Data Augmentation of Backscatter X-ray Images for Deep Learning-Based Automatic Cargo Inspection

2022.05.19 AAI Seminar

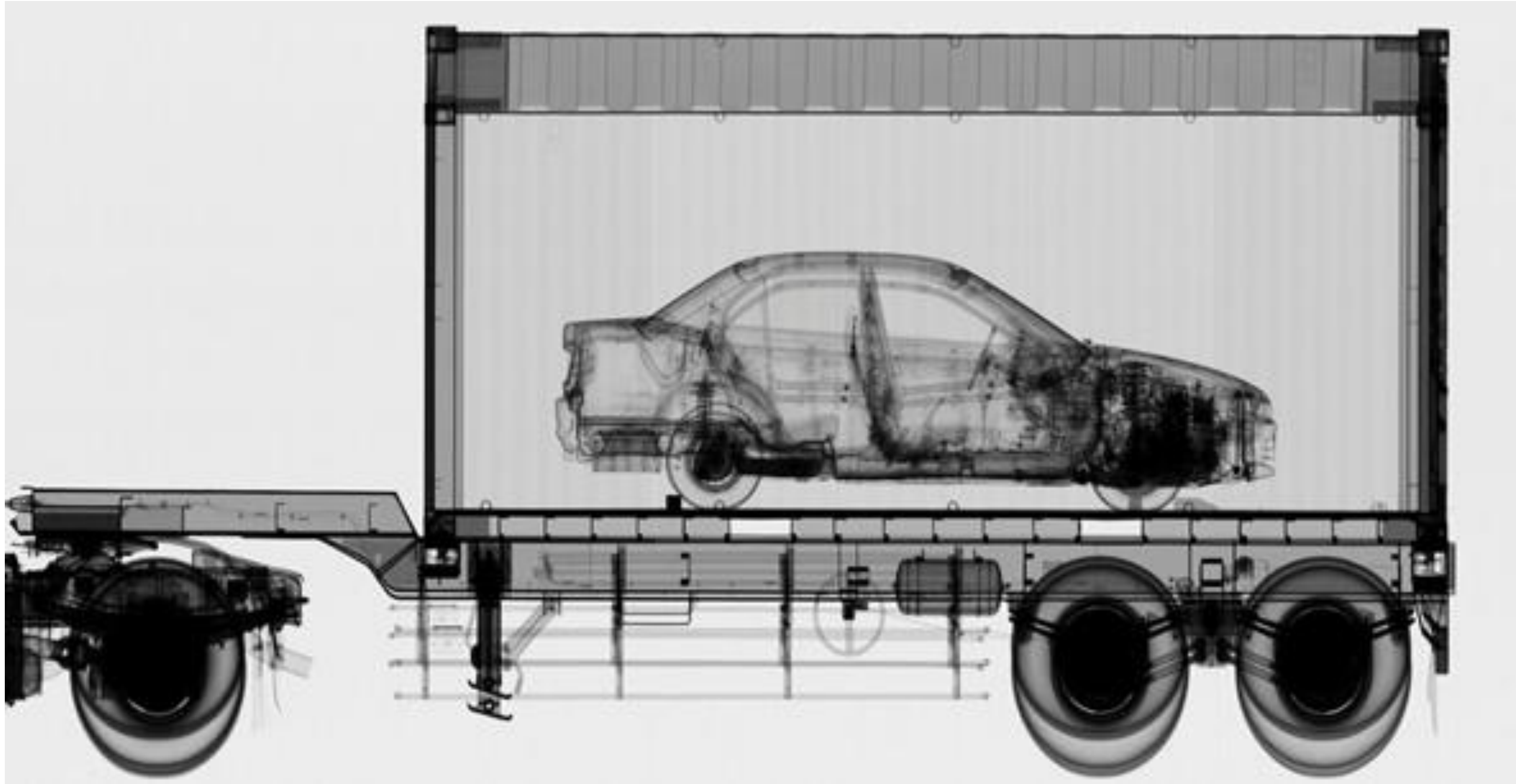
AAI Lab. 김민준
AAI Lab. 송승우
AAI Lab. 최창수

Hanbat National University, Dept. of Computer Engineering

Contents

Research Purpose

X-ray 이미지는 비파괴적으로 물체의 내부를 확인 할 수 있기 때문에 세관 검사에 활용 되었음



Research Purpose

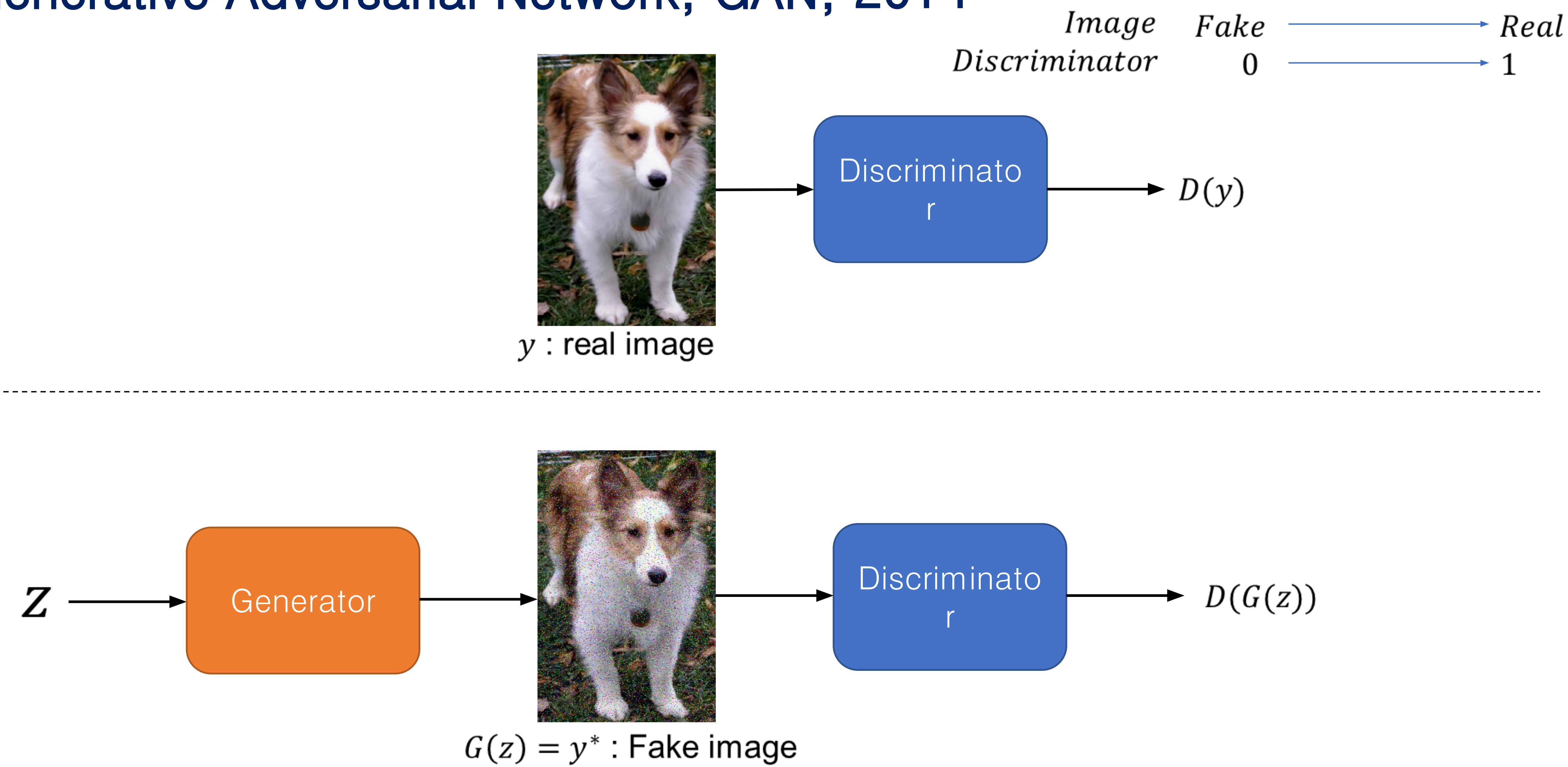
인력과 장비가 부족한 탓에 국내 주요 항만에서
엑스레이 스캐너로 검사한 컨테이너 화물은
1.6%에 불과

인공지능(AI) 기술을 활용한 자동 검사 방법 고려

화물의 X-ray(BSX) 데이터 부족

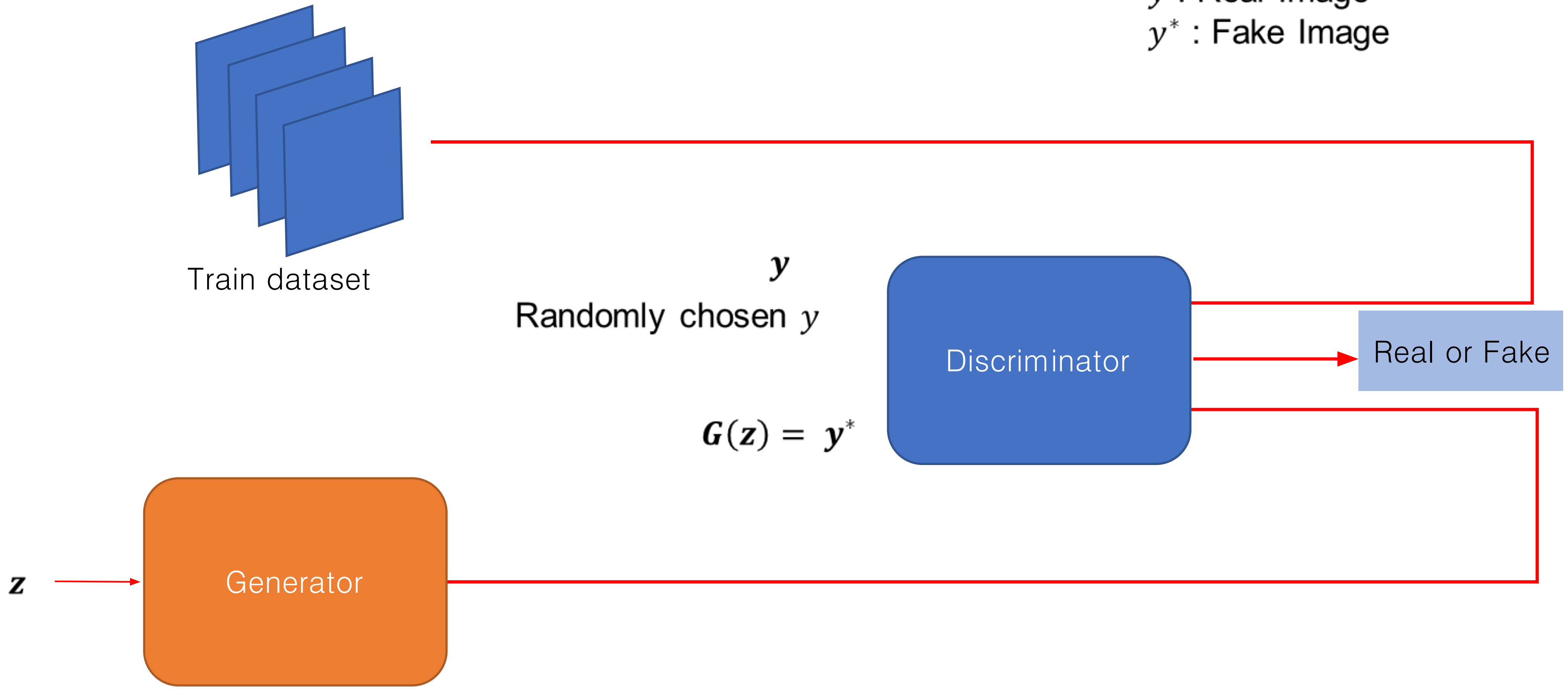
https://www.youtube.com/watch?v=7wBfElexJI0

