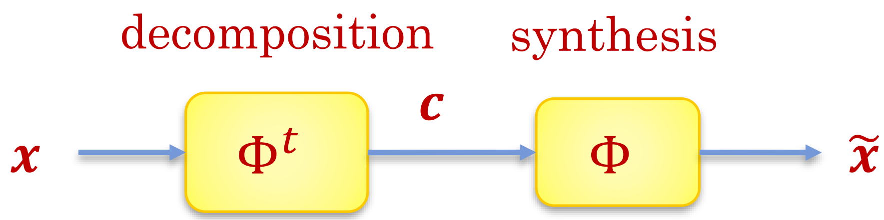
Deep Learning

– GAN

林伯慎 Prof. Bor-Shen Lin
bslin@cs.ntust.edu.tw

Vector Space Decomposition and Synthesis



- Assume $\Phi = \{\boldsymbol{\phi}_i\}_{i=1}^{n}$ is an orthonormal set, $\boldsymbol{x}$ is a vector.
- Decomposition : $c_i = \langle \boldsymbol{x}, \boldsymbol{\phi}_i \rangle$ for $i = 1,2,\ldots,n$.
  - c_i the amount of projection of x in the direction of $\boldsymbol{\phi}_i$.
  - $\boldsymbol{c} = \Phi^t \boldsymbol{x}$ is the decomposition of vector x.
- Synthesis : $\tilde{\boldsymbol{x}} = \sum_{i=1}^{n} c_i \boldsymbol{\phi}_i = \Phi\Phi^t \boldsymbol{x}$.
  - $\tilde{\boldsymbol{x}}$ is the reconstruction of $\boldsymbol{x}$ with reconstruction loss $L_2(\boldsymbol{x}, \tilde{\boldsymbol{x}})$.
  - If Φ is a basis, $L_2(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = 0$.

Analysis

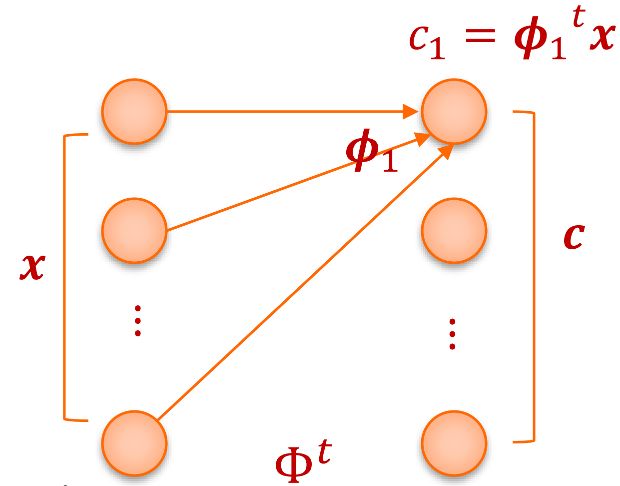
- If $\Phi = \{\boldsymbol{\phi}_i\}_{i=1}^{n}$ is a orthonormal vectors in a vector space, and $\boldsymbol{x}$ is a vector in the vector space.
- $c_i = \langle \boldsymbol{x}, \boldsymbol{\phi}_i \rangle$ for $i = 1,2,\dots,n$.
  - c_i is the projection of vector x on the direction of $\boldsymbol{\phi}_i$.
  - Decomposition of the vector **x** in the subspace

$$\boldsymbol{c} = \Phi^t \boldsymbol{x} = [\boldsymbol{\phi}_1 \boldsymbol{\phi}_2 \dots \boldsymbol{\phi}_n]^t \boldsymbol{x}$$

$$\begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} {\boldsymbol{\phi}_1}^t \\ \vdots \\ {\boldsymbol{\phi}_n}^t \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} {\boldsymbol{\phi}_1}^t \boldsymbol{x} \\ \vdots \\ {\boldsymbol{\phi}_n}^t \boldsymbol{x} \end{bmatrix}$$

$$c_i = {\boldsymbol{\phi}_i}^t \boldsymbol{x} = \langle \boldsymbol{x}, \boldsymbol{\phi}_i \rangle$$

  - Φ as an analysis network
  - $\boldsymbol{\phi}_i$ connection weights of neuron i

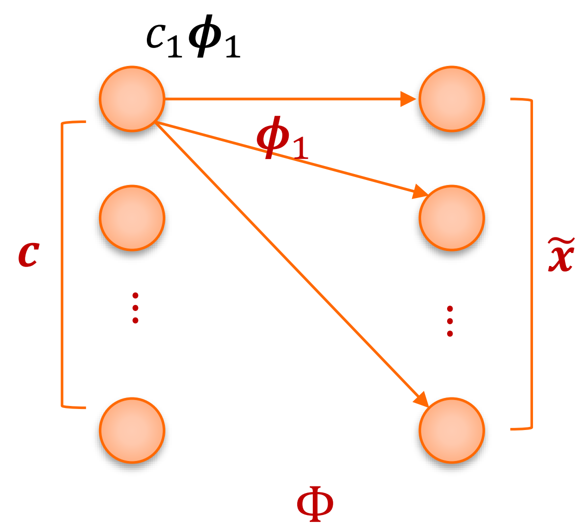


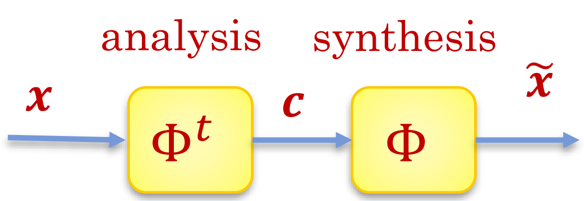
SYNTHESIS

- $\tilde{\boldsymbol{x}} = \sum_{i=1}^{n} c_i \boldsymbol{\phi}_i = [\boldsymbol{\phi}_1 \boldsymbol{\phi}_2 \ldots \boldsymbol{\phi}_n] \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix}$

