-upc-2016

Adversarial 학습

- Notations
  - $p_{data}(x)$: 실제 데이터 x 에서 뽑아낸 확률 분포
  - $p_z(\boldsymbol{z})$: 랜덤 노이즈로부터 뽑아낸 확률 분포
  - $G(z;\theta_g)$: Generator 함수 (θ_g 를 파라미터로 갖는 뉴럴 네트워크)
  - $D(x;\theta_d)$: Discriminator 함수, 출력값은 스칼라 값
    - 입력이 실제 데이터에 왔다면 1을 출력하고 아니면 0을 출력

- Loss 함수

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

Loss 함수

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

Loss 함수

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

x는 실제 데이터로 부터 생성

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

Loss 함수

D(real)의 확률

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

x는 실제 데이터로 부터 생성

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

Loss 함수

D(real)의 확률

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

x는 실제 데이터로 부터 생성　z 는 노이즈로부터 생성

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

Loss 함수

D(real)의 확률

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

x는 실제 데이터로 부터 생성

z 는 노이즈로부터 생성

Generator가 생성한 fake

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

Loss 함수

D(real)의 확률 D(fake)의 확률

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

x는 실제 데이터로 부터 생성 z 는 노이즈로부터 생성 Generator가 생성한 fake

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

Loss 함수

Discriminator는 식을 **최대화**
Generator는 식을 **최소화**

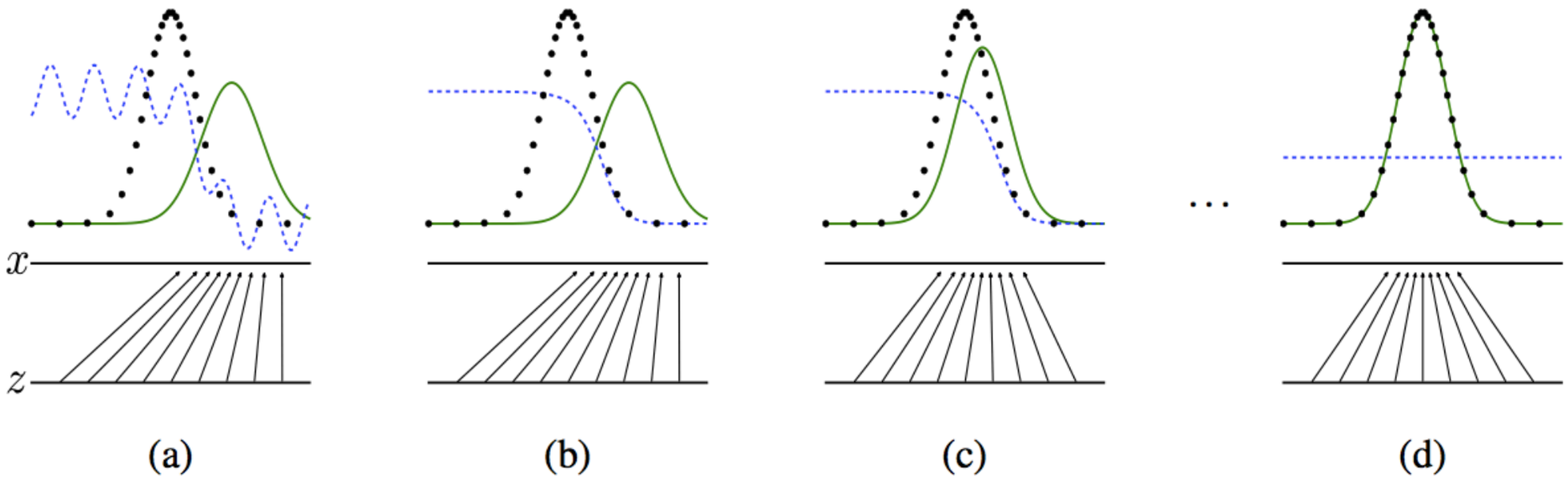
D(real)의 확률 D(fake)의 확률

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

x는 실제 데이터로 부터 생성 z 는 노이즈로부터 생성 Generator가 생성한 fake

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

GAN



(a) (b) (c) (d)

Figure 1: Generative adversarial nets are trained by simultaneously updating the **d**iscriminative distribution (D, blue, dashed line) so that it discriminates between samples from the data generating distribution (black, dotted line) $p_{\boldsymbol{x}}$ from those of the **g**enerative distribution p_g (G) (green, solid line). The lower horizontal line is the domain from which $\boldsymbol{z}$ is sampled, in this case uniformly. The horizontal line above is part of the domain of $\boldsymbol{x}$. The upward arrows show how the mapping $\boldsymbol{x} = G(\boldsymbol{z})$ imposes the non-uniform distribution p_g on transformed samples. G contracts in regions of high density and expands in regions of low density of p_g. (a) Consider an adversarial pair near convergence: p_g is similar to p_{data} and D is a partially accurate classifier. (b) In the inner loop of the algorithm D is trained to discriminate samples from data, converging to $D^*(\boldsymbol{x}) = \frac{p_{\text{data}}(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$. (c) After an update to G, gradient of D has guided $G(\boldsymbol{z})$ to flow to regions that are more likely to be classified as data. (d) After several steps of training, if G and D have enough capacity, they will reach a point at which both cannot improve because $p_g = p_{\text{data}}$. The discriminator is unable to differentiate between the two distributions, i.e. $D(\boldsymbol{x}) = \frac{1}{2}$.

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

GAN 학습

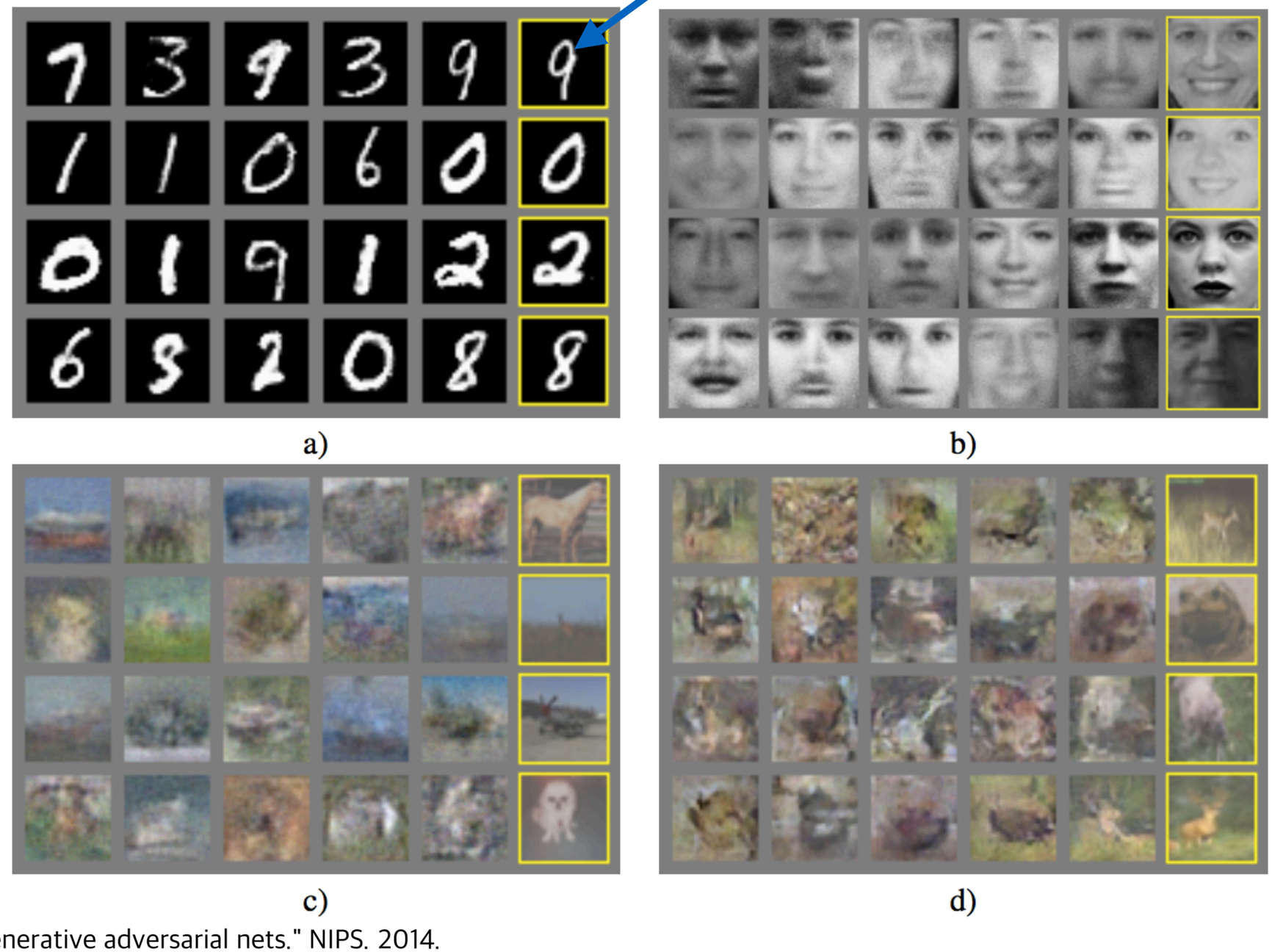
- 실제 구현시 Loss 함수를 약간 변형해서 사용
  - 초기 단계에서 G는 매우 안좋은 결과를 낼 가능성이 높음
  - 따라서 D는 거의 모든 결과에 대해 쉽게 0을 출력할 수 있음
  - $\log(1 - D(G(\boldsymbol{z})))$ 는 초기 단계에서 거의 0이 므로 $(\log(1) = 0)$ generator가 학습이 잘 되지 않음

- Generator가 $\log(1 - D(G(\boldsymbol{z})))$ 를 최소화 하는 대신, $\log D(G(\boldsymbol{z}))$ 를 최대화 하는 방향으로 학습

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

GAN 생성 결과

모델이 Overfit 되어 학습 데이터를 복붙하는 것이 아님을 증명하기 위해 생성된 데이터와 가장 유사한 학습 데이터를 시각화



a) b) c) d)

Goodfellow, Ian, et al. "Generative adversarial nets." NIPS. 2014.

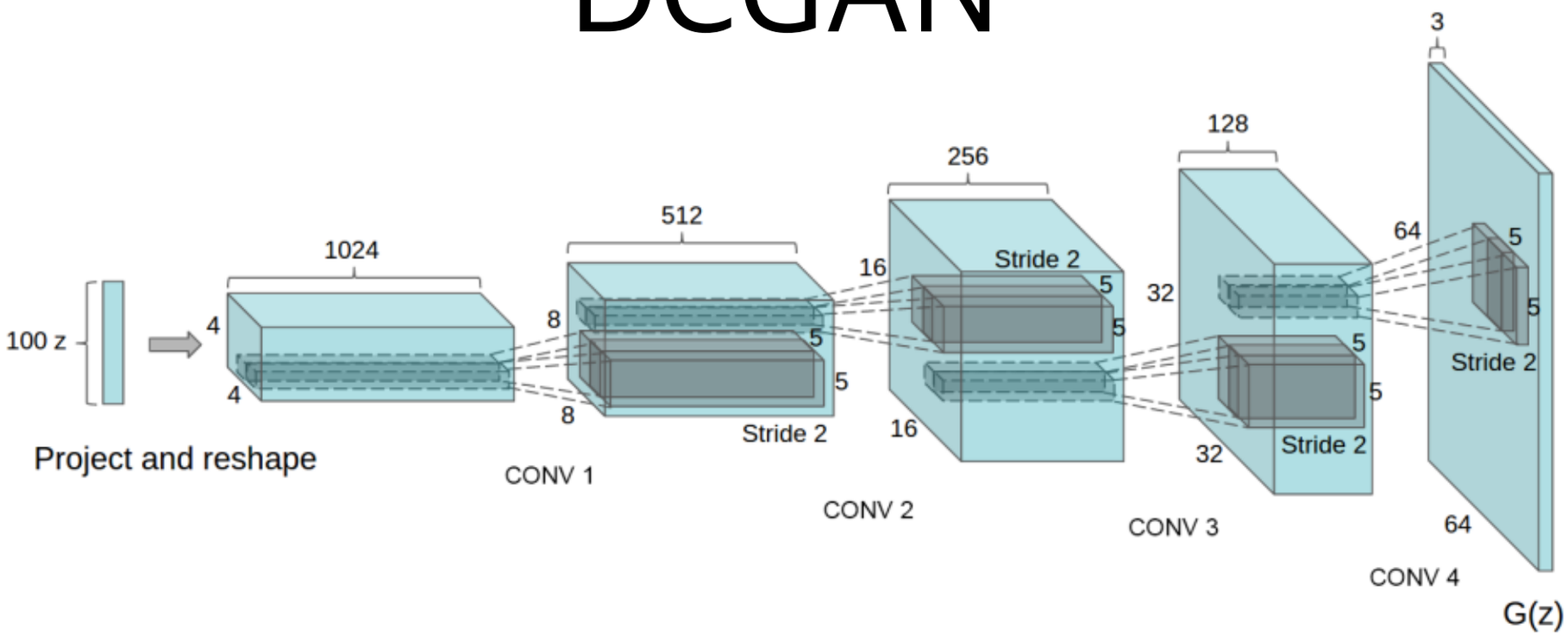
Deep Convolutional GAN

- GAN의 단점
  - 학습이 매우 불안정하고, learning rate의 영향을 많이 받음
  - Learning rate를 높게 주면 거의 모든 경우 학습 실패

- DCGAN
  - GAN 모델을 만들 때 고려할 수 있는 가이드라인 제시
  - 거의 대부분의 상황에서 일반 GAN보다 학습이 잘 되는 장점

Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." ICLR. 2016.

DCGAN

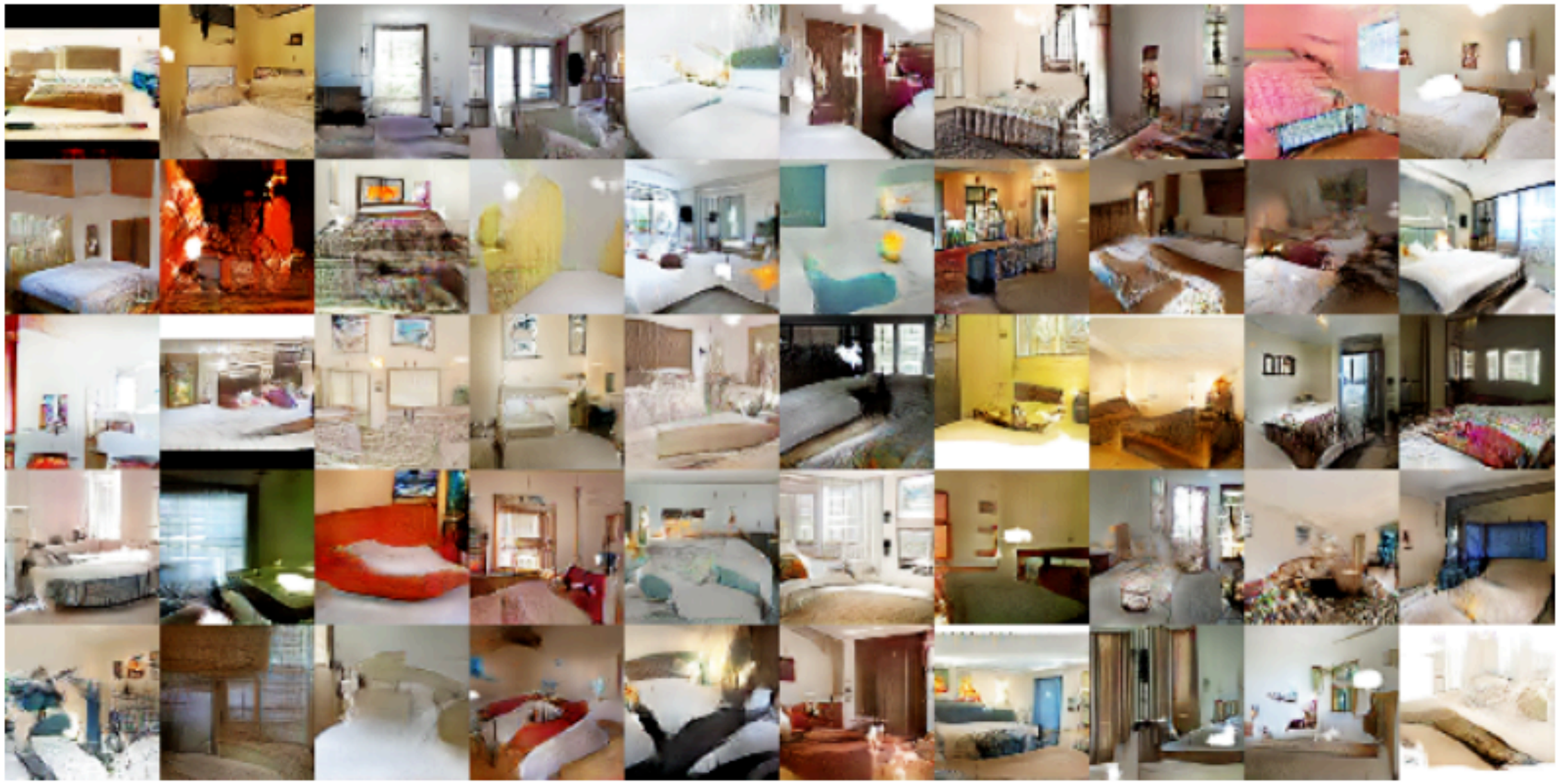


- **풀링 레이어를 삭제**하고 컨볼루션 레이어 혹은 transpose 컨볼루션 레이어를 통해 특징 맵 크기 감소 / 증가
- Generator와 discriminator에 **배치 정규화** 사용
  (Generator의 출력 레이어와 discriminator의 입력 레이어는 사용 x)
- **Fully-connected 레이어 삭제**, 대신 Global average pooling 사용
- Generator에는 활성 함수로 **ReLU**를 사용하고 마지막 레이어는 tanh
- Discriminator에는 **leaky ReLU** 사용

Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." ICLR. 2016.

DCGAN 생성 결과

- LSUN 데이터셋



Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." ICLR. 2016.

DCGAN 생성 결과

- CelebA 데이터셋

Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." ICLR. 2016.

GAN이 중요한 이유

- GAN을 semi-supervised 학습에 응용 가능
  - Unsupervised 하게 GAN을 학습시킨 뒤, discriminator의 중간 레이어 출력값을 통해 특징을 추출 [1]
- Word2vec과 마찬가지로 이미지도 벡터 연산 가능
- 생성된 이미지를 학습 데이터로 사용 가능 [2]

[1] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." ICLR. 2016.
[2] Shrivastava, Ashish, et al. "Learning from simulated and unsupervised images through adversarial training." CVPR. 2017.

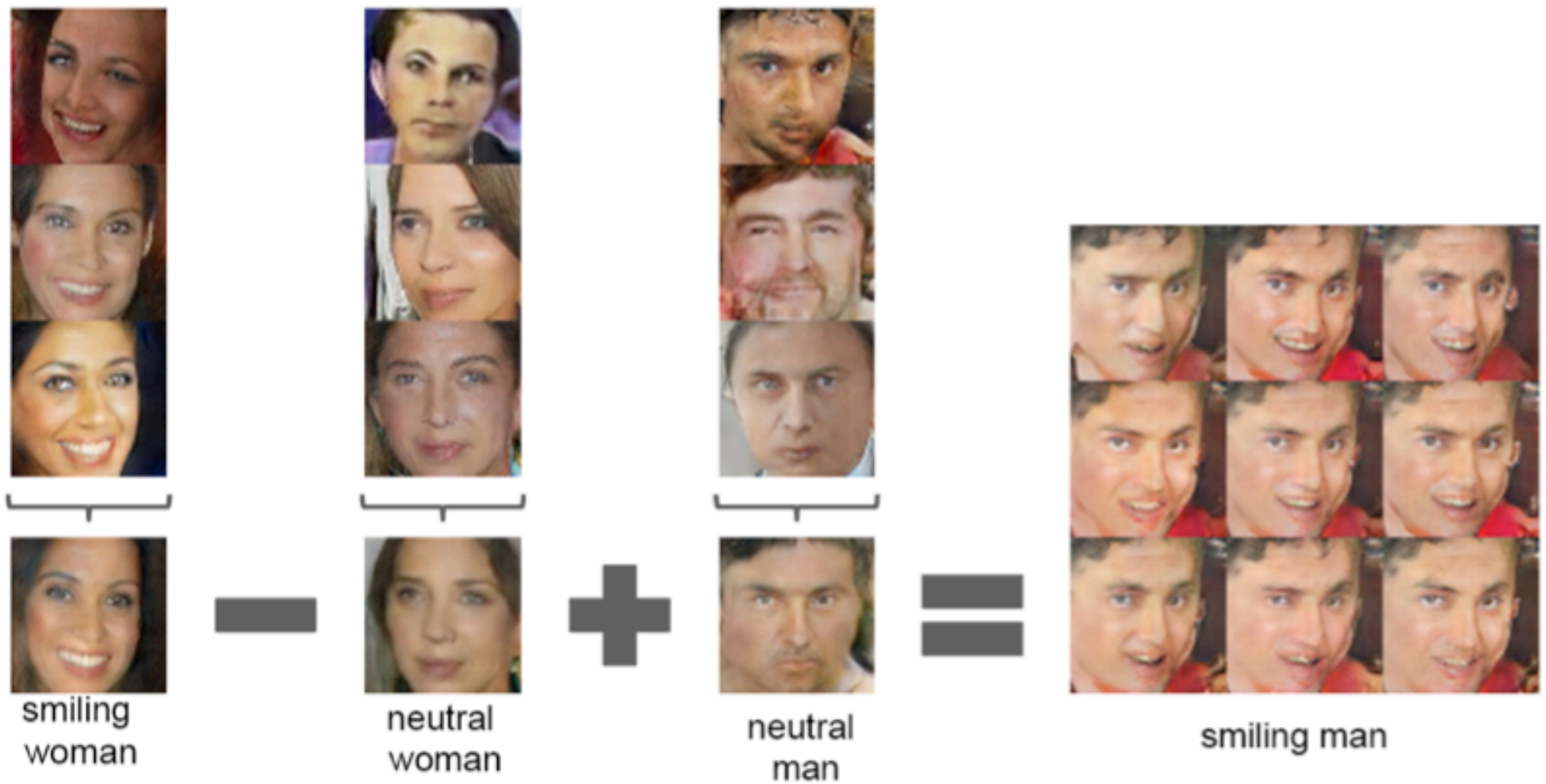
DCGAN

- 노이즈 z를 천천히 변화 시키면서 이미지 생성

Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." ICLR. 2016.

DCGAN

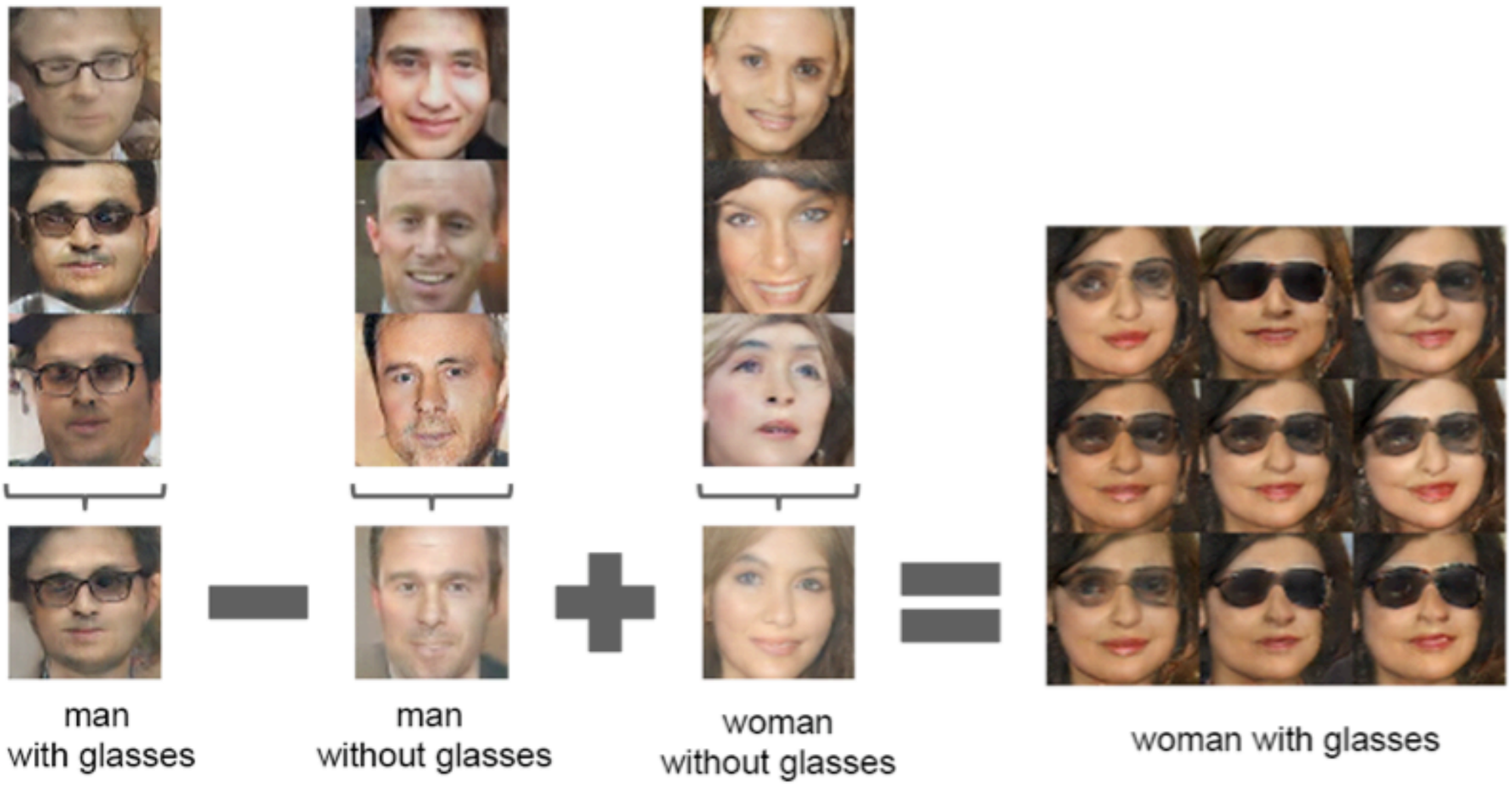
- 노이즈 z 값의 연산을 통해 이미지의 벡터 연산



Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." ICLR. 2016.

DCGAN

- 노이즈 z 값의 연산을 통해 이미지의 벡터 연산

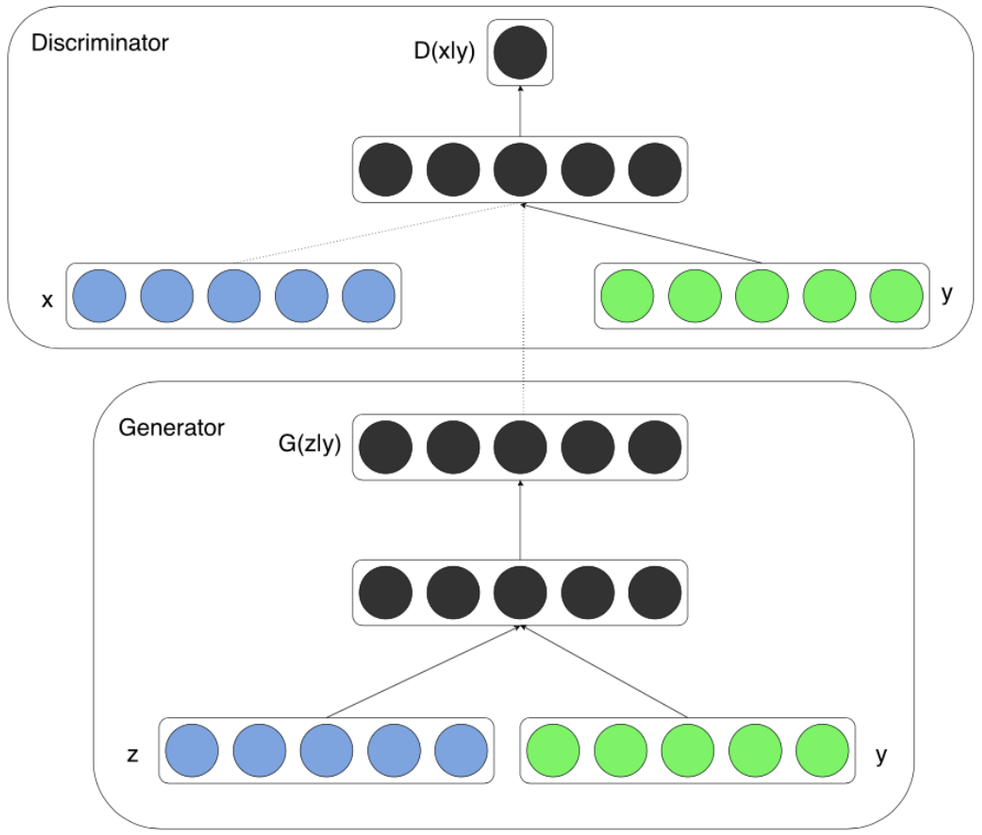


Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." ICLR. 2016.