Generative Adversarial Network, GAN, 2014



Generative Adversarial Network, GAN – Structure

y : Real Image
y^* : Fake Image

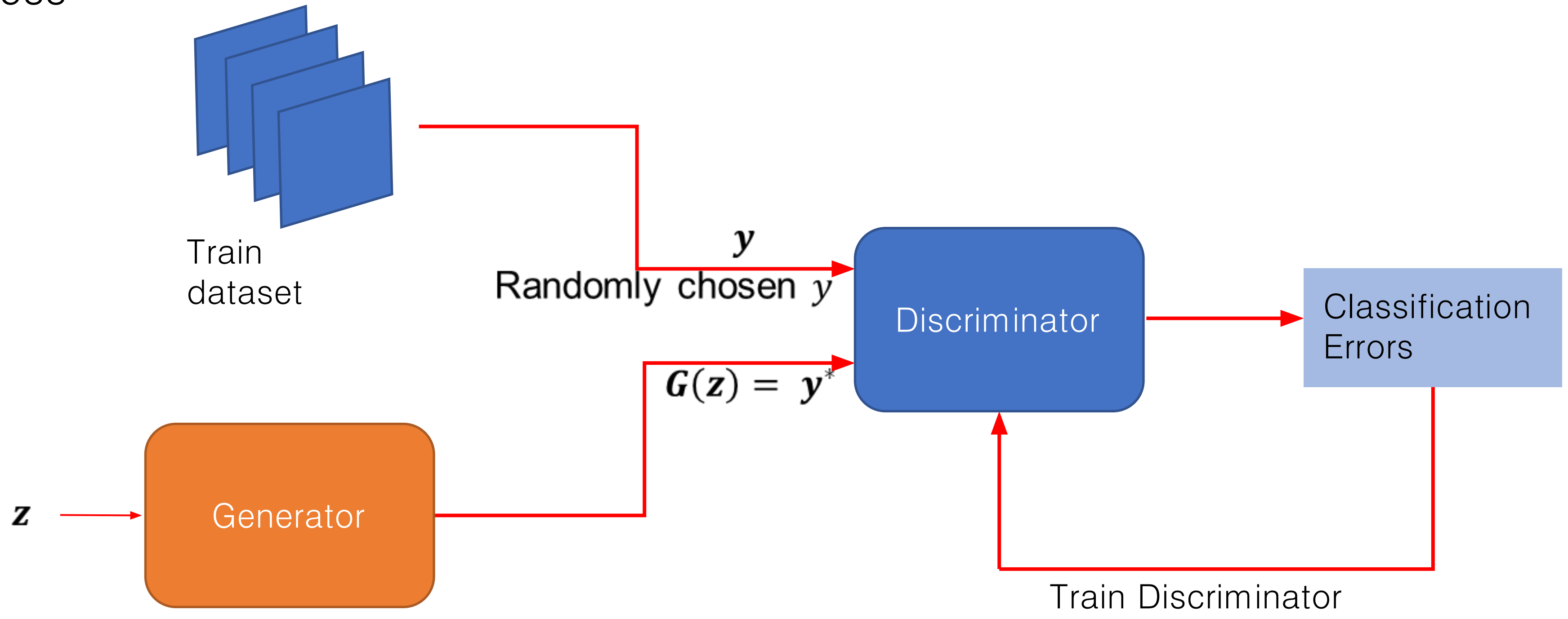


Generative Adversarial Network, GAN – Discriminator

- Discriminator : 진짜 데이터와 생성자가 만드는 가짜 데이터를 구별하는 것을 목표로 함

y : Real Image
y^* : Fake Image

- Discriminator Train Process



AAI Lab
Applied Artificial Intelligence

Generative Adversarial Network, GAN – Discriminator

Discriminator Train Process

1. 훈련 데이터에서 학습에 사용할 데이터 y를 랜덤하게 선택한다.
2. 랜덤 잡음 벡터(Latent vector) z를 얻어서 Generator를 통해 가짜 데이터 x^*를 생성한다.
3. Discriminator를 이용하여 y와 y^* classification
4. Classification loss를 계산하고 이를 통해 학습한다.(loss를 최소화 한다.)

→ Binary Cross Entropy

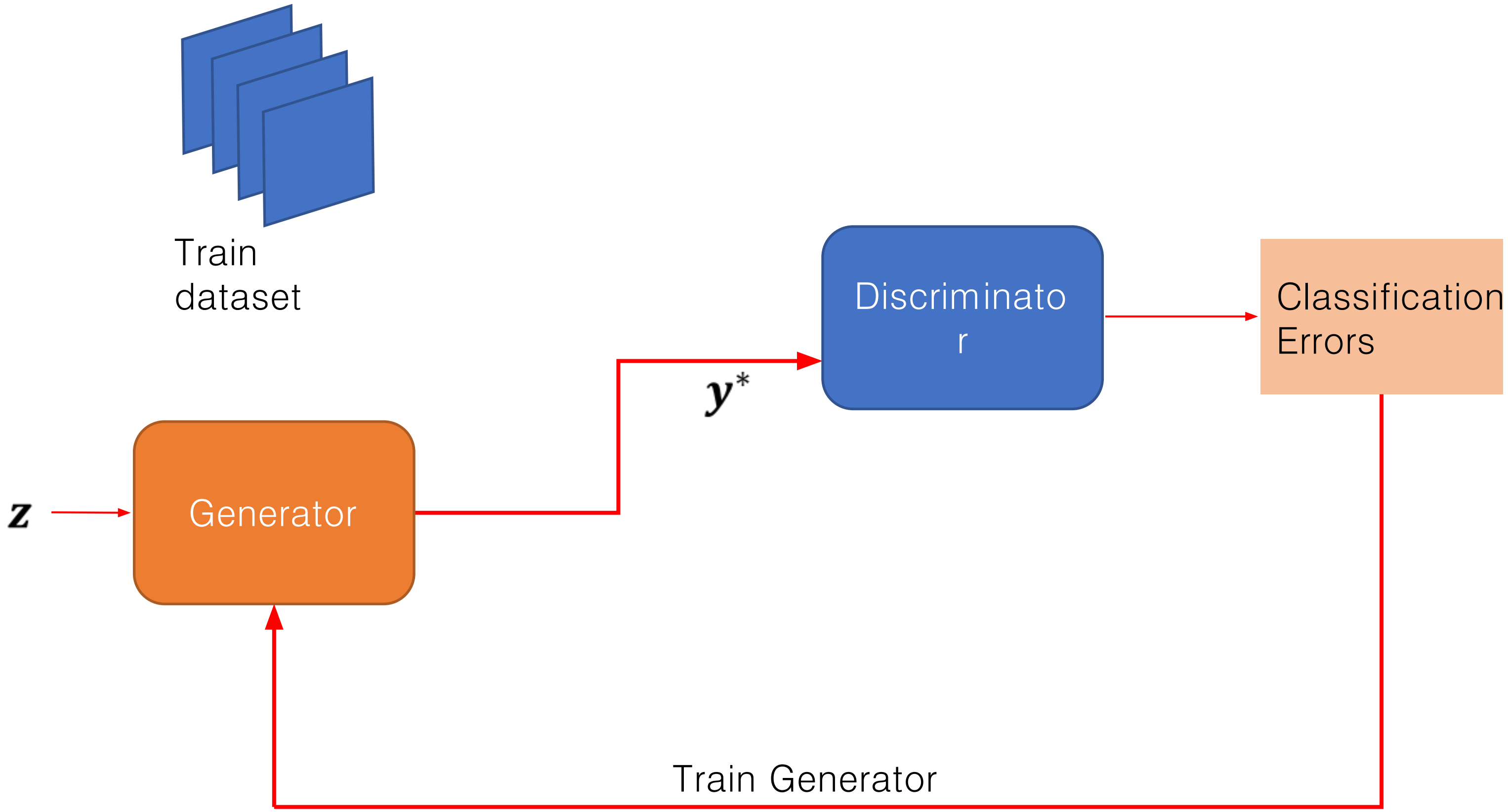
Discriminator의 loss function : $-\frac{1}{n}\sum_{i=1}^{n}(t_i \log(p_i) + (1-t_i)\log(1-p_i))$

AAI Lab
Applied Artificial Intelligence

Generative Adversarial Network, GAN – Generator

- Generator : 진짜 데이터와 구별이 안되는 데이터를 생성해 판별자가 구분을 못하도록 만드는 것을 목표로 함

- Generator Train Process

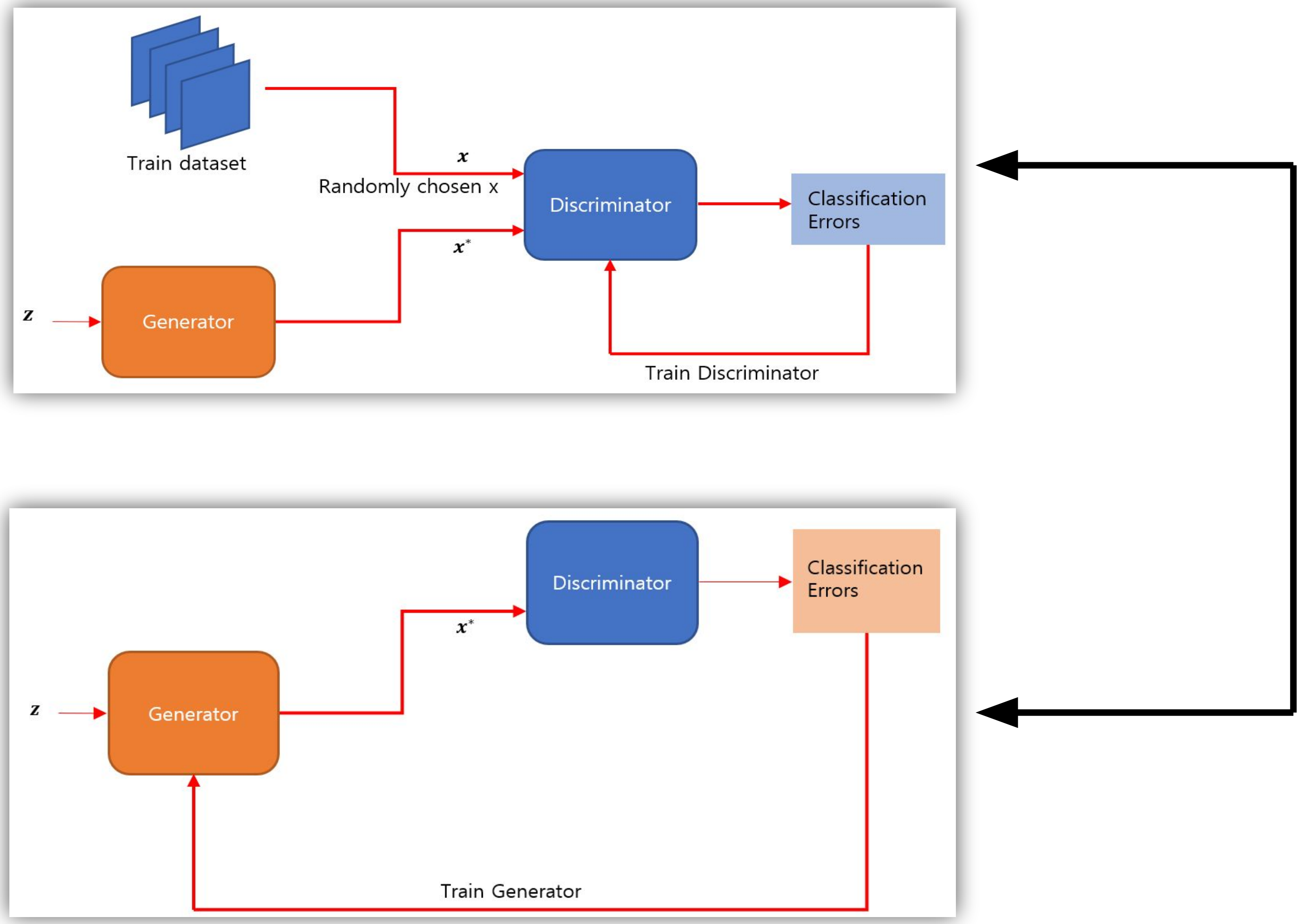


Generative Adversarial Network, GAN – Generator

Generator Train Process

1. 랜덤 잡음 벡터(Latent vector) z를 얻어서 생성자를 통해 가짜 샘플 y^*를 만든다.
2. Discriminator를 통해 y^*를 분류한다.
3. Classification loss를 계산하고 오차를 최대화 하는 방향으로 학습한다.

Generative Adversarial Network, GAN – Training process



두 과정을 반복하며 훈련

Generative Adversarial Network, GAN – Objective function

$$\min_G \max_D V(D,G) = E_{y\sim P_{data}(y)}[logD(y)] + E_{z\sim p_z(z)}\left[\log\{1 - D\left(G(z)\right)\}\right]$$

- ✓ Discriminator는 Real image를 1로, Fake image를 0으로 판별하는 것이 목적

  →$D(y) = 1,\ D\left(G(Z)\right) = 0$

- ✓ Generator는 Discriminator가 1로 판별하는 이미지를 생성하는 것이 목적

  →$D\left(G(Z)\right) = 1$

pix2pix, 2016

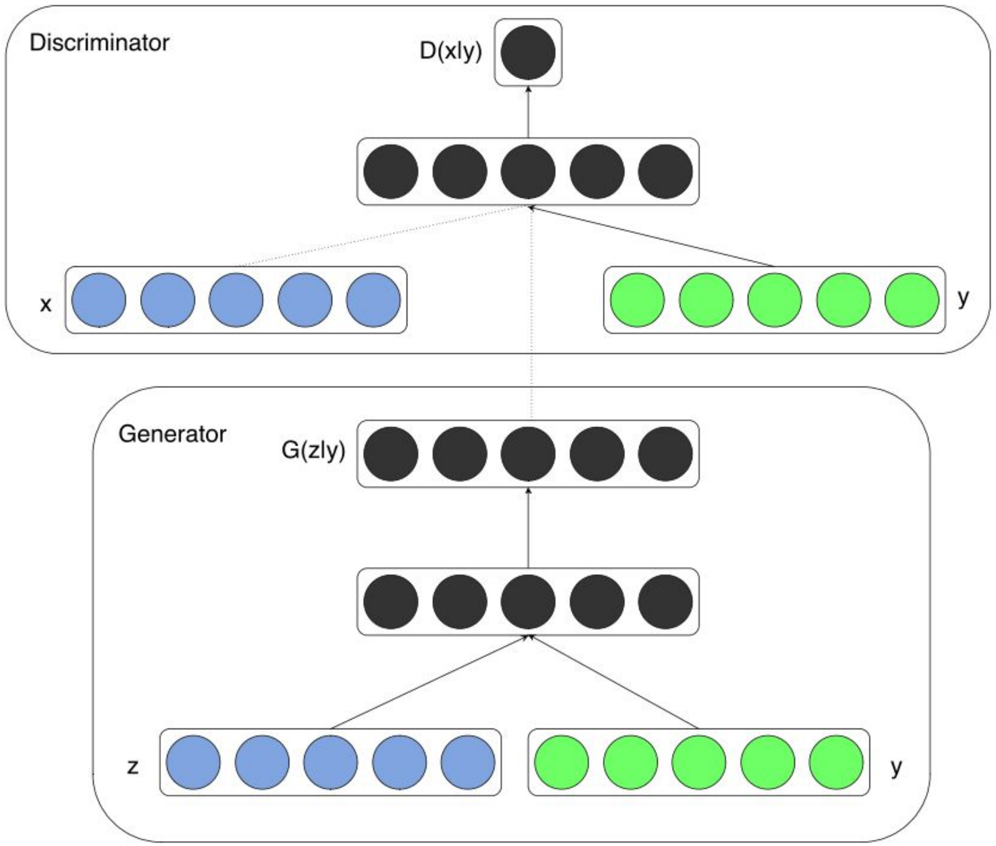
generator and discriminator are conditioned on some **extra information** y