  $= \Phi \boldsymbol{c} = \Phi \Phi^t \boldsymbol{x}.$

  - Reconstruction of the vector $\boldsymbol{x}$ in linear subspace spanned by Φ.
  - Reconstruction error: $L_2(\boldsymbol{x}, \tilde{\boldsymbol{x}})$.
  - When Φ is a basis of the vector space, $L_2(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = 0$.
  - $\boldsymbol{c}$ is a representation of $\boldsymbol{x}$.



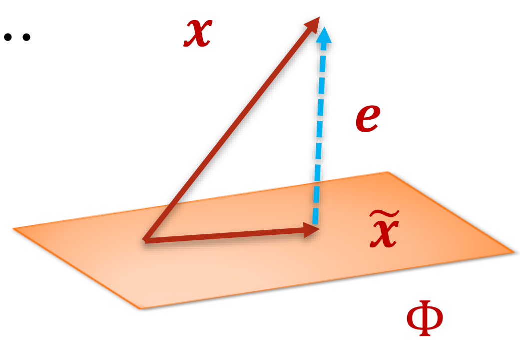


EXAMPLE: DFT / IDFT

- Discrete Fourier transform
  - Decomposition of discrete-time signal $x[n]$ of length N on a subspace with basis $\Phi = \{e^{j\omega n}\}$.
  - FT: $X(\omega) = \langle x[n], e^{j\omega n} \rangle$ continuous spectrum
  - DFT: $X[k] = \left\langle x[n], e^{j\frac{2\pi kn}{N}} \right\rangle$ discrete spectrum
  - Ingredients of $x[n]$ at different frequency (ω)
- Inverse Discrete Fourier transform
  - Reconstruction of signal using features and basis Φ.
  - IFT: $\tilde{x}[n] = \frac{1}{2\pi} \int X(\omega) e^{j\omega n} d\omega$
  - IDFT: $\tilde{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\frac{2\pi kn}{N}}$

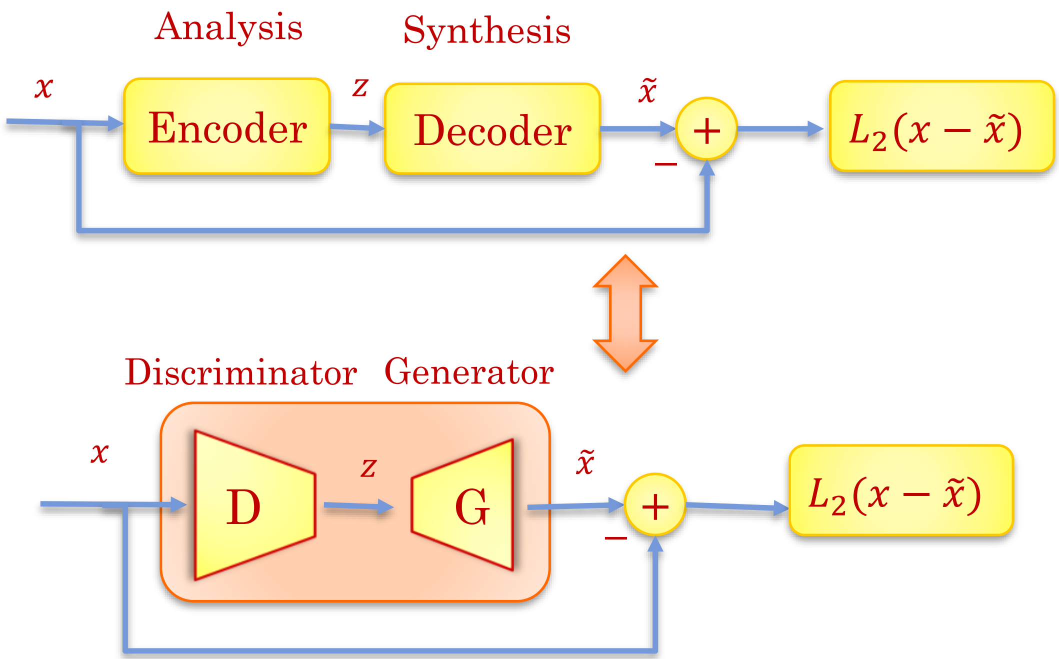
Decomposition

- Car
  - A car → 1 handler,4 wheels,…
- Hamberger
  - A hamberger → water, starch, mineral, …
- 3D vector projected onto 2D plane
  - Error vector perpendicular to the plane
  - Projection is the reconstruction