Conditional GAN

- 지금까지 GAN은 generator가 노이즈를 입력으로 받음
  - 우리가 원하는 대로 이미지를 생성 불가 (숫자 2를 생성해라)

- Conditional GAN $D(x|c)\ or\ G(z|c)$
  - Generator와 discriminator에 추가적인 정보 c 를 주입
  - 추가적인 정보: 클래스 정보, 물체의 위치, 이미지 캡션 등등



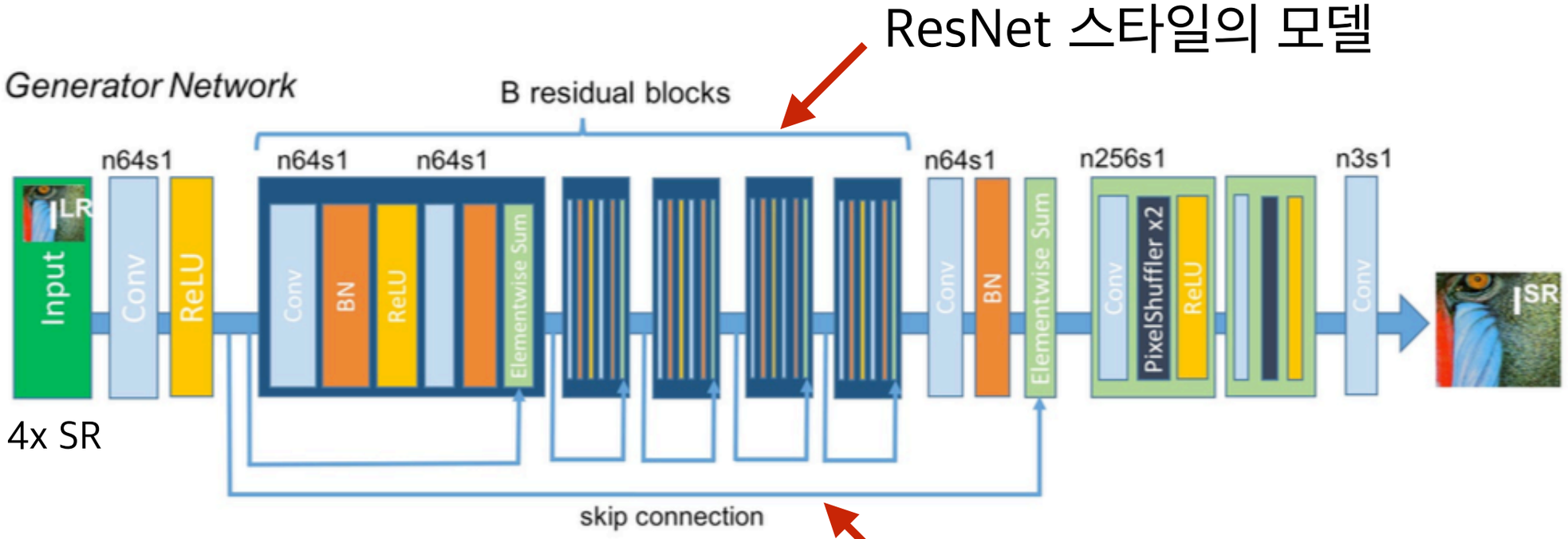
Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." arXiv preprint arXiv:1411.1784 (2014).

GAN 응용 사례

- Super-resolution (SRGAN)
- Text-to-image 합성
  - GAN-CLS, GAWWN, StackGAN
- Image-to-image 변환
  - pix2pix, DiscoGAN, SimGAN

Super-resolution

SRResNet



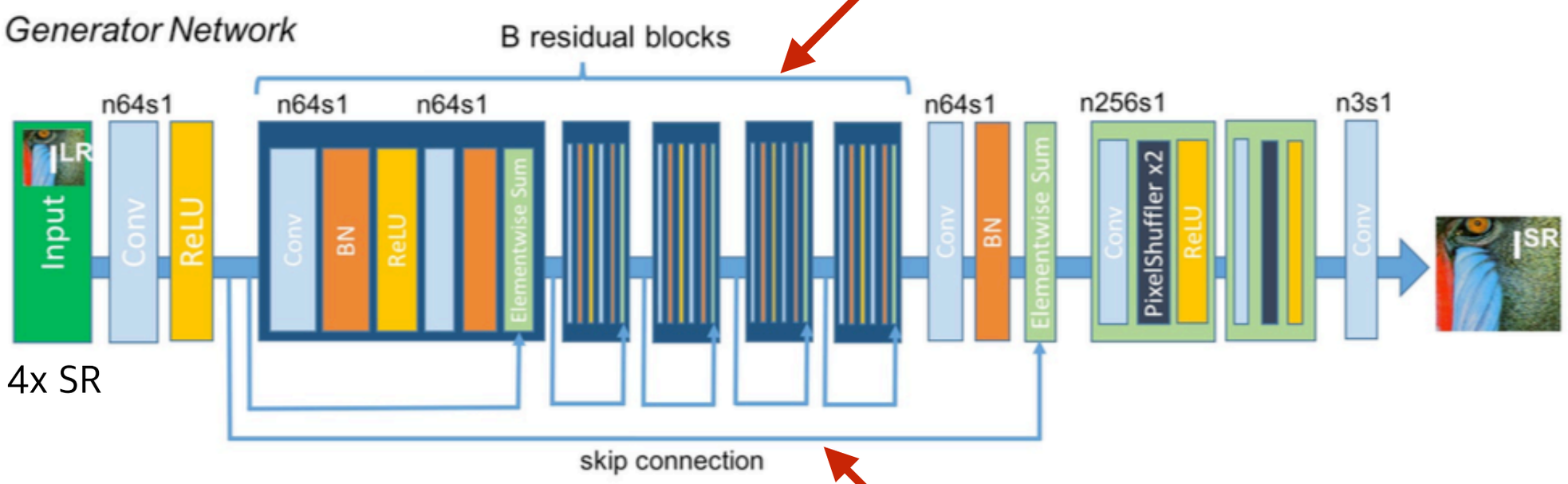
ResNet 스타일의 모델

skip-connection을 사용하면 학습 시 수렴이 빨라짐

Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.

SRResNet

ResNet 스타일의 모델

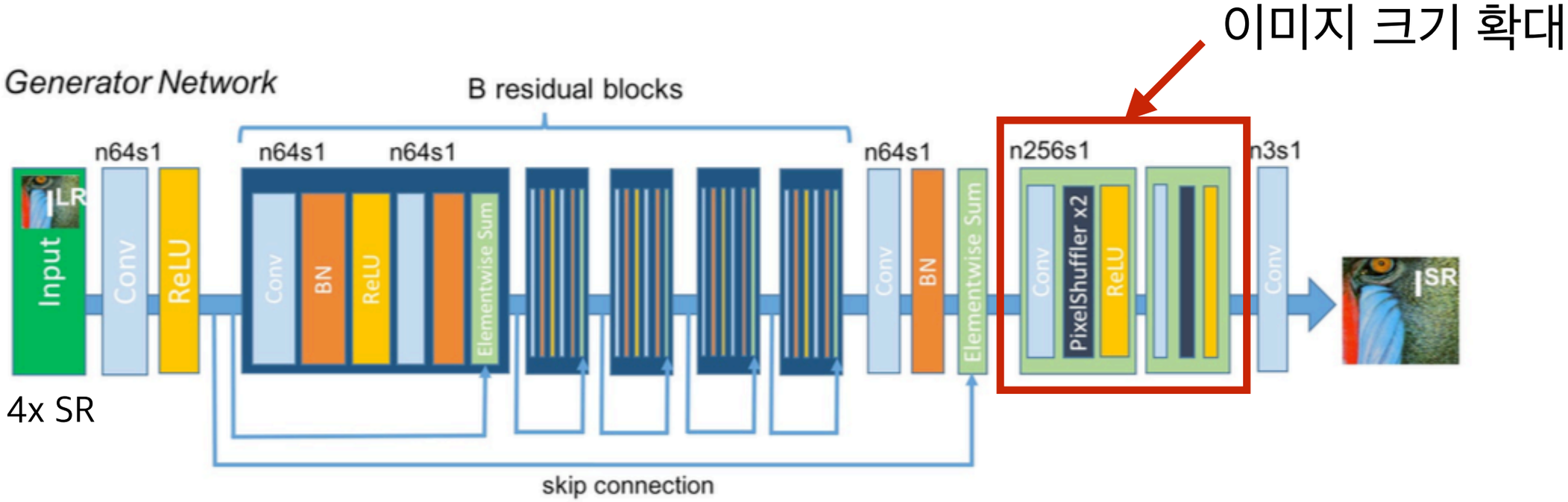


skip-connection을 사용하면 학습 시 수렴이 빨라짐

Loss 함수: $$l^{SR}_{MSE} = \frac{1}{r^2WH}\sum_{x=1}^{rW}\sum_{y=1}^{rH}(I^{HR}_{x,y} - G_{\theta_G}(I^{LR})_{x,y})^2$$

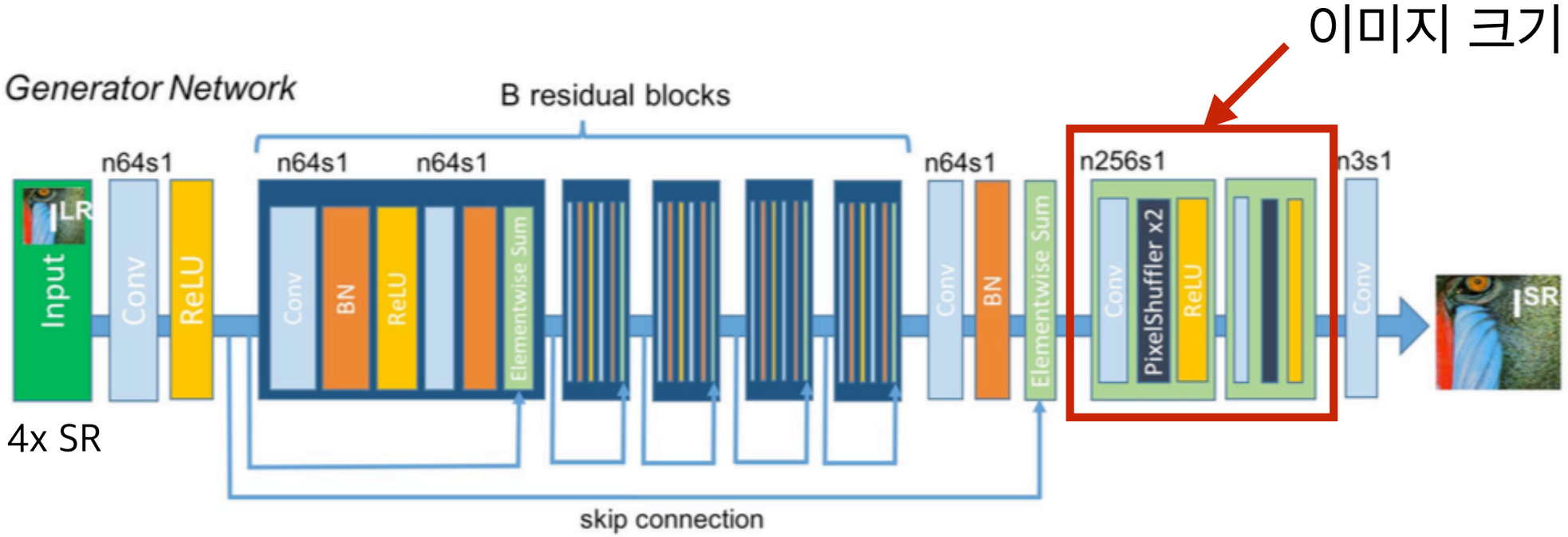
Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.

SRResNet



Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.
Odena, Augustus, Vincent Dumoulin, and Chris Olah. "Deconvolution and checkerboard artifacts." Distill 1.10 (2016): e3.

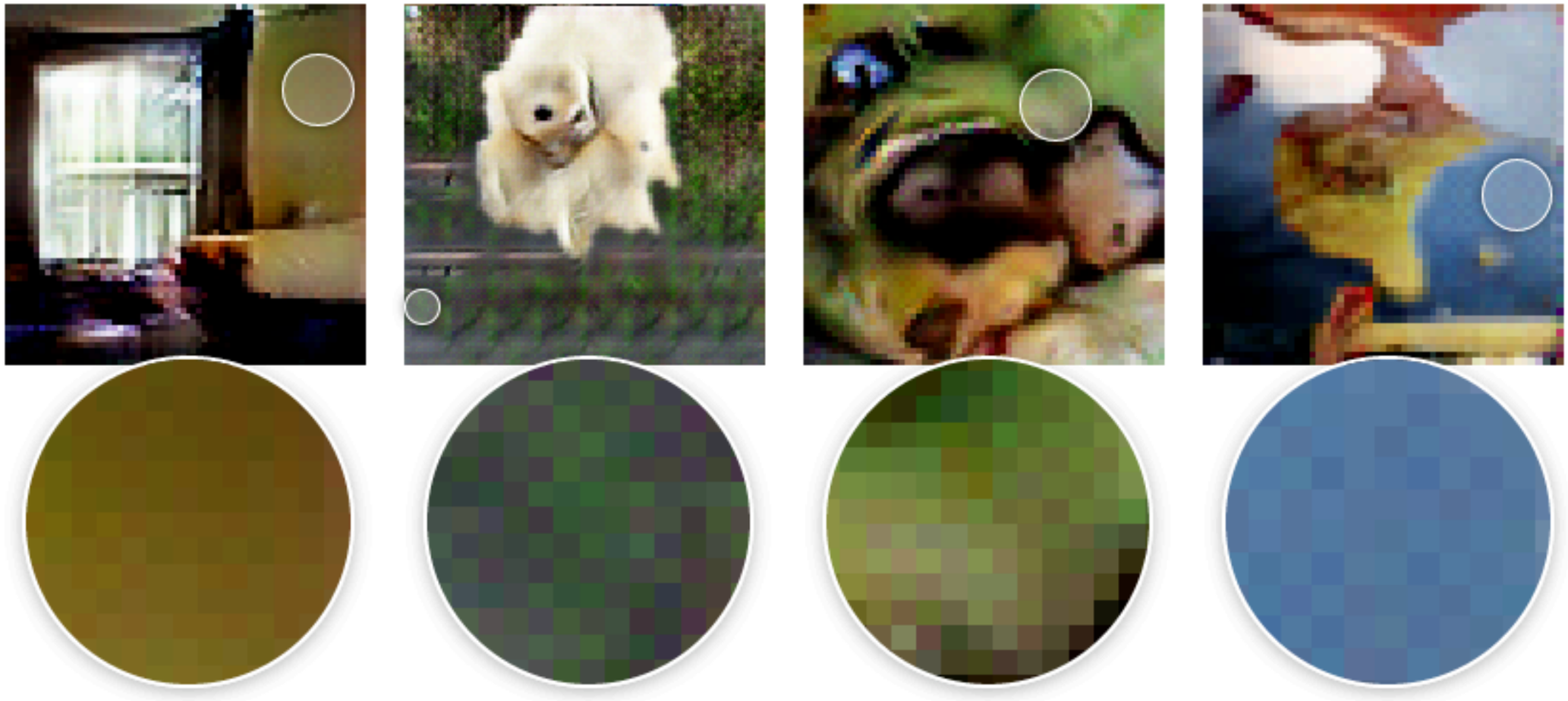
SRResNet



- SRResNet 은 이미지 확대시 Transpose 컨볼루션 사용 x
  - Transpose 컨볼루션은 **체커보드 아티팩트(artifact)**를 생성
  - 대신 **PixelShuffler** 라는 특수 레이어를 사용해서 이미지 크기를 확대

Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.
Odena, Augustus, Vincent Dumoulin, and Chris Olah. "Deconvolution and checkerboard artifacts." Distill 1.10 (2016): e3.

체커보드 아티팩트



Radford, et al., 2015 [1]

Salimans et al., 2016 [2]

Donahue, et al., 2016 [3]

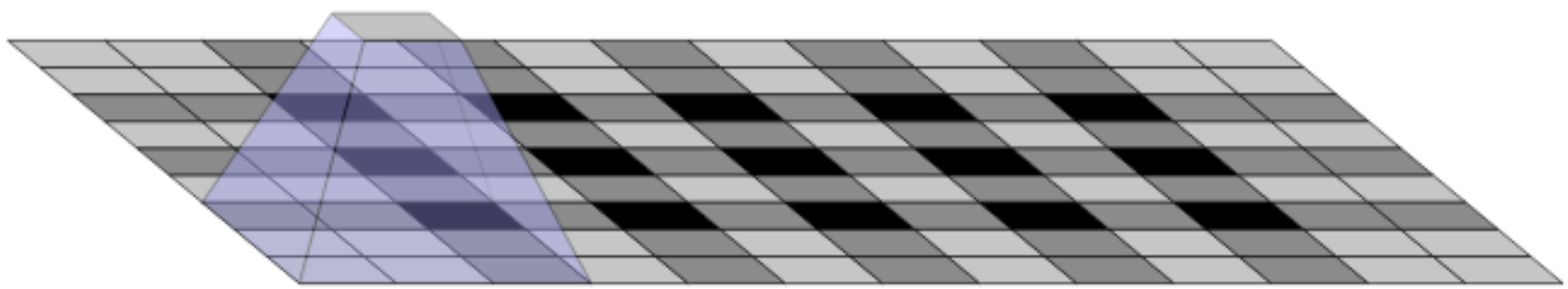
Dumoulin, et al., 2016 [4]

Odena, Augustus, Vincent Dumoulin, and Chris Olah. "Deconvolution and checkerboard artifacts." Distill 1.10 (2016): e3.

체커보드 아티팩트

- Transpose 컨볼루션의 문제점
  - Stride를 2 이상 주면 **겹치는 영역**이 생김
  - 겹치는 영역에서 값을 더해주기 때문에 이미지 상에서 픽셀이 부각됨

- 다른 업샘플링 방법들
  - Sub-pixel shuffler
  - NN-resize (일반적인 이미지 확대 방법을 적용하고 컨볼루션 레이어 통과)



Odena, Augustus, Vincent Dumoulin, and Chris Olah. "Deconvolution and checkerboard artifacts." Distill 1.10 (2016): e3.
Shi, Wenzhe, et al. "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network." CVPR. 2016.

MSE Loss (pixel loss)

- SRResNet의 Loss 함수는 **MSE**를 사용
- MSE 기반 SR 방법의 문제점
  - 정량적 성능 평가 (PSNR, SSIM..) 시 점수가 높게 나옴
  - (정량적 평가 척도가 MSE 기반이기 때문)
  - 하지만 사람이 보기에 이미지가 뿌옇게 보이는 현상 (**blur**) 발생
  - 이는 MSE 함수가 **픽셀 차이들을 평균 내기** 때문에

Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.

MSE Loss (pixel loss)

- SRResNet의 Loss 함수는 **MSE**를 사용

- MSE 기반 SR 방법의 문제점
  - 정량적 성능 평가 (PSNR, SSIM..) 시 점수가 높게 나옴
  - (정량적 평가 척도가 MSE 기반이기 때문)
  - 하지만 사람이 보기에 이미지가 뿌옇게 보이는 현상 (**blur**) 발생
  - 이는 MSE 함수가 **픽셀 차이들을 평균 내기** 때문에

$$l_{MSE}^{SR} = \frac{1}{r^2WH}\sum_{x=1}^{rW}\sum_{y=1}^{rH}(I_{x,y}^{HR} - G_{\theta_G}(I^{LR})_{x,y})^2$$

Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.

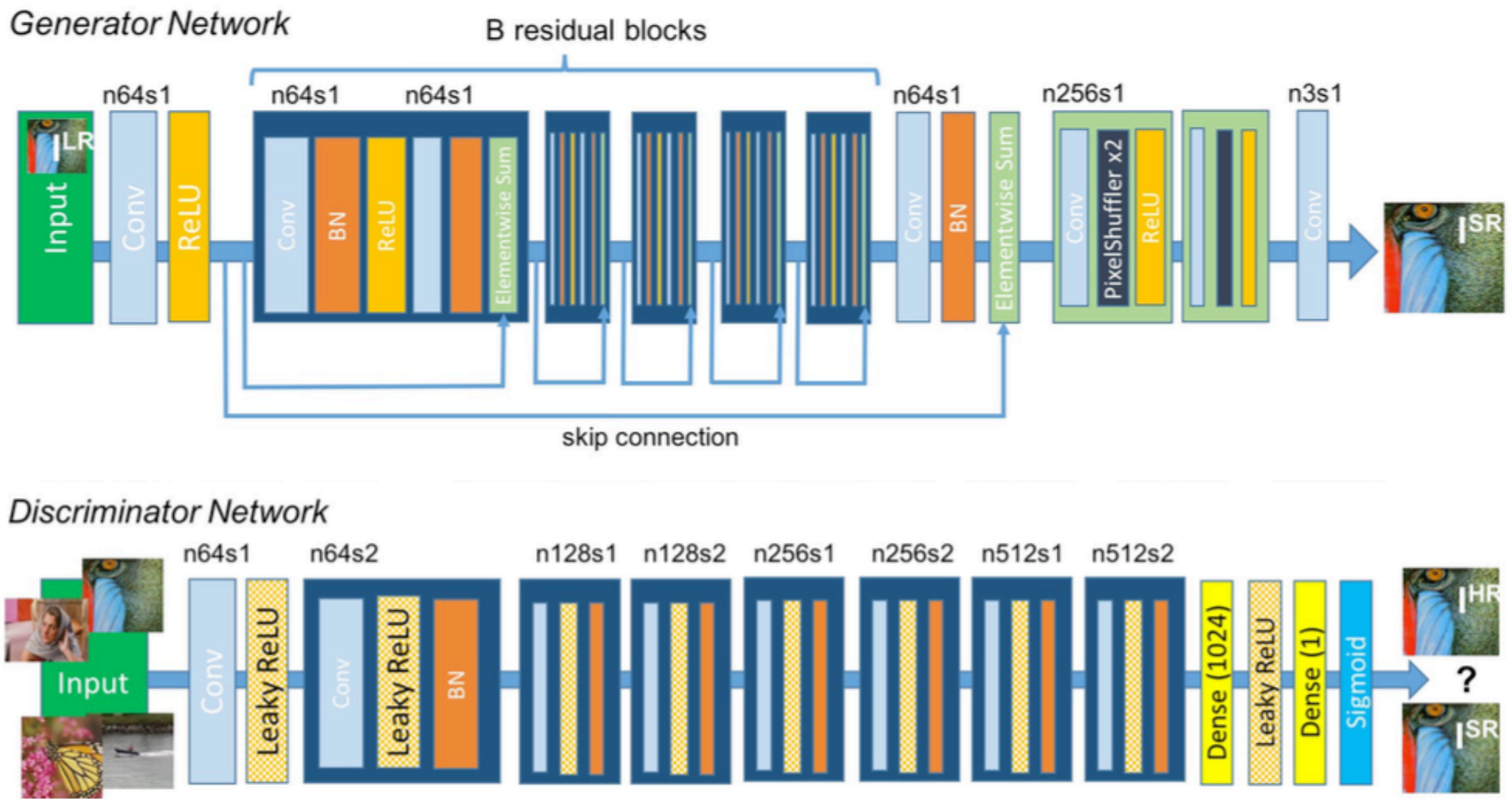
Content Loss

- MSE 기반 Loss의 문제점을 해결하기 위한 Loss [2]
  - 두 장의 이미지를 사전 학습된 VGG 모델에 넣고 특징 추출
  - 추출된 특징을 통해 MSE Loss를 계산
  - 특징 추출은 컨볼루션 레이어의 결과를 사용 (ReLU 활성 함수 이전 값)
  - 예) $\varphi_{4,5}$ = 4번째 컨볼루션 다음, 5번째 풀링 레이어 이전

$$l^{SR}_{VGG/i.j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2$$

Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.
Johnson, Justin, Alexandre Alahi, and Li Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution." ECCV, 2016.

SRGAN



Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.

Adversarial Loss

- 일반 GAN Loss 함수 $l_{Gen}^{SR} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$

- SRGAN의 Loss 함수
  - Adversarial loss와 Content loss를 결합해서 사용

$$l^{SR} = \underbrace{\underbrace{l_{\mathrm{X}}^{SR}}_{\text{content loss}} + \underbrace{10^{-3} l_{Gen}^{SR}}_{\text{adversarial loss}}}_{\text{perceptual loss (for VGG based content losses)}}$$

Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.

결과

	SRResNet-		SRGAN-		
Set5	MSE	VGG22	MSE	VGG22	VGG54
PSNR	32.05	30.51	30.64	29.84	29.40
SSIM	0.9019	0.8803	0.8701	0.8468	0.8472
MOS	3.37	3.46	3.77	3.78	3.58
Set14					
PSNR	28.49	27.19	26.92	26.44	26.02
SSIM	0.8184	0.7807	0.7611	0.7518	0.7397
MOS	2.98	3.15*	3.43	3.57	3.72*

PSNR / SSIM:

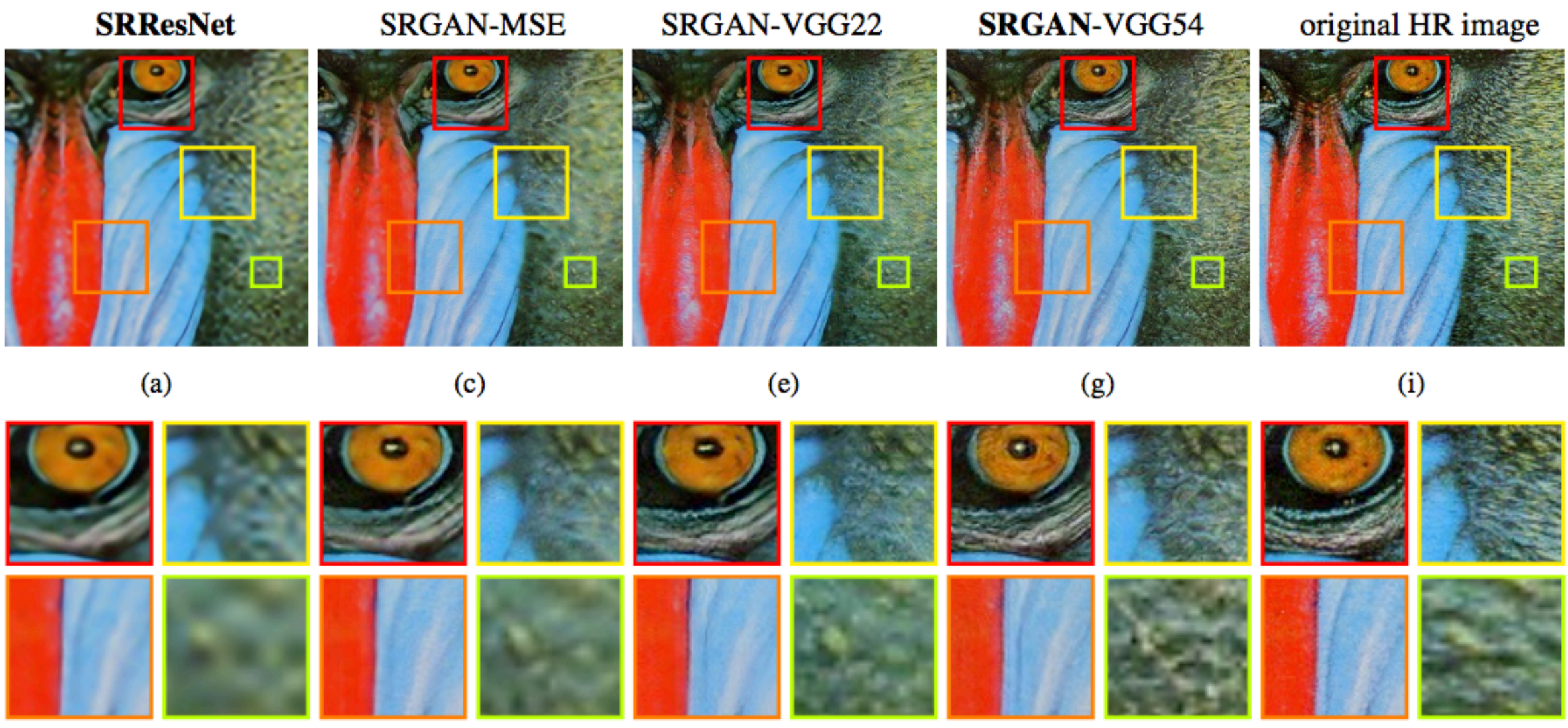
- 픽셀들을 이용해서 계산하는 정량적인 이미지 차이 계산 방법
- 단순 계산이라 간단하게 구할 수 있지만, 사람의 인지와 차이를 보임

MOS (Mean Opinion Score)

- 여러 사람들에게 이미지에 대해 평가를 내리게 한 뒤 평균 계산한 값
- 가장 좋은 척도, 하지만 계산하기 어려움

Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.

결과 예시



Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.

Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." CVPR. 2017.

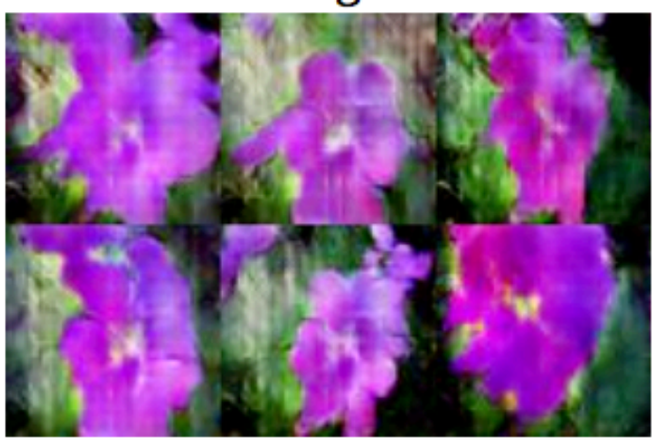
text-to-image 합성

Text-to-image 합성

this small bird has a pink breast and crown, and black primaries and secondaries.

this magnificent fellow is almost all black with a red crest, and white cheek patch.

