BE530 – Medical Deep Learning

- Representative GANs -

Byoung-Dai Lee

Division of AI Computer Science and Engineering

Kyonggi University

KYONGGI UNIVERSITY

- Perceptual Loss
  - Perceptual Losses for Real-Time Style Transfer and Super Resolution (2016)
- Multimodal Unsupervised Image-to-Image Translation (MUNIT)
  - Multimodal Unsupervised Image-to-Image Translation (2018)
- Cycle-Consistent Network
  - CycleGAN: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks (2017)

Perceptual Loss

- Key Objectives
  - To train feed-forward transformation networks for image transformation tasks using perceptual loss functions that depend on high-level features from a pre-trained loss network

- Perceptual loss function
  - To measure high-level perceptual and semantic differences between images

- Insights
  - Convolutional neural networks pretrained for image classification have already learned to encode the perceptual and semantic information we would like to measure in our loss functions

Perceptual Loss (cont.)

- System overview
  - An image transformation network + a loss network

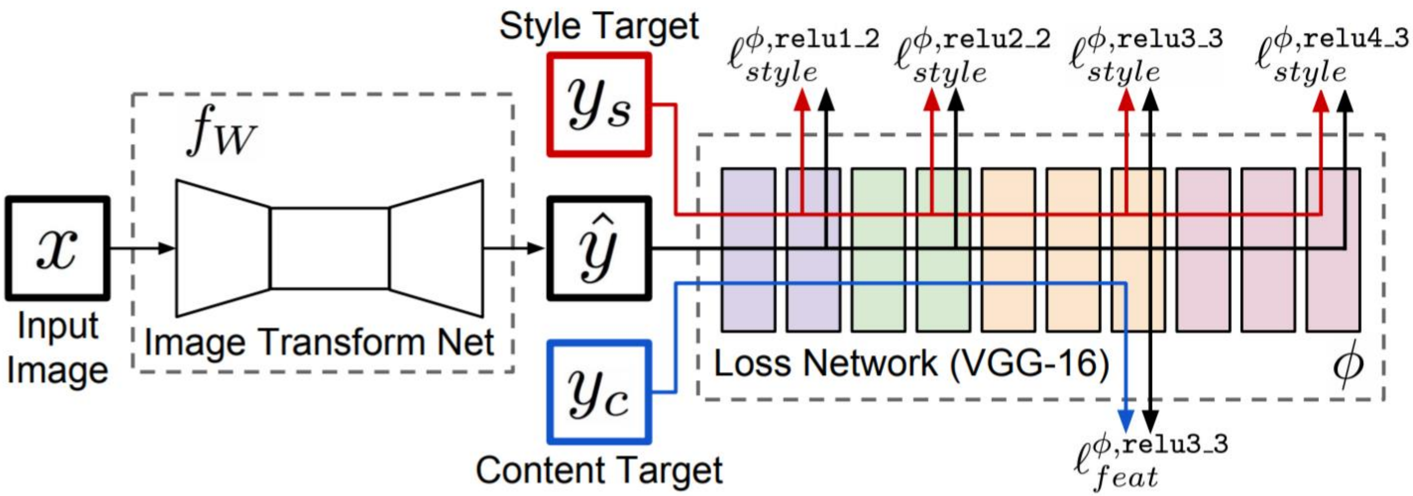


Fig. 2. System overview. We train an *image transformation network* to transform input images into output images. We use a *loss network* pretrained for image classification to define *perceptual loss functions* that measure perceptual differences in content and style between images. The loss network remains fixed during the training process.

Perceptual Loss (cont.)

Loss network

The loss network ϕ is used to define a *feature reconstruction loss* ℓ_{feat}^{ϕ} and a *style reconstruction loss* ℓ_{style}^{ϕ} that measure differences in content and style between images. For each input image x we have a *content target* y_c and a *style target* y_s. For style transfer, the content target y_c is the input image x and the output image $\hat{y}$ should combine the content of $x = y_c$ with the style of y_s; we train one network per style target. For single-image super-resolution, the input image x is a low-resolution input, the content target y_c is the ground-truth high-resolution image, and the style reconstruction loss is not used; we train one network per super-resolution factor.