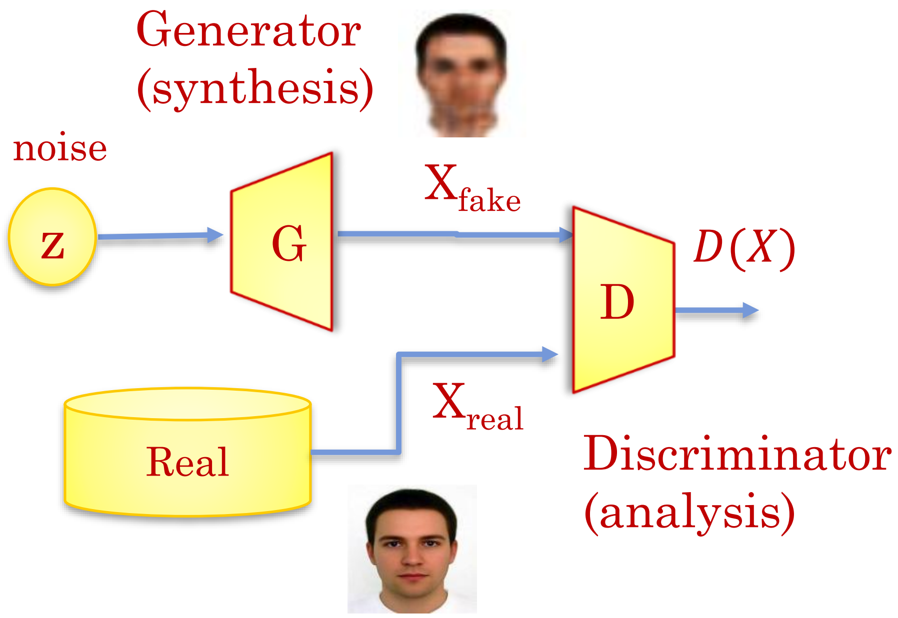
- Fourier analysis
  - Decomposing the signal with a set of cosine functions
  - 「Fourier transform」 decomposition of signal
  - 「Inverse Fourier transform」 reconstruction of signal

AUTO-ENCODER



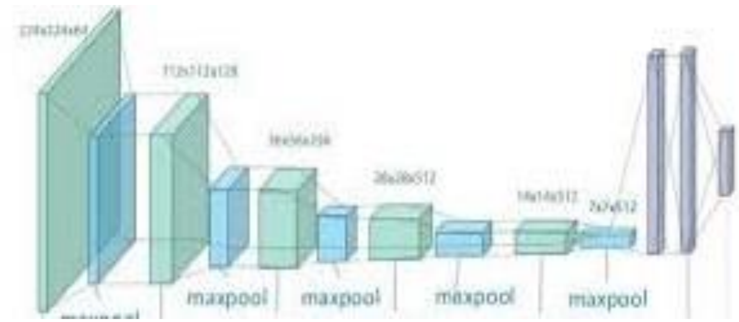
- Self estimate of a vector to minimize $L_2(x - \tilde{x})$
- D/G could be FNN, CNN/DCNN, RNN or others
- Representation learning (unsupervised)
  - z is the feature of x

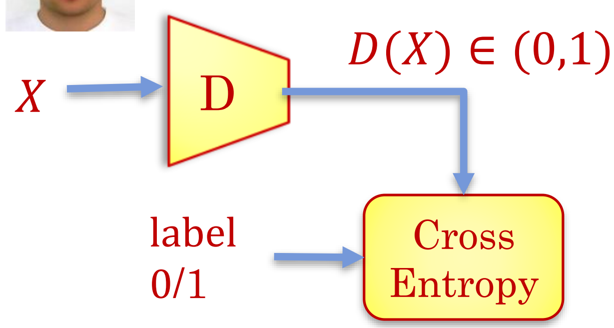
GAN
(Generative Adversarial Network)