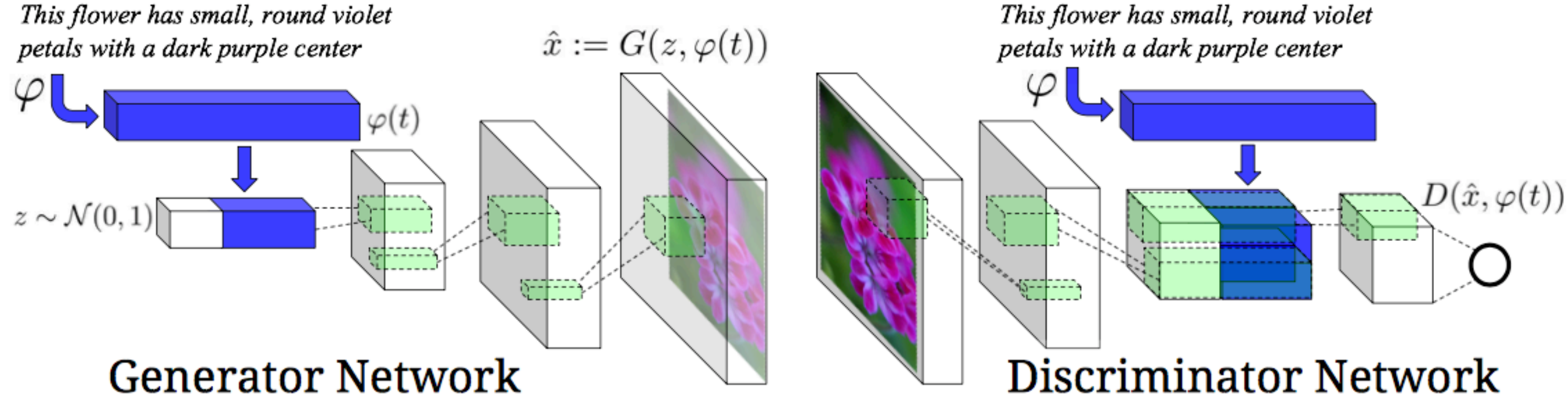
the flower has petals that are bright pinkish purple with white stigma



this white and yellow flower have thin white petals and a round yellow stamen

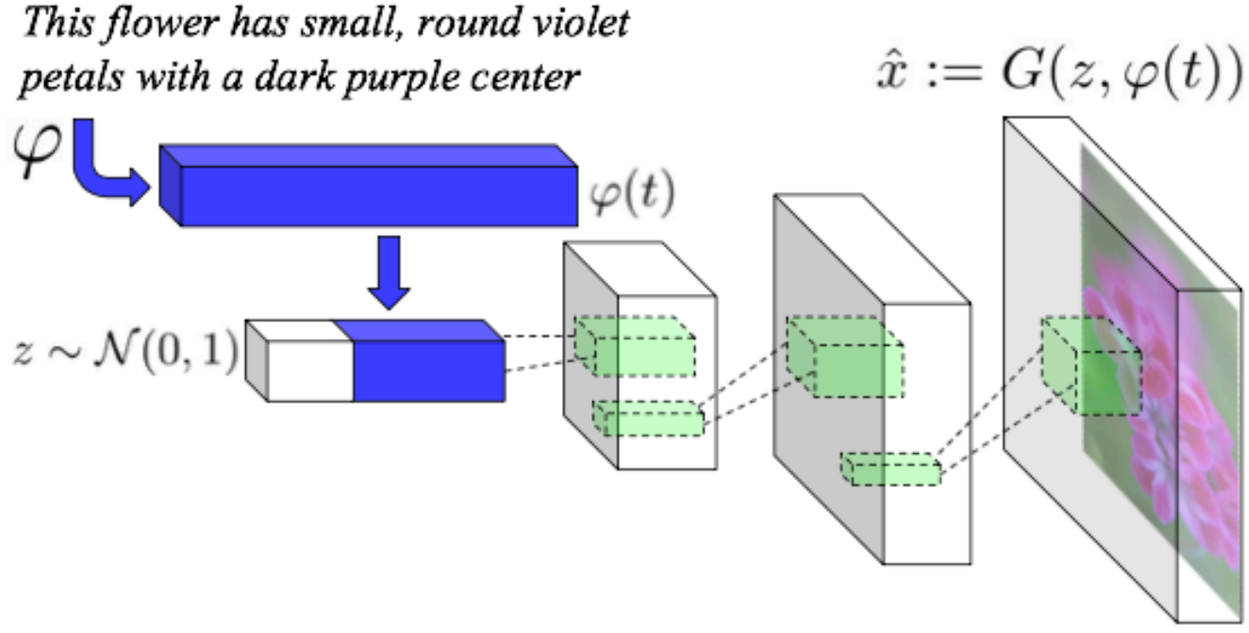
Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

Text-to-image 합성



Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

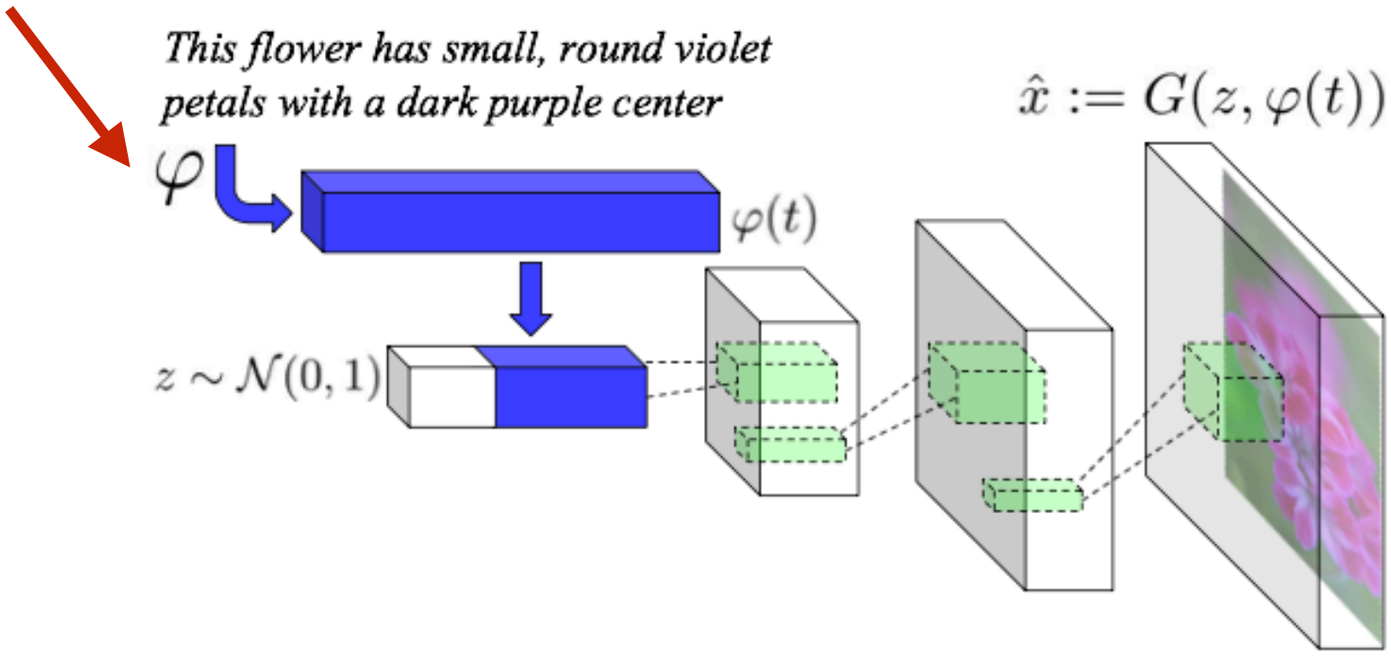
Generator



Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

Generator

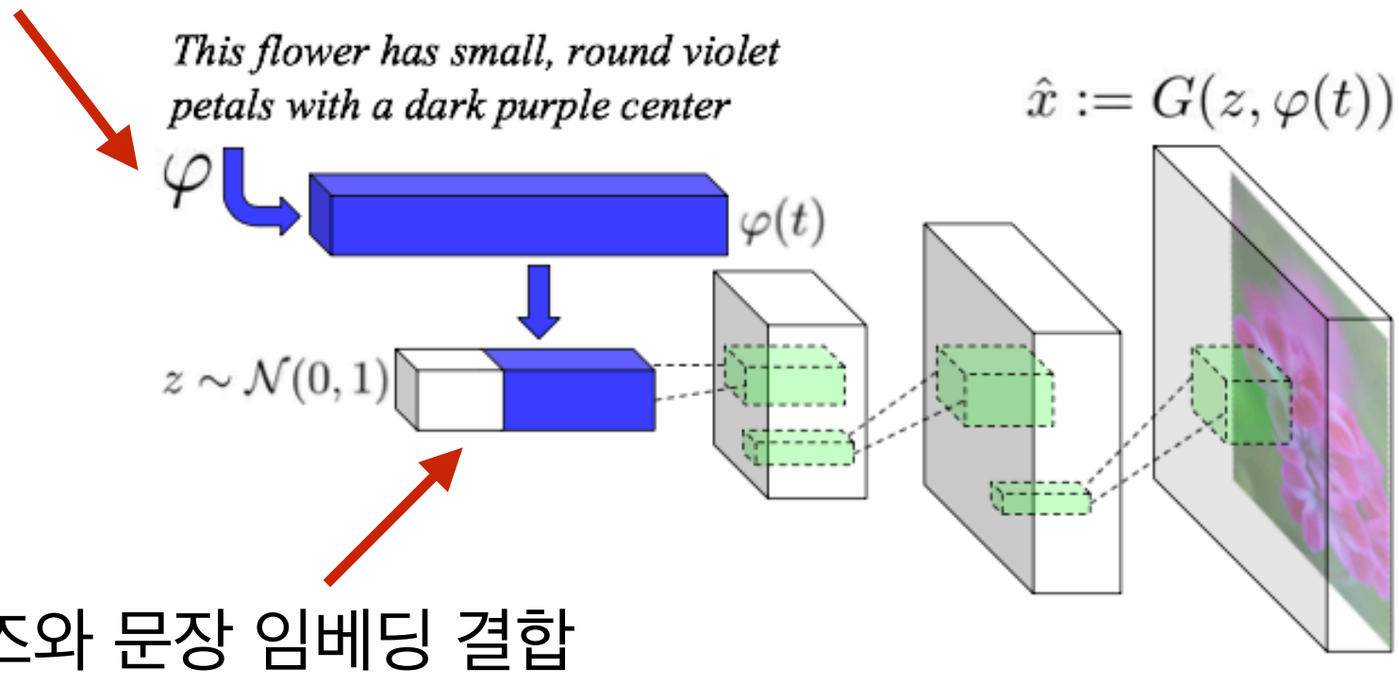
문장 임베딩 (사전 학습된 모델 사용 e.g. skip-thought)
+ FC 레이어를 통한 차원 축소 (e.g. 4096 -> 1024)



Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

Generator

문장 임베딩 (사전 학습된 모델 사용 e.g. skip-thought)
+ FC 레이어를 통한 차원 축소 (e.g. 4096 -> 1024)

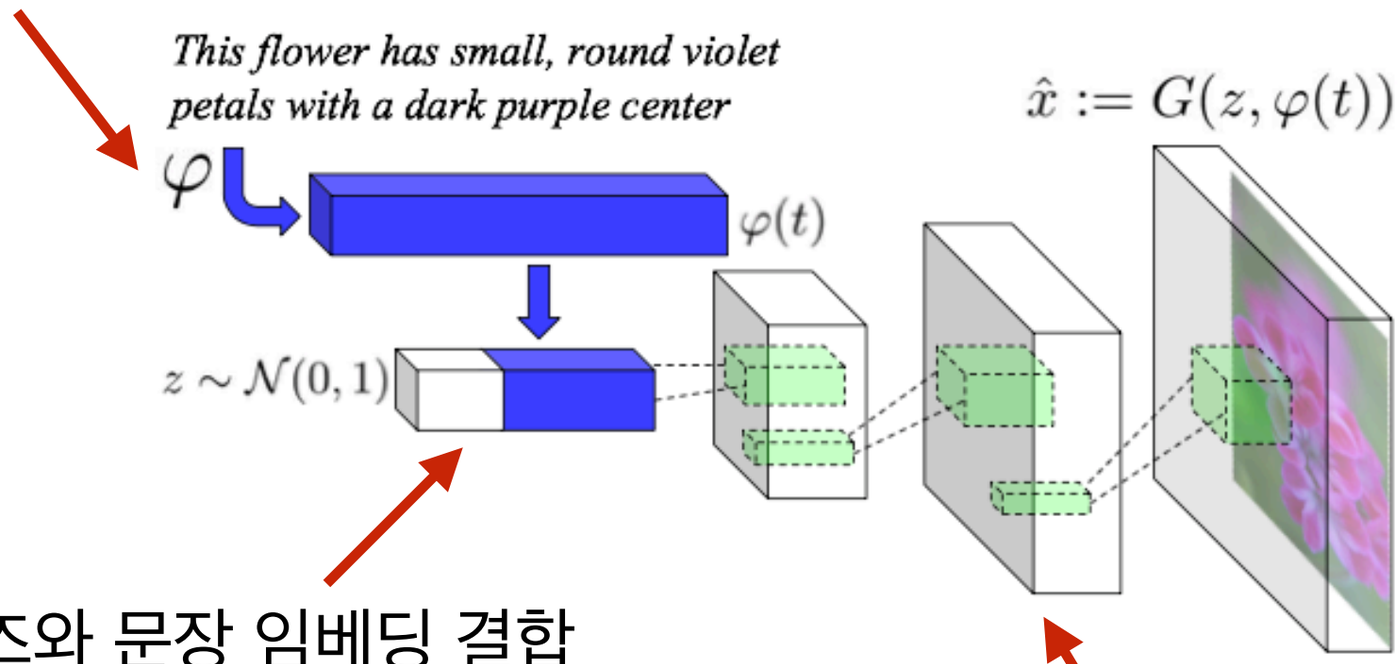


랜덤 노이즈와 문장 임베딩 결합

Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

Generator

문장 임베딩 (사전 학습된 모델 사용 e.g. skip-thought)
+ FC 레이어를 통한 차원 축소 (e.g. 4096 -> 1024)

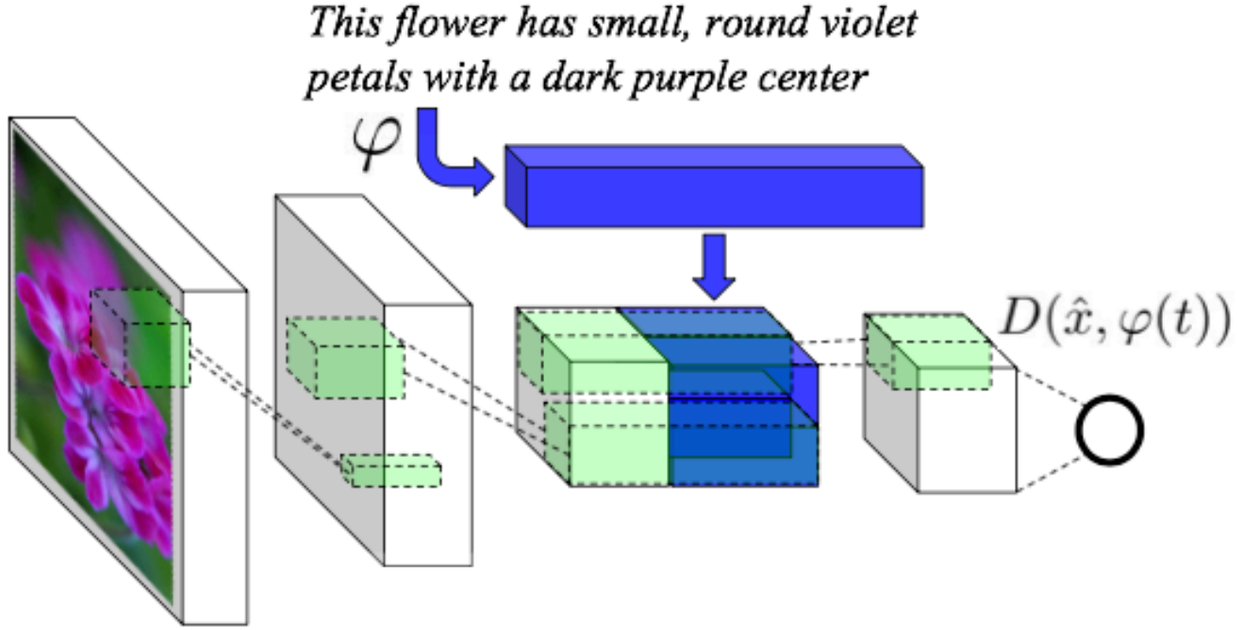


랜덤 노이즈와 문장 임베딩 결합

이미지 생성 (DCGAN과 유사한 과정)

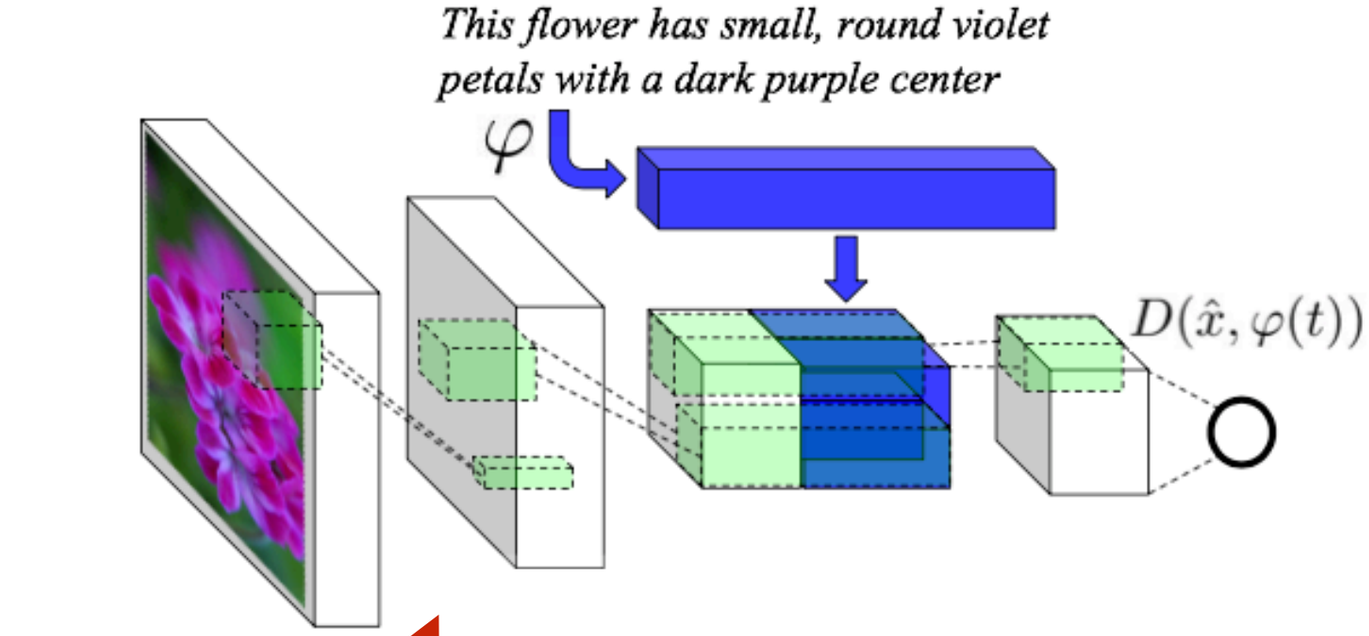
Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

Discriminator



Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

Discriminator

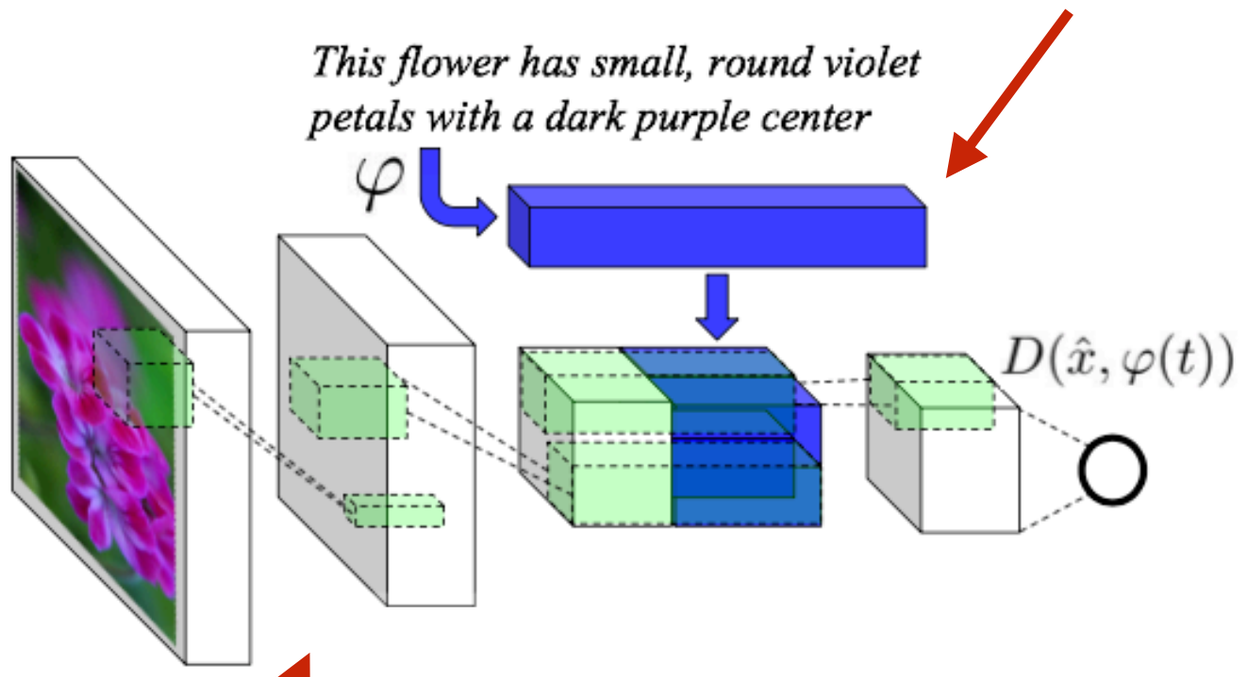


이미지 축소
(DCGAN과 유사한 과정)

Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

Discriminator

문장 임베딩 +
FC 레이어를 통한 차원 축소

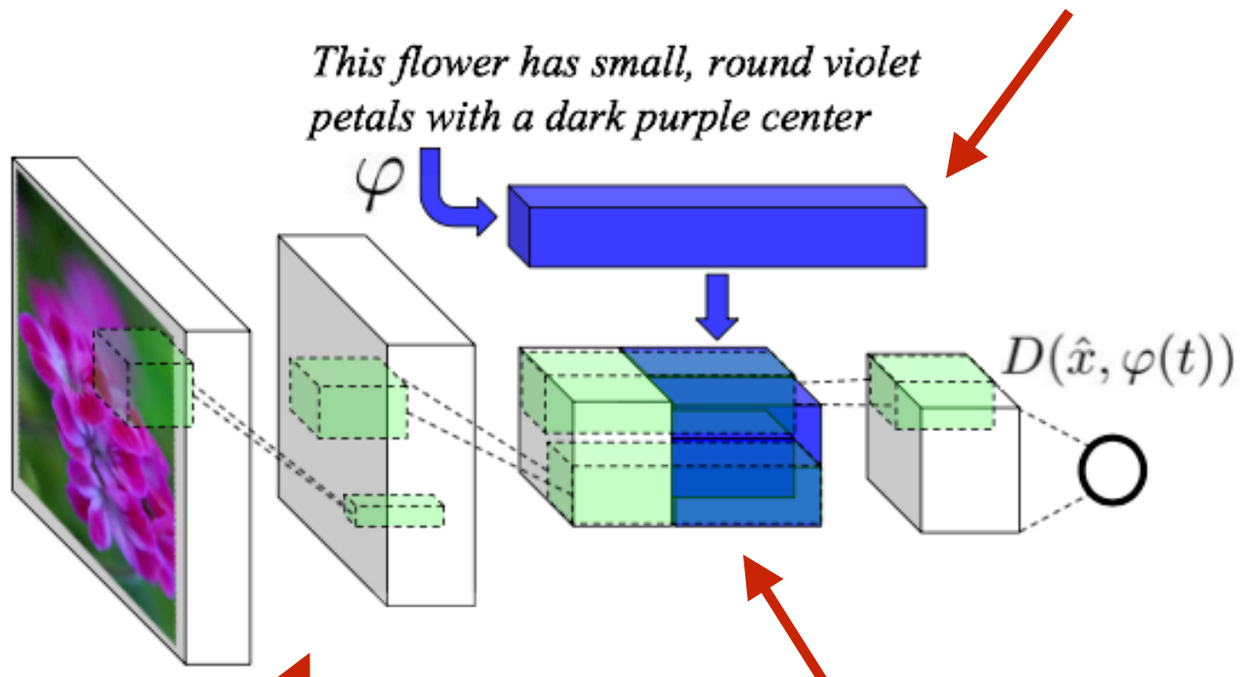


이미지 축소
(DCGAN과 유사한 과정)

Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

Discriminator

문장 임베딩 +
FC 레이어를 통한 차원 축소

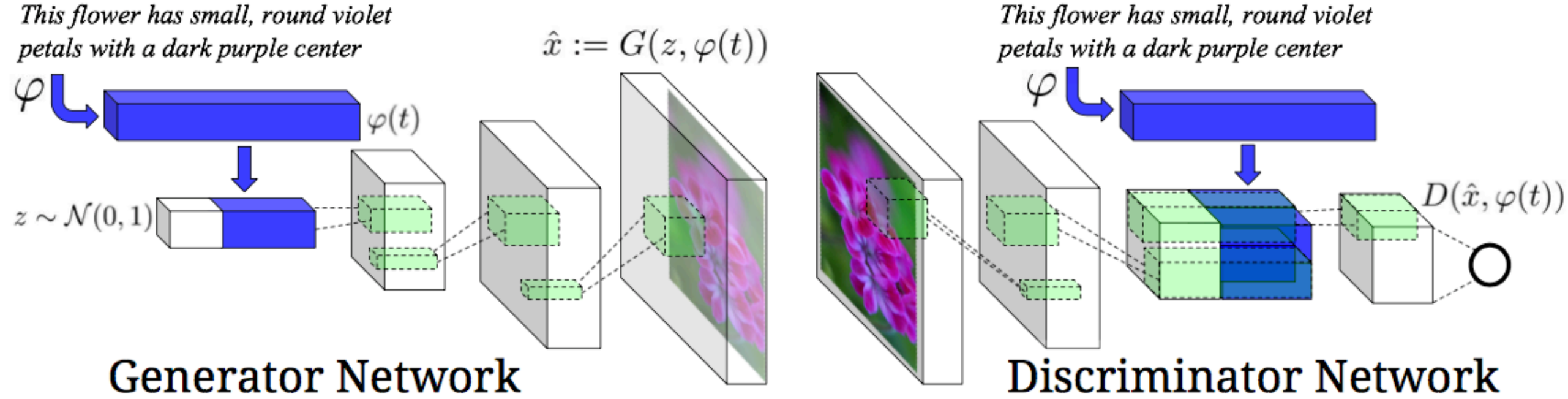


이미지 축소
(DCGAN과 유사한 과정)

이미지 특징맵과
문장 임베딩 결합

Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

Text-to-image 합성



Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

GAN-CLS

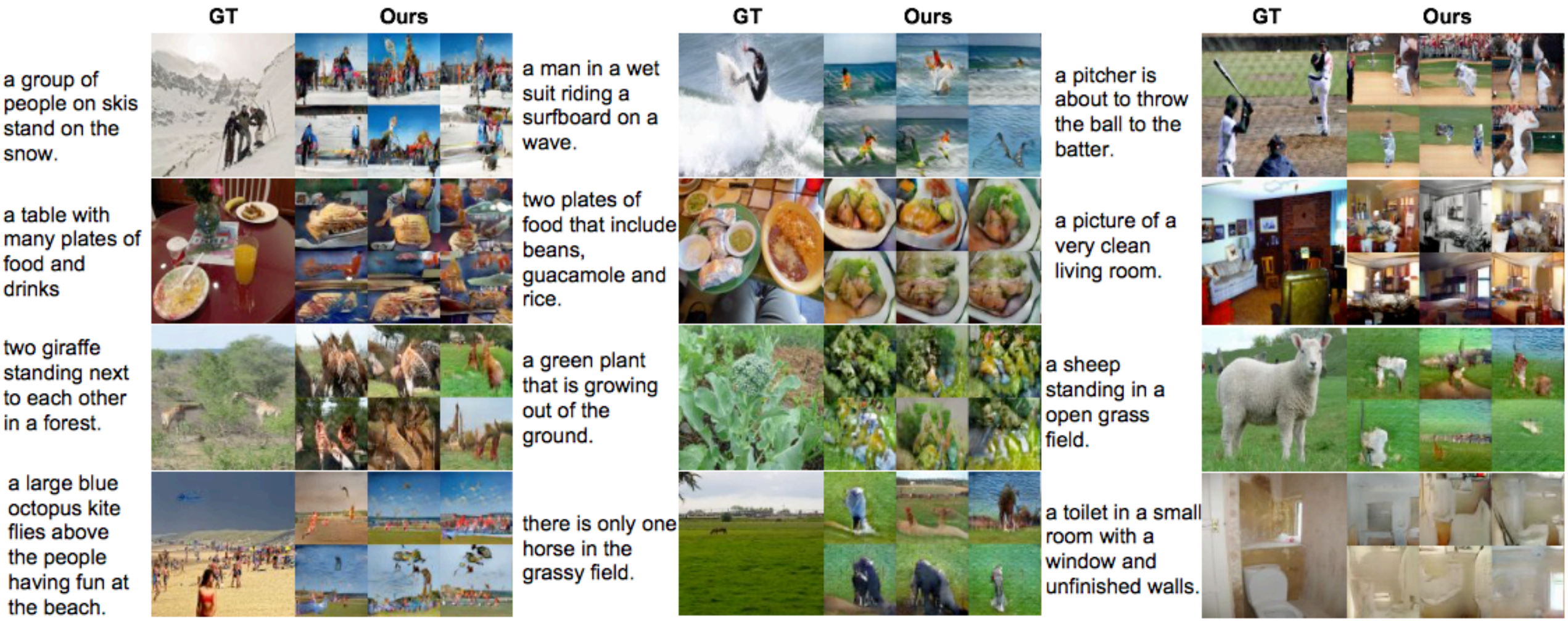
- 기존 discriminator는 이미지가 실제냐 아니냐만 판별
  - 이미지와 텍스트 짝이 맞는지도 판별해야 정확한 결과가 나옴

- Matching-aware discriminator
  - 실제 이미지-맞는 텍스트, 실제 이미지-틀린 텍스트, 가짜 이미지-맞는 텍스트
  - 세 가지 시나리오를 통해 discriminator loss 계산 및 학습

$$
\begin{aligned}
s_r &\leftarrow D(x, h) \ \{\text{real image, right text}\} \\
s_w &\leftarrow D(x, \hat{h}) \ \{\text{real image, wrong text}\} \\
s_f &\leftarrow D(\hat{x}, h) \ \{\text{fake image, right text}\} \\
\mathcal{L}_D &\leftarrow \log(s_r) + (\log(1 - s_w) + \log(1 - s_f))/2
\end{aligned}
$$

Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

결과 (MS-COCO)



Reed, Scott, et al. "Generative Adversarial Text to Image Synthesis." ICML. 2016.