'조건 이미지' 를 추가하여 임의로 생성되는 출력을 제어하기 위한 방법

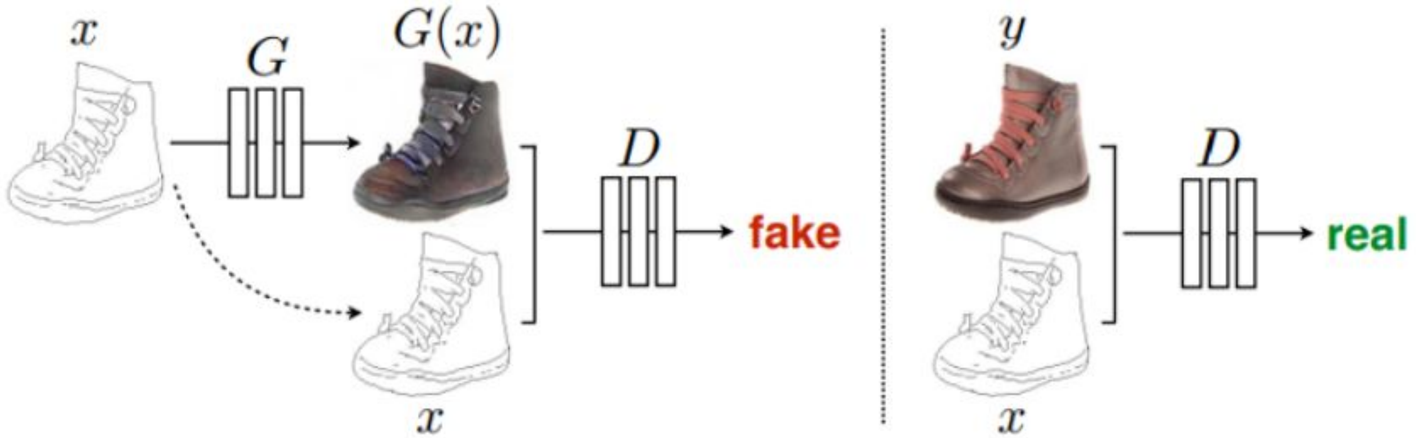
3.2 Conditional Adversarial Nets

Generative adversarial nets can be extended to a conditional model if both the generator and discriminator are conditioned on some extra information y. y could be any kind of auxiliary information, such as class labels or data from other modalities. We can perform the conditioning by feeding y into the both the discriminator and generator as additional input layer.

M.Mirza, Conditional GAN



pix2pix – objective function

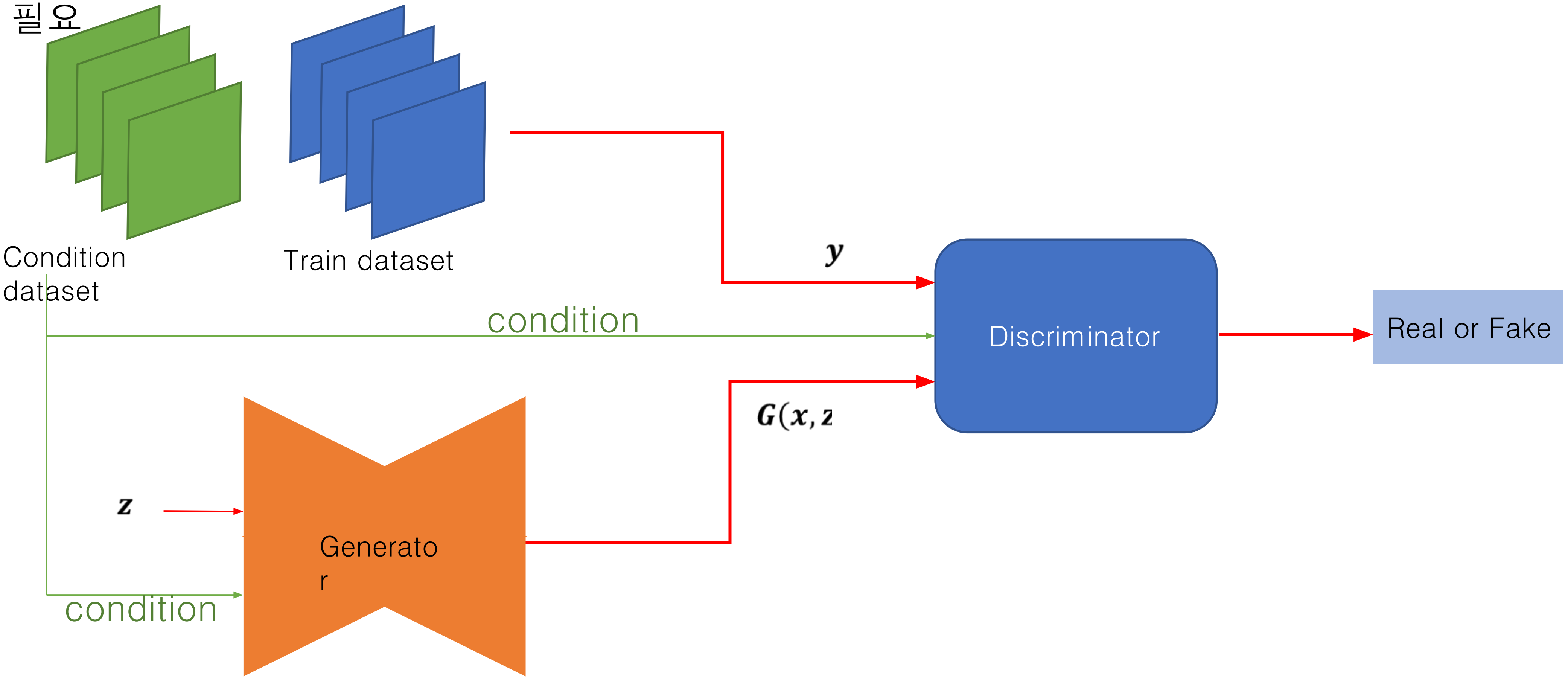


조건 이미지　　실제 이미지　　변환된 이미지

pix2pix – Structure

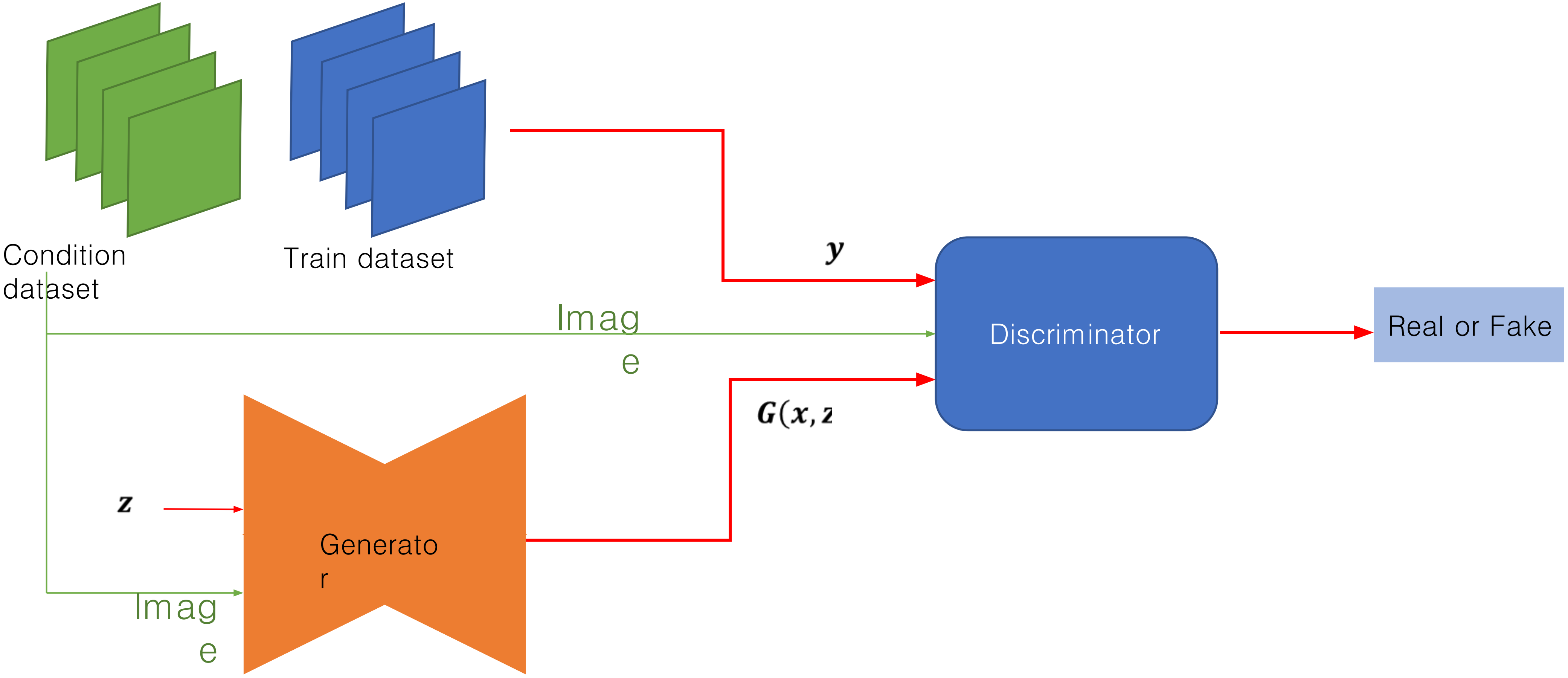
y : Real Image
y^* : Fake Image
z : Latent vector