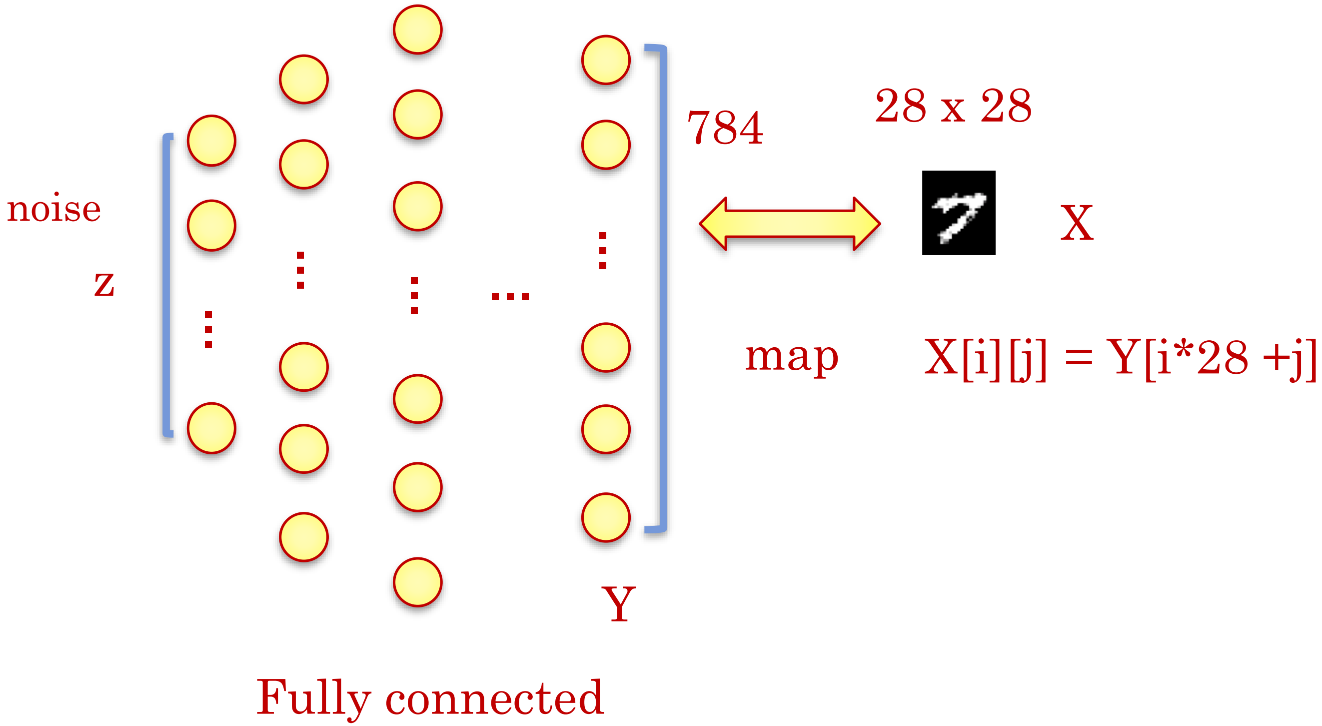
Discriminator

- Binary Classifier
  - Tell if an object is of a specific type or not
  - Positive/negative samples
  - e.g. CNN
- Example: Face detection
  - Positives: any face photos
  - Negatives: any non-face photos

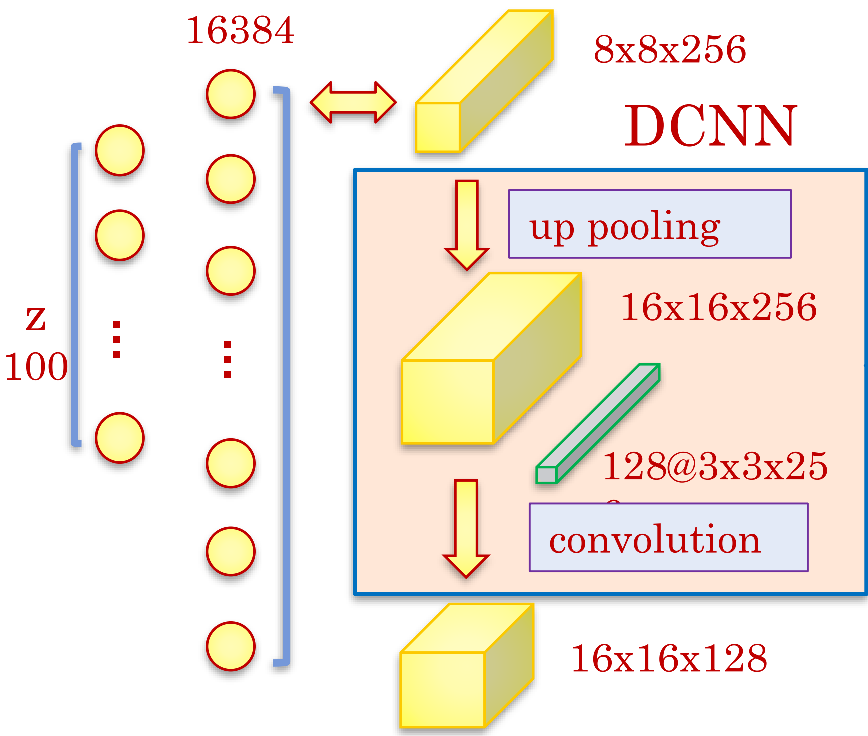


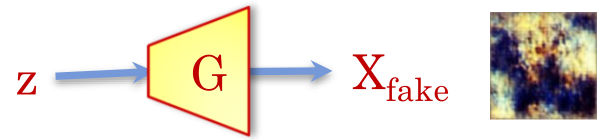


FNN Generator



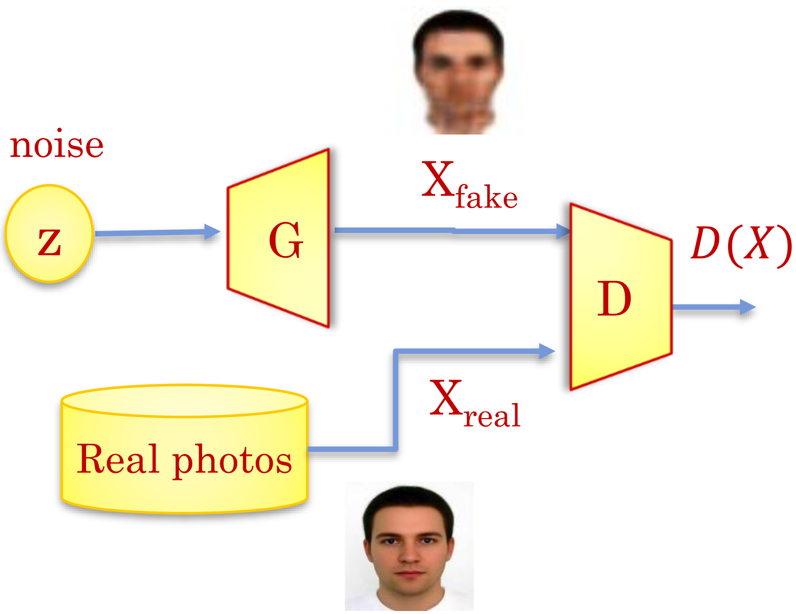
DCNN Generator





Layer Operation	**Input**	**Output**
Fully Connected 16,384 x 100	100	16,384
Up pooling+ **Conv** 128@3x3x256	8x8x256	16x16x128
Up pooling+ **Conv** 64@3x3x128	16x16x128	32x32x64
Up pooling+ **Conv** 3@3x3x64	32x32x64	64x64x3