Image transformation network

- Style transfer
  - Input/Output images – 256×256×3 color images

Perceptual Loss (cont.)

- Perceptual loss function
  - <u>Feature reconstruction loss</u>
    - To verify if the output image has similar feature representations
    - The (squared, normalized) Euclidean distance between feature representations of the layer of the loss network

$$\ell_{feat}^{\phi,j}(\hat{y}, y) = \frac{1}{C_j H_j W_j} \|\phi_j(\hat{y}) - \phi_j(y)\|_2^2$$

$\phi_j(x)$ be the activations of the jth layer of the network

$\hat{y} = f_W(x)$

    - The feature reconstruction loss penalizes the output image when it deviates in content from the target image
      - Using a feature reconstruction loss for training the image transformation networks encourages the output image to be perceptually similar to the target image
      - To minimize the feature reconstruction loss for early layers tends to produce images that are visually indistinguishable from the target image

KYONGGI UNIVERSITY

Perceptual Loss (cont.)

- Perceptual loss function (cont.)
  - Style reconstruction loss
    - It penalizes differences in style; colors, textures, common patterns, etc.
    - Gram matrix

G(i,j)는 채널(i), 채널(j)의 feature map(H×W)간의 내적.
따라서 G(i,j)가 큰 값을 가진다는 것은 두 채널이
동시에 activation된다는 것을 의미함

$G_j^\phi(x)$ to be the $C_j \times C_j$ matrix

$$G_j^\phi(x)_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(x)_{h,w,c} \phi_j(x)_{h,w,c'}.$$

    - Style reconstruction loss

$$\ell_{style}^{\phi,j}(\hat{y}, y) = \|G_j^\phi(\hat{y}) - G_j^\phi(y)\|_F^2$$

    - Minimizing the style reconstruction loss preserves stylistic feature from the target image, but does not preserve its spatial structure

Perceptual Loss (cont.)

- Simple loss functions that depend only on low-level pixel information
  - Pixel Loss
    - The (normalized) Euclidean distance between the output image and the target image

$$\ell_{pixel}(\hat{y}, y) = \|\hat{y} - y\|_2^2 / CHW$$

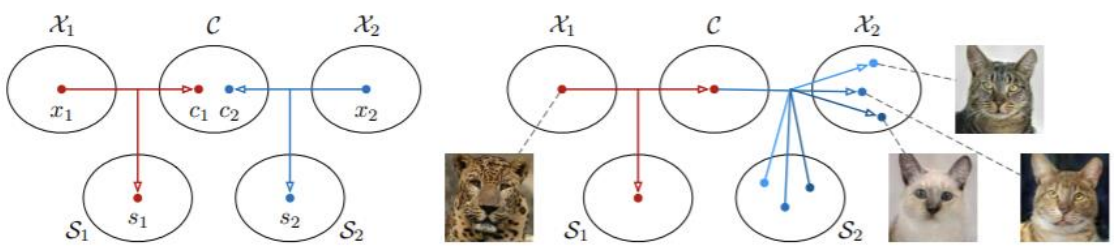
  - Total Variation Regularization
    - To encourage spatial smoothness in the output image

MUNIT

- **Key Objective**
  - Unsupervised image-to-image translation
  - Multimodal mapping
    - The model can generate diverse outputs from a given source domain image
- **Learning disentangled representation**
  - Disentangling content from style
    - Content – The underling spatial structure
    - Style – The rendering of the structure

MUNIT (cont.)

- Assumptions
  - The latent space of images can be decomposed into a content space and a style space
  - Images in different domains share a common content space but not the style space
  - The content code encodes the information that should be preserved during translation, while the style code represents remaining variations that are not contained in the input image

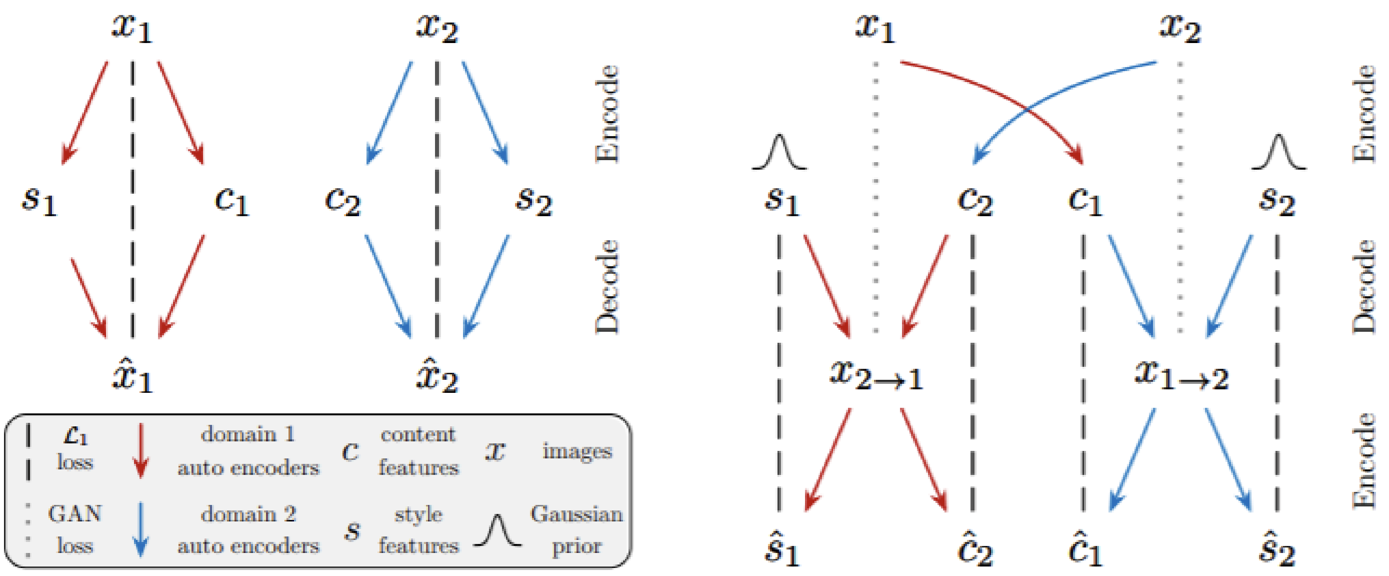


(a) Auto-encoding (b) Translation

Fig. 1. An illustration of our method. (a) Images in each domain $\mathcal{X}_i$ are encoded to a shared content space $\mathcal{C}$ and a domain-specific style space $\mathcal{S}_i$. Each encoder has an inverse decoder omitted from this figure. (b) To translate an image in $\mathcal{X}_1$ (*e.g.*, a leopard) to $\mathcal{X}_2$ (*e.g.*, domestic cats), we recombine the content code of the input with a random style code in the target style space. Different style codes lead to different outputs.

MUNIT (cont.)

- Model



(a) Within-domain reconstruction (b) Cross-domain translation

Fig. 2 shows an overview of our model and its learning process. Similar to Liu *et al.* [15], our translation model consists of an encoder E_i and a decoder G_i for each domain $\mathcal{X}_i$ $(i = 1, 2)$. As shown in Fig. 2 (a), the latent code of each auto-encoder is factorized into a content code c_i and a style code s_i, where $(c_i, s_i) = (E_i^c(x_i), E_i^s(x_i)) = E_i(x_i)$. Image-to-image translation is performed by swapping encoder-decoder pairs, as illustrated in Fig. 2 (b). For example, to translate an image $x_1 \in \mathcal{X}_1$ to $\mathcal{X}_2$, we first extract its content latent code $c_1 = E_1^c(x_1)$ and randomly draw a style latent code s_2 from the prior distribution $q(s_2) \sim \mathcal{N}(0, \mathbf{I})$. We then use G_2 to produce the final output image $x_{1\to2} = G_2(c_1, s_2)$.

KYONGGI UN

MUNIT (cont.)