{train, condition} 의 데이터 쌍이 필요



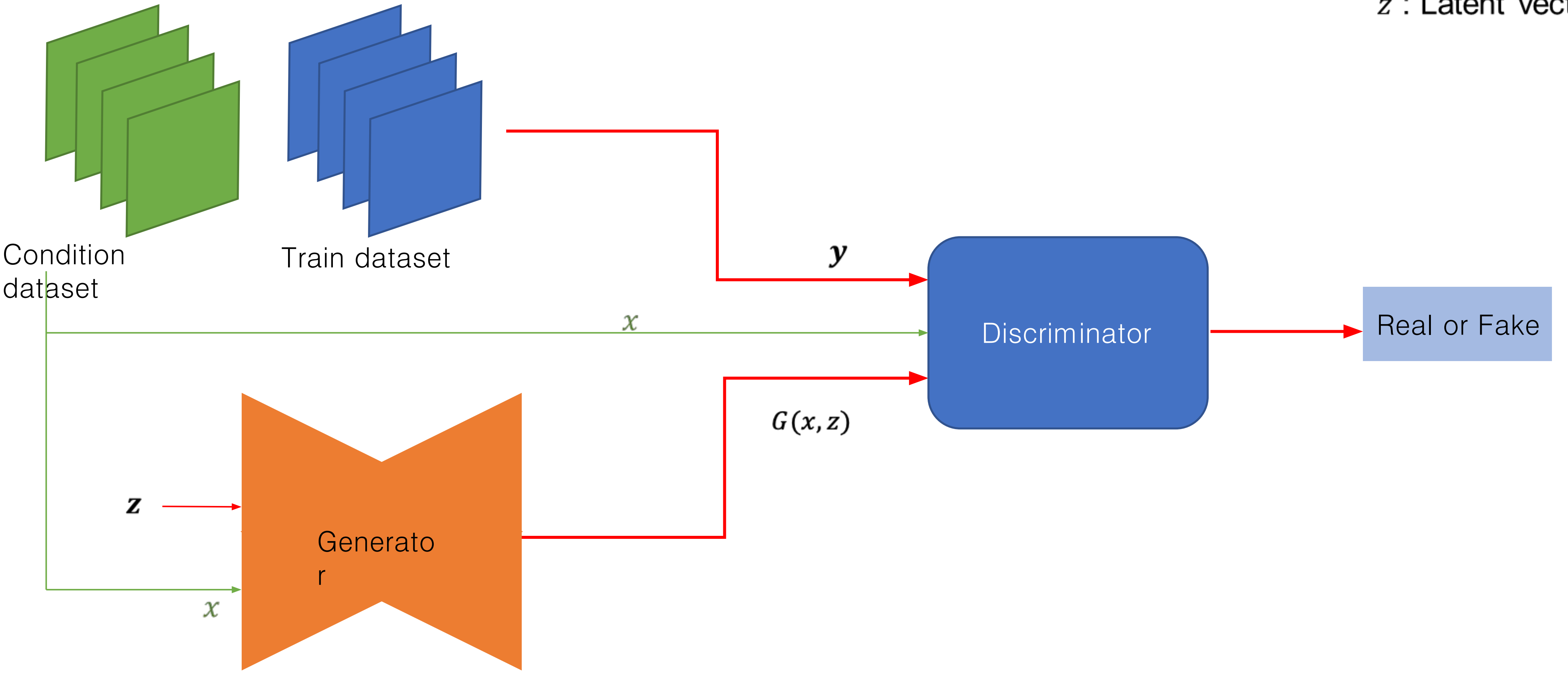
pix2pix – Structure

y : Real Image
y^* : Fake Image
z : Latent vector



pix2pix – Structure

y : Real Image
y^* : Fake Image
x : Condition
z : Latent vector



Generator : U-Net 구조

pix2pix – objective function

- Objective function of Conditional GAN

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

- Objective function of pix2pix

$$\arg\min_{G}\max_{D}\mathcal{L}_{cGAN}(G,D) + \lambda\mathcal{L}_{L1}(G)$$

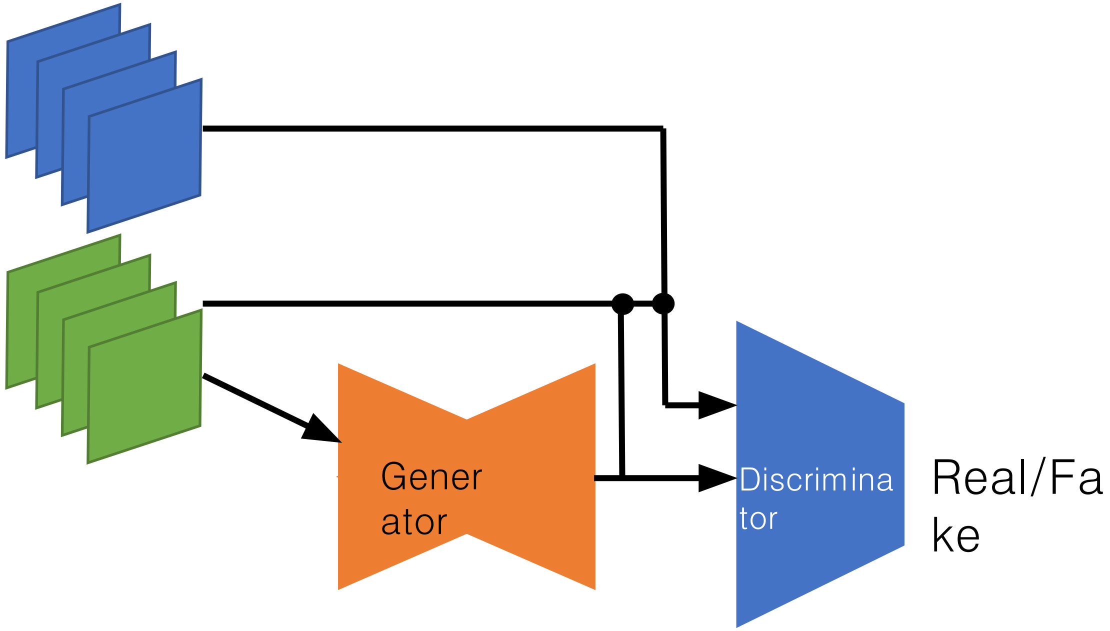
$$\mathcal{L}_{L1} = \mathbb{E}_{x,y,z}[\|y - G(x,z)\|_1]$$

- L1 distance → 실제 이미지와 더욱 비슷하게 만듦(더욱 선명한 결과)

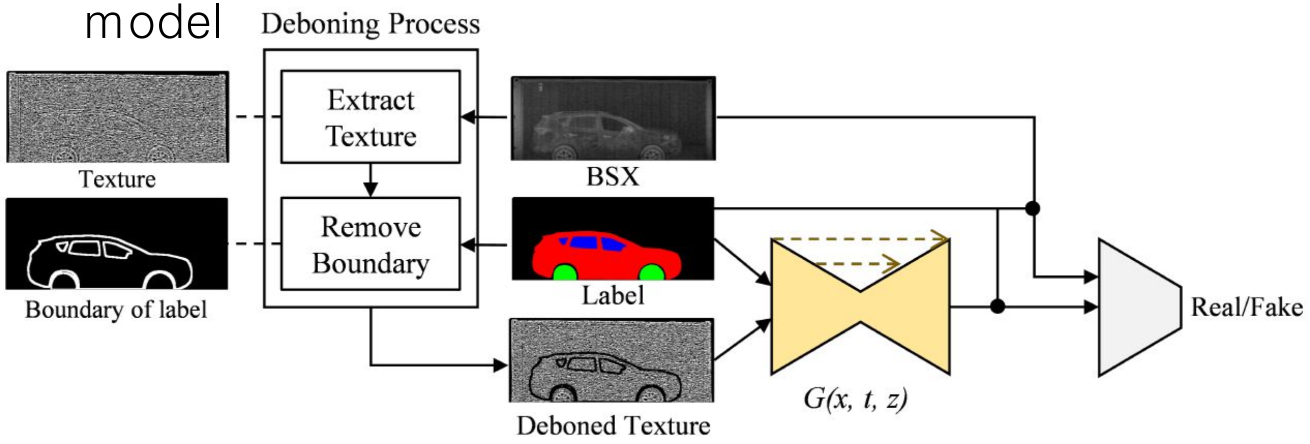
AAI Lab
Applied Artificial Intelligence

Proposed Model – Compare with pix2pix

pix2pix

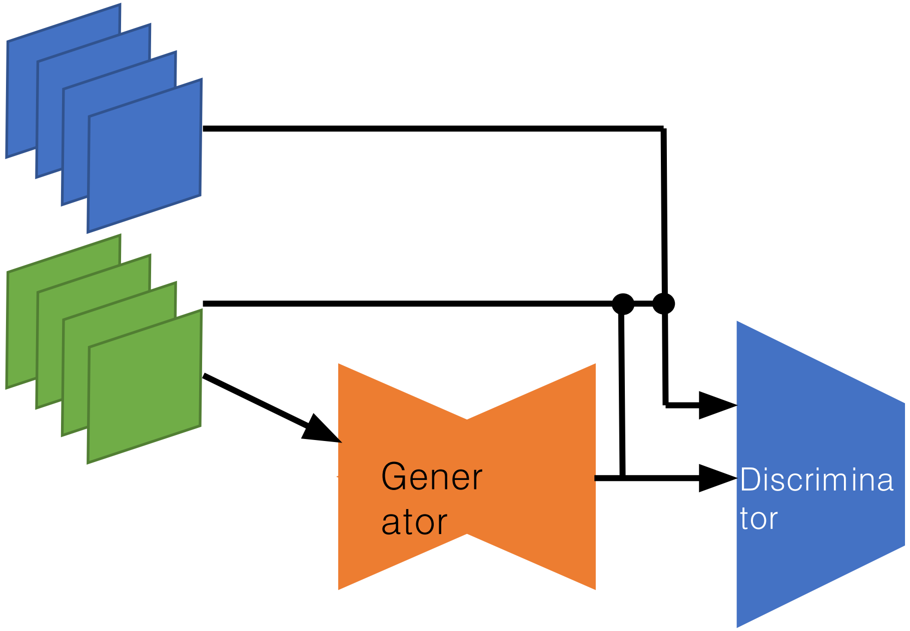


Proposed model



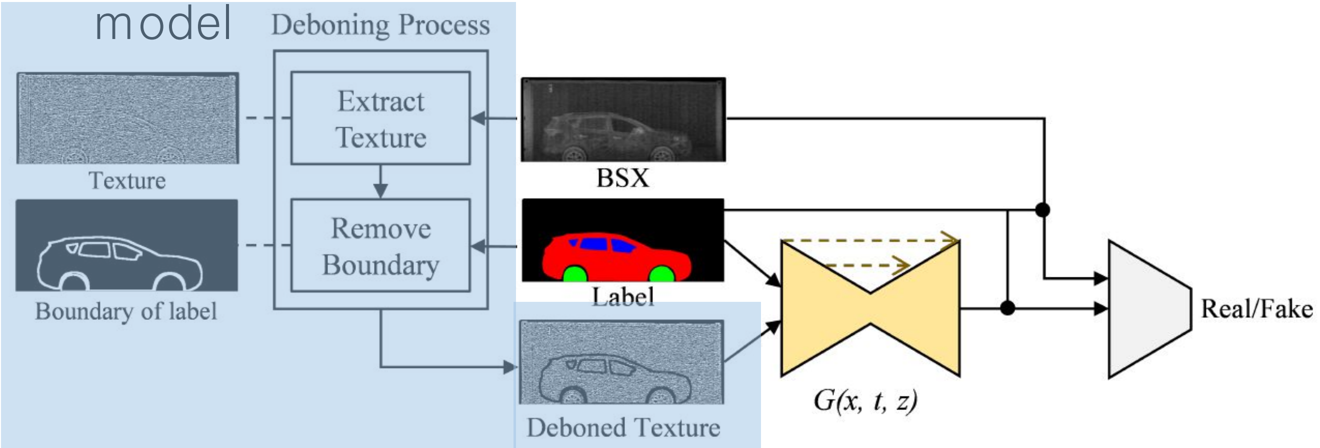
Proposed Model – Compare with pix2pix

pix2pix



Real/Fa
ke

Proposed
model



pix2pix – objective function

- Objective function of Conditional GAN (+texture)

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

- Objective function

$$\arg\min_{G}\max_{D}\mathcal{L}_{cGAN}(G,D) + \lambda\mathcal{L}_{L1}(G)$$

$$\mathcal{L}_{L1} = \mathbb{E}_{x,y,z}[\|y - G(x,t,z)\|_1]$$

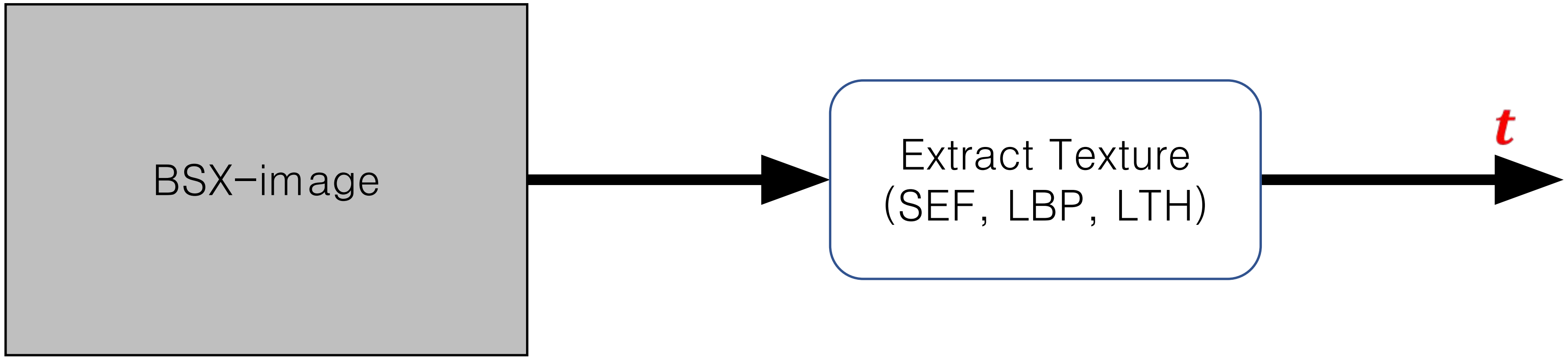
pix2pix – Limit

pix2pix BSX-image

Real BSX-image

✔ 실험에서 pix2pix는 real BSX-image의 노이즈 패턴을 재현하지 못했다.