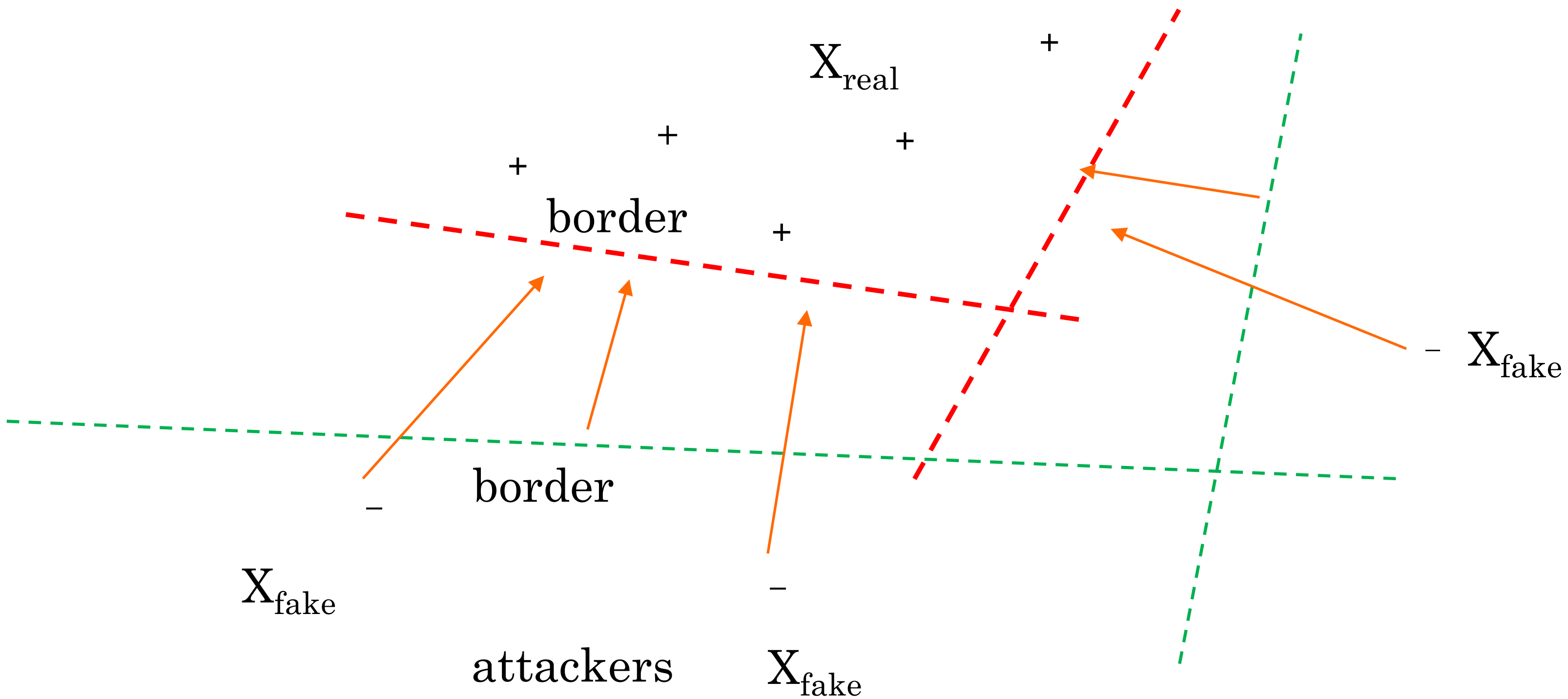
Training of GAN



GAN Learning

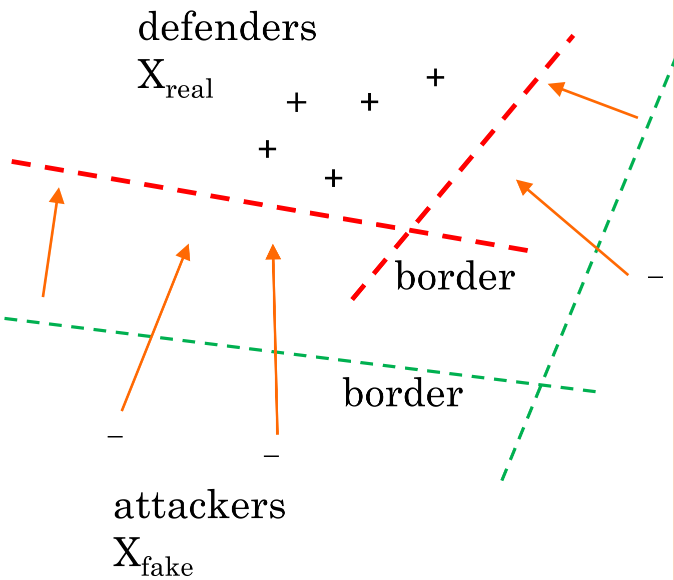
1. X_{real} : its goal is to be accepted by D when learning D (gold as 1) $\max_D(\log(D(X_{real})))$.
2. X_{fake} : : its goal is to be rejected by D when learning **D** (gold as 0) $\max_D\left(\log(1 - D(X_{fake}))\right)$.
3. X_{fake} : its goal is to be pretend to be real and accepted by D, so Dset gold as 1 to generate gradient for G to learn **G** (D is NOT updated) $\max_G D(G(z))$.

How Does GAN Work?



Discussions

- Discriminator is a binary classifier with positive samples ONLY. Negative samples are produced by Generator.
- If Generator is not good enough,
  - Generated X_{fake} are too far away from X_{real}, which makes the decision boundary lousy.
  - You cannot train a troop with weak imaginary enemies..
- When Generator becomes tough,
  - Generated samples come closer to the positive samples, and the decision boundary shrink backward towards the positive samples.
  - Train Olympic athletics in real games.



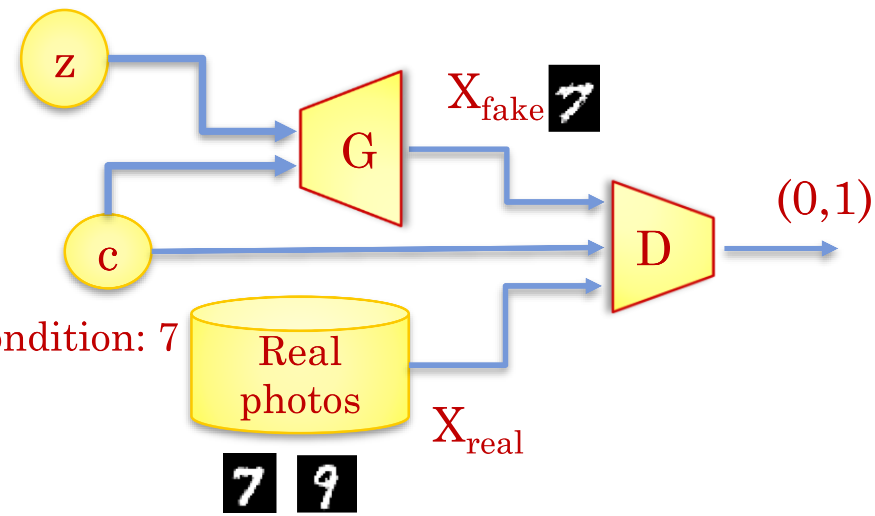
GOALS OF GAN

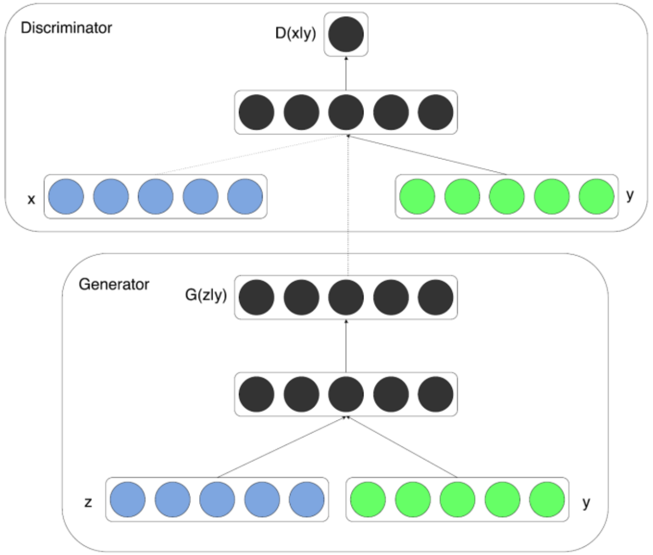
- May be train discriminator(D) or generator(G).
- When the goal is to train the discriminator
  - It means it is possible to train discriminator with GAN when only positive samples are available.
  - Make use of generator to produce more negative samples so as to better train discriminator
- When the goal is to train the generator
  - It is possible to generate something similar to the positive samples (reals) but with variation(through using noise z)
  - It is not expected to generate exact the same things
  - mode collapse

    → when changing z, no difference (loss allows M-to-1)

    → cannot control the characteristics of the generated output

CONDITIONAL GAN (C-GAN)



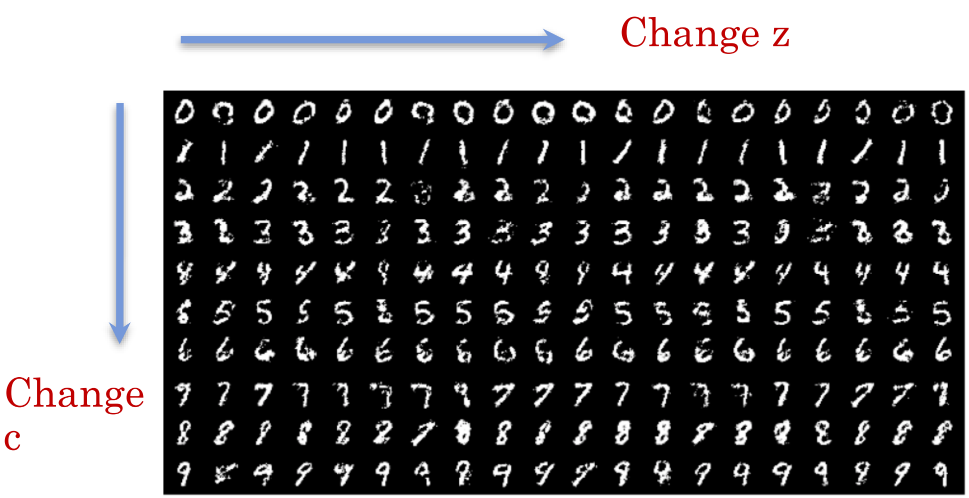


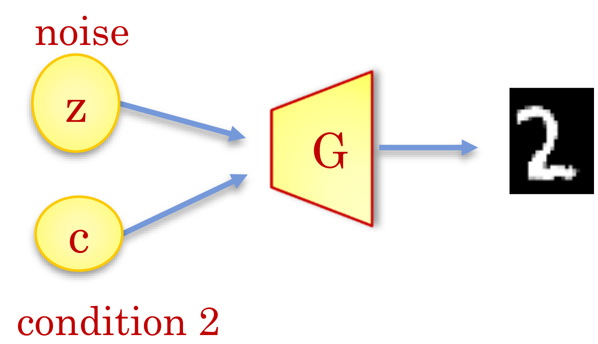
Implement (FC)

Cited from C-GAN by M Mirza

- Training inputs: image+condition
- Use c to control condition and z to produce variation
- **Conditions: label, image, or text**

C-GAN Example- MNIST



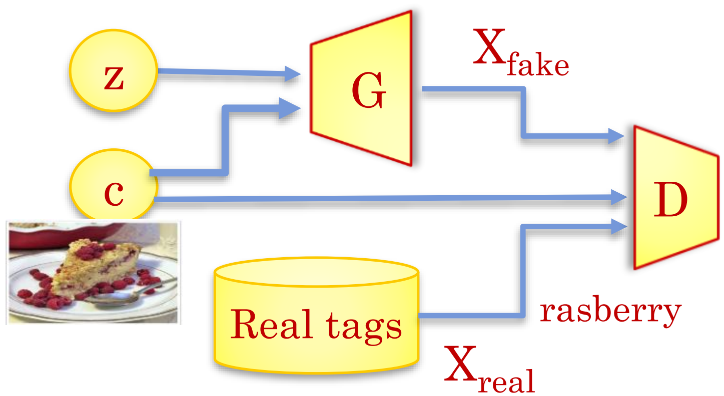


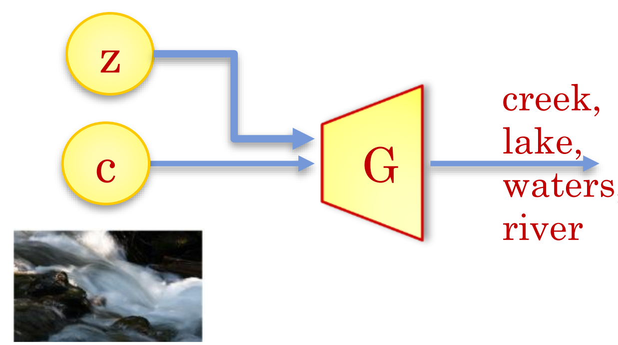
- Label as condition

Cited from C-GAN by M Mirza

C-GAN Example – Auto Tagging

	User tags + annotations	Generated tags
	montanha, trem, inverno, frio, people, male, plant life, tree, structures, transport, car	taxi, passenger, line, transportation, railway station, passengers, railways, signals, rail, rails
	food, raspberry, delicious, homemade	chicken, fattening, cooked, peanut, cream, cookie, house made, bread, biscuit, bakes
	water, river	creek, lake, along, near, river, rocky, treeline, valley, woods, waters
	people, portrait, female, baby, indoor	love, people, posing, girl, young, strangers, pretty, women, happy, life

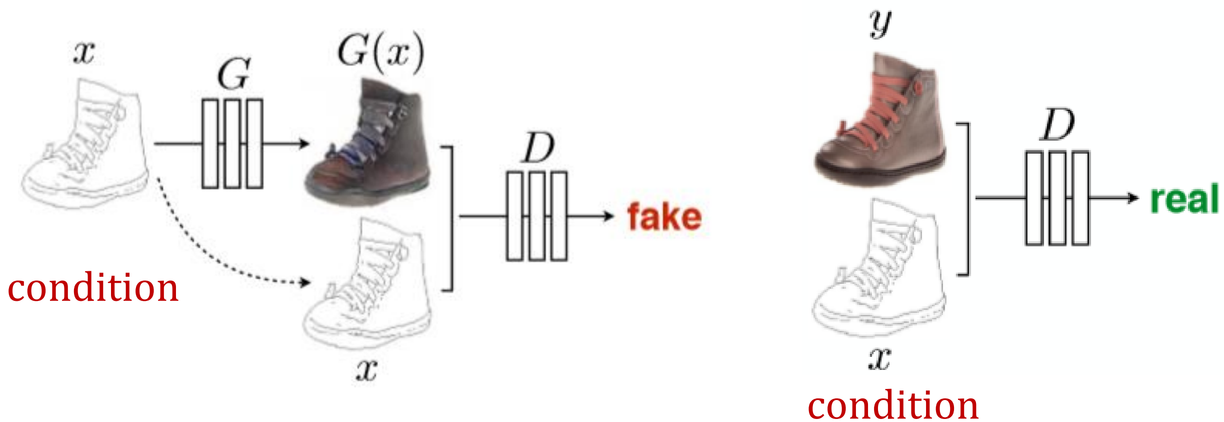




Cited from C-GAN by M Mirza

TAIWAN TECH National Taiwan University of Science and Technology

C-GAN for Image-to-Image Translation

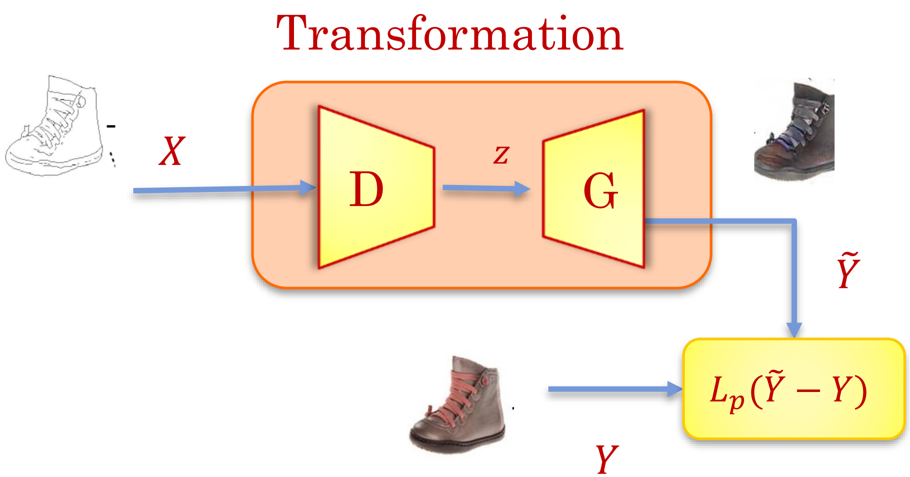


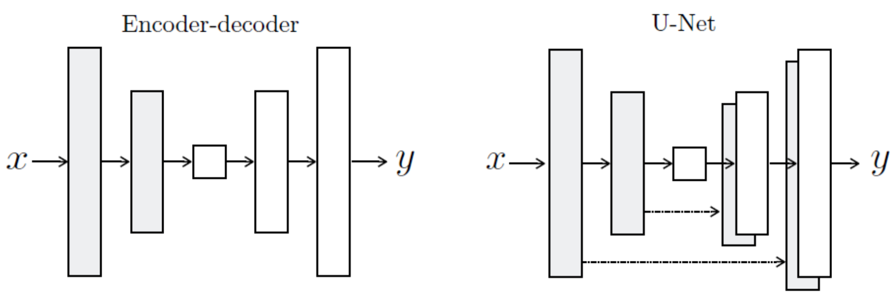
- Cited from *Image-to-Image Translation with Conditional Adversarial Networks*
- D使用PatchGAN: 判斷任意NxN的patch為real/fake
  - 減小Xreal空間,且有更多正樣本

Domain Transformation

- Auto-Encoder
- Variational Auto-Encoder (VAE)
- GAN/cGAN Transformer
- Cycle Consistent GAN
- Star GAN

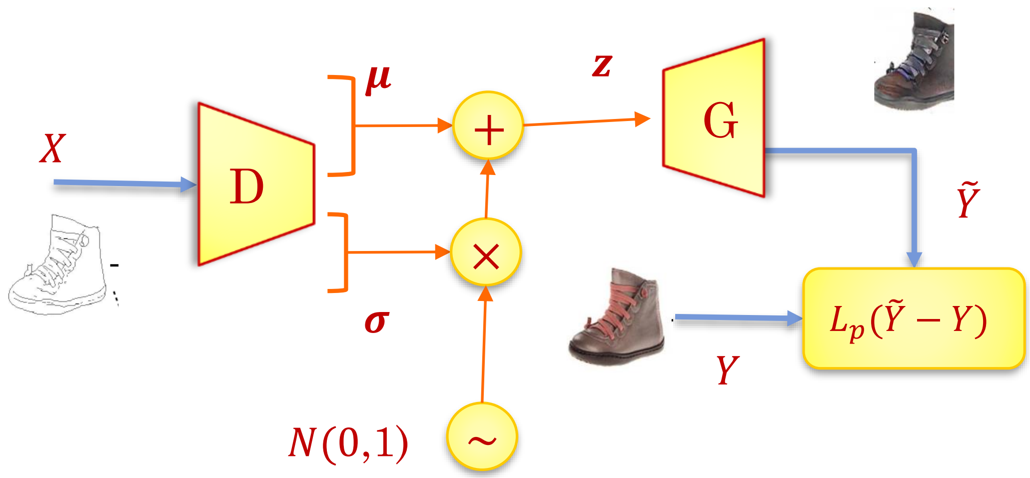
AUTO-ENCODER, AE (TRANSFORMATION)





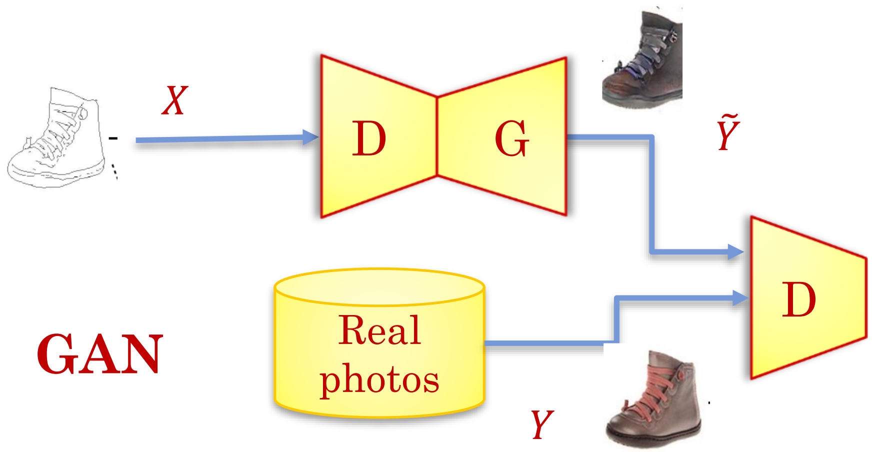
- Encoder-decoder
  - Unet/ResNet
- Learn transformation
  - Need **paired data** $\{(X_i, Y_i)\}$
  - $\min L_1(Y - \tilde{Y})$
- Example
  - Gray-to-color

VARIATIONAL AUTO-ENCODER