GAWWN

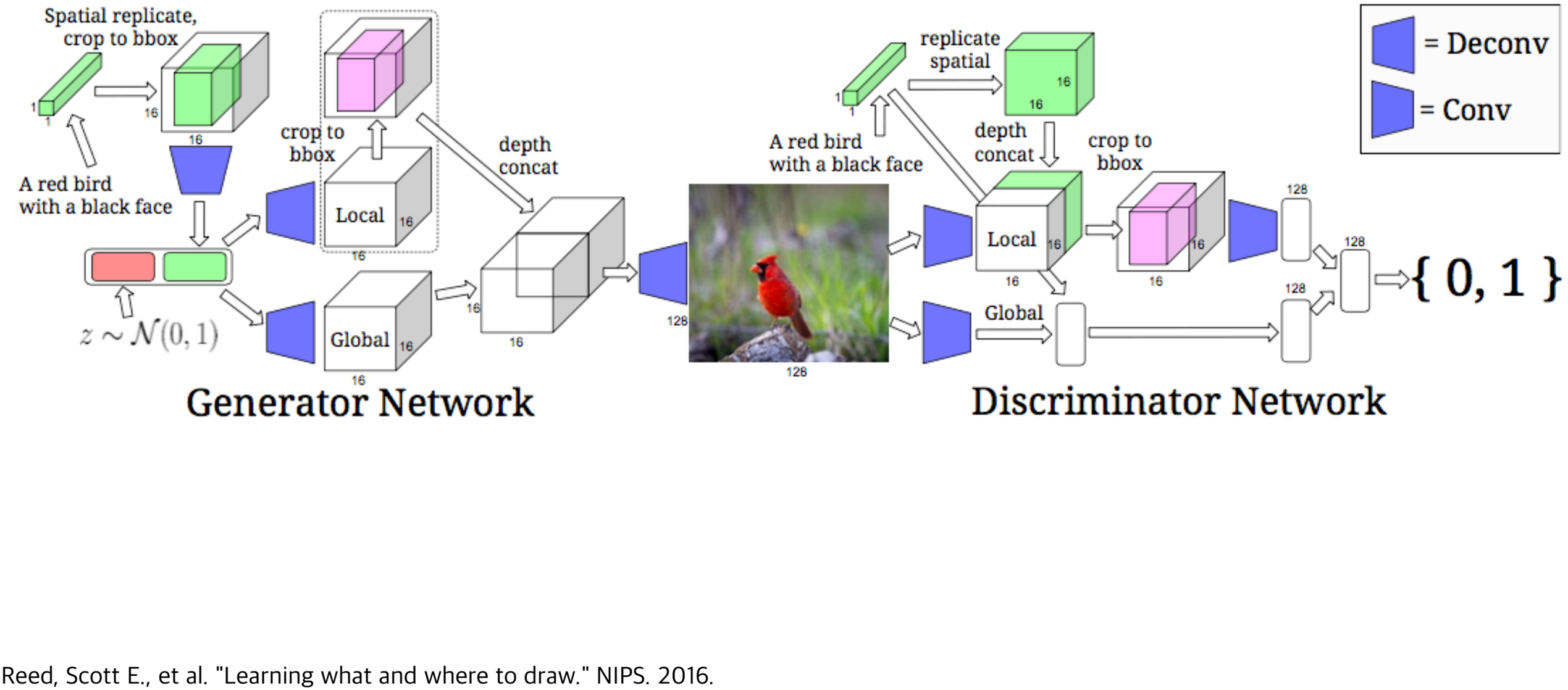
- 바운딩 박스 추가(conditional) 정보 text-to-image 모델
  - 랜덤 **노이즈**, **텍스트**와 **바운딩 박스**를 입력

- 키 포인트 추가(conditional) 정보 text-to-image 모델
  - 랜덤 **노이즈**, **텍스트**와 물체의 파트 위치에 해당하는 **키 포인트**를 입력

- 키 포인트 정보 생성 모델
  - 사용자에게 모든 키 포인트를 입력받게 하는 일은 매우 비 효율적
    (새 데이터셋은 키 포인트가 15개정도)
  - 유저가 입력하지 않은 키 포인트를 생성하기

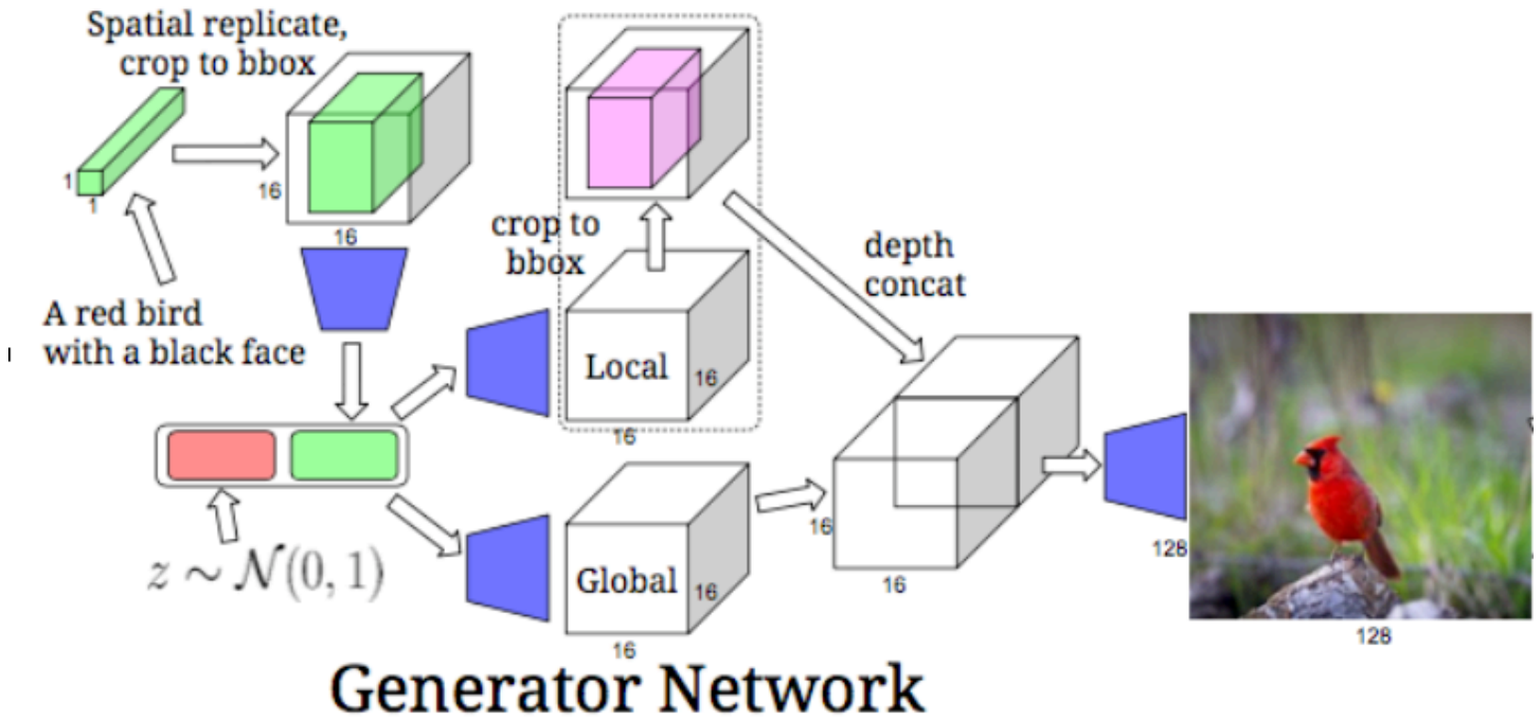
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 모델



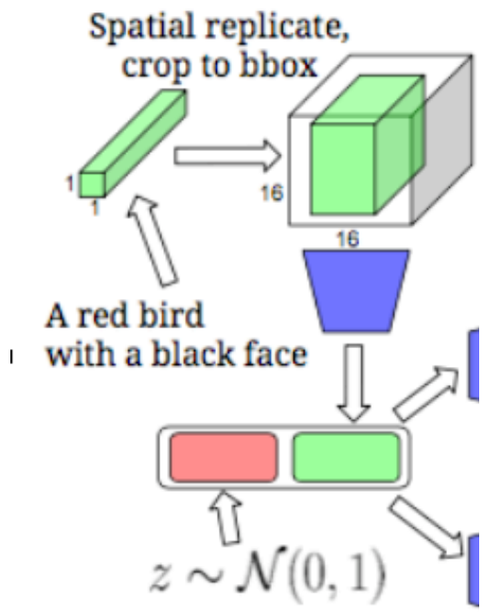
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator



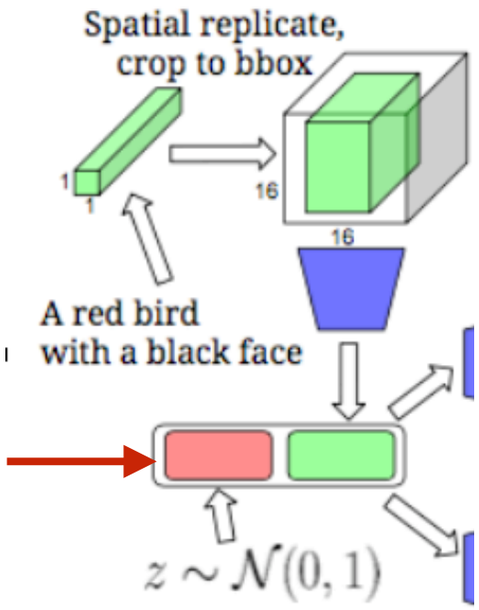
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator



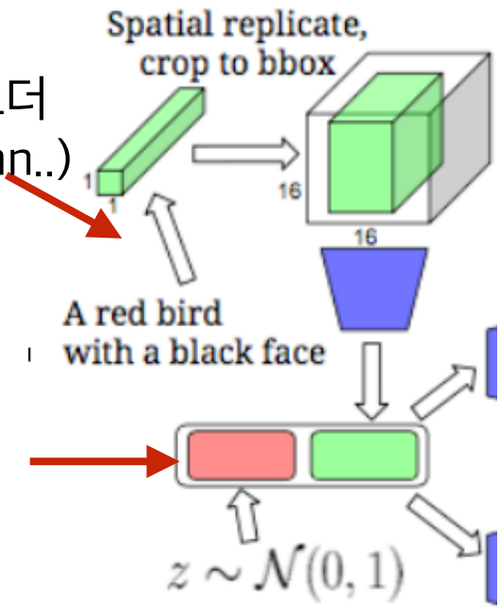
100D 랜덤 노이즈

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator

사전 학습된 텍스트 인코더
(skip-thought, char-cnn..)

100D 랜덤 노이즈



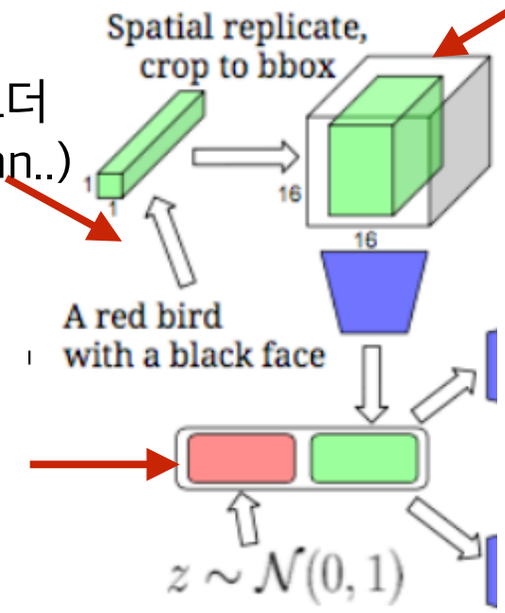
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator

바운딩 박스 정보와 크기를 맞추기 위해
spatial 하게 텍스트 임베딩 결과를 복제

사전 학습된 텍스트 인코더
(skip-thought, char-cnn..)

100D 랜덤 노이즈



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

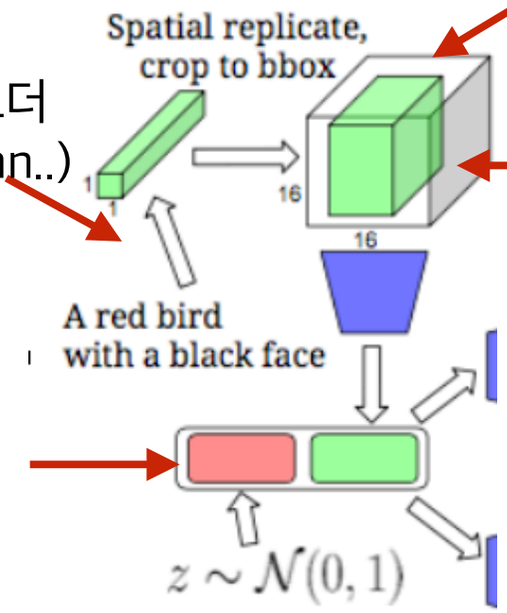
바운딩 박스 Generator

바운딩 박스 정보와 크기를 맞추기 위해
spatial 하게 텍스트 임베딩 결과를 복제

사전 학습된 텍스트 인코더
(skip-thought, char-cnn..)

바운딩 박스 크기로 텍스트 임베딩 크롭
(바운딩 박스 외부는 0으로 채워짐)

100D 랜덤 노이즈



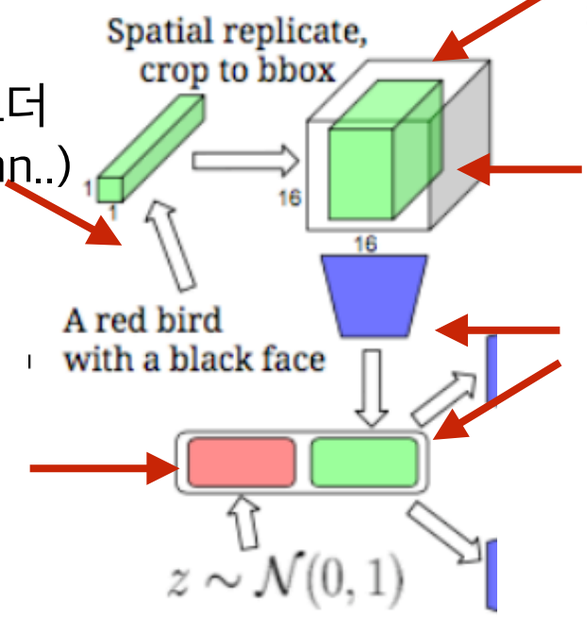
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator

바운딩 박스 정보와 크기를 맞추기 위해
spatial 하게 텍스트 임베딩 결과를 복제

사전 학습된 텍스트 인코더
(skip-thought, char-cnn..)

바운딩 박스 크기로 텍스트 임베딩 크롭
(바운딩 박스 외부는 0으로 채워짐)

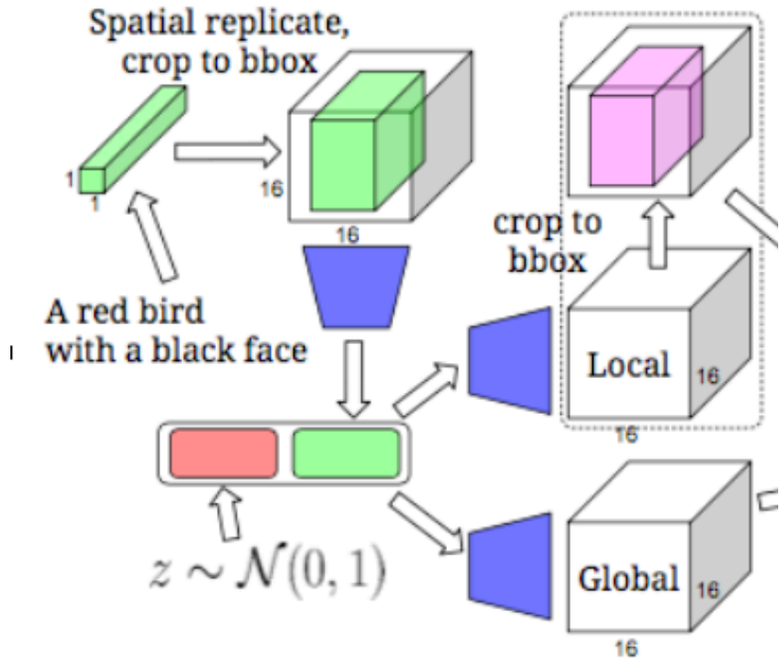


컨볼루션과 풀링을 통해 128x1 차원으로
만들고, 노이즈와 같이 concatenate 결합

100D 랜덤 노이즈

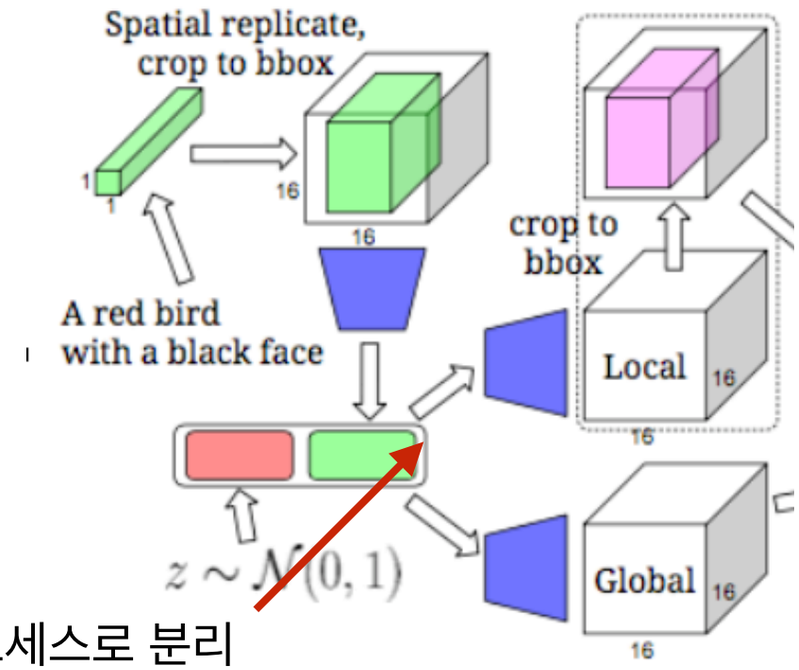
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

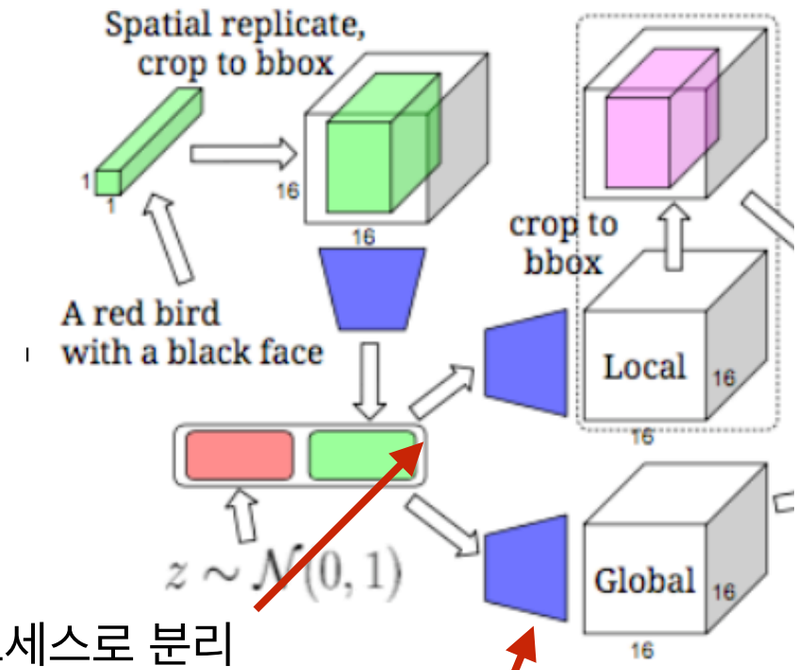
바운딩 박스 Generator



로컬/글로벌 프로세스로 분리

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator

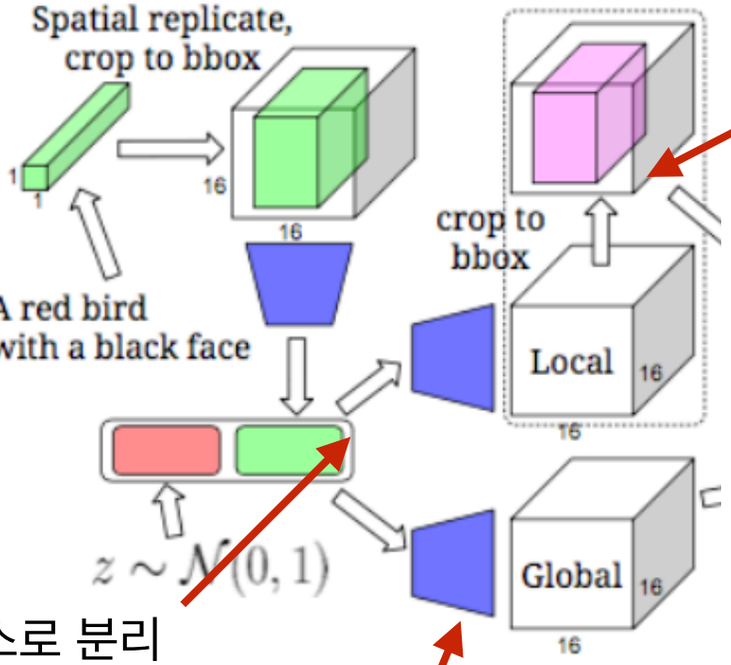


로컬/글로벌 프로세스로 분리

16x16으로 맞추기 위해 로컬/글로벌 프로세스
모두 transpose 컨볼루션 진행

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator



로컬 프로세스는 16x16 정보를 바운딩 박스에 의해 크롭

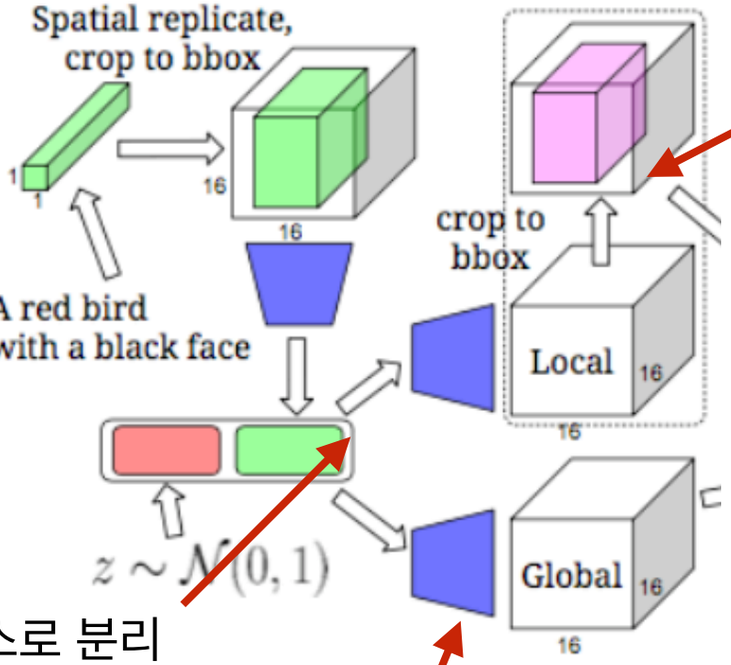
로컬/글로벌 프로세스로 분리

16x16으로 맞추기 위해 로컬/글로벌 프로세스 모두 transpose 컨볼루션 진행

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator

- 로컬 프로세스는 물체의 디테일한 부분을 초점을 맞추고,
- 글로벌 프로세스는 덜 중요한 배경 등을 생성하는 것으로 보임



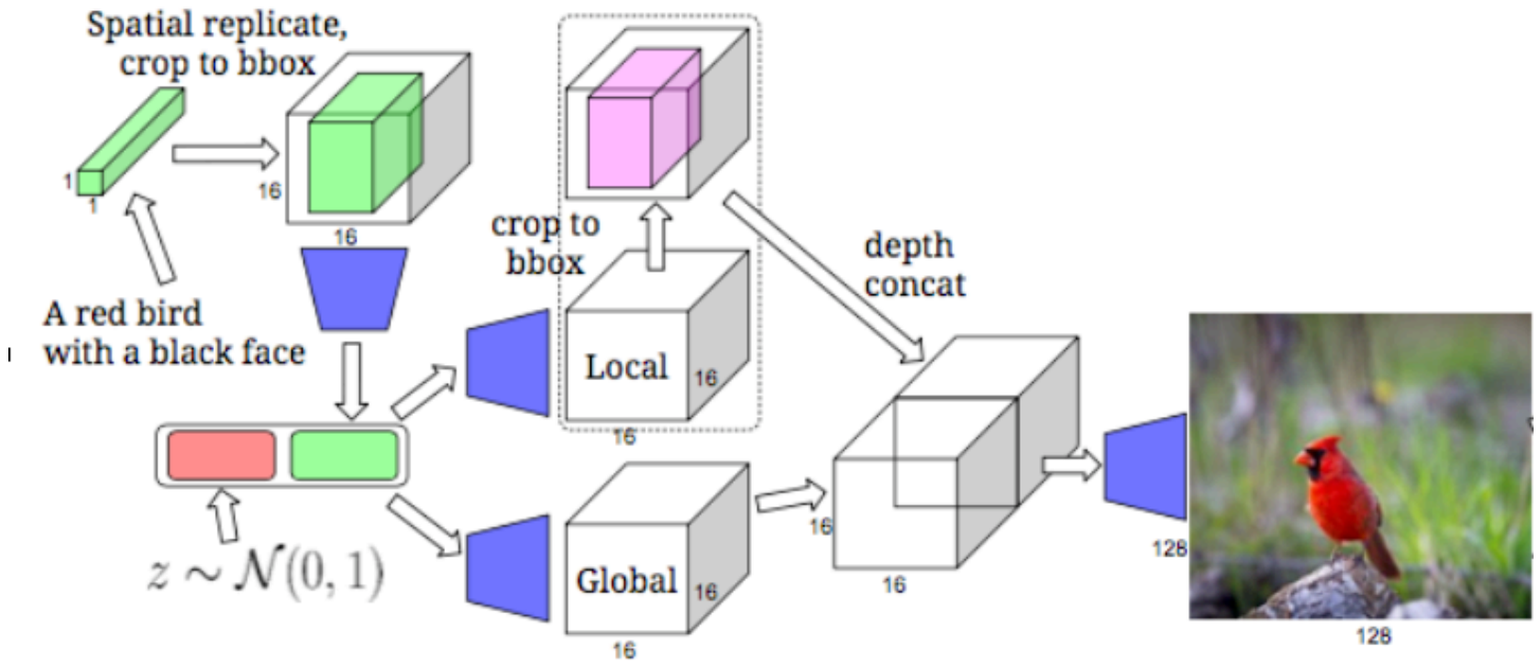
로컬 프로세스는 16x16 정보를 바운딩 박스에 의해 크롭

로컬/글로벌 프로세스로 분리

16x16으로 맞추기 위해 로컬/글로벌 프로세스 모두 transpose 컨볼루션 진행

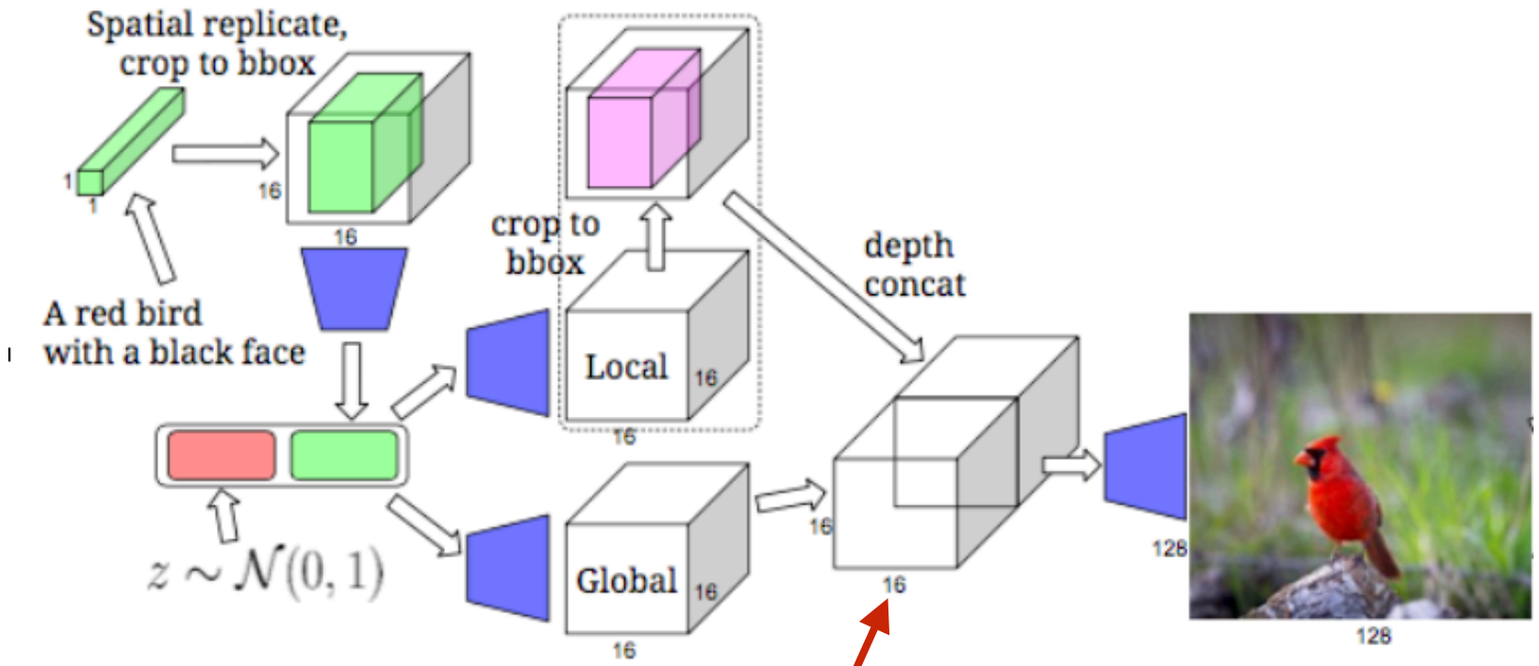
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