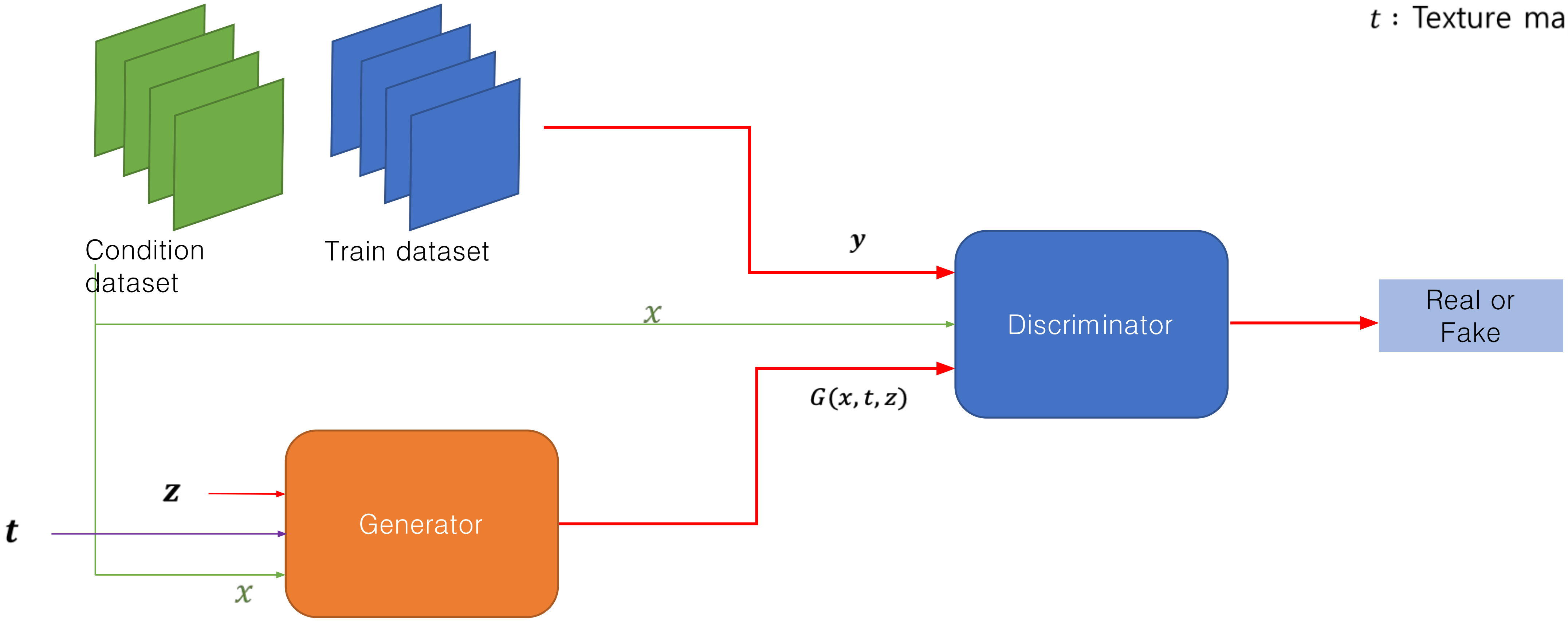
Proposed process – Extract Texture



- BSX-image의 texture map 생성

Proposed process – Extract Texture

y : Real Image
y^* : Fake Image
x : Condition
z : Latent vector
t : Texture map



Proposed process – function

y = real image
b = random background

$$L_{cGAN}(G, D) = E_{x,y}[\log D(x, y)] + E_{x,z,\boldsymbol{t}}[\log 1 - D(x, G(x, \boldsymbol{t}, z))]$$

$$L_1(G) = E_{x,y,z,t}[\ \|y - G(x, \boldsymbol{t}, z)\|_1\ ]$$

$$\boldsymbol{t} = \begin{cases} ExtractTexture(y)\ for\ training \\ ExtractTexture(b)\ for\ testing \end{cases}$$

AAI Lab
Applied Artificial Intelligence

SEF(Sobel edge filter)

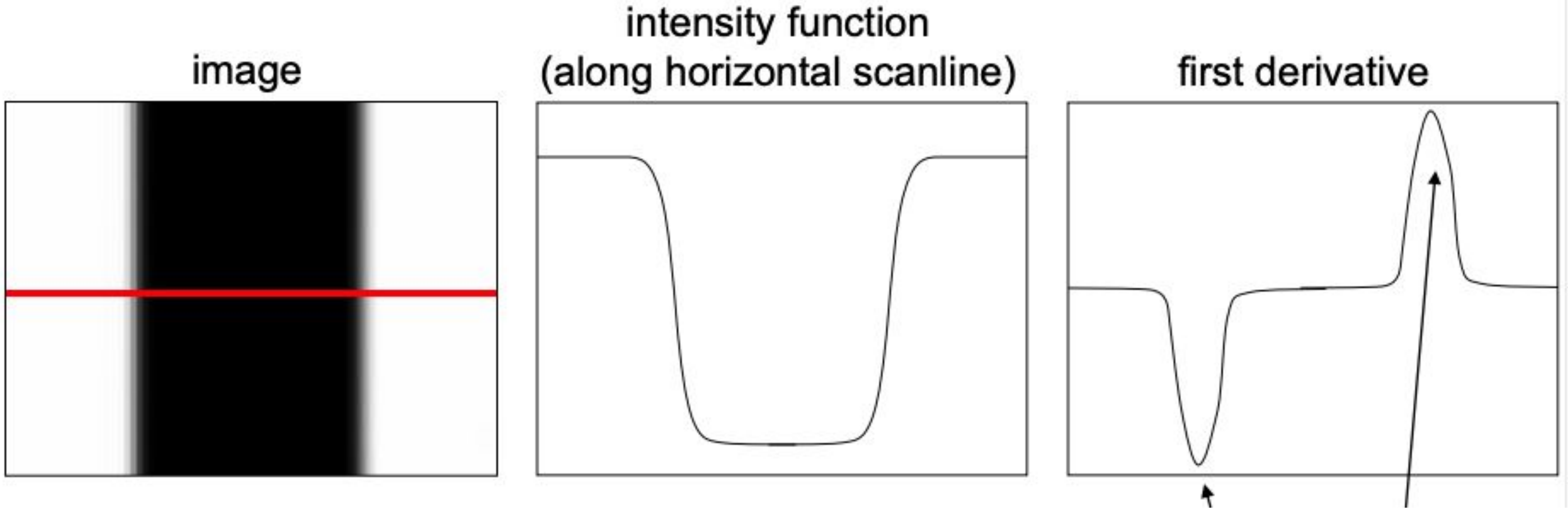
$$G_w = \begin{vmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{vmatrix} \otimes I$$

$$G_h = \begin{vmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{vmatrix} \otimes I$$

$$G = \sqrt{G_w^2 + G_{h'}^2}$$

AAI Lab
Applied Artificial Intelligence

SEF(Sobel edge filter)



SEF(Sobel edge filter) – Finite differences

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Definition of a derivative using forward difference

$$f'(x) = \frac{f(x+0.5h) - f(x-0.5h)}{h}$$

use central difference

$$f'(x) = \frac{f(x+1) - f(x-1)}{2}$$

Remove limit and set h = 2

-1	0	1

1D derivative filter(x–direction)

AAI Lab
Applied Artificial Intelligence

SEF(Sobel edge filter)

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

Sobel filter

Blurring

1D derivative filter (x-direction)

SEF(Sobel edge filter) – Filtering with Average filter

$$h[m,n]=\sum_{k,l} g[k,l]\, f[m+k,n+l]$$

output filter image (signal)

$g[\cdot,\cdot]$

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

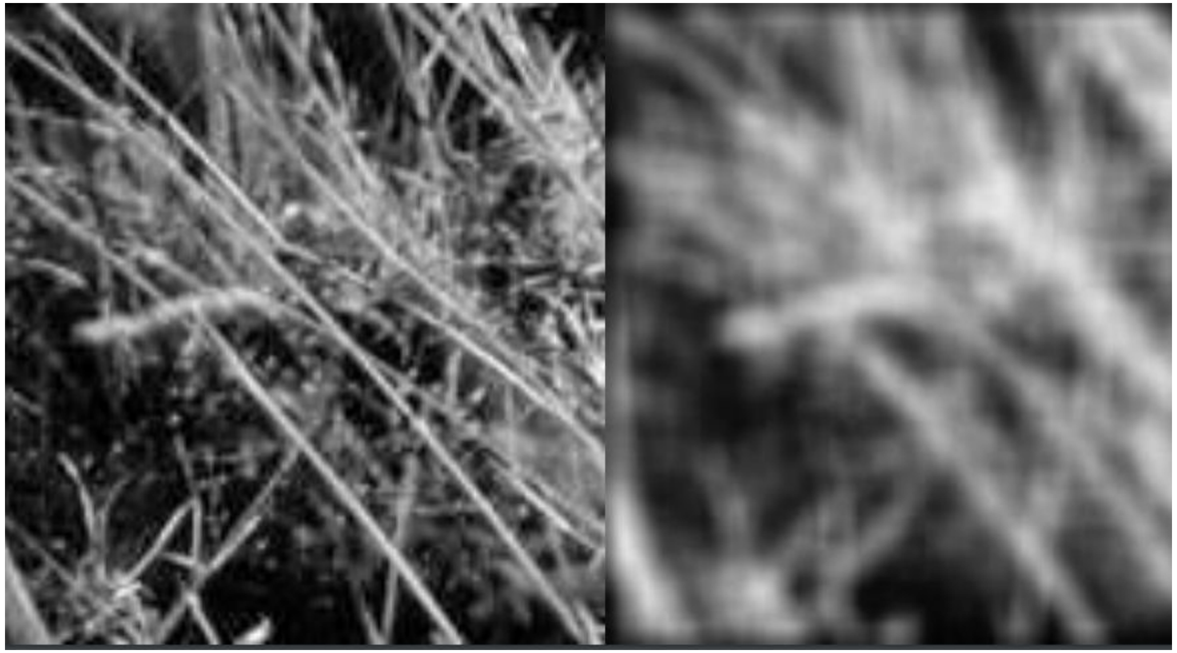
$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

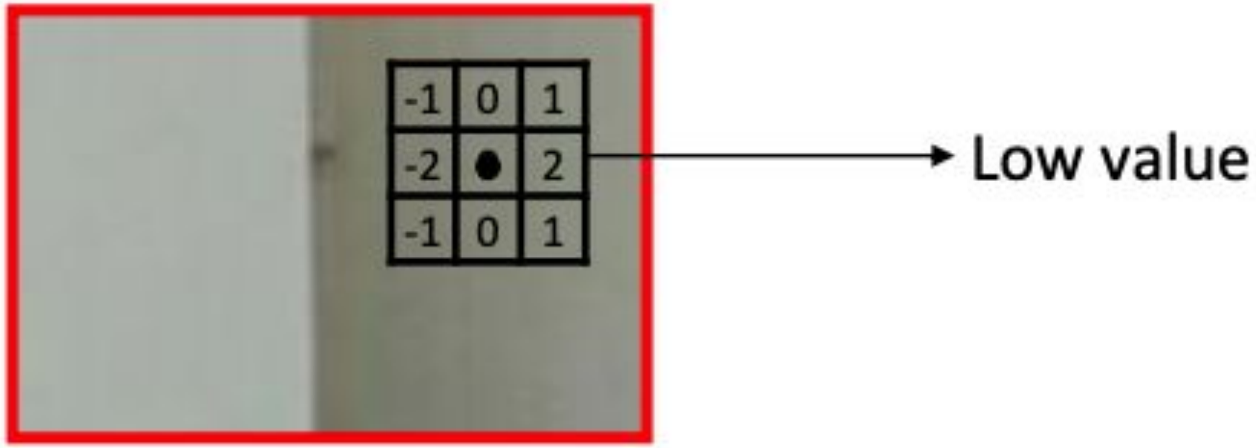
$h[.,.]$

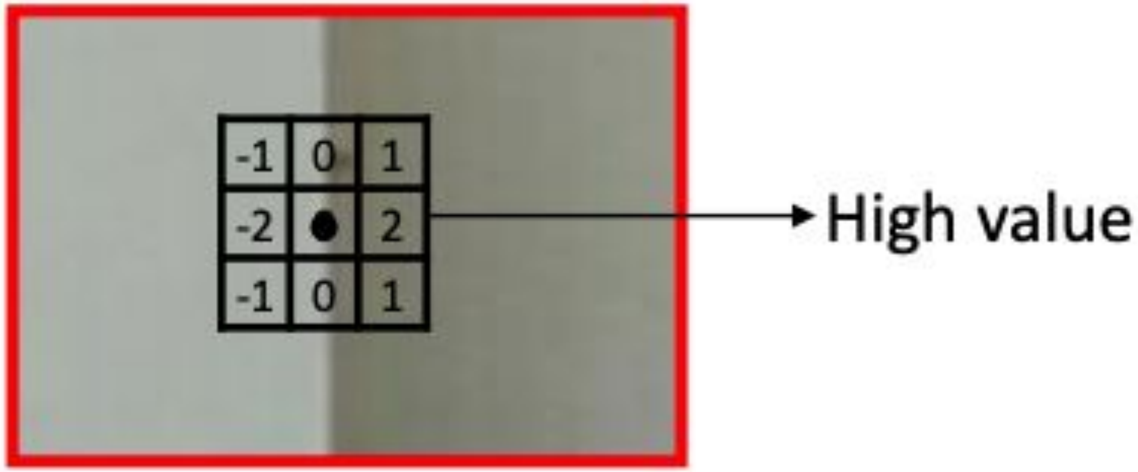
	0	10	20						

SEF(Sobel edge filter) – Average filter



SEF(Sobel edge filter) – filtering





SEF(Sobel edge filter) – Vertical and horizontal Sobel filter

Vertical Sobel filter:

