# Machine Learning Loss Functions

Han, Yoseob

Theoretical Division,
T-5 Applied Mathematics and Plasma Physics,
Los Alamos National Laboratory (LANL)
Los alamos, NM 87545, USA

E-mail: hanyosub@gmail.com

February 9, 2020

# 1 Supervised learning

In a supervised learning scheme, our goal is finding an optimal generator $G$ constructed by trainable parameters $\theta_g$ and the optimal generator $G$ induces a **minimum value of loss function** $\mathcal{L}(G)$ as expressed in Eq. 1.

$$G^* = \arg\min_G \mathcal{L}(G). \tag{1}$$

## 1.1 L1 Loss (= Mean Absolute Error Loss; MAE Loss)

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y}[|y - G(x;\theta_g)|], \tag{2}$$

where $G$ is generator, and $\theta_g$ is trainable parameters such as convolution kernel ($\omega$) and bias($b$). $x$ and $y$ are input and target data, respectively.

## 1.2 L2 loss (= Mean Squared Error Loss; MSE Loss)

$$\mathcal{L}_{L2}(G) = \mathbb{E}_{x,y}[||y - G(x;\theta_g)||_2^2], \tag{3}$$

where $G$ is generator, and $\theta_g$ is trainable parameters such as convolution kernel ($\omega$) and bias($b$). $x$ and $y$ are input and target data, respectively.

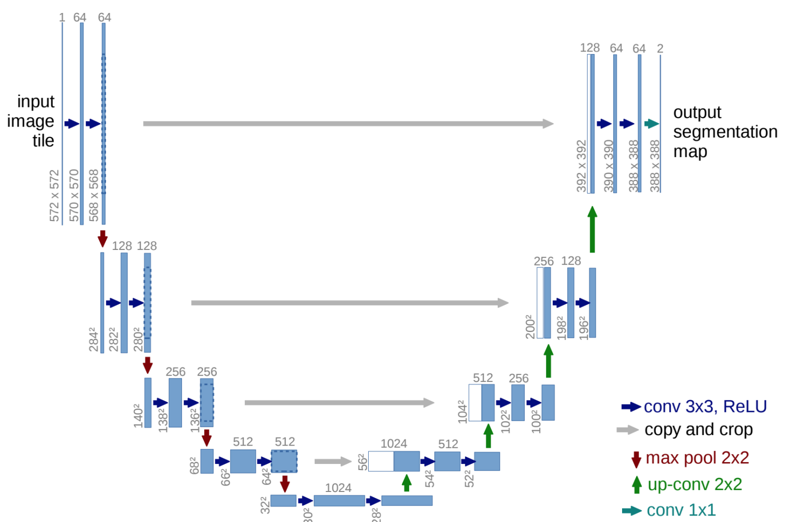


**Fig. 1.** U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

Figure 1: U-Net [1] is one of examples for supervised learning.

## 2 Unsupervised learning

In a unsupervised learning scheme, our goal is finding an optimal generator $G$ and discriminator $D$ constructed by trainable parameters $\theta_g$ and $\theta_D$, respectively. The optimal generator $G$ induces a minimum value of loss function $\mathcal{L}(G)$, but the optimal discriminator $D$ induces a maximum value of loss. The optimization problem related with between generator $G$ and discriminator $D$ is called by **minimax game** as expressed in Eq. 4.

$$G^*, D^* = \arg\min_G \max_D \mathcal{L}(G, D). \tag{4}$$

### 2.1 Generative Adversarial Network (GAN) [2, 3]

$$\begin{aligned} \mathcal{L}_{GAN}(G, D) &= \mathbb{E}_{x\sim p_{data}(x)}[\log D(x;\theta_d)] \\ &+ \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z;\theta_g);\theta_d))], \end{aligned} \tag{5}$$

where $G$ and $D$ are generator and discriminator, respectively, and its $\theta_g$ and $\theta_d$ are trainable parameters such as convolution kernel ($\omega$) and bias($b$). $z$ and $y$ are input (Gaussian and/or normal noise) and target (image) data, respectively, and its $p_{data}(x)$ and $p_z(z)$ are data distributions.