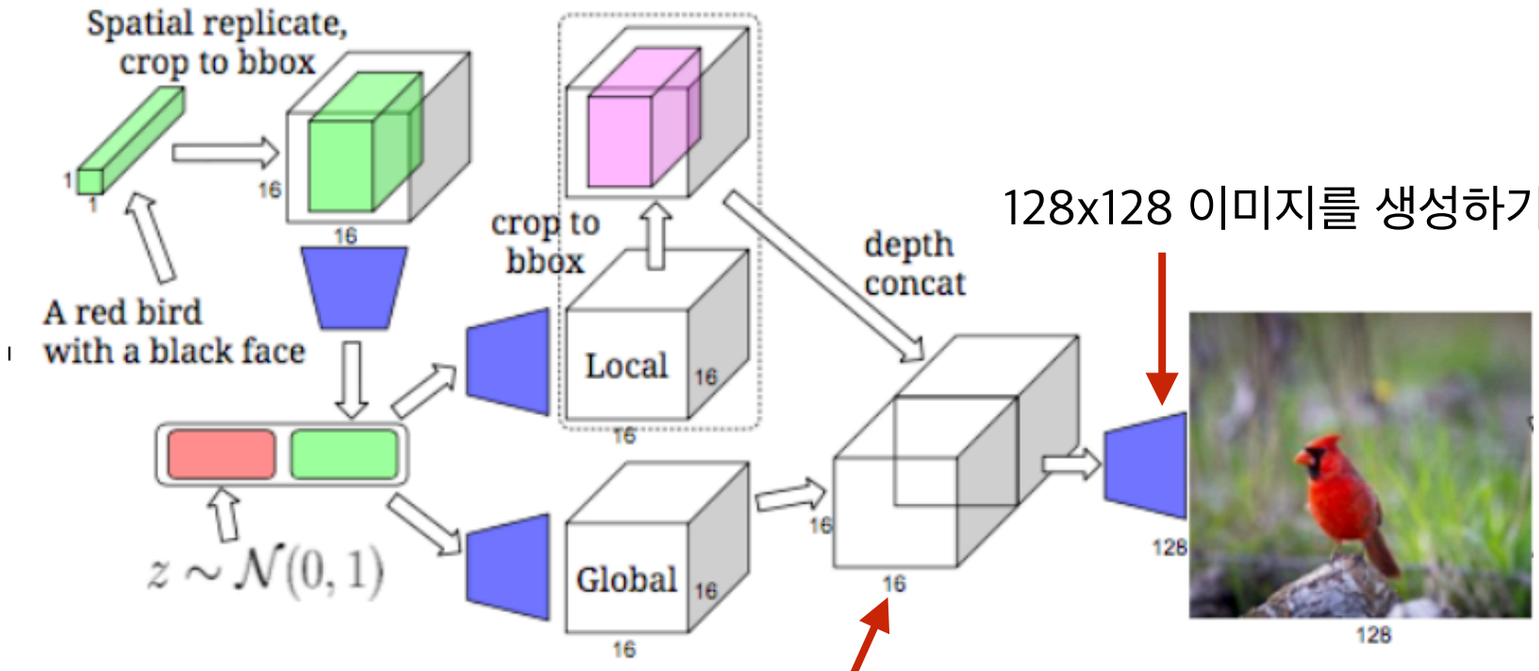
바운딩 박스 Generator



글로벌 프로세스의 결과와 크롭된 로컬 프로세스의 결과를 depth-wise concatenate로 합침

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Generator

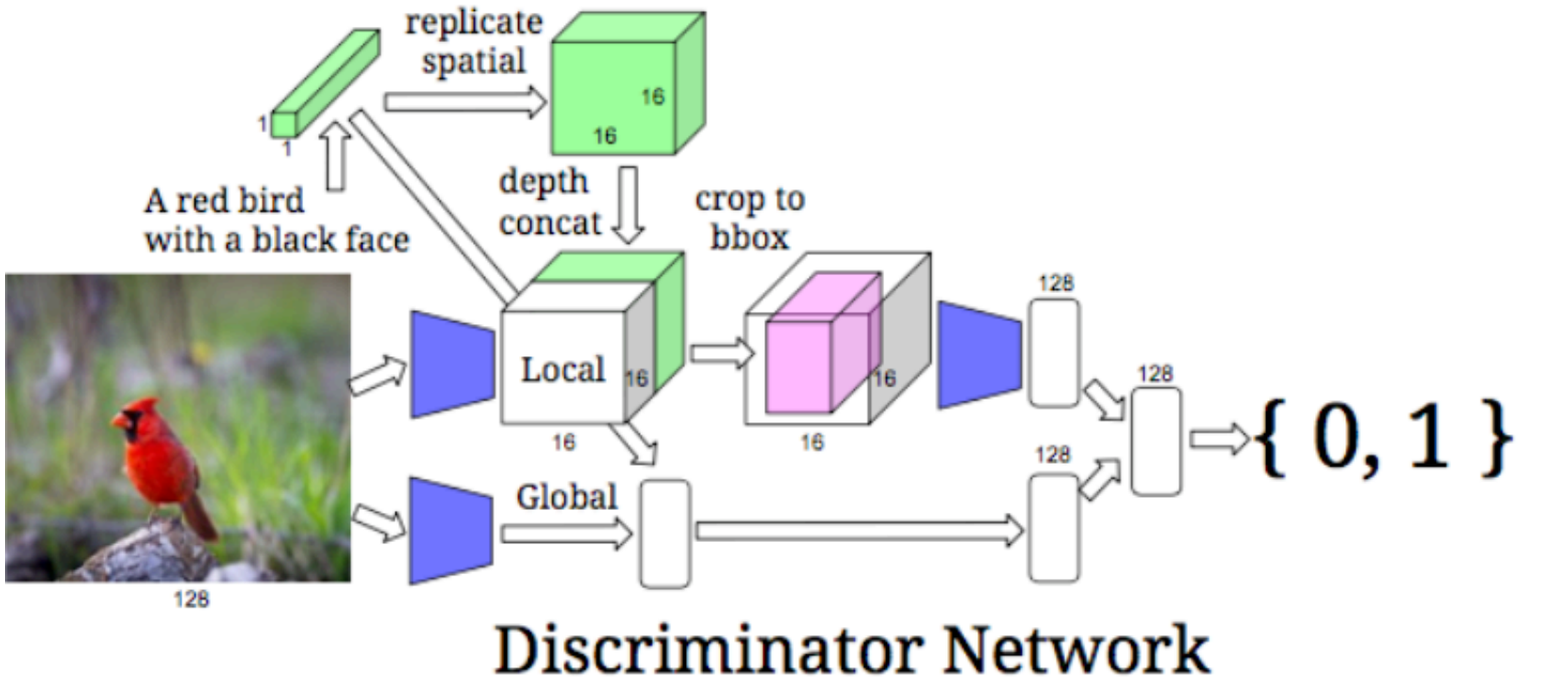


128x128 이미지를 생성하기 위한 단계

글로벌 프로세스의 결과와 크롭된 로컬 프로세스의 결과를 depth-wise concatenate로 합침

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Discriminator

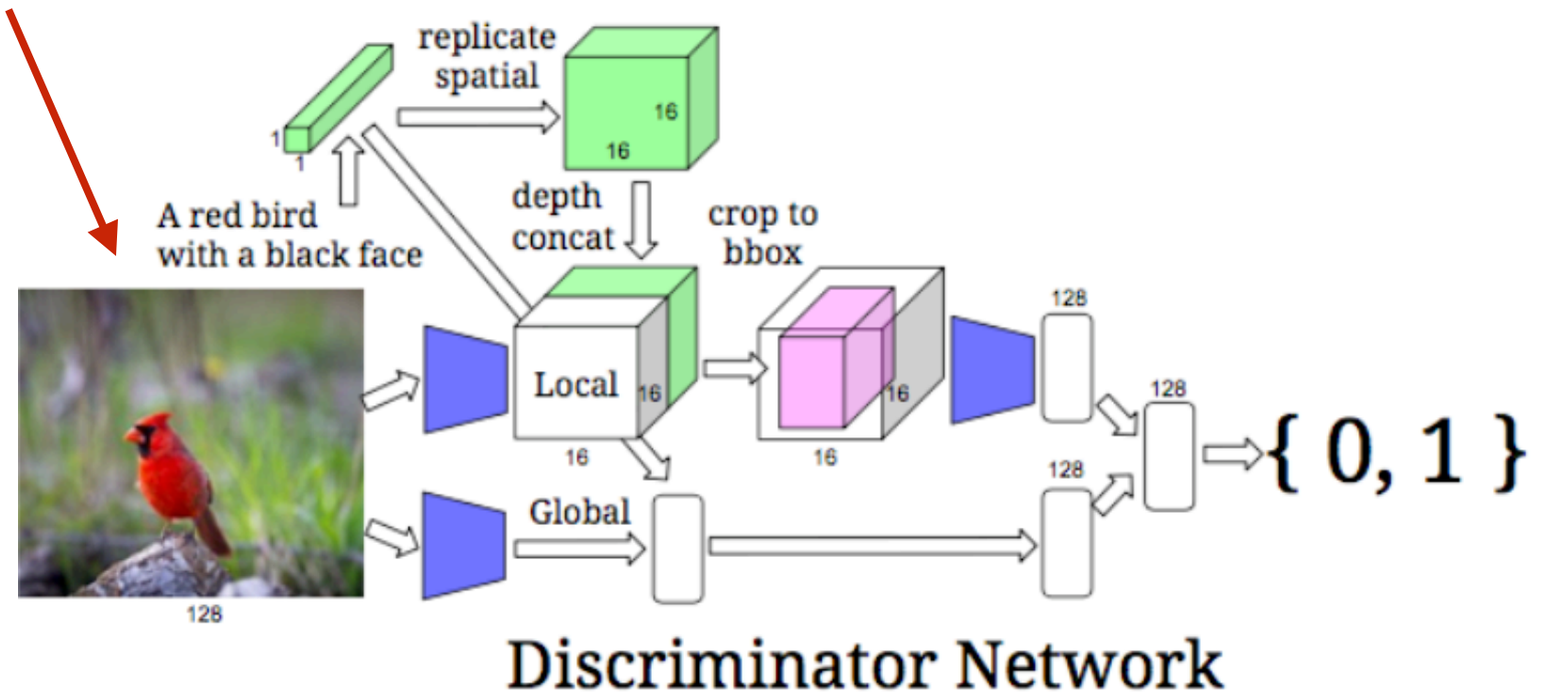


Discriminator Network

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Discriminator

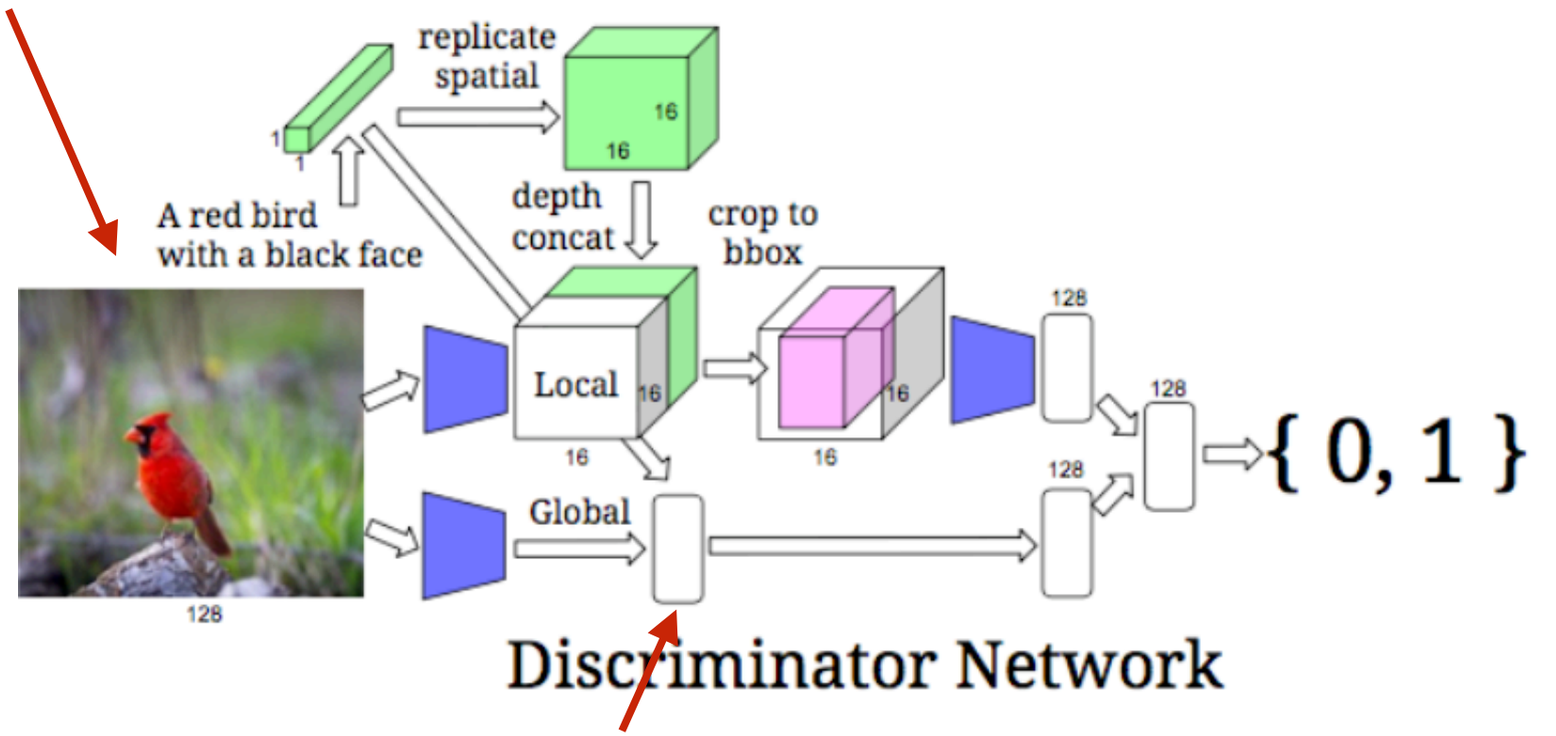
이미지와 연관된 캡션이 discriminator의 입력으로 들어감



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Discriminator

이미지와 연관된 캡션이 discriminator의 입력으로 들어감

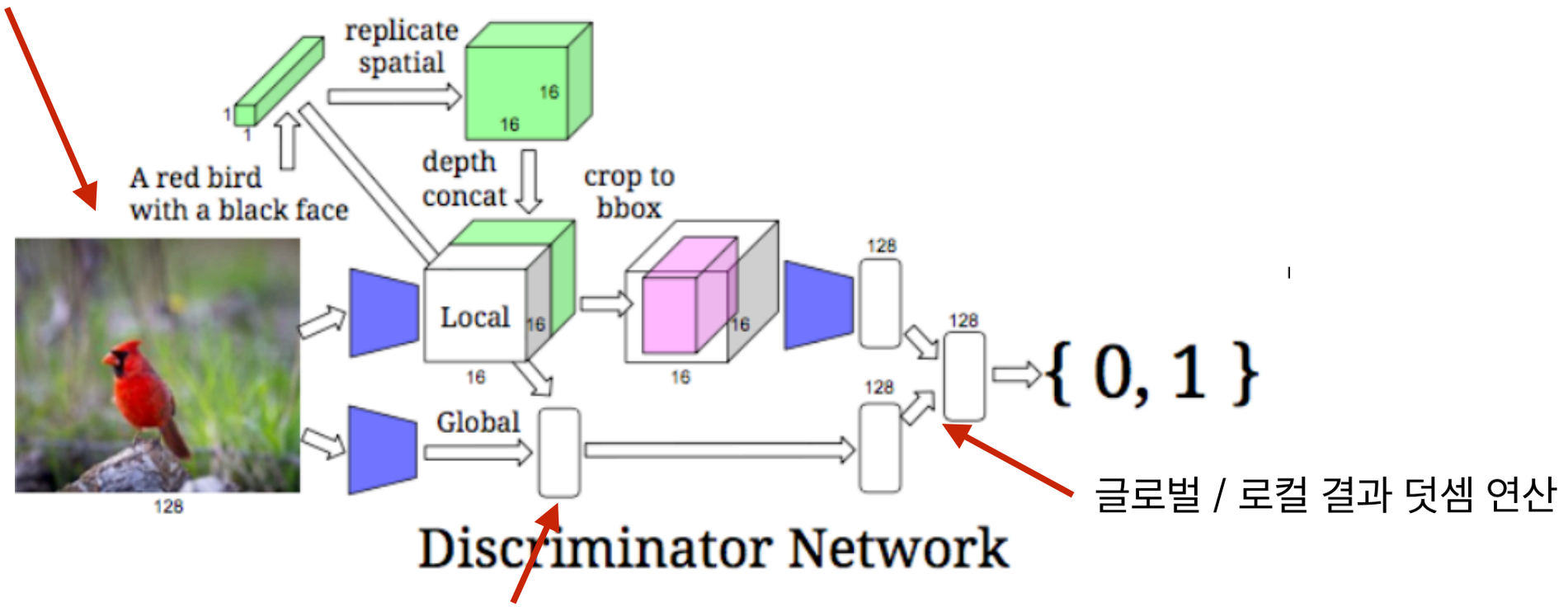


텍스트 임베딩과 글로벌 결과 덧셈 연산 (addition)

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 Discriminator

이미지와 연관된 캡션이 discriminator의 입력으로 들어감

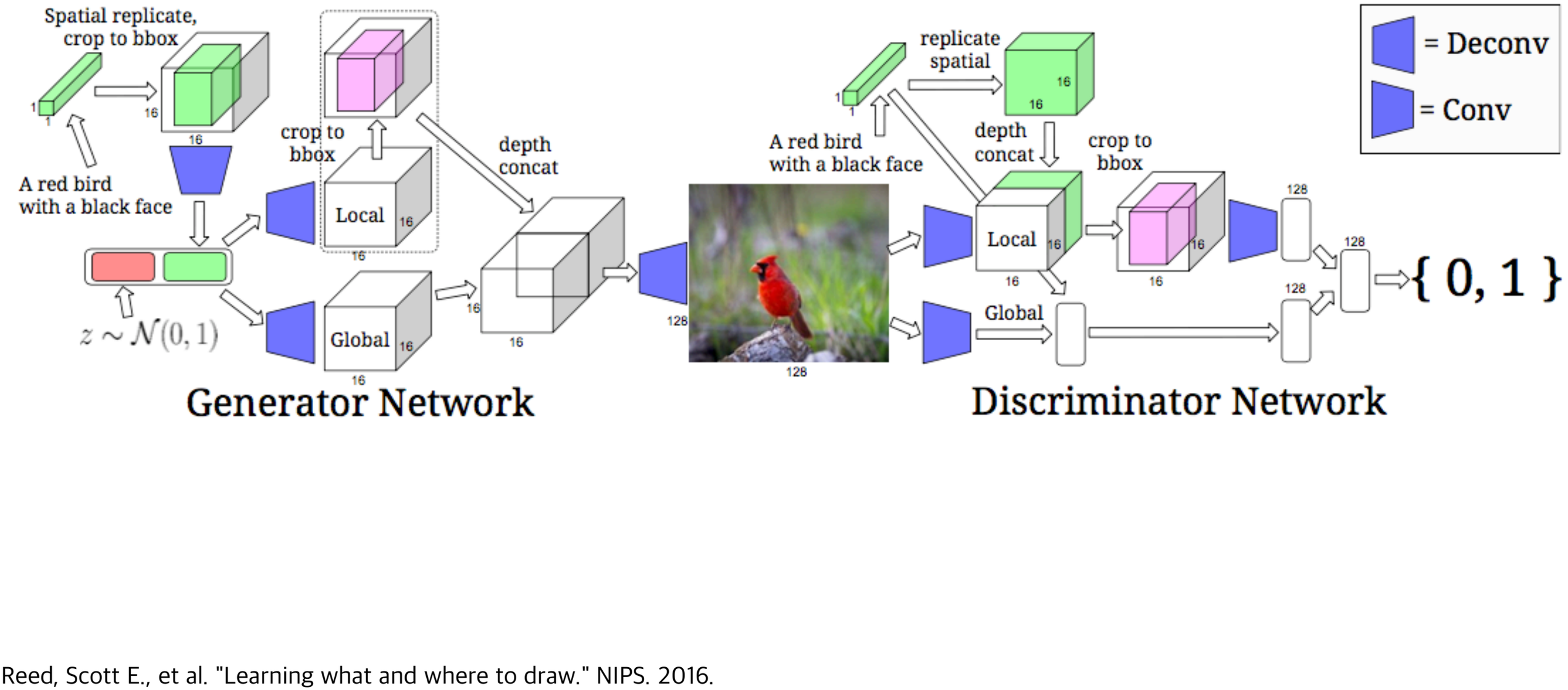


글로벌 / 로컬 결과 덧셈 연산

텍스트 임베딩과 글로벌 결과 덧셈 연산 (addition)

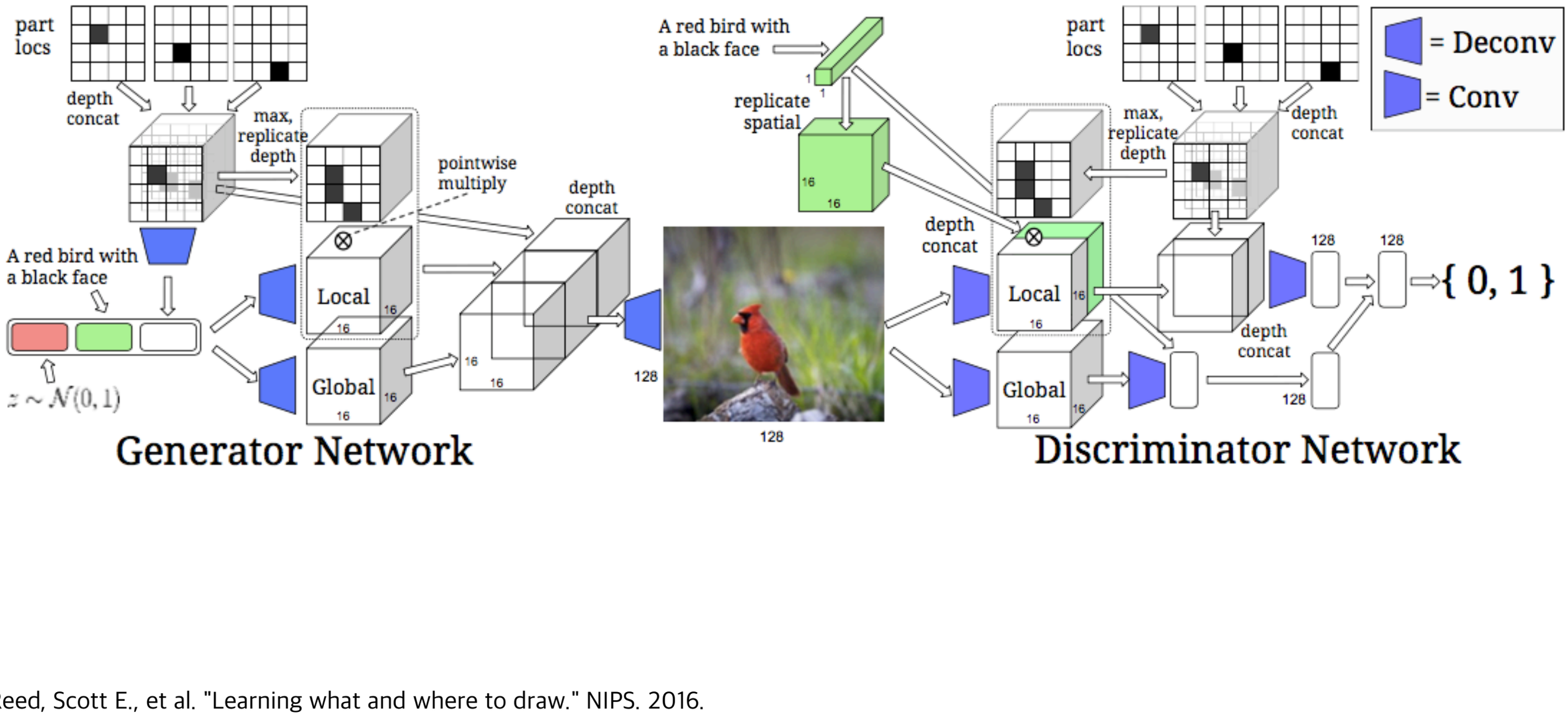
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

바운딩 박스 모델



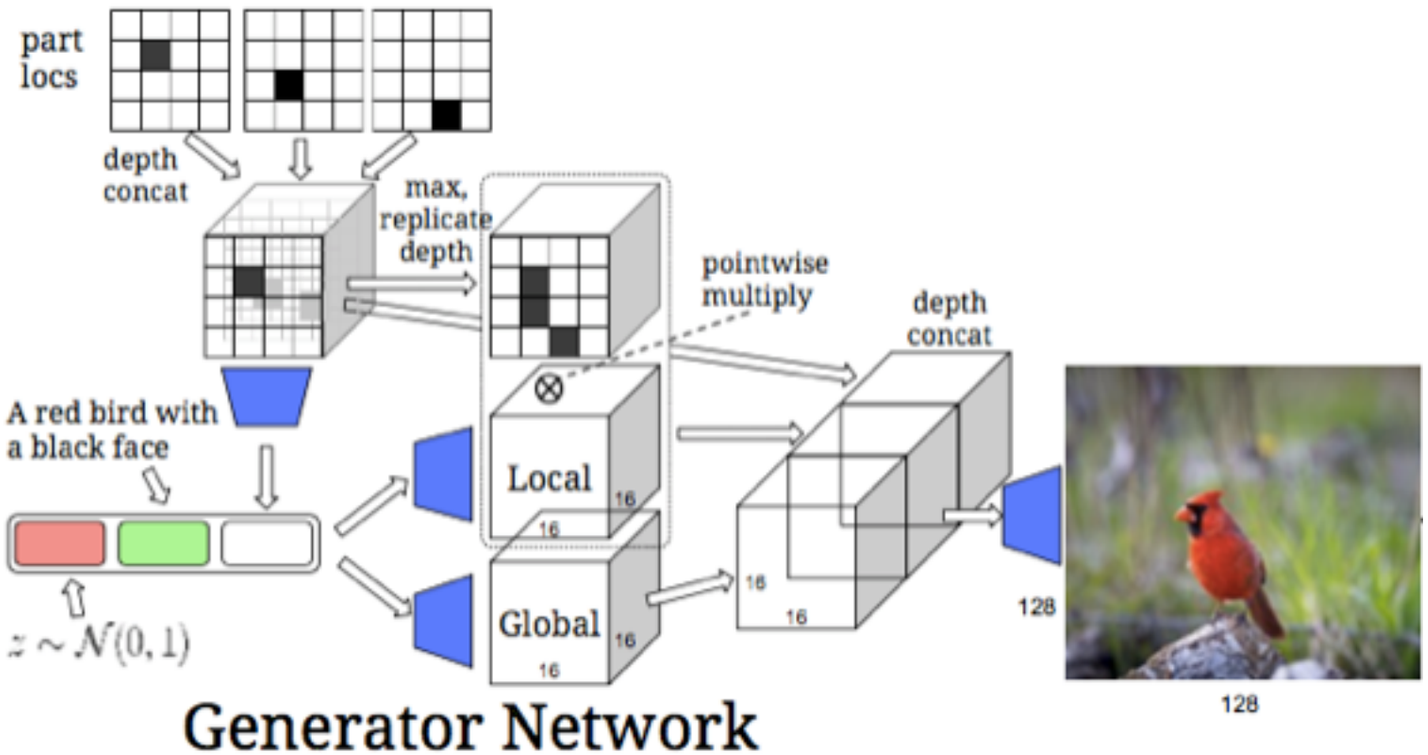
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 모델



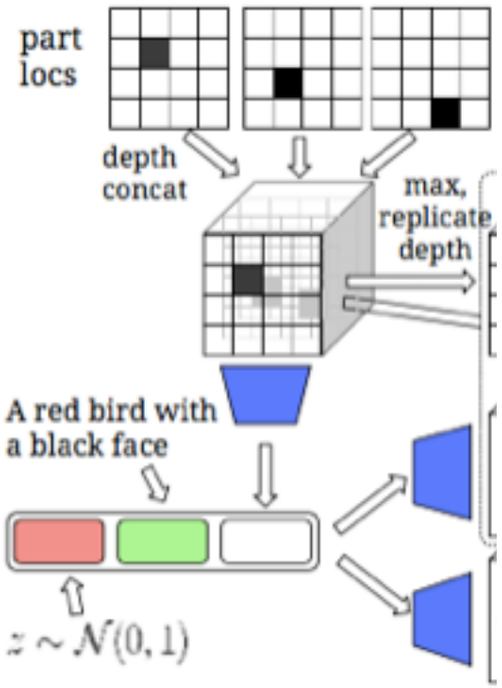
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Generator



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

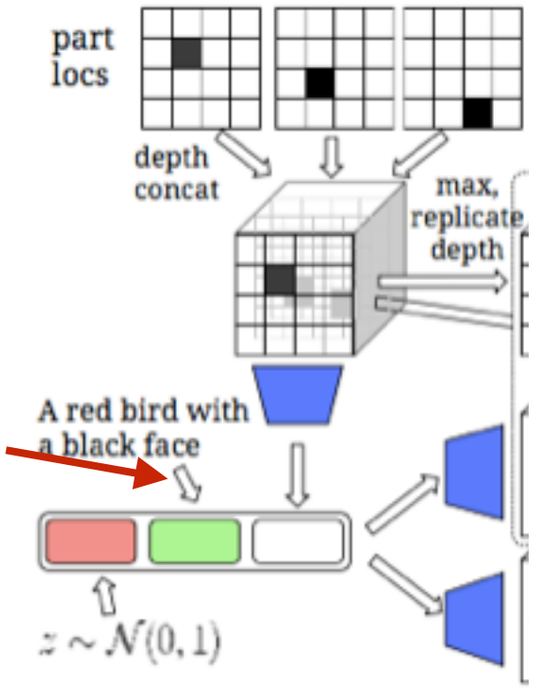
키포인트 Generator



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Generator

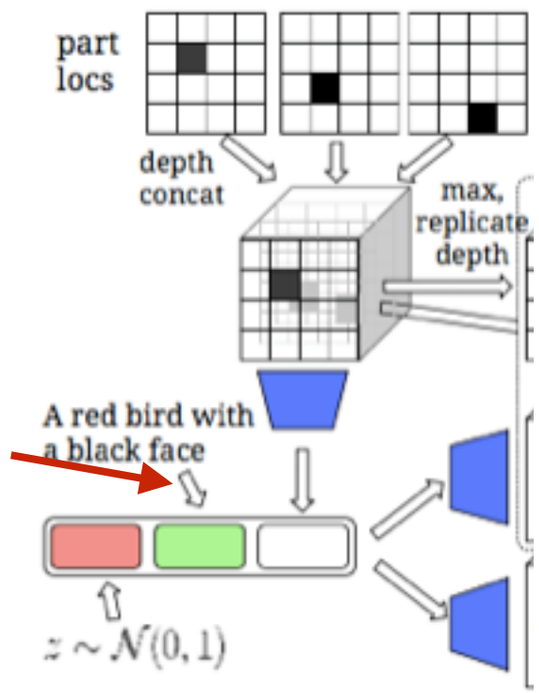
1024D로 텍스트 임베딩



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Generator

키포인트는 16x16xK 크기의 특징맵으로 인코딩 되어 있으며,
K는 모든 파트의 개수 (몸체=0, 날개=1 …)
(여기서는 간단하게 표현하기 위해 4x4로 그려짐)