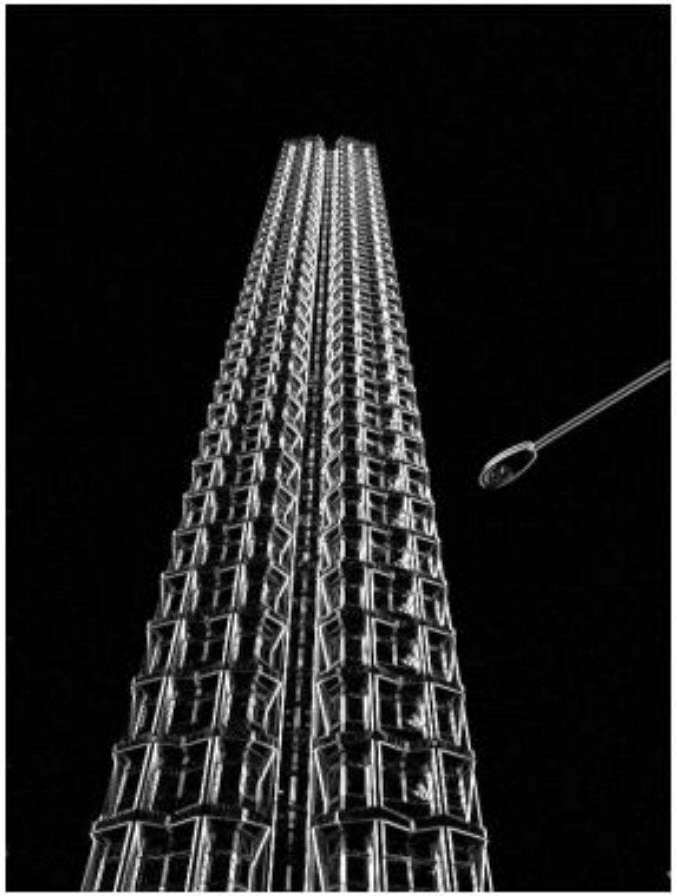
$$\begin{bmatrix}-1&0&1\\-2&0&2\\-1&0&1\end{bmatrix} = \underbrace{\begin{bmatrix}1\\2\\1\end{bmatrix}}_{\text{Blurring}} * \underbrace{\begin{bmatrix}-1&0&1\end{bmatrix}}_{\text{1D derivative filter (x-direction)}}$$

horizontal Sobel filter:

$$\begin{bmatrix}-1&-2&-1\\0&0&0\\1&2&1\end{bmatrix} = \underbrace{\begin{bmatrix}-1\\0\\1\end{bmatrix}}_{\text{1D derivative filter (y-direction)}} * \underbrace{\begin{bmatrix}1&2&1\end{bmatrix}}_{\text{Blurring}}$$

SEF(Sobel edge filter) – Sobel filter example

original



vertical Sobel filter

horizontal Sobel filter

SEF(Sobel edge filter) – Image Gradient

$$G = \sqrt{G_w^2 + G_h^2},$$

$$\nabla I = (I_x, I_y) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)$$

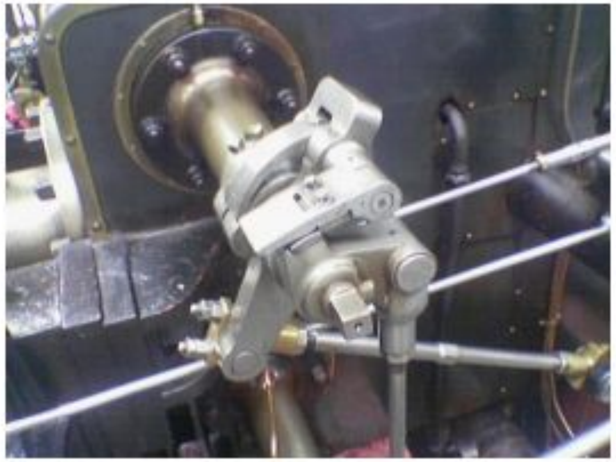
2D gradient of an image

$$\| \nabla I \| = \sqrt{I_x^2 + I_y^2}$$

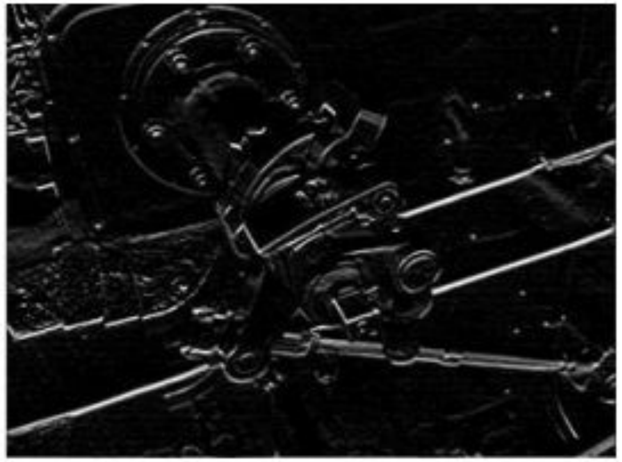
The gradient magnitude (edge strength)

SEF(Sobel edge filter)

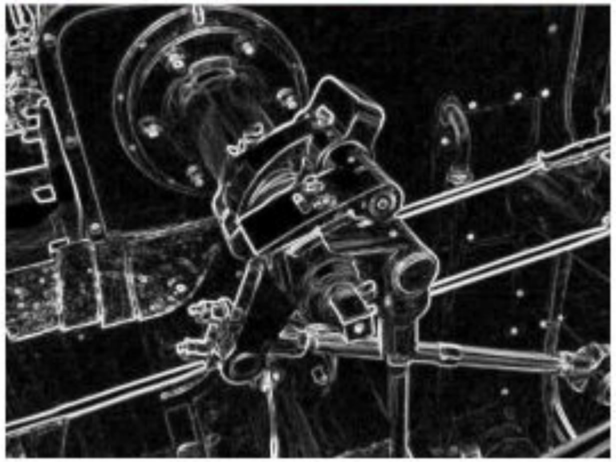
original



vertical derivative



gradient amplitude



horizontal derivative

Local Binary Pattern

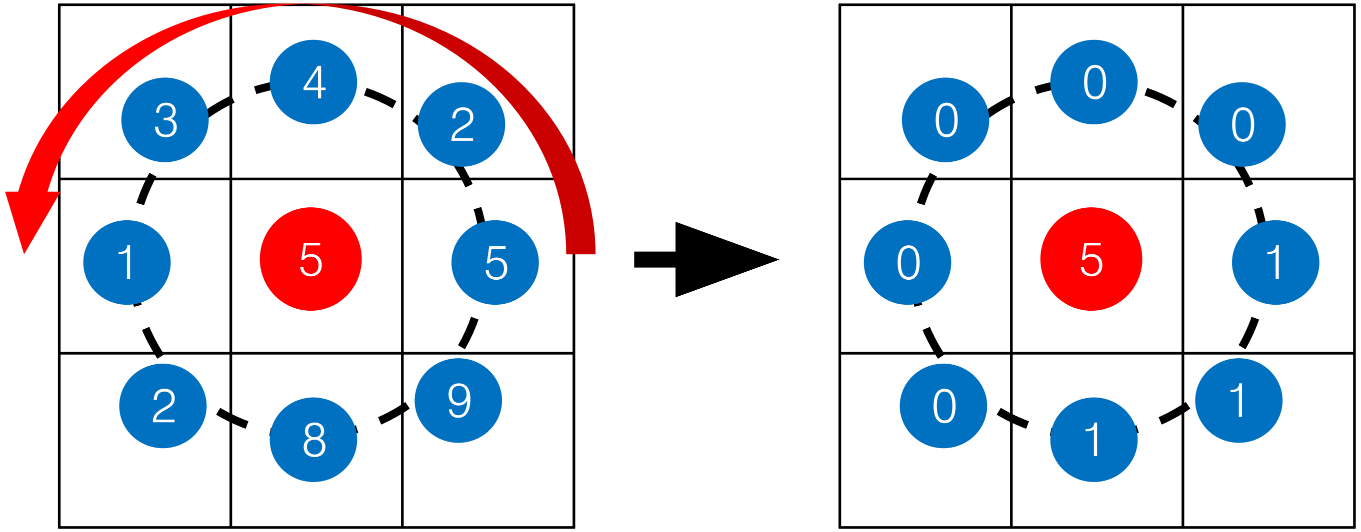
$$L_{p,R}(r_c, c_c) = \sum_{p=0}^{P-1} s(g_p - g_c)2^p$$

$$s(q) = \begin{cases} 1, q \geq 0 \\ 0, otherwise \end{cases}$$

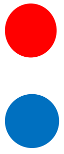
$$g_p = I(r_p, c_p), p = 0, \dots, P-1$$