$G$ generates a fake sample $\tilde{x} = G(z;\theta_g)$ in $p_{data}(x)$ domain from a noise $z$ in $p_z(z)$ domain. For true data $x \sim p_{data}(x)$ and synthesized data $\tilde{x} = G(z;\theta_g)$, $D$ distinguishes whether a given data belongs to $p_{data}(x)$ domain.

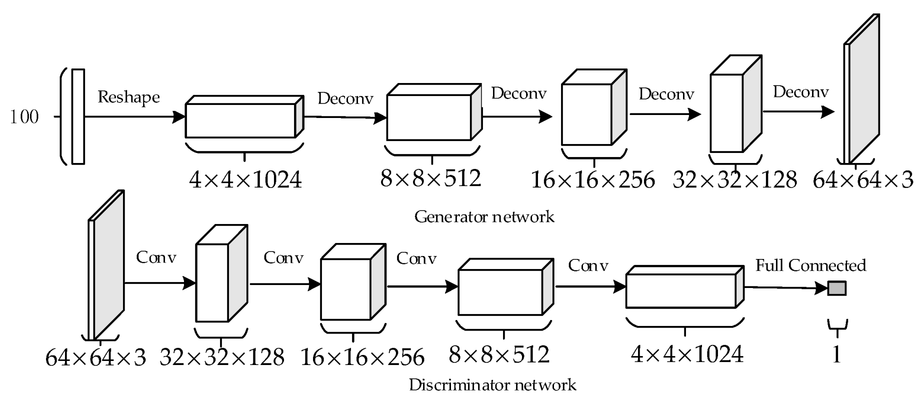


Figure 2: Standard GAN [2, 3]. (top) Generator network architecture ($G$), and (bottom) Discriminator ($D$) network architecture.

## 2.2 pix2pix: Conditional GAN (cGAN) [4]

$$\mathcal{L}_{\text{pix2pix}}(G, D) = \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G), \tag{6}$$

where $\mathcal{L}_{cGAN}(G, D)$ is an objective function of a conditional GAN and $\mathcal{L}_{L1}(G)$ is an objective function of a L1 loss. $\lambda$ is hyper-parameter that control the relative importance of the two objectives. $\mathcal{L}_{cGAN}(G, D)$ and $\mathcal{L}_{L1}(G)$ are defined by Eqs. 7 and 8, respectively.

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y; \theta_d)] + \mathbb{E}_x[\log(1 - D(x, G(x; \theta_g); \theta_d))], \tag{7}$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y}[|y - G(x; \theta_g)|], \tag{8}$$

where $G$ and $D$ are generator and discriminator, respectively, and its $\theta_g$ and $\theta_d$ are trainable parameters such as convolution kernel ($\omega$) and bias($b$). $x$ and $y$ are input and target data, respectively.

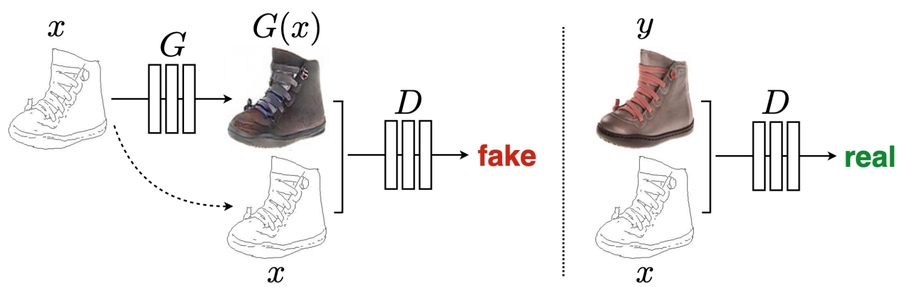


Figure 2: Training a conditional GAN to map edges→photo. The discriminator, $D$, learns to classify between fake (synthesized by the generator) and real {edge, photo} tuples. The generator, $G$, learns to fool the discriminator. Unlike an unconditional GAN, both the generator and discriminator observe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

Figure 3: Overview of pix2pix [4]. Actually, pix2pix [4] is not an unsupervised learning because they need a paired dataset.