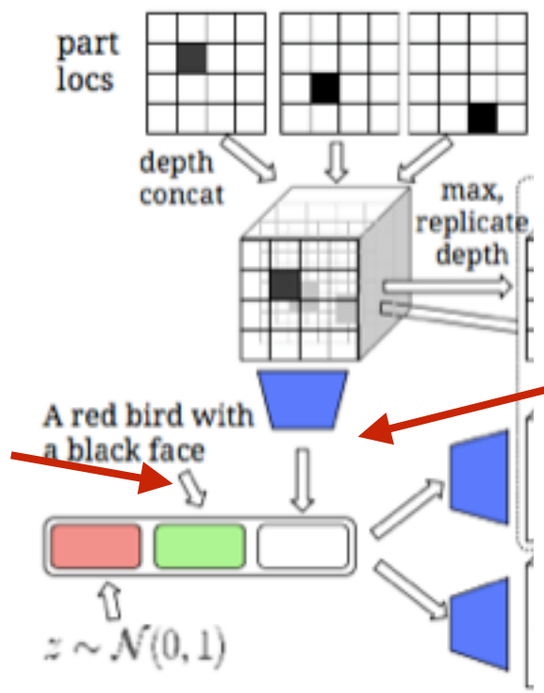


1024D로 텍스트 임베딩

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Generator

키포인트는 16x16xK 크기의 특징맵으로 인코딩 되어 있으며,
K는 모든 파트의 개수 (몸체=0, 날개=1 …)
(여기서는 간단하게 표현하기 위해 4x4로 그려짐)

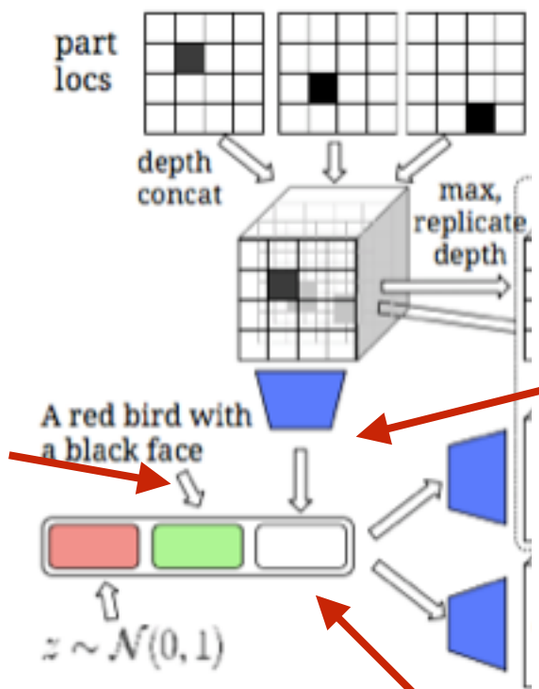


1024D로 텍스트 임베딩

특징맵 크기를 줄이기 위해 stride-2 컨볼루션을 진행하고,
노이즈와 텍스트 임베딩과 같이 결합

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Generator



키포인트는 16x16xK 크기의 특징맵으로 인코딩 되어 있으며,
K는 모든 파트의 개수 (몸체=0, 날개=1 …)
(여기서는 간단하게 표현하기 위해 4x4로 그려짐)

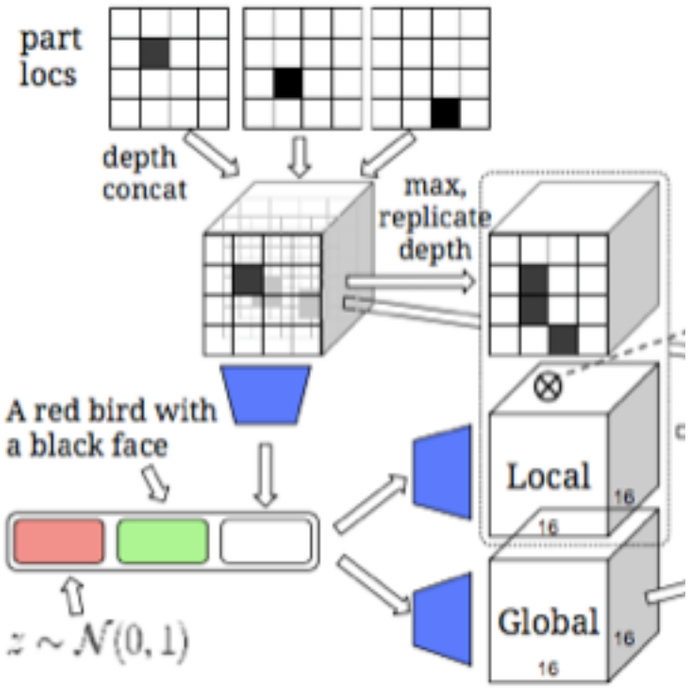
1024D로 텍스트 임베딩

특징맵 크기를 줄이기 위해 stride-2 컨볼루션을 진행하고,
노이즈와 텍스트 임베딩과 같이 결합

이 정보는 물체의 대략적인 정보와 파트에 대한
위치 정보를 담고 있게 됨

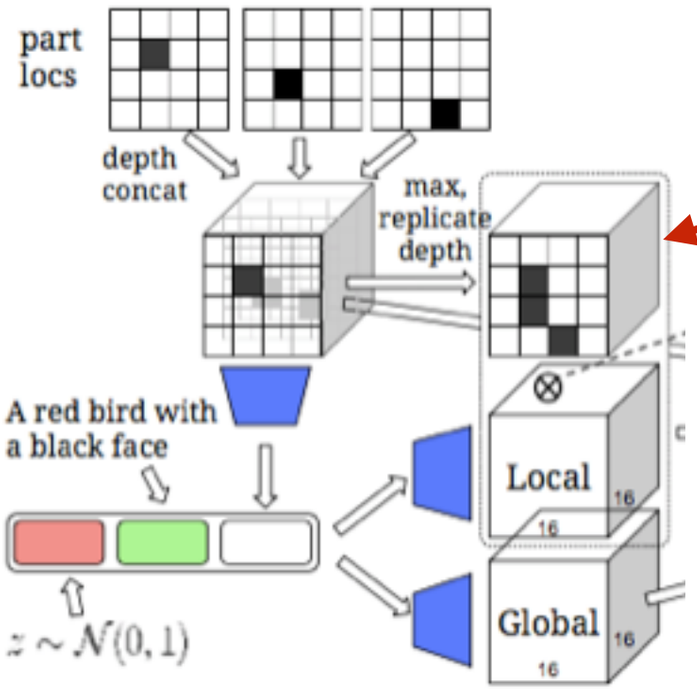
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Generator



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

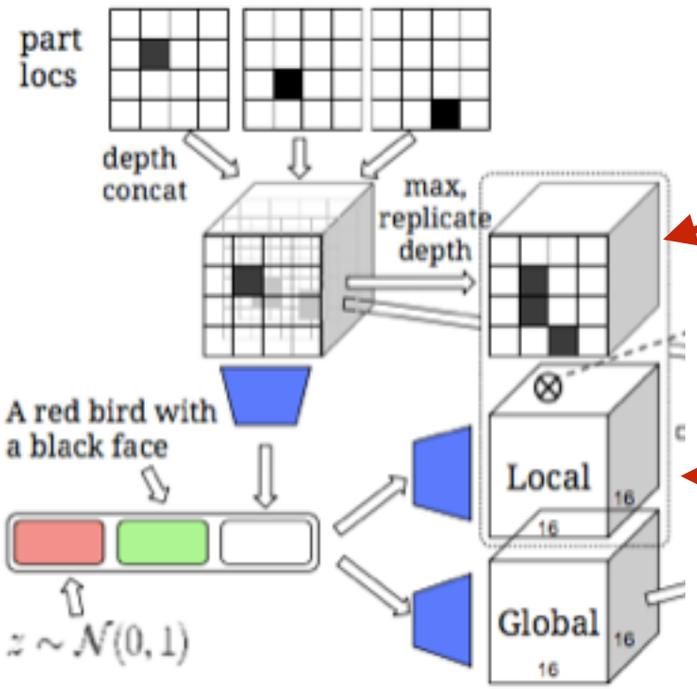
키포인트 Generator



MAX 연산에 의해 키포인트 특징들을 16x16x1로 만들고 다음 진행을 위해 depth-wise로 값들을 복사
이 특징맵에서 값이 있는 부분은 파트의 유무를 나타냄

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Generator

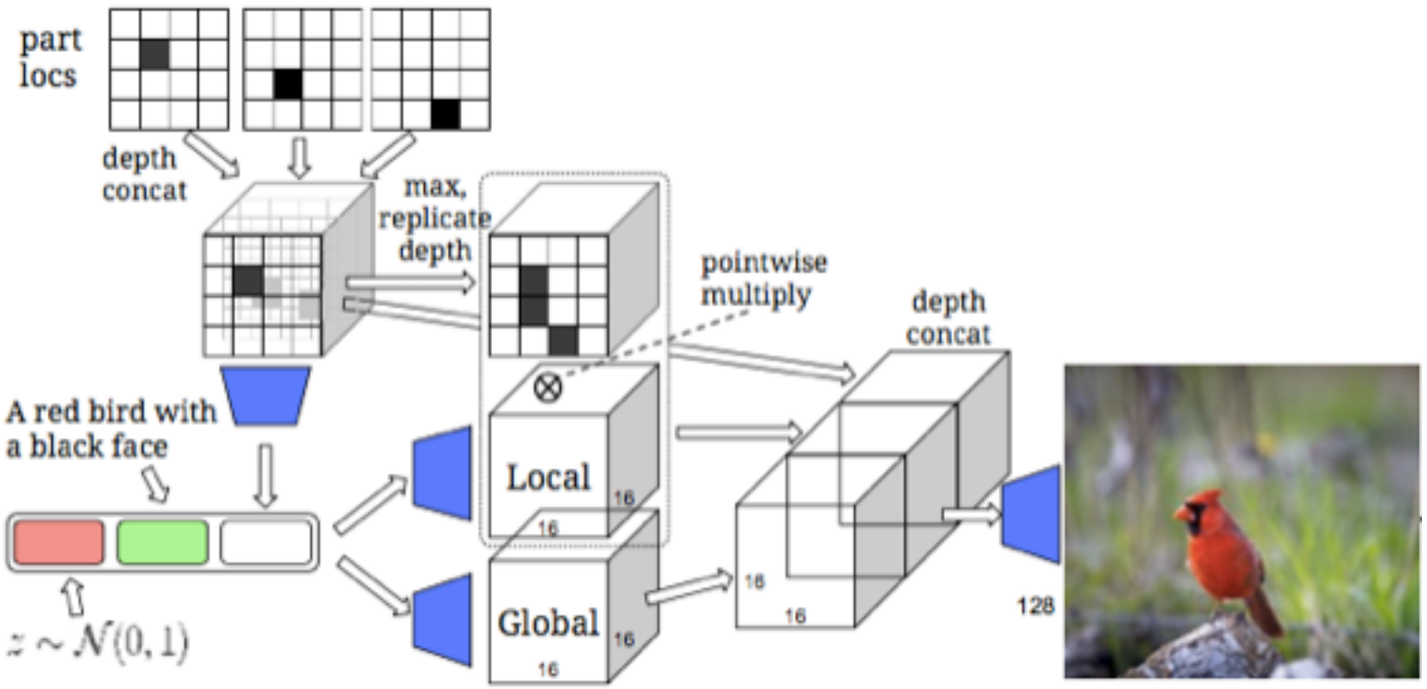


MAX 연산에 의해 키포인트 특징들을 16x16x1로 만들고 다음 진행을 위해 depth-wise로 값들을 복사
이 특징맵에서 값이 있는 부분은 파트의 유무를 나타냄

로컬 프로세스의 결과값은 키포인트 정보와 곱셈 연산 수행
(파트가 있는 영역만 강조하기 위해서)

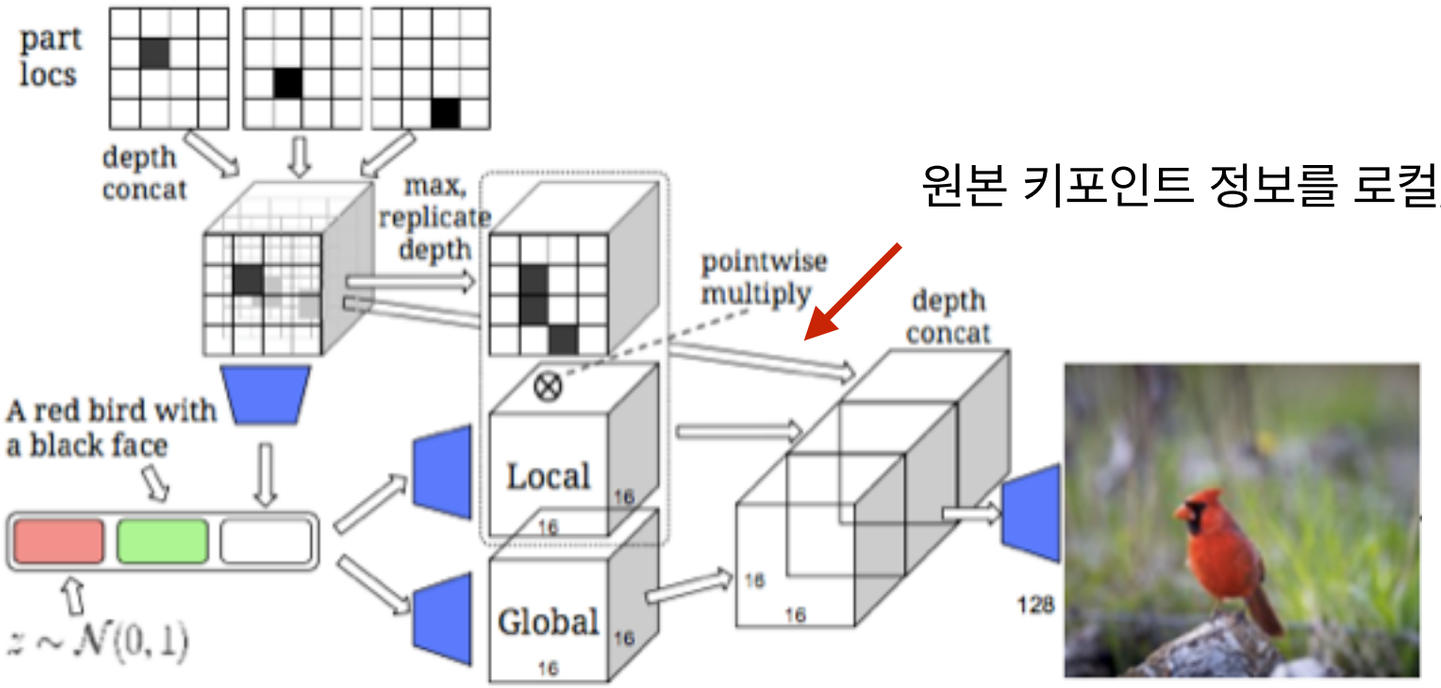
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Generator



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

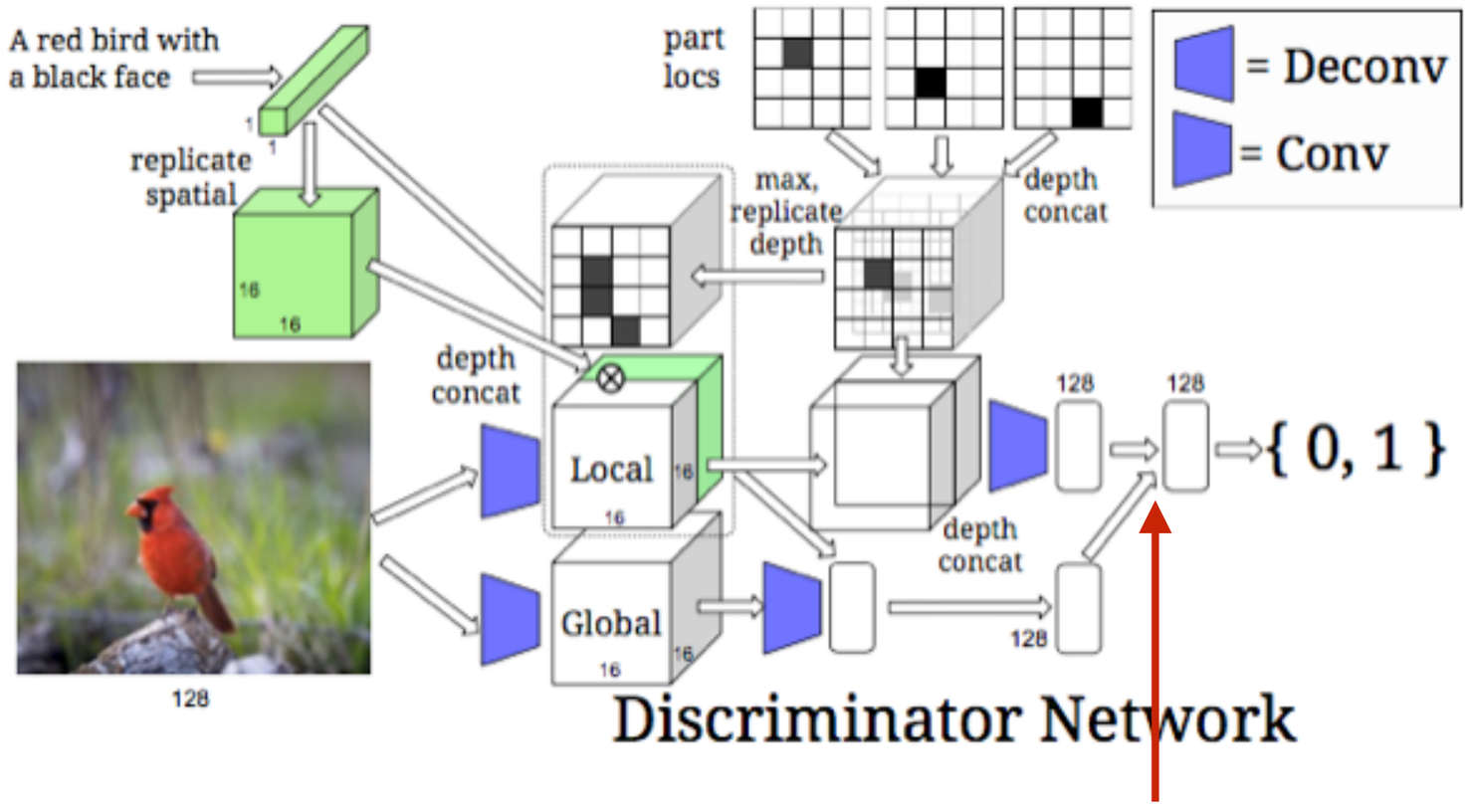
키포인트 Generator



원본 키포인트 정보를 로컬/글로벌 정보와 합침

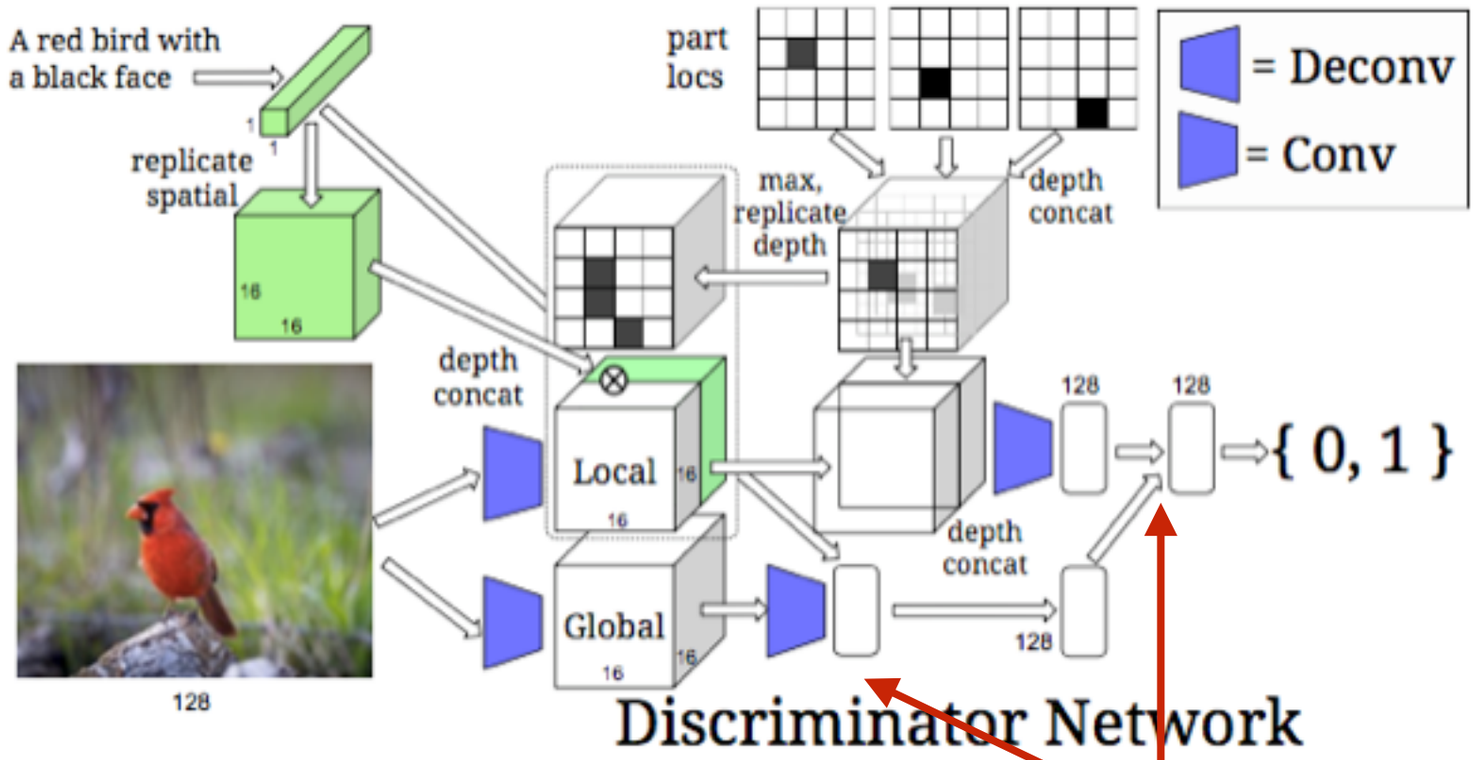
Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Discriminator



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 Discriminator



Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

키포인트 생성 모델

- 별도의 GAN을 통해서 관측되지 않은 키포인트 정보를 생성
  - **k:** 키포인트
  - **s:** 키포인트 파트가 존재하면 1인 스위치 변수
  - **f:** 3-레이어 뉴럴 네트워크

$$G_k(z, \mathbf{t}, \mathbf{k}, \mathbf{s}) := \mathbf{s} \odot \mathbf{k} + (1 - \mathbf{s}) \odot f(z, \mathbf{t}, \mathbf{k})$$

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

Figure 4: Controlling the bird's position using bounding box coordinates. and previously-unseen text.

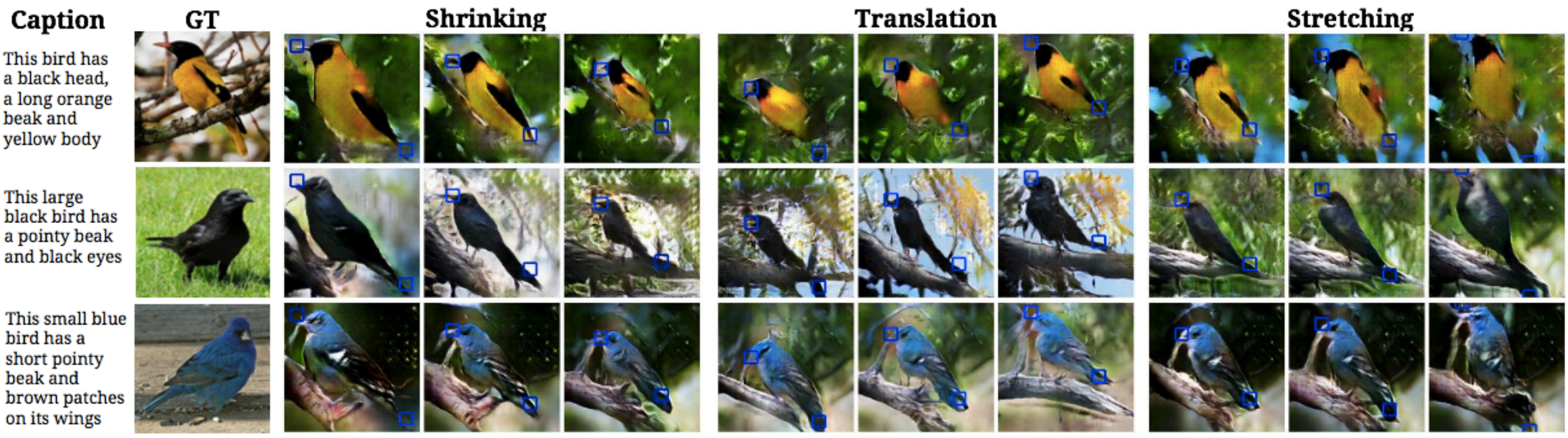


Figure 6: Controlling the bird's position using keypoint coordinates. Here we only interpolated the beak and tail positions, and sampled the rest conditioned on these two.

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

Figure 7: Keypoint- and text-conditional bird generation in which the keypoints are generated conditioned on unseen text. The small blue boxes indicate the generated keypoint locations.

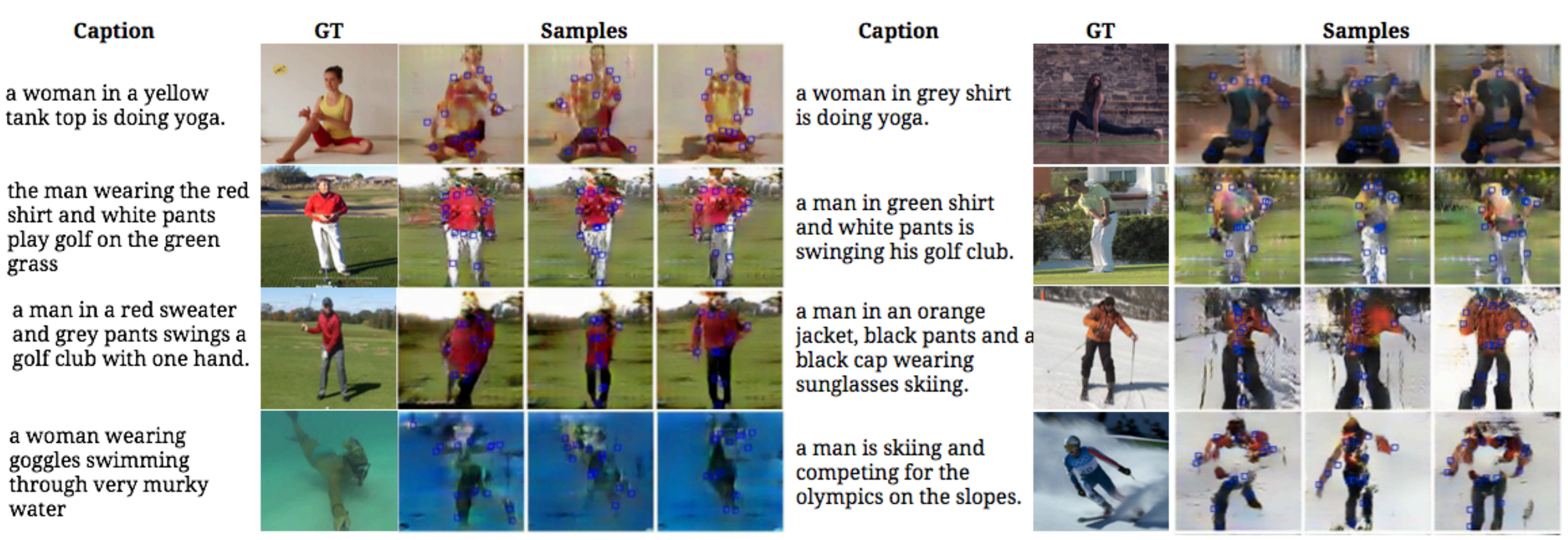


Figure 9: Generating humans. Both the keypoints and the image are generated from unseen text.

Reed, Scott E., et al. "Learning what and where to draw." NIPS. 2016.