$$r_p = r_c - R\sin\left(\frac{2\pi p}{P}\right)$$

$$c_p = c_c + R\cos\left(\frac{2\pi p}{P}\right)$$

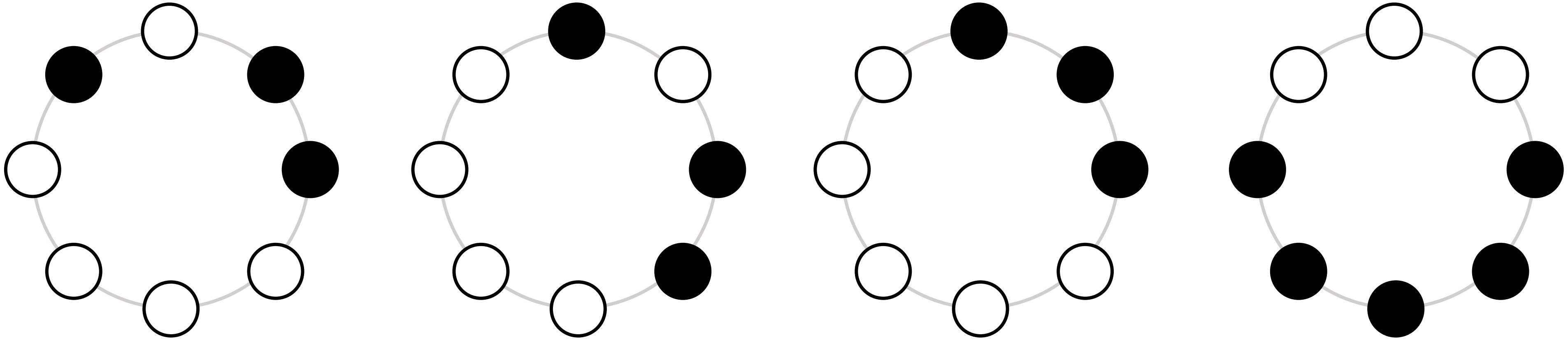


$$L_{8,1} = 11000001 = 193$$



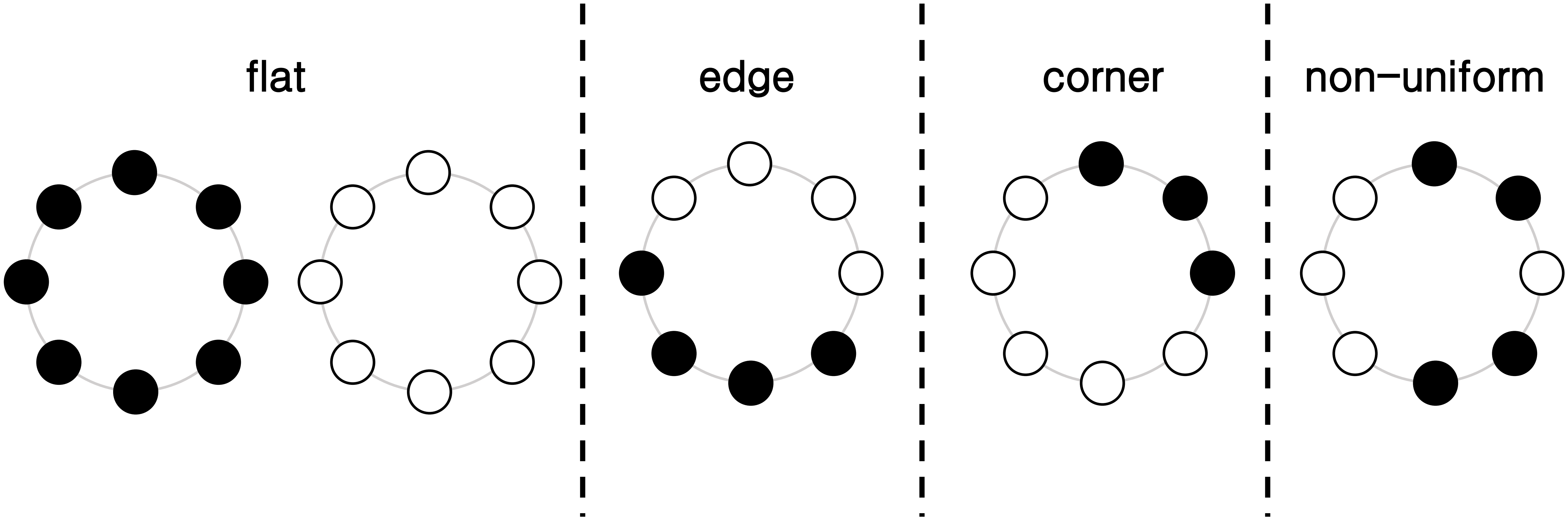
AAI Lab
Applied Artificial Intelligence

Local Binary Pattern

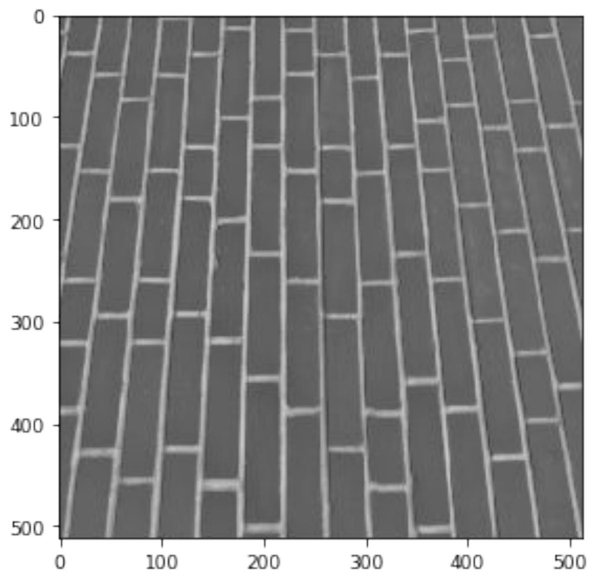


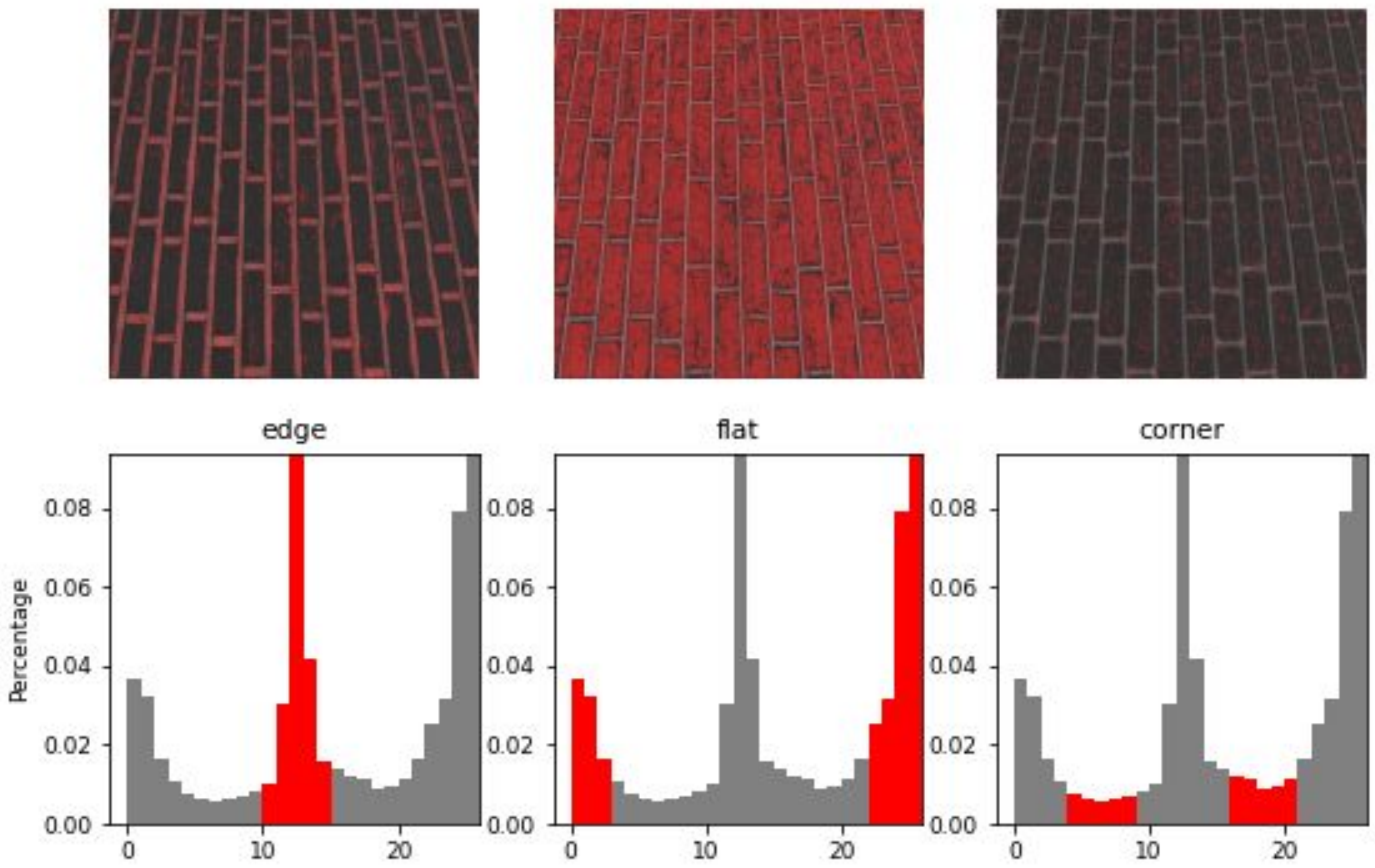
‘00001011’ = 13 ‘10000101’ = 134 ‘00000111’ =7 ‘11110001’ = 241

LBP(Local Binary Pattern)

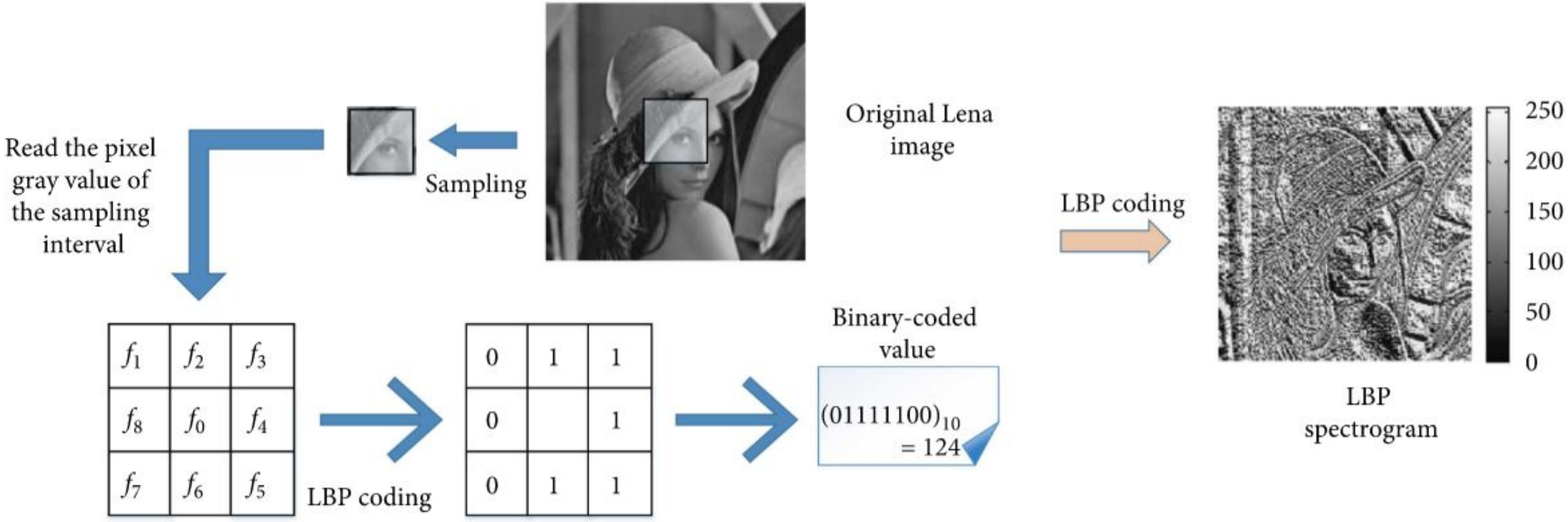


LBP(Local Binary Pattern)





LBP(Local Binary Pattern)



https://www.hindawi.com/journals/sv/2020/983016 2/

LTH(Local thresholding)

Separate objectsand and backgrounds based on the difference in contrast 에 주로 사용되지만 textures를 표현하는데도 사용된다. – 발표할 때 지움

Morpology기법에 Threshold를 설정해 이미지의 미세한 구조와 텍스처 특성을 뽑을 수 있음

$$T(r_c, c_c) = \begin{cases} 1, I(r_c, c_c) \geq m(r_c, c_c) \\ 0, \text{otherwise} \end{cases}$$

$$m(r_c, c_c) = \frac{1}{K^2} \sum_{i=c_c-\lfloor\frac{k}{2}\rfloor}^{c_c+\lfloor\frac{k}{2}\rfloor} \sum_{j=r_c-\lfloor\frac{k}{2}\rfloor}^{r_c+\lfloor\frac{k}{2}\rfloor} I(i,j)$$

AAI Lab
Applied Artificial Intelligence

LTH(Local thresholding)

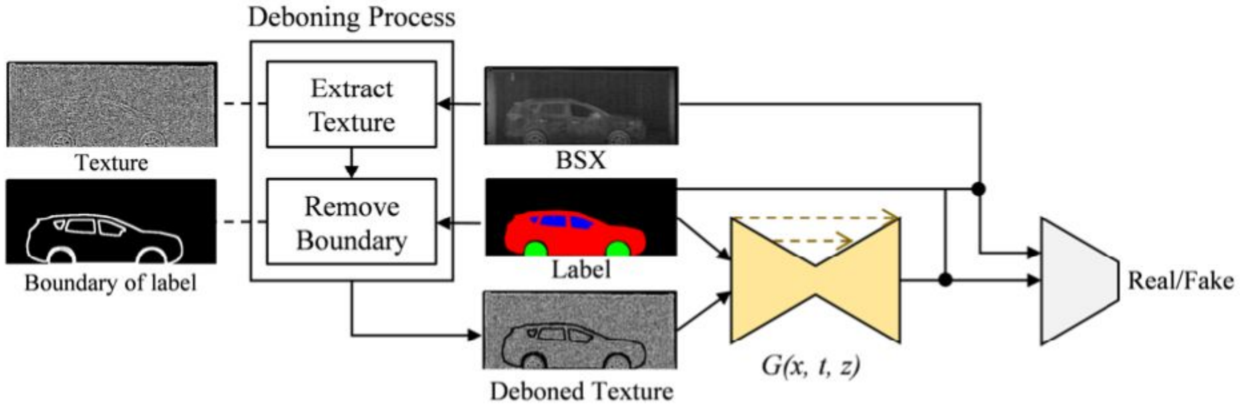
$$m(r_c, c_c) = \frac{1}{K^2} \sum_{i=c_c-\lfloor\frac{k}{2}\rfloor}^{c_c+\lceil\frac{k}{2}\rceil} \sum_{j=r_c-\lfloor\frac{k}{2}\rfloor}^{r_c+\lceil\frac{k}{2}\rceil} I(i,j)$$

$$k = 3 \rightarrow m(r_c, c_c) = \frac{1}{3^2} \sum_{i=c_c-1}^{c_c+2} \sum_{j=r_c-1}^{r_c+2} I(i,j)$$

$$T(r_c, c_c) = \begin{cases} 1, I(r_c, c_c) \geq m(r_c, c_c) \\ 0, \text{otherwise} \end{cases}$$

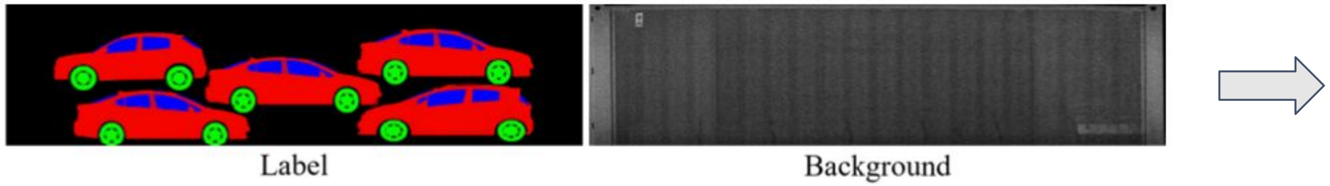
T(r_0,c_0)	T(r_0,c_1)	T(r_0,c_2)	…						
	I(r_c-1,c_c-1)	I(r_c-1,c_c)	I(r_c-1,c_c+1)	I(r_c-1,c_c+2)					
$\frac{1}{K^2}$ *	I(r_c,c_c-1)	**I(r_c,c_c)**	I(r_c,c_c+1)	I(r_c,c_c+2)					
	I(r_c+1,c_c-1)	I(r_c+1,c_c)	I(r_c+1,c_c+1)	I(r_c+1,c_c+2)					
	I(r_c+2,c_c-1)	I(r_c+2,c_c)	I(r_c+2,c_c+1)	I(r_c+2,c_c+2)					
						…	T(r_(n-2),c_(n-2))	T(r_(n-1),c_(n-1))	T(r_n,c_n)

Deboning



Proposed method to train the generative adversarial network by using the BSX texture image as an input.