### 2.3 CycleGAN [5]

$$\begin{aligned}\mathcal{L}_{\text{cycleGAN}}(G_{X\to Y}, G_{Y\to X}, D_X, D_Y) &= \mathcal{L}_{GAN}(G_{X\to Y}, D_Y) \\ &+ \mathcal{L}_{GAN}(G_{Y\to X}, D_X) \\ &+ \lambda\mathcal{L}_{cyc}(G_{X\to Y}, G_{Y\to X}) \\ &+ \rho\mathcal{L}_{identity}(G_{X\to Y}, G_{Y\to X}), \quad (9)\end{aligned}$$

where $\mathcal{L}_{GAN}(G_{X\to Y}, D_Y)$ and $\mathcal{L}_{GAN}(G_{Y\to X}, D_X)$ are an objective function of a GAN, $\mathcal{L}_{cyc}(G_{X\to Y}, G_{Y\to X})$ is an objective function of a cycle consistency loss and $\mathcal{L}_{identity}(G_{X\to Y}, G_{X\to Y})$ is an objective function of an identity loss. $\lambda$ and $\rho$ are hyper-parameters that control the relative importance. $\mathcal{L}_{GAN}(G, D)$, $\mathcal{L}_{cyc}(G_{X\to Y}, G_{Y\to X})$, and $\mathcal{L}_{identity}(G_{X\to Y}, G_{Y\to X})$ are defined by Eqs. 10, 11, and 12, respectively.

$$\begin{aligned}\mathcal{L}_{GAN}(G_{X\to Y}, D_Y) &= \mathbb{E}_{y\sim p_{data}(y)}[\log D_Y(y;\theta_d^y)] \quad (10a)\\ &+ \mathbb{E}_{x\sim p_{data}(x)}[\log(1 - D_Y(G_{X\to Y}(x;\theta_g^{X\to Y});\theta_d^y))],\end{aligned}$$

$$\begin{aligned}\mathcal{L}_{GAN}(G_{Y\to X}, D_X) &= \mathbb{E}_{x\sim p_{data}(x)}[\log D_X(x;\theta_d^x)] \quad (10b)\\ &+ \mathbb{E}_{y\sim p_{data}(y)}[\log(1 - D_X(G_{Y\to X}(y;\theta_g^{Y\to X});\theta_d^x))],\end{aligned}$$

$$\begin{aligned}&\mathcal{L}_{cyc}(G_{X\to Y}, G_{Y\to X}) \quad (11)\\ &= \mathbb{E}_{x\sim p_{data}(x)}[|G_{Y\to X}(G_{X\to Y}(x;\theta_g^{X\to Y});\theta_g^{Y\to X}) - x|]\\ &+ \mathbb{E}_{y\sim p_{data}(y)}[|G_{X\to Y}(G_{Y\to X}(y;\theta_g^{Y\to X});\theta_g^{X\to Y}) - y|],\end{aligned}$$

$$\begin{aligned}\mathcal{L}_{identity}(G_{X\to Y}, G_{Y\to X}) &= \mathbb{E}_{y\sim p_{data}(y)}[|G_{Y\to X}(x;\theta_g^{Y\to X}) - x|] \quad (12)\\ &+ \mathbb{E}_{x\sim p_{data}(x)}[|G_{X\to Y}(y;\theta_g^{X\to Y}) - y|],\end{aligned}$$

where $G$ and $D$ are generator and discriminator, respectively, and its $\theta_g$ and $\theta_d$ are trainable parameters such as convolution kernel ($\omega$) and bias($b$). $x$ and $y$ are data for each difference classes, respectively, and $p_{data}(x)$ and $p_{data}(y)$ are its data distributions.

$G_{X\to Y}$ generates a fake sample $\tilde{y} = G_{X\to Y}(x;\theta_g^{X\to Y})$ in $p_{data}(y)$ domain from a true sample $x$ in $p_{data}(x)$ domain, while $G_{Y\to X}$ generates a fake sample $\tilde{x} = G_{Y\to X}(y;\theta_g^{Y\to X})$ in $p_{data}(x)$ domain from a true sample $y$ in $p_{data}(y)$ domain. For true data $x \sim p_{data}(x)$ and synthesized data $\tilde{x} = G_{Y\to X}(y;\theta_g^{Y\to X})$, $D_X$ distinguishes whether a given data belongs to $p_{data}(x)$ domain. On the contrary, true data $y \sim p_{data}(y)$ and synthesized data $\tilde{y} = G_{X\to Y}(x;\theta_g^{X\to Y})$ are classified by $D_Y$ whether a given data belongs to $p_{data}(y)$ domain.