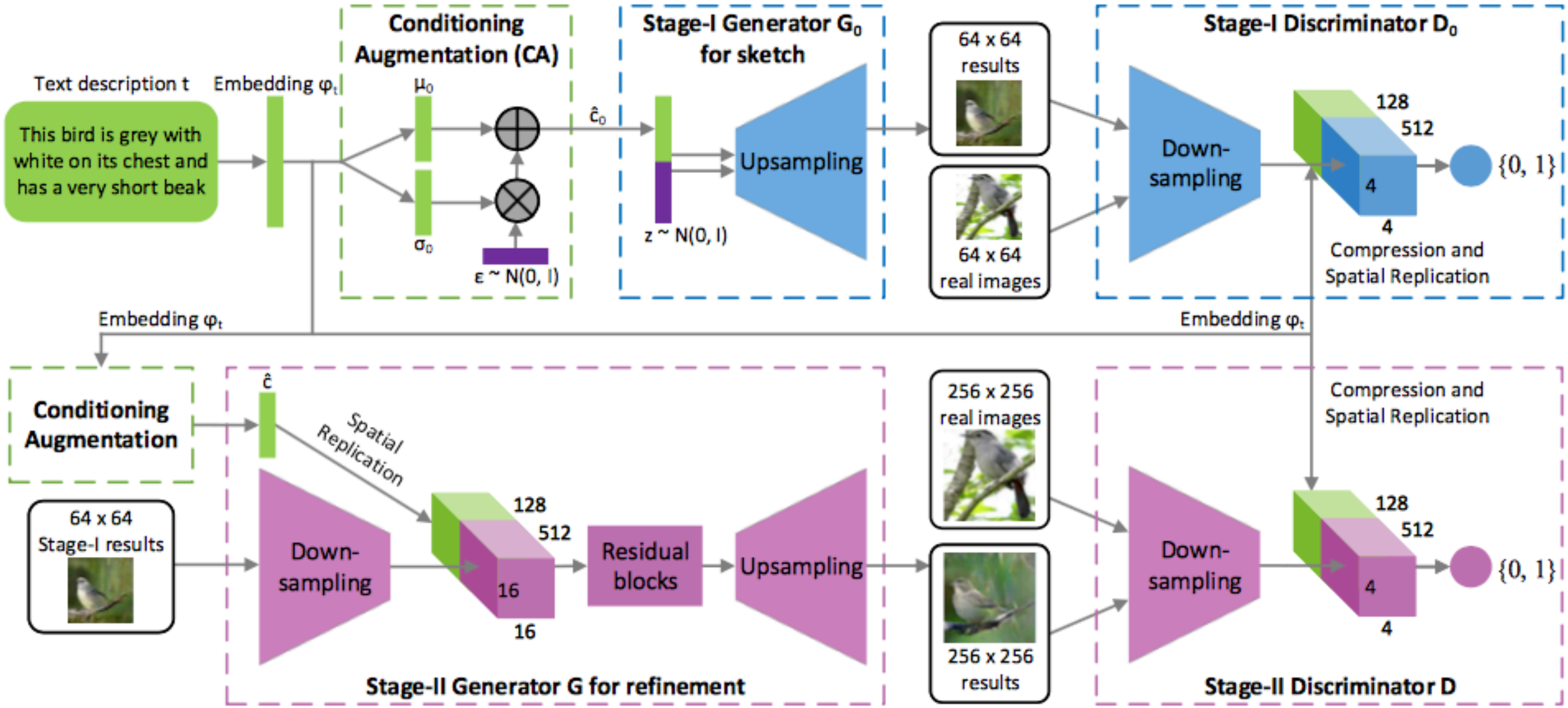
StackGAN

- GAN-CLS 방법은 고해상도 이미지를 잘 생성하지 못함 (최대 128x128)

- StackGAN
  - 한 번에 고해상도 이미지를 생성하지 않고, **여러 단계에 걸쳐 생성**
  - 첫번째 generator는 이미지의 전체적인 구조를 생성하고,
  - 두번째 generator는 고해상도의 디테일한 이미지를 생성

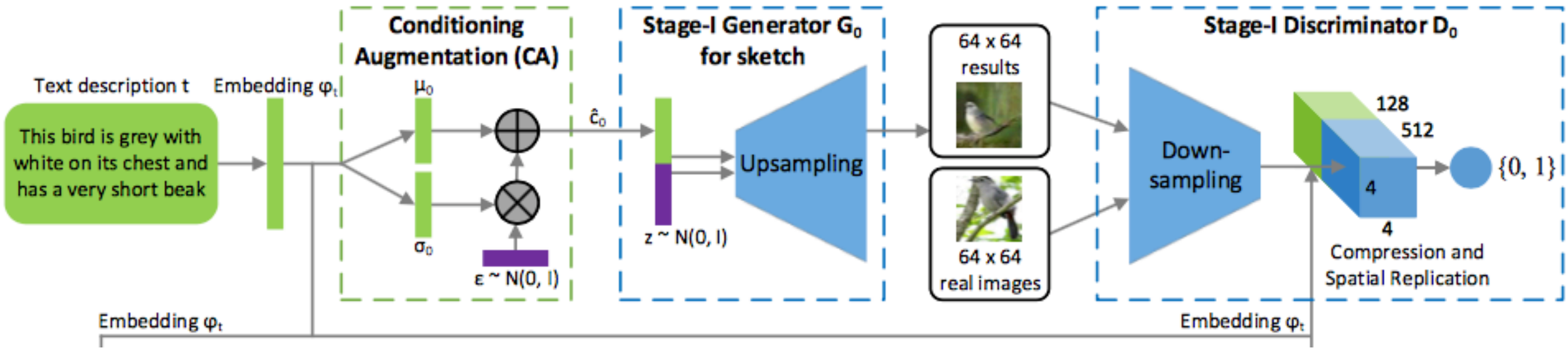
Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV. 2017.

StackGAN



Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV. 2017.

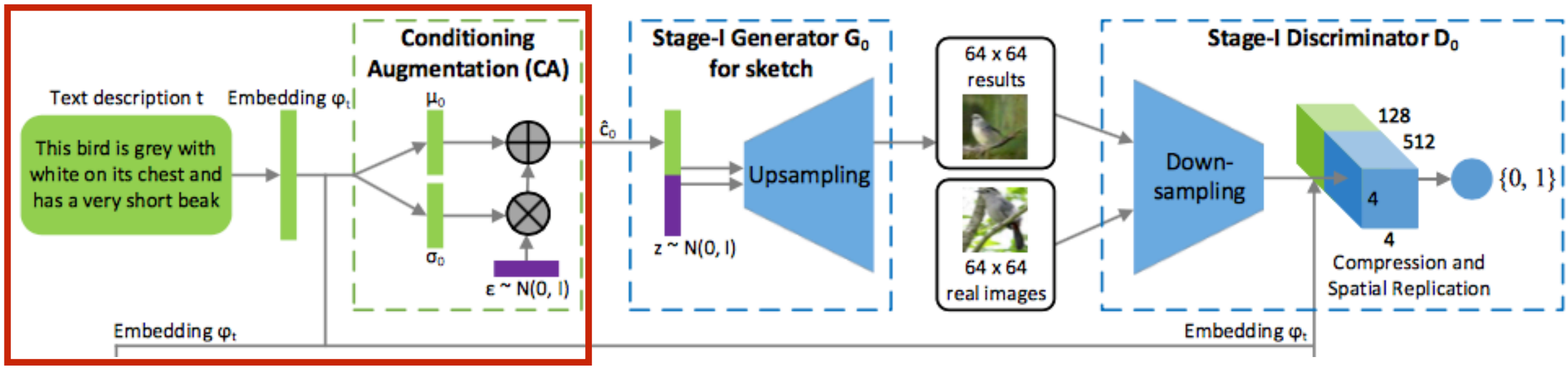
StackGAN – stage 1



Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV. 2017.

StackGAN – stage 1

텍스트 임베딩 모듈

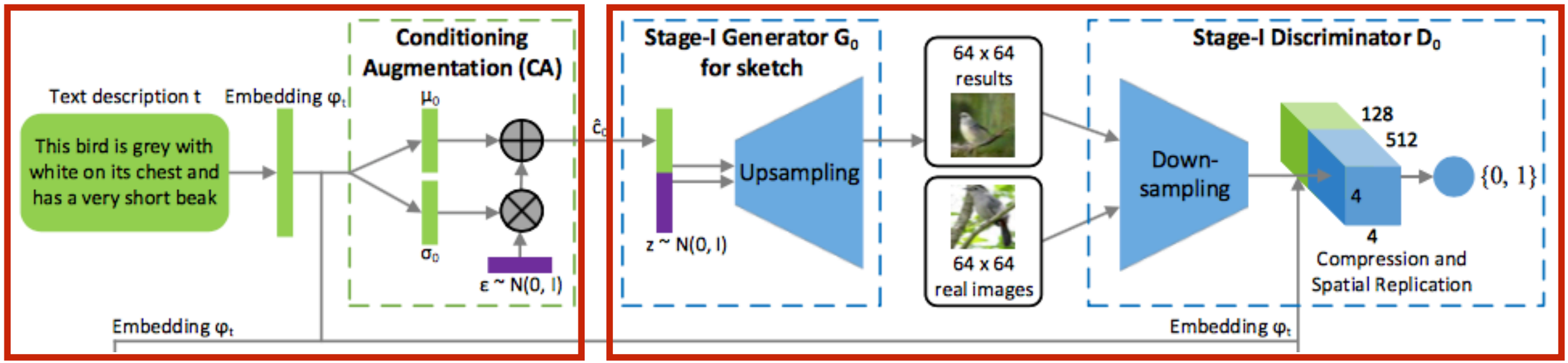


Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV. 2017.

StackGAN – stage 1

텍스트 임베딩 모듈

저해상도 이미지 생성 (GAN-CLS와 동일)

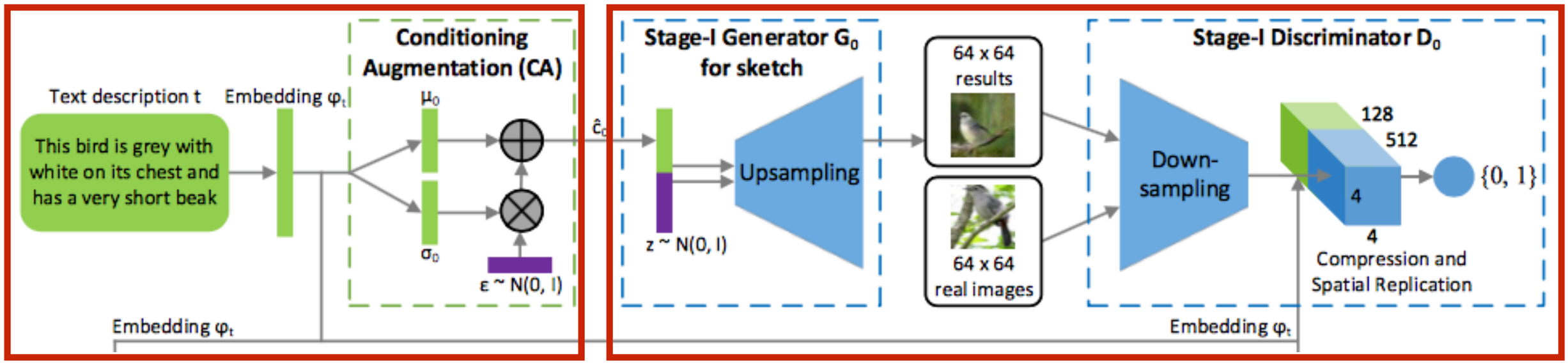


Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV. 2017.

StackGAN - stage 1

텍스트 임베딩 모듈

저해상도 이미지 생성 (GAN-CLS와 동일)

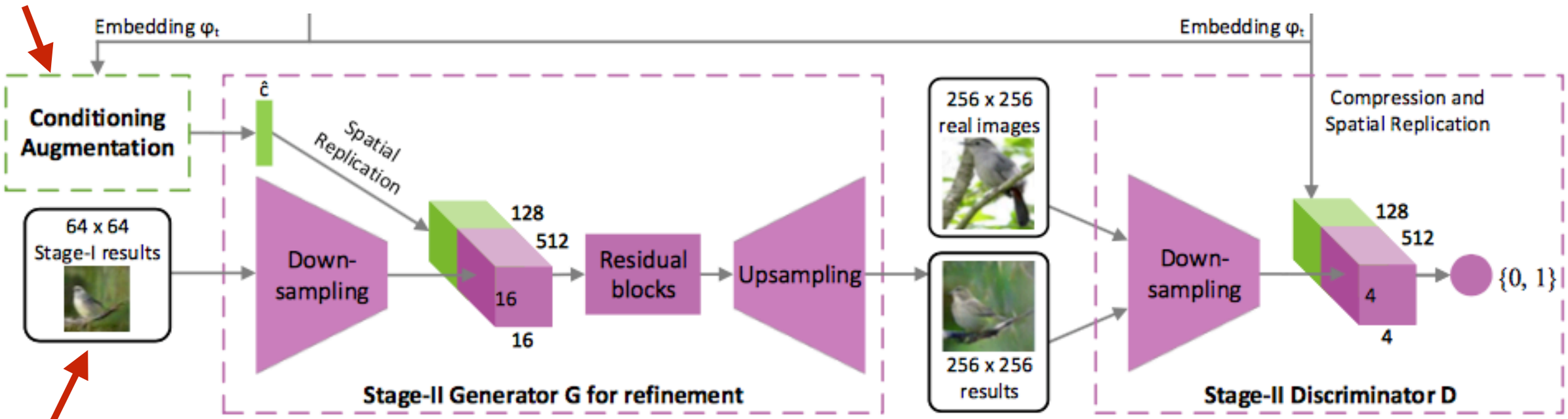


여기선 다루지 않지만,
텍스트 임베딩의 경우 VAE의 parameterization trick을 응용한
Conditioning Augmentation을 사용하면 성능 향상

Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV. 2017.

StackGAN – stage 2

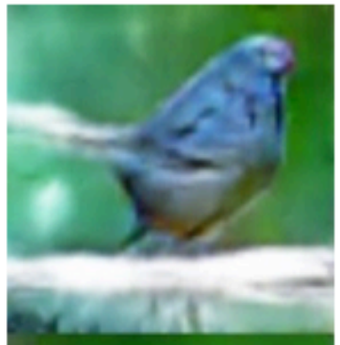
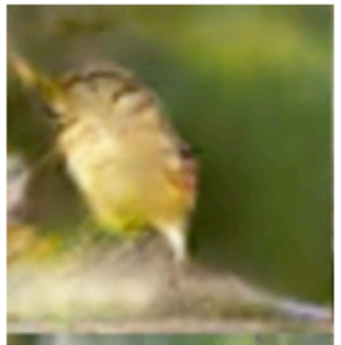
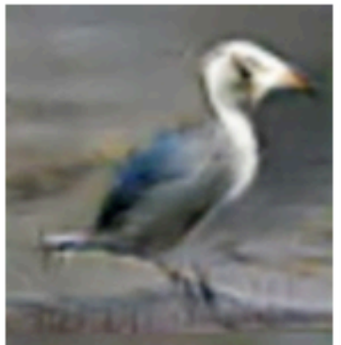
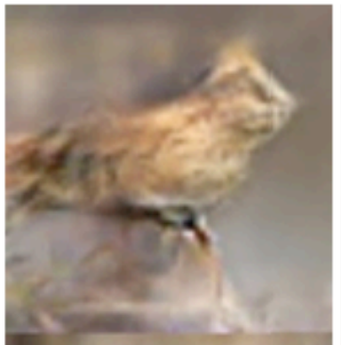
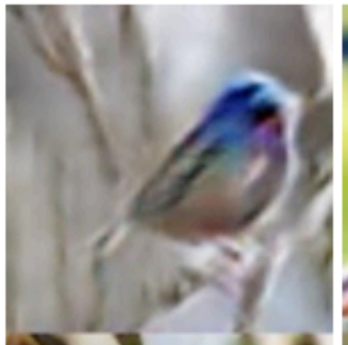
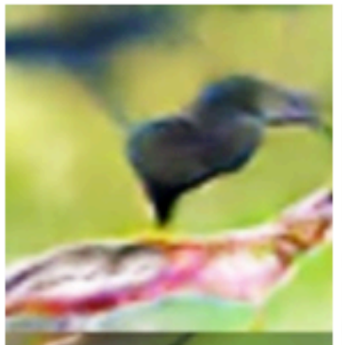
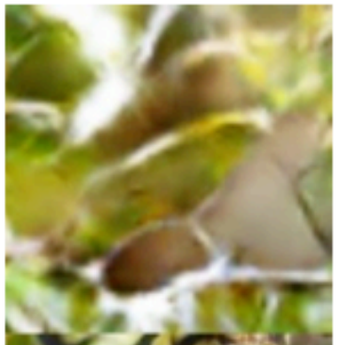
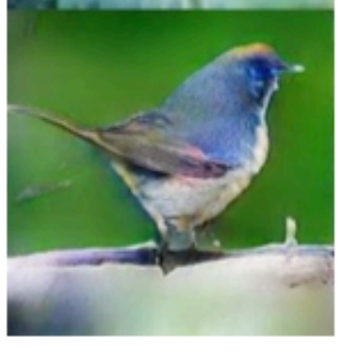
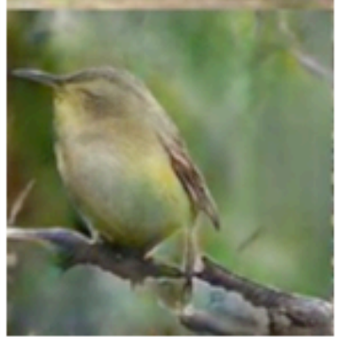
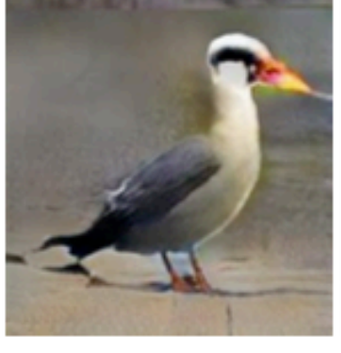
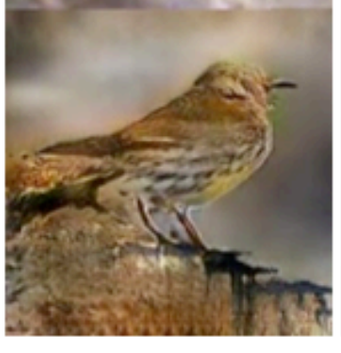
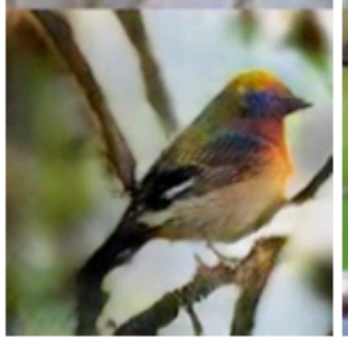
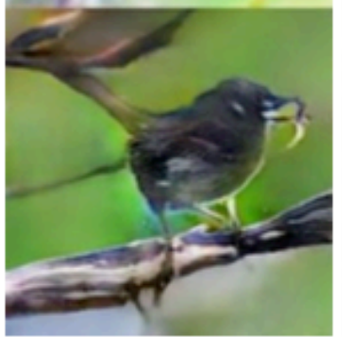
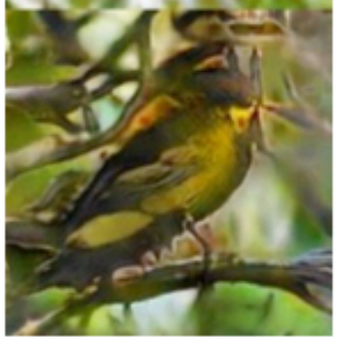
텍스트 임베딩 모듈



미리 생성된
저해상도 이미지

Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV. 2017.

결과

Text description	This bird is blue with white and has a very short beak	This bird has wings that are brown and has a yellow belly	A white bird with a black crown and yellow beak	This bird is white, black, and brown in color, with a brown beak	The bird has small beak, with reddish brown crown and gray belly	This is a small, black bird with a white breast and white on the wingbars.	This bird is white black and yellow in color, with a short black beak
Stage-I images							
Stage-II images							

Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV. 2017.

결과

A flower with small pink petals and a massive central orange and black stamen cluster

Stage-I images

Stage-II images

This flower has white petals with a yellow tip and a yellow pistil

Stage-I images

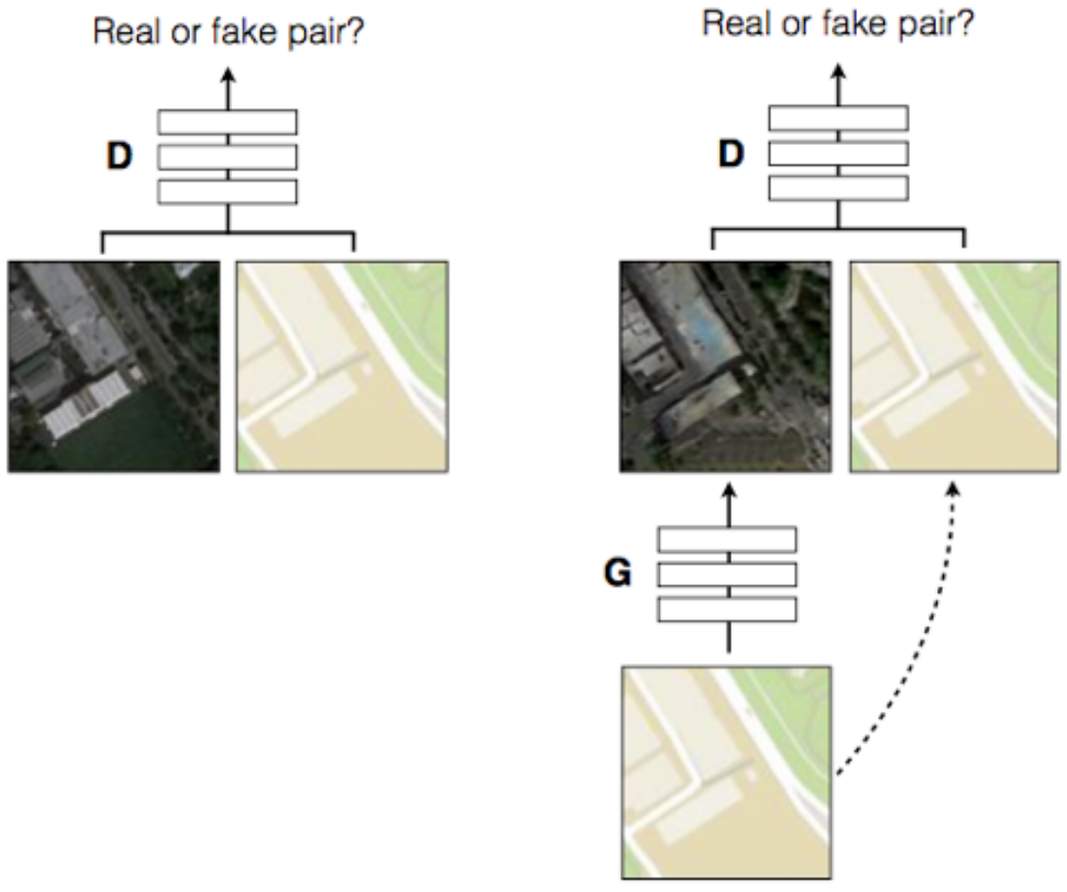
Stage-II images

Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV. 2017.

Image-to-image 변환

pix2pix

- 이미지를 다른 도메인의 이미지로 **변환**하는 알고리즘
- 이를 위해 두 도메인 간의 이미지를 **서로 짝을 지어놓은 데이터**가 필요
- 예) 위성 이미지 - 지도 이미지
- GAN을 통해 해결



Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." CVPR. 2017.

Loss 함수

- Conditional GAN loss 함수

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

- L1 loss 함수 $\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y\sim p_{data}(x,y),z\sim p_z(z)}[\|y - G(x,z)\|_1].$
  - L2 함수는 이미지가 뿌옇게 보이는 현상이 더 심하게 발생

- 최종 Loss는 GAN loss와 L1 loss의 결합

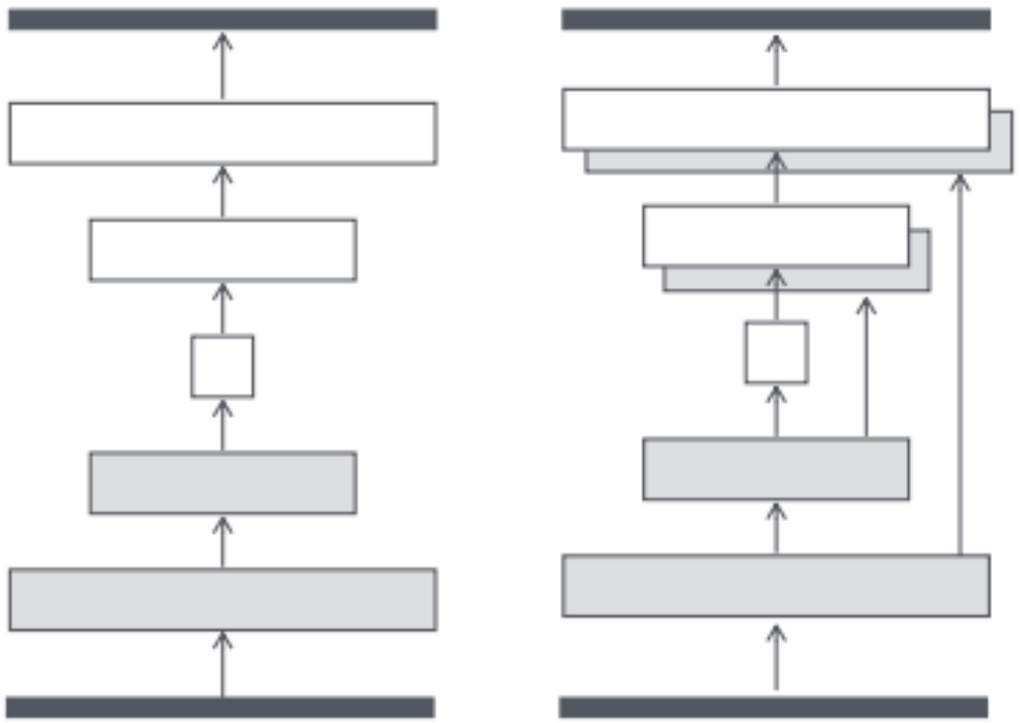
$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda \mathcal{L}_{L1}(G).$$

Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." CVPR. 2017.

네트워크 아키텍쳐

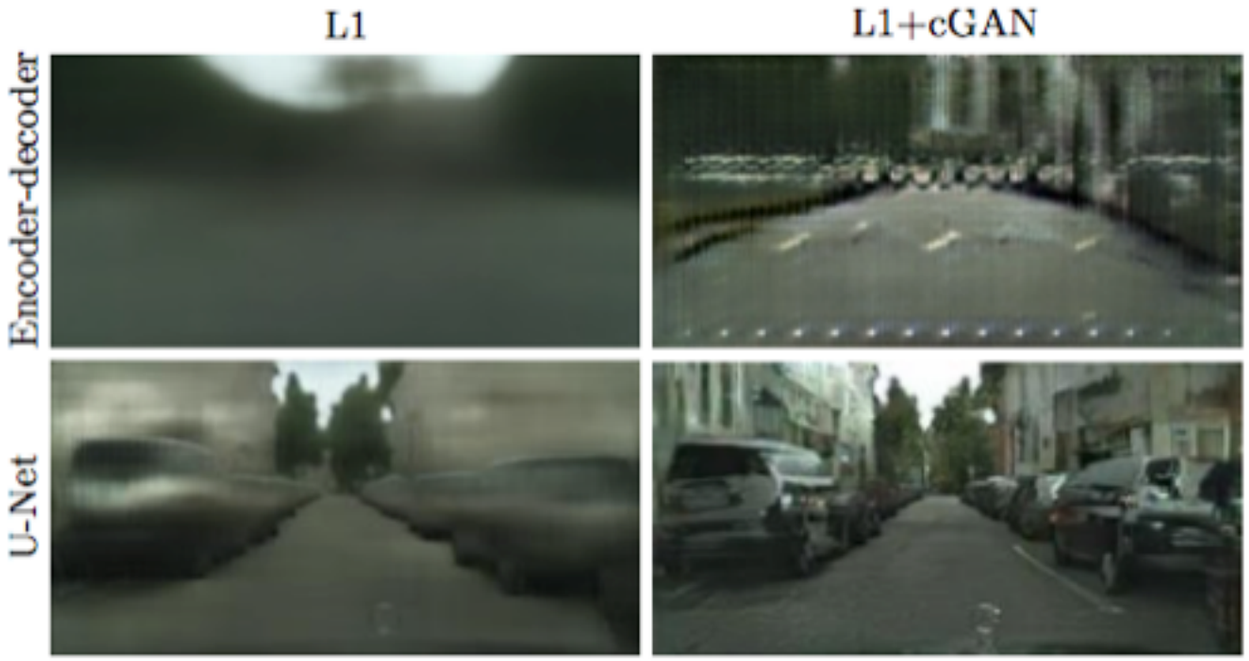
- Encoder-decoder 모델
  - 입력 이미지를 컨볼루션 연산을 통해 크기를 줄여나가고 (**encoder**)
  - 줄여진 이미지를 다시 늘려나가는 과정 (**decoder**)
  - 이미지 전체적인 정보는 잘 유지하지만 low-level 정보가 손실될 수 있음

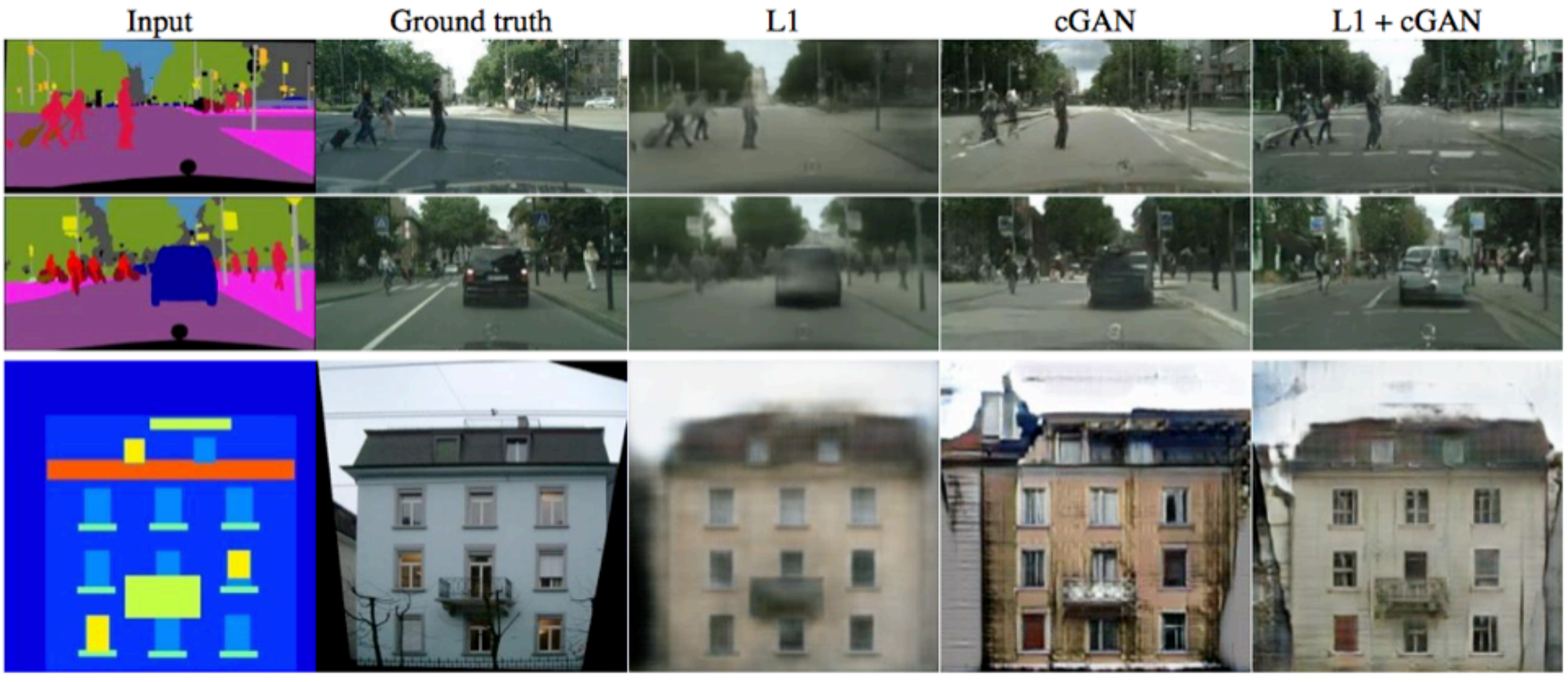
- U-Net 구조
  - Encoder-decoder 구조에 **skip-connection** 추가
  - Low-level 정보도 skip-connection을 통해 전파됨



Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." CVPR. 2017.

결과

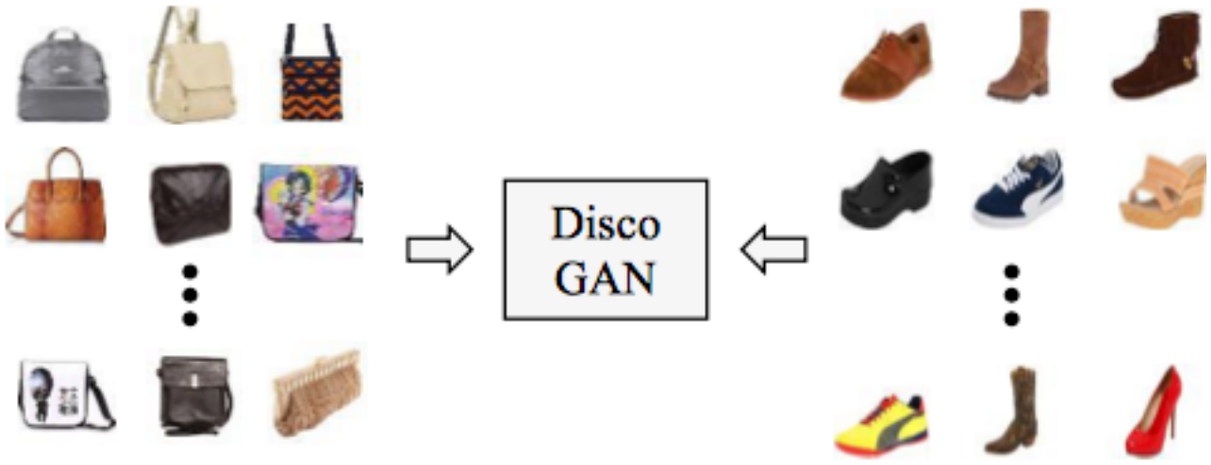




Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." CVPR. 2017.

Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." CVPR. 2017.

DiscoGAN



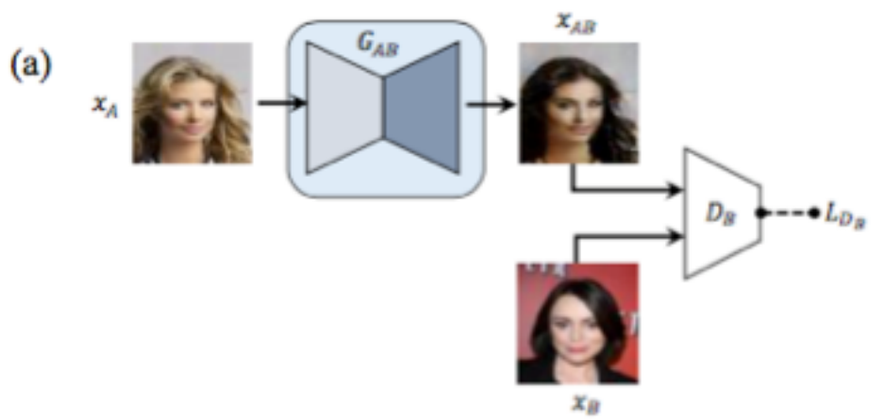
- pix2pix의 단점
  - 데이터 구성 시 서로 짝으로 이루어진 레이블이 존재해야 함 (paired data)
    - 많은 수의 데이터를 위와 같이 만들기는 어려운 작업

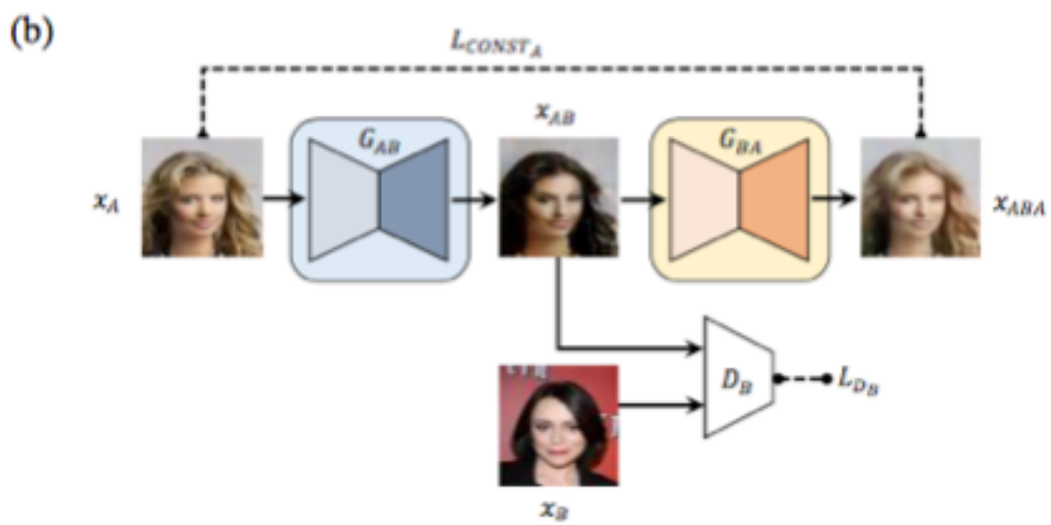
- DiscoGAN
  - **별도의 정보 없이** (unpaired data) 두 도메인 사이의 정보를 하는 모델
  - pix2pix 보다 성능이 좋지 않지만, 일반적으로 사용하기에 범용성이 좋음

Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." ICML. 2017.

Reconstruction Loss

- 기존의 GAN 모델 (a)는 도메인 A 에서 도메인 B로 향하는 변환만 학습
- 양방향의 변환을 학습하기 위해 별도의 generator를 생성 (b)
- 도메인 A->B->A 인 이미지를 생성하고, 원본 이미지와 비교 (e.g. L2 loss)





- G_{AB}: 도메인 A -> B로 변환하는 generator
- G_{ABA}: 도메인 A -> B -> A로 변환하는 generator

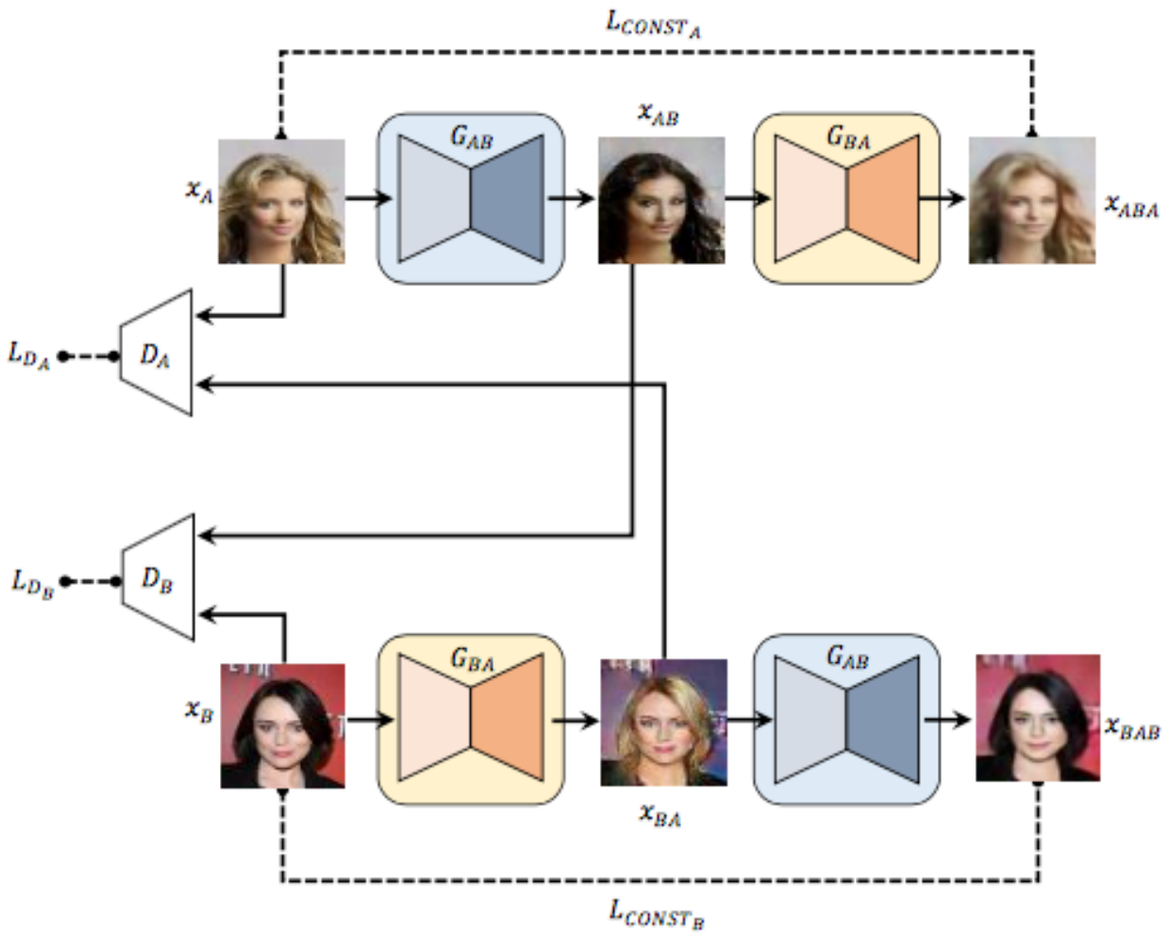
Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." ICML. 2017.

DiscoGAN

- Reconstruction loss에 **추가적인 discriminator**를 붙인 모델

$$L_G = L_{G_{AB}} + L_{G_{BA}}$$
$$= L_{GAN_B} + L_{CONST_A} + L_{GAN_A} + L_{CONST_B}$$

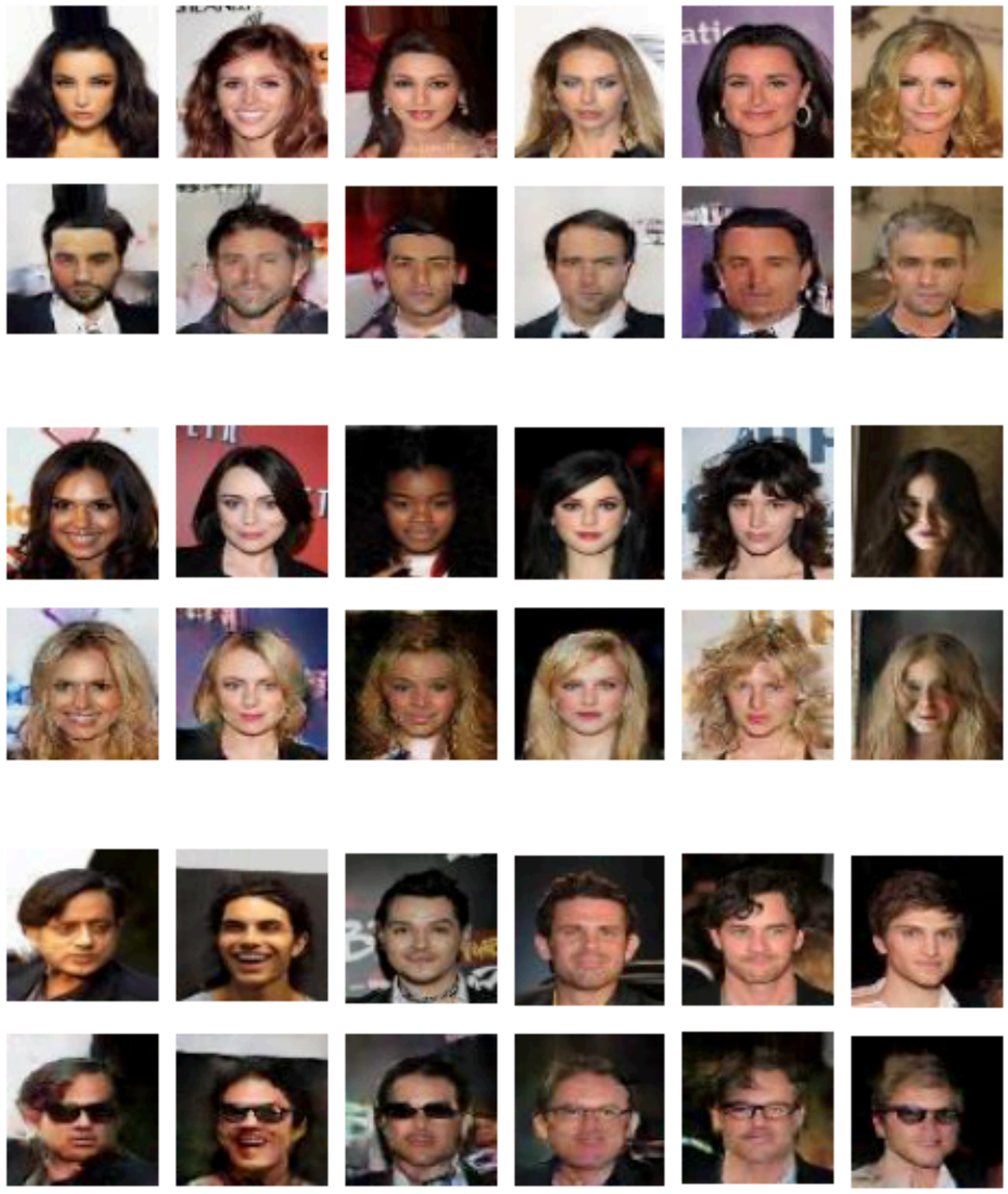
$$L_D = L_{D_A} + L_{D_B}$$



- 두 도메인간의 **one-to-one 변환** 가능

Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." ICML. 2017.

결과



Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." ICML. 2017.

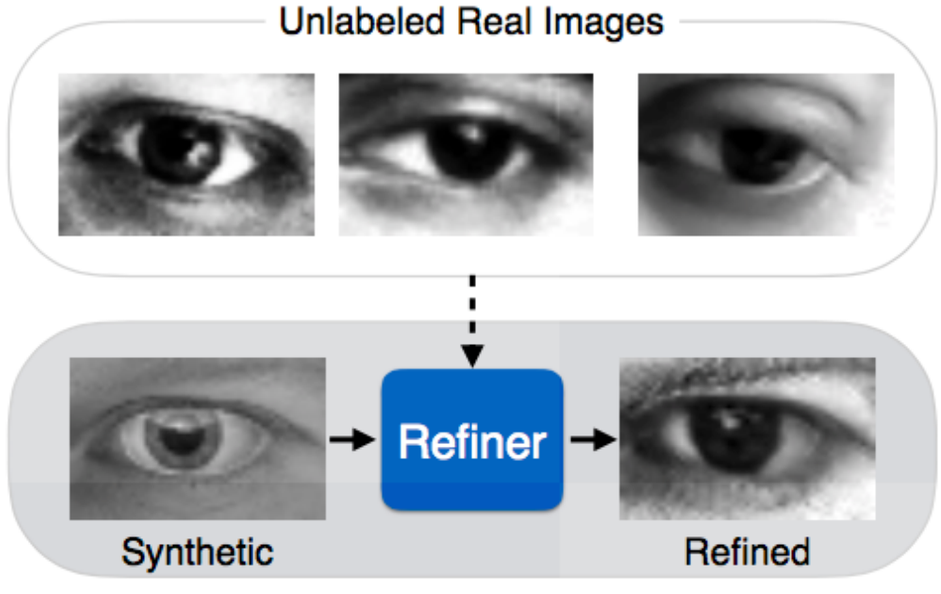
결과

Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." ICML. 2017.

S+U Learning

- 이런 경우?
  - **레이블이 없는 실제 데이터**는 많지만, 이를 전부 레이블링 하기 힘든 경우
  - 별도의 방법을 통해서 데이터를 **인위적으로 합성** 가능할 때 (e.g. Unity3D)

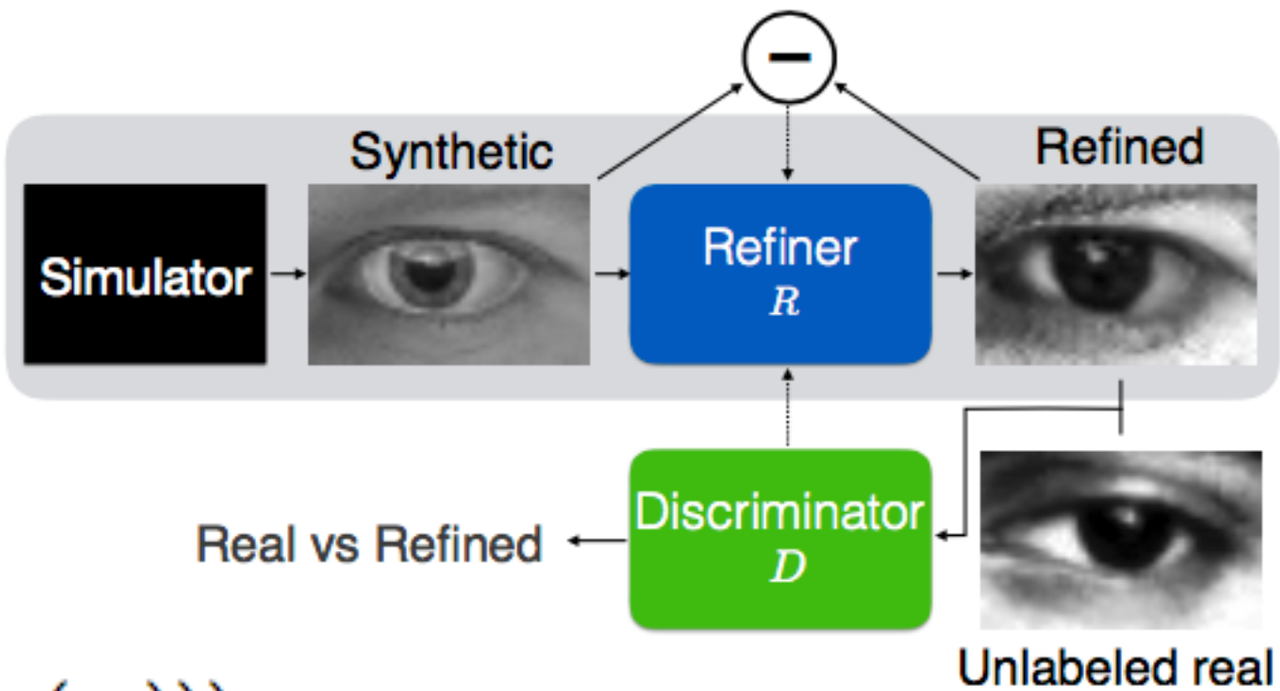
- 만들어진 데이터를 실제 데이터처럼 변환할 수 있을까?
  - **GAN**!!
  - Simulated + Unsupervised Learning



Shrivastava, Ashish, et al. "Learning from simulated and unsupervised images through adversarial training." CVPR. 2017

S+U Learning

- SimGAN은 **refiner** 네트워크를 통해 만들어진 데이터를 실제 데이터처럼 만들어 주는 모델
  - y: 레이블이 없는 실제 데이터
  - x: 만들어진 데이터



$$\mathcal{L}_R(\boldsymbol{\theta}) = \sum_i \ell_{\text{real}}(\boldsymbol{\theta}; \mathbf{x}_i, \mathcal{Y})$$

$$\ell_{\text{real}}(\boldsymbol{\theta}; \mathbf{x}_i, \mathcal{Y}) = -\log(1 - D_{\boldsymbol{\phi}}(R_{\boldsymbol{\theta}}(\mathbf{x}_i))).$$

일반 GAN Loss

Shrivastava, Ashish, et al. "Learning from simulated and unsupervised images through adversarial training." CVPR. 2017

S+U Learning

- 문제점:
  - 인조 데이터에 있는 레이블 정보는 변환 과정에서도 유지되어야 함
- 해결책:
  - Loss 함수에 self-regularized 텀 추가하기

$$\mathcal{L}_R(\boldsymbol{\theta}) = -\sum_i \log(1 - D_{\boldsymbol{\phi}}(R_{\boldsymbol{\theta}}(\mathbf{x}_i))) + \lambda \|\psi(R_{\boldsymbol{\theta}}(\mathbf{x}_i)) - \psi(\mathbf{x}_i)\|_1.$$

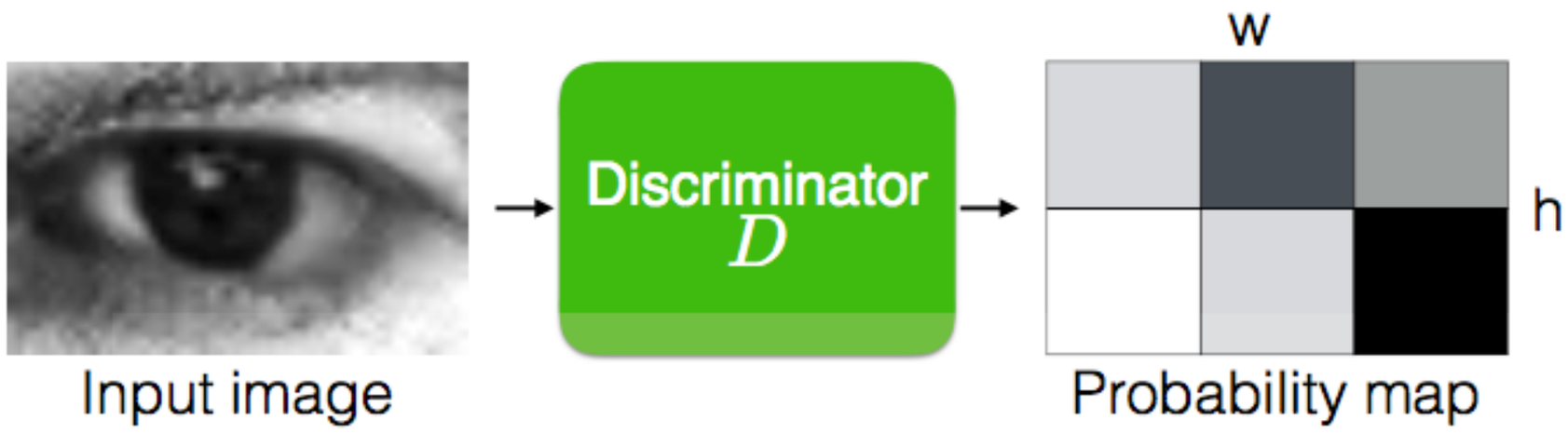
  - 기본 설정은 identity mapping $(\psi(\mathbf{x}) = \mathbf{x})$.
    하지만 다른 함수를 사용할 수 있음 (e.g. VGG content loss)

Shrivastava, Ashish, et al. "Learning from simulated and unsupervised images through adversarial training." CVPR. 2017

S+U Learning

- 문제점:
  - Generator가 discriminator를 속이기 위해 특정 위치만 과도하게 강조하는 경향을 보임 (속이기만 하면 되니까..)

- 해결책:
  - Discriminator가 **부분 영역**에 대해서 adversarial loss를 출력하도록 변경
  - 최종 discriminator loss는 부분 영역 loss들을 더하면 됨

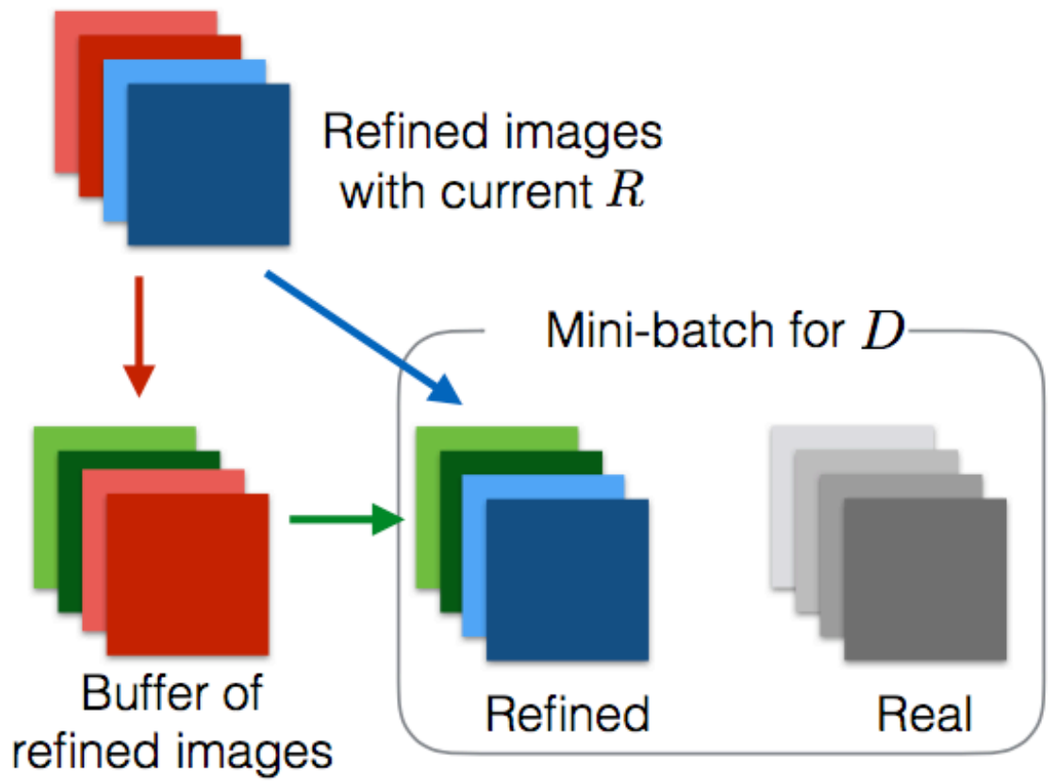


Shrivastava, Ashish, et al. "Learning from simulated and unsupervised images through adversarial training." CVPR. 2017

S+U Learning

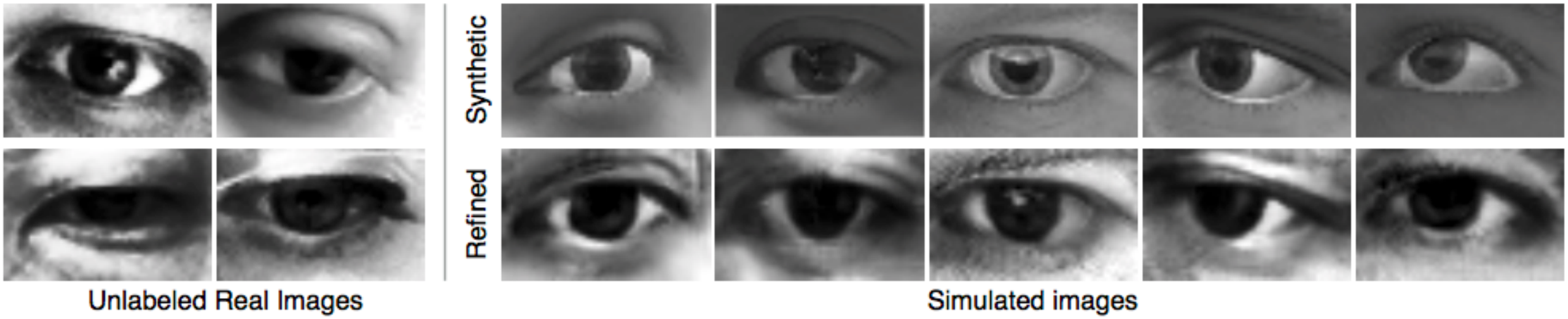
- 문제점:
  - Discriminator가 **최근에 변환한 이미지에 대해서만** 초점을 맞춤
  - 이러면 (1) 모델 수렴이 안되고 (2) 이미 discriminator가 이상하다고 기억 하지 못하는 아티팩트를 generator가 생성함 (진작에 까먹어서..)

- 해결책:
  - 히스토리 버퍼를 만들기
  - 랜덤하게 batch_size/2 만큼 샘플을 히스토리 버퍼에서 뽑은 후 변환
  - 매 이터레이션마다 batch_size/2 만큼 버퍼를 새로 생성한 이미지로 교체



Shrivastava, Ashish, et al. "Learning from simulated and unsupervised images through adversarial training." CVPR. 2017

S+U Learning



	Selected as real	Selected as synt
Ground truth real	224	276
Ground truth synt	207	293

Table 1. Results of the 'Visual Turing test' user study for classifying real vs refined images. The average human classification accuracy was 51.7% (chance = 50%).

Shrivastava, Ashish, et al. "Learning from simulated and unsupervised images through adversarial training." CVPR. 2017