Data Required for Inference



Deboned Texture

Experiment - Deboning

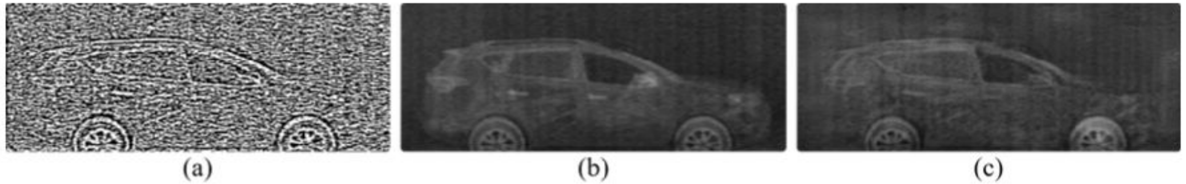
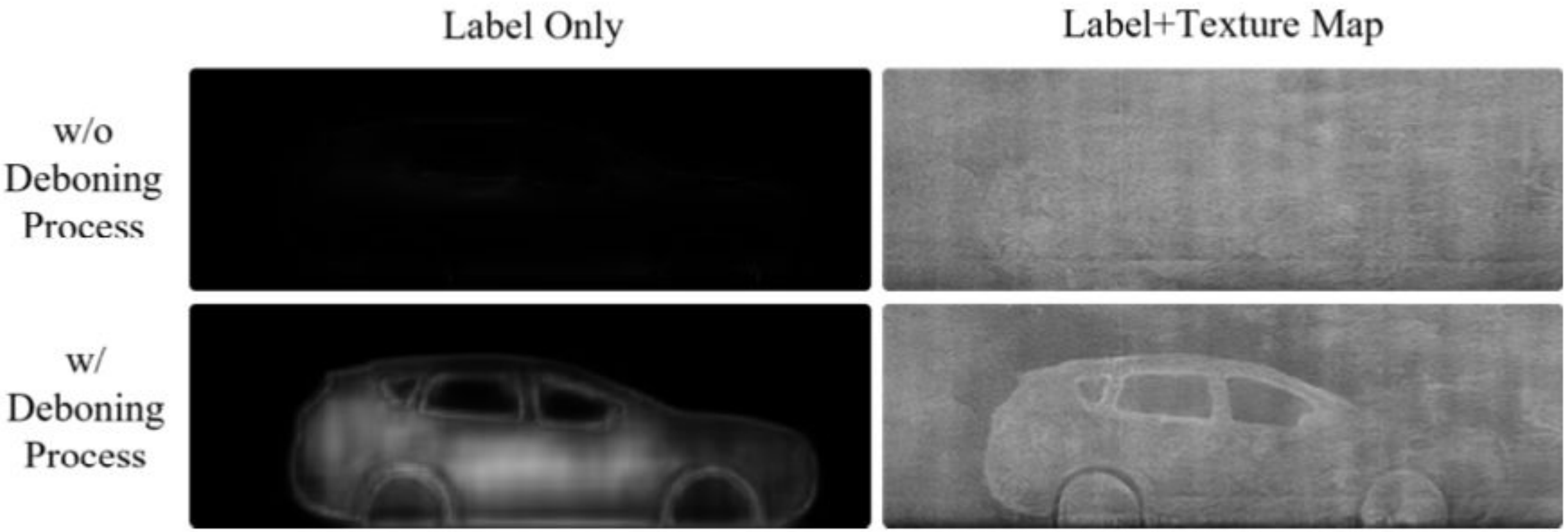
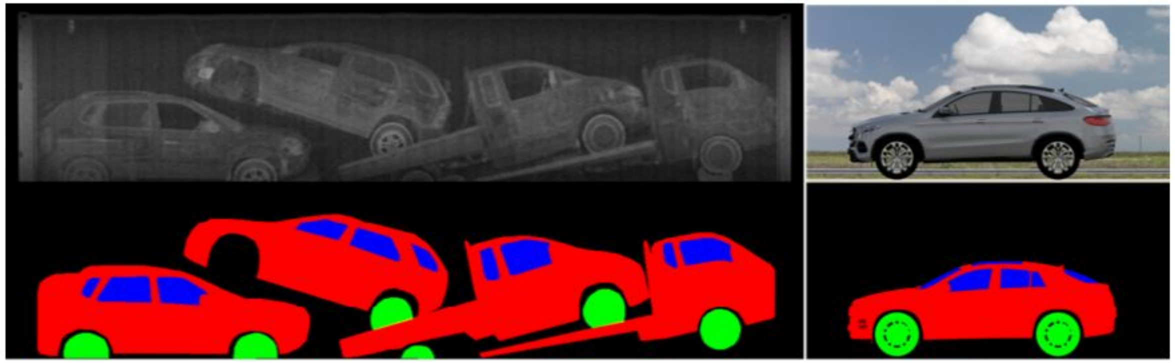


Figure 3. Image translation results when only the texture input was fed into the generator. The texture input (**a**) was extracted from a real BSX image (**b**), and the result was (**c**).



Experiment - Dataset

국세청의 1776개의 bsx 스캔 이미지, (Train - 1136 , Validation – 356, Test – 284)
1000개의 3D 자동차 모델 – only for label in testing phase



(a) (b)

Figure 5. Sample images of datasets: (**a**) Real BSX image and the corresponding segmentation label from the BSX-car dataset. (**b**) Rendered image of a 3D model and the corresponding segmentation label from the 3D-car dataset.

Experiments – FID

FID는 대상 도메인의 실제 이미지 모음 통계와 생성된 이미지 모음 통계를 비교해 평가를 진행

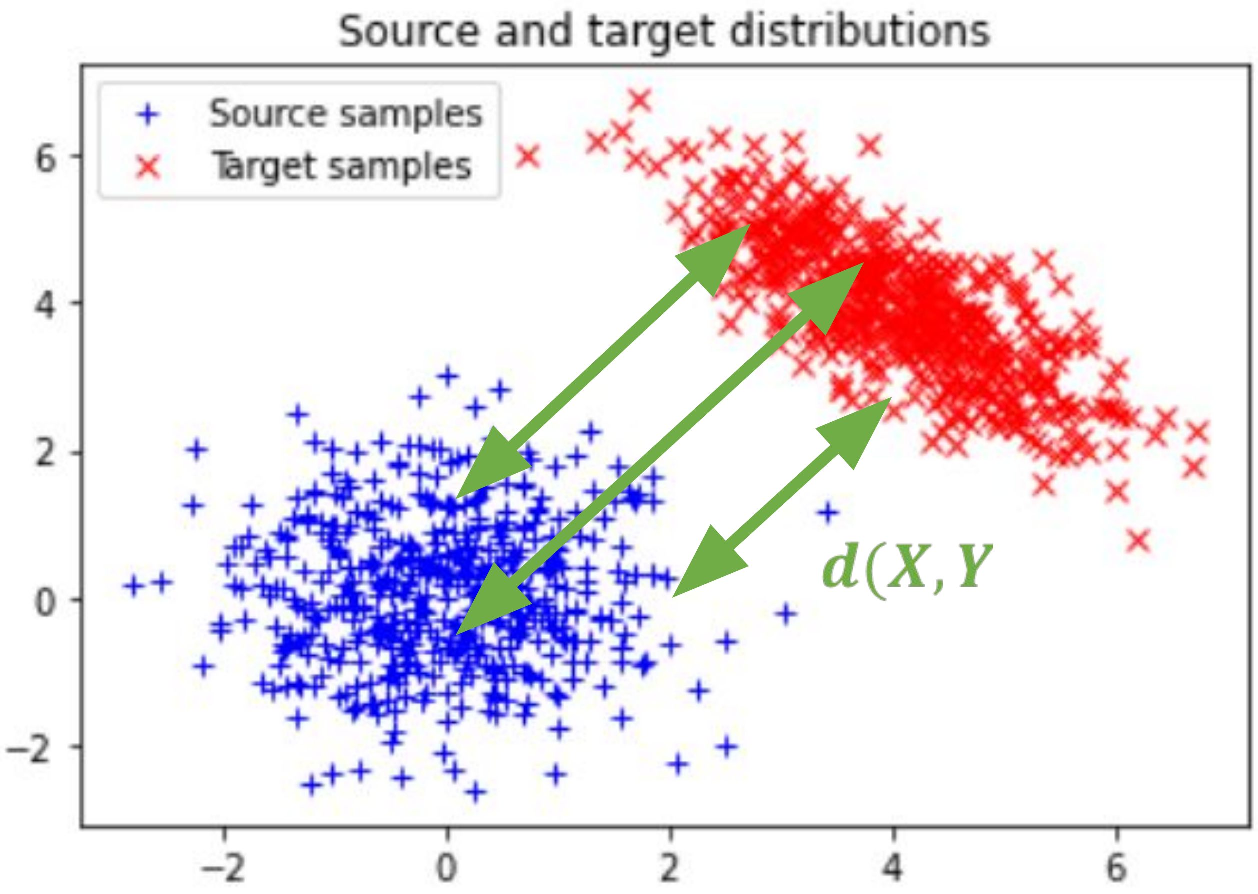
FID 점수는 GAN에 의해 생성된 이미지의 품지를 평가하는데 사용되며, 낮은 점수는 고품질 이미지와 연관 됨

FID는 대상 도메인의 실제 이미지 모음 통계와 생성된 이미지 모음 통계를 비교해 평가를 진행

FID 점수가 낮을수록 Generator에 의해 생성된 이미지의 품질이 더 높고 실제와 유사함을 나타맴

Lower FID는 synthetic과 real data 분포 사이의 거리가 더 작다는 것을 의미

Wassertein distance



Wassertein distance

$d(X,Y)$의 기대 값이 가장 작게 나오는 확률 분포

→ Wassertein distance가 작을 수록 좋은 값이다.

Experiments

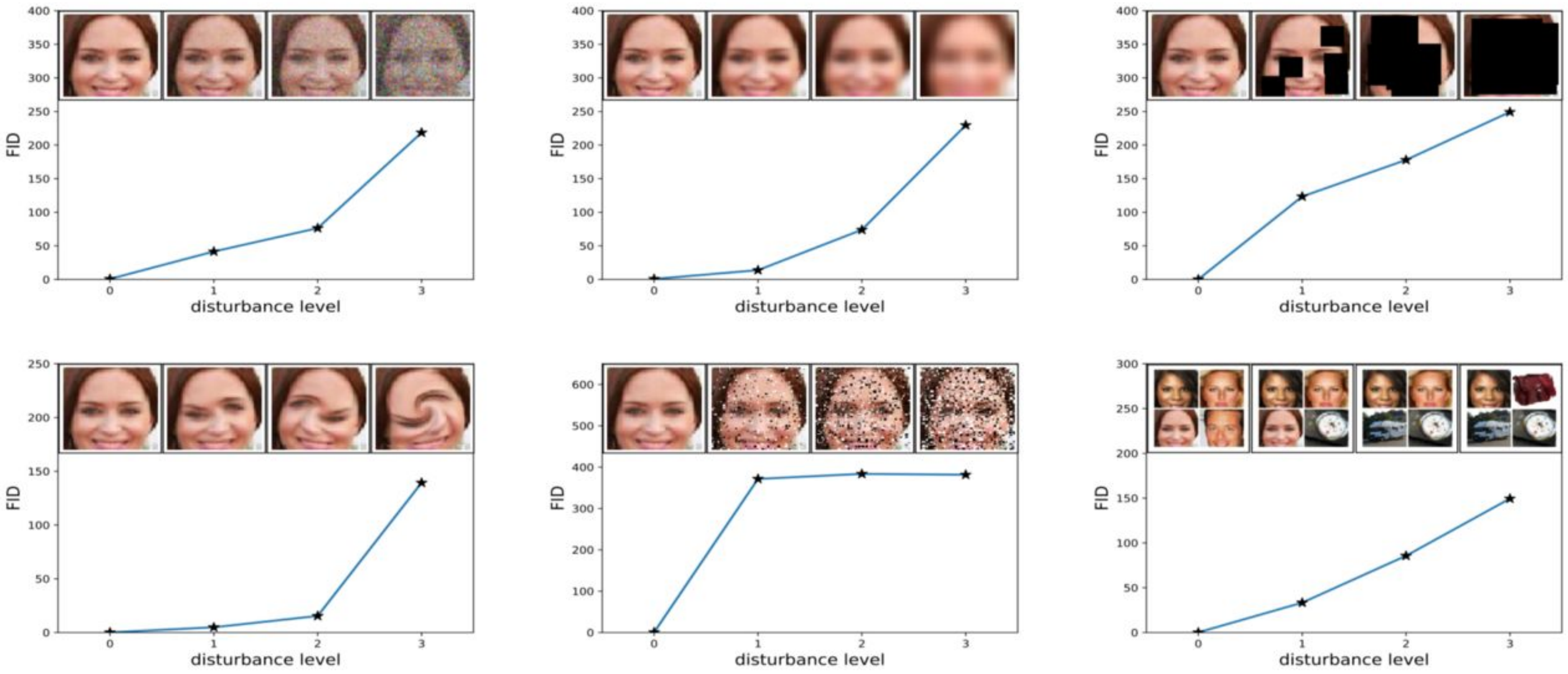


Figure 3: FID is evaluated for **upper left:** Gaussian noise, **upper middle:** Gaussian blur, **upper right:** implanted black rectangles, **lower left:** swirled images, **lower middle:** salt and pepper noise, and **lower right:** CelebA dataset contaminated by ImageNet images. The disturbance level rises from zero and increases to the highest level. The FID captures the disturbance level very well by monotonically increasing.

AAI Lab
Applied Artificial Intelligence

Experiments

Table 1. FID and SWD of images generated according to the texture map.

Generated Image	FID	SWD
Fake (pix2pix)	43.9	1393.97
Fake (SEF)	42.1	1283.53
Fake (LTH)	28.1	1133.91
Fake (LBP)	**27.6**	**892.97**

Table 2. Effect of fake images on the segmentation performance.

	Accuracy		mIoU	
Training Dataset	L_{3D}	L_{BSX}	L_{3D}	L_{BSX}
Real	0.909		0.715	
Real + Fake (pix2pix)	0.921	0.915	0.764	0.726
Real + Fake (SEF)	0.914	0.914	0.748	0.739
Real + Fake (LTH)	**0.925**	**0.921**	**0.773**	0.754
Real + Fake (LBP)	0.920	0.918	0.746	**0.757**
Fake(pix2pix)	0.677	0.793	0.250	0.468
Fake(SEF)	0.708	0.798	0.293	0.489
Fake(LTH)	**0.772**	**0.867**	**0.407**	**0.621**
Fake(LBP)	0.748	0.839	0.395	0.563

Experiments