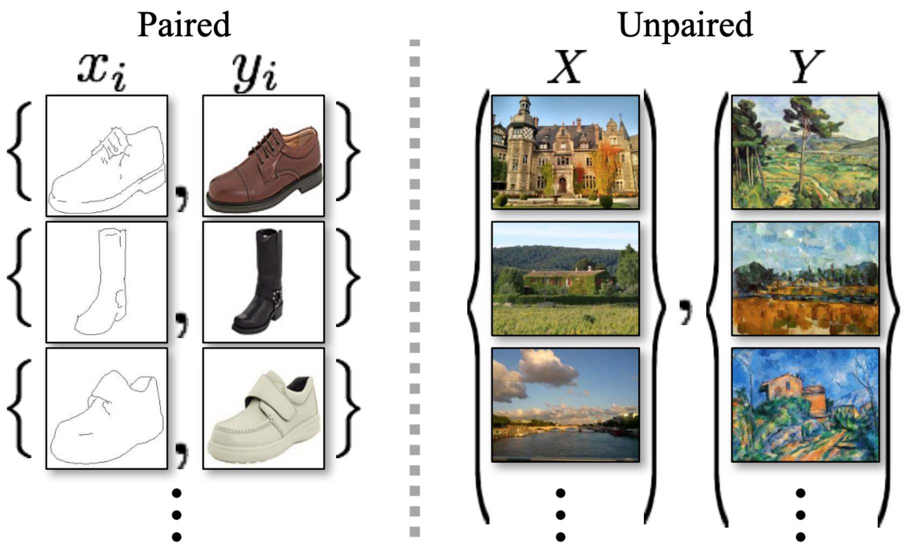


Figure 2: *Paired* training data (left) consists of training examples $\{x_i, y_i\}_{i=1}^N$, where the correspondence between $x_i$ and $y_i$ exists [22]. We instead consider *unpaired* training data (right), consisting of a source set $\{x_i\}_{i=1}^N$ ($x_i \in X$) and a target set $\{y_j\}_{j=1}$ ($y_j \in Y$), with no information provided as to which $x_i$ matches which $y_j$.

Figure 4: Example of unpaired data distributions $x \sim p_{data}(x)$ and $y \sim p_{data}(y)$.

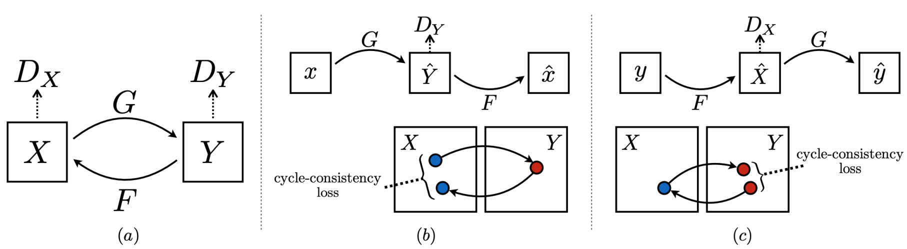


Figure 3: (a) Our model contains two mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$, and associated adversarial discriminators $D_Y$ and $D_X$. $D_Y$ encourages $G$ to translate $X$ into outputs indistinguishable from domain $Y$, and vice versa for $D_X$ and $F$. To further regularize the mappings, we introduce two *cycle consistency losses* that capture the intuition that if we translate from one domain to the other and back again we should arrive at where we started: (b) forward cycle-consistency loss: $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$, and (c) backward cycle-consistency loss: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$

Figure 5: Overview of cyclegan [5].