- Loss function
  - Bidirectional reconstruction loss
    - Image reconstruction

$$\mathcal{L}^{x_1}_{\text{recon}} = \mathbb{E}_{x_1 \sim p(x_1)}[||G_1(E^c_1(x_1), E^s_1(x_1)) - x_1||_1]$$

    - Latent reconstruction

$$\mathcal{L}^{c_1}_{\text{recon}} = \mathbb{E}_{c_1 \sim p(c_1), s_2 \sim q(s_2)}[||E^c_2(G_2(c_1, s_2)) - c_1||_1]$$
$$\mathcal{L}^{s_2}_{\text{recon}} = \mathbb{E}_{c_1 \sim p(c_1), s_2 \sim q(s_2)}[||E^s_2(G_2(c_1, s_2)) - s_2||_1]$$

  - Adversarial loss

$$\mathcal{L}^{x_2}_{\text{GAN}} = \mathbb{E}_{c_1 \sim p(c_1), s_2 \sim q(s_2)}[\log(1 - D_2(G_2(c_1, s_2)))] + \mathbb{E}_{x_2 \sim p(x_2)}[\log D_2(x_2)]$$

  - Total loss

$$\min_{E_1, E_2, G_1, G_2} \max_{D_1, D_2} \mathcal{L}(E_1, E_2, G_1, G_2, D_1, D_2) = \mathcal{L}^{x_1}_{\text{GAN}} + \mathcal{L}^{x_2}_{\text{GAN}} +$$
$$\lambda_x(\mathcal{L}^{x_1}_{\text{recon}} + \mathcal{L}^{x_2}_{\text{recon}}) + \lambda_c(\mathcal{L}^{c_1}_{\text{recon}} + \mathcal{L}^{c_2}_{\text{recon}}) + \lambda_s(\mathcal{L}^{s_1}_{\text{recon}} + \mathcal{L}^{s_2}_{\text{recon}})$$

MUNIT (cont.)

- The architecture of auto−encoder

$$\mathrm{AdaIN}(x,y) = \sigma(y)\left(\frac{x-\mu(x)}{\sigma(y)}\right)+\mu(y)$$

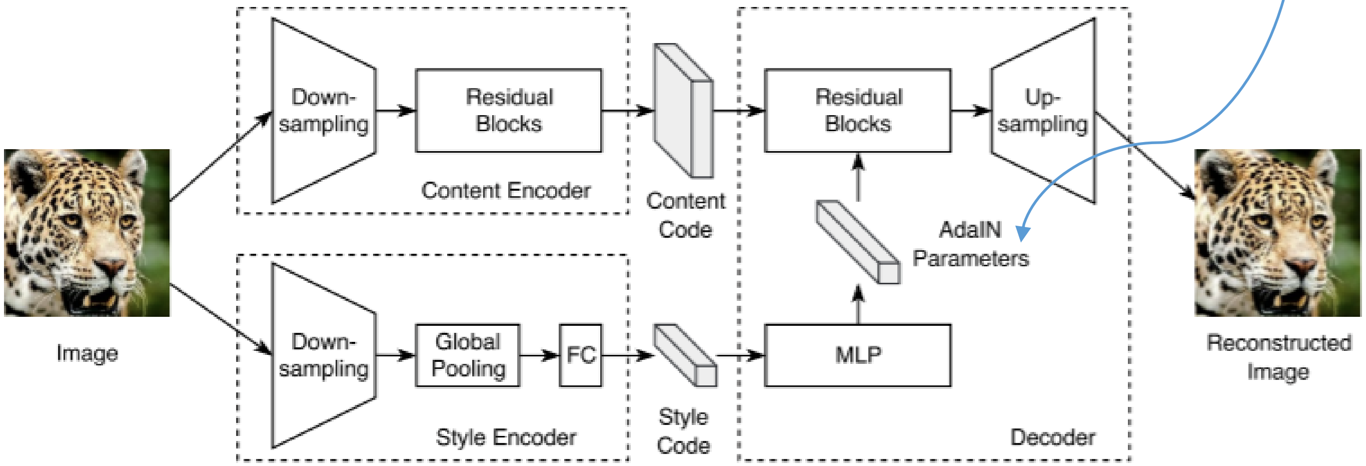
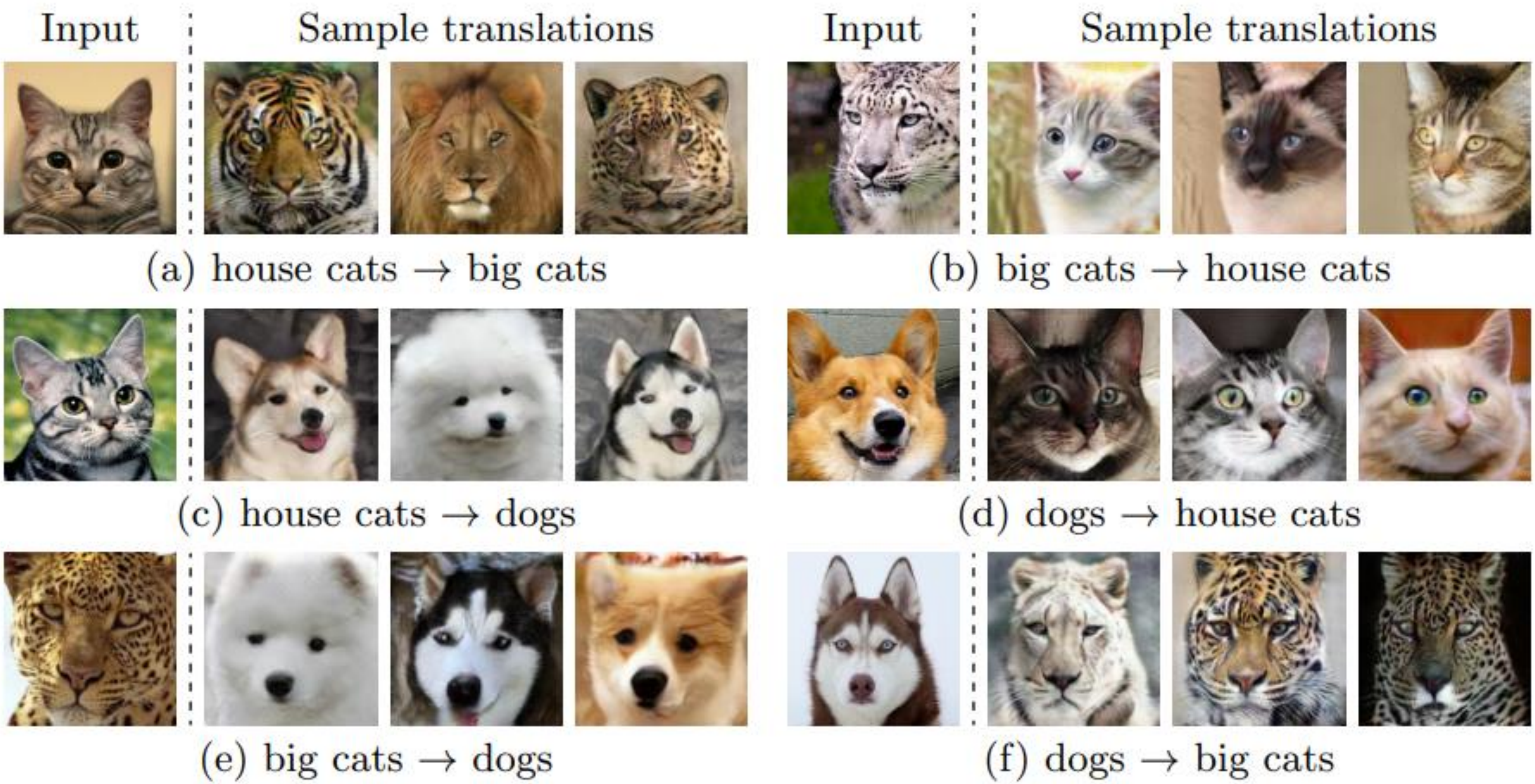


Fig. 3. Our auto-encoder architecture. The content encoder consists of several strided convolutional layers followed by residual blocks. The style encoder contains several strided convolutional layers followed by a global average pooling layer and a fully connected layer. The decoder uses a MLP to produce a set of AdaIN [54] parameters from the style code. The content code is then processed by residual blocks with AdaIN layers, and finally decoded to the image space by upsampling and convolutional layers.

MUNIT (cont.)

(a) edges ↔ shoes (b) edges ↔ handbags

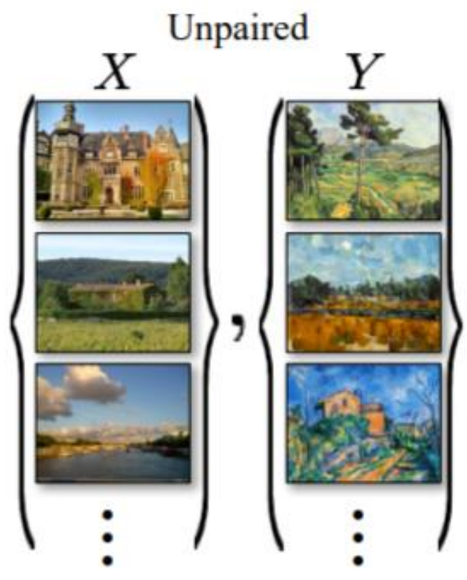
MUNIT (cont.)



Input
Sample translations
Input
Sample translations
(a) house cats → big cats
(b) big cats → house cats
(c) house cats → dogs
(d) dogs → house cats
(e) big cats → dogs
(f) dogs → big cats

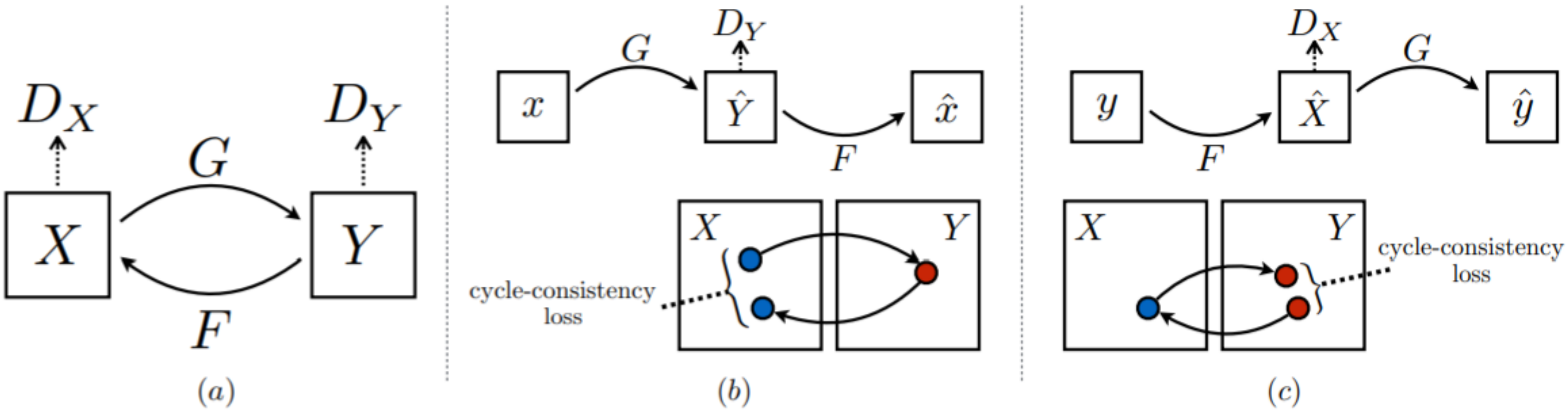
Cycle-Consistent GAN

- Key Objective
  - Capturing special characteristics of one image collection and figuring out how these characteristics could be translated into the other image collection, all in the absence of any paired training examples



Cycle–Consistent GAN (cont.)

- Model



- Loss function

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) =& \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda \mathcal{L}_{\text{cyc}}(G, F),\end{aligned}$$

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) =& \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] \\ &+ \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))],\end{aligned}$$

$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G, F) =& \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] \\ &+ \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1].\end{aligned}$$

KYONGGI UNIVERSITY

Cycle-Consistent GAN (cont.)

Implementation

- Network architecture
  - Generator – based on the DNN of the perceptual losses paper
  - Discriminator – 70×70 PatchGAN
- Input image resolutions
  - 128×128 or 256×256

Training details

- To replace the negative log likelihood objective by a least-square loss

$$\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))],$$

→

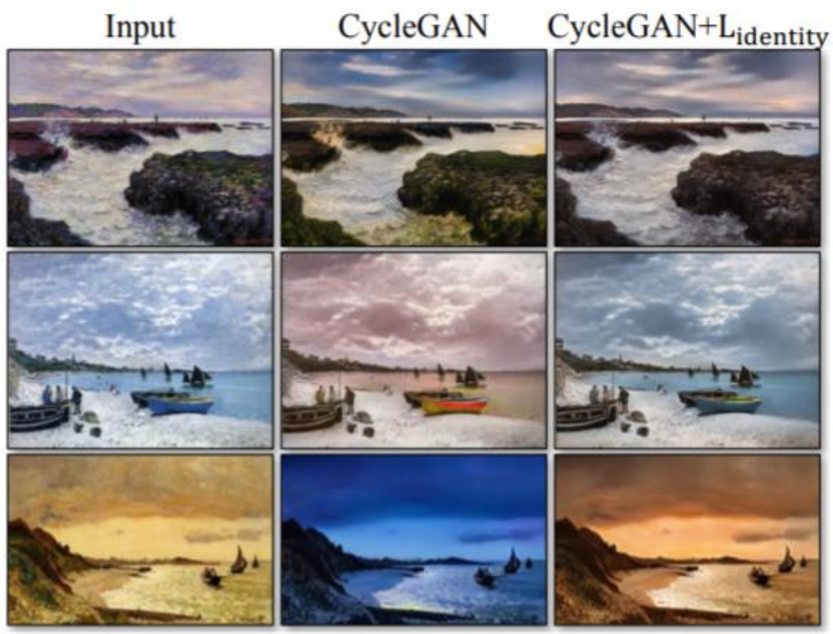
we train the G to minimize $\mathbb{E}_{x \sim p_{\text{data}}(x)}[(D(G(x)) - 1)^2]$ and train the D to minimize $\mathbb{E}_{y \sim p_{\text{data}}(y)}[(D(y) - 1)^2] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[D(G(x))^2]$.

- To update the discriminator
  - Using a history of generated images rather than the ones produced by the latest generators
  - Previously created 50 images are kept in the buffer

Cycle-Consistent GAN (cont.)

- Identity Mapping
  - G(x) → y로 변환하는 맵핑의 경우, y가 G의 입력으로 사용될 경우 새로운 값으로 변환하지 않고 원래 자신, 즉 y를 반환하게 함으로써 입력 이미지의 컬러값을 유지하도록 함

$$\mathcal{L}_{\text{identity}}(G, F) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(y) - y\|_1] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(x) - x\|_1]$$



Cycle-Consistent GAN (cont.)

- Limitations
  - Color and texture changes → showed good performance
  - Geometric changes → little success
  - The distribution characteristics of the training datasets
    - "A person riding a horse" 이미지가 학습데이터에 없을 경우 유사한 이미지가 입력으로 제공될 경우 translation되는 이미지의 품질은 나쁠 수 있음

References

- Perceptual Losses for Real-Time Style Transfer and Super Resolution, https://arxiv.org/abs/1603.08155
- CycleGAN: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, https://arxiv.org/abs/1703.10593
- Multimodal Unsupervised Image-to-Image Translation, https://arxiv.org/pdf/1804.04732.pdf
- Arbitrary Style Transfer in Real-Time With Adaptive Instance Normalization, https://openaccess.thecvf.com/content_ICCV_2017/papers/Huang_Arbitrary_Style_Transfer_ICCV_2017_paper.pdf


ANY
QUESTIONS?