## 2.4 StarGAN [6]

$$\begin{aligned}\mathcal{L}_{\text{starGAN}}(G, D^{(cls,src)}) &= \mathcal{L}_{GAN}(G, D^{(src)}) \\ &+ \lambda_{cls}\mathcal{L}_{cls}(G, D^{(cls)}) + \lambda_{rec}\mathcal{L}_{rec}(G), \end{aligned} \tag{13}$$

where $\mathcal{L}_{GAN}(G, D^{(src)})$ is an objective function of a GAN, $\mathcal{L}_{cls}(G, D^{(cls)})$ is a multi-class classification loss and $\mathcal{L}_{rec}(G)$ is a reconstruction loss (= cycle consistency loss). $\lambda_{cls}$ and $\lambda_{rec}$ are hyper-parameters that control the relative importance. $\mathcal{L}_{GAN}(G, D^{(src)})$, $\mathcal{L}_{cls}(G, D^{(cls)})$, and $\mathcal{L}_{rec}(G)$ are defined by Eqs. 14, 15, and 16, respectively.

$$\begin{aligned}\mathcal{L}_{GAN}(G, D^{(src)}) &= \mathbb{E}_x[\log D(x; \theta_d^{(src)})] \\ &+ \mathbb{E}_{x,c}[\log(1 - D(G(x, c; \theta_g); \theta_d^{(src)}))], \end{aligned} \tag{14}$$

$$\mathcal{L}_{cls}(G, D^{(cls)}) = \mathcal{L}_{cls}^{real}(G, D^{(cls)}) + \mathcal{L}_{cls}^{fake}(G, D^{(cls)}),$$

$$\mathcal{L}_{cls}^{real}(G, D^{(cls)}) = \mathbb{E}_{x,c'}[-\log D^{(cls)}(c'|x; \theta_d^{(cls)})], \tag{15a}$$

$$\mathcal{L}_{cls}^{fake}(G, D^{(cls)}) = \mathbb{E}_{x,c}[-\log D^{(cls)}(c|G(x, c; \theta_g); \theta_d^{(cls)})], \tag{15b}$$

$$\mathcal{L}_{rec}(G) = \mathbb{E}_{x,c,c'}[|G(G(x, c; \theta_g), c'; \theta_g) - x|], \tag{16}$$

where $G$ and $D^{(cls,src)}$ are generator and discriminator, respectively, and its $\theta_g$, $\theta_d^{(cls)}$ and $\theta_d^{(src)}$ are trainable parameters such as convolution kernel ($\omega$) and bias($b$). Specifically, $\theta_d^{(cls)}$ and $\theta_d^{(src)}$ share all parameters except the end of layer. At the end of layer, $D^{(cls)}$ and $D^{(src)}$ are separated using different convolution layers.

$G$ generates a fake sample $\tilde{x} = G(x, c; \theta_g)$ in $p_{data}(c)$ domain from a input $x$ in $p_{data}(c')$ domain. For true data $x \sim p_{data}(c')$ and synthesized data $\tilde{x} = G(x, c; \theta_g)$, $D^{(src)}$ distinguishes whether a given data belongs to real domain.

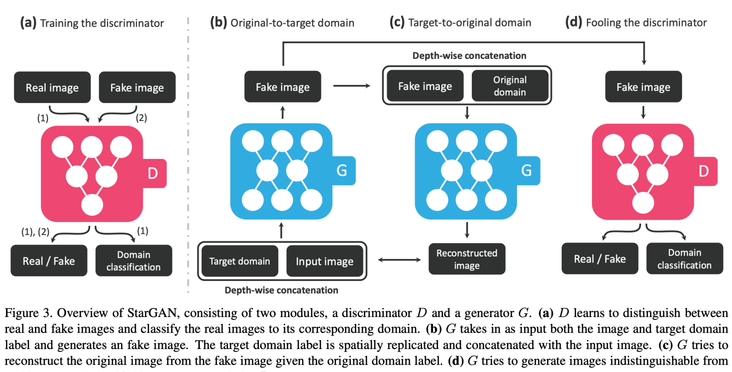


Figure 6: Overview of StarGAN [6].

## References


[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[3] Alec Radford, Luke Metz, and Soumith Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[4] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[5] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[6] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8789–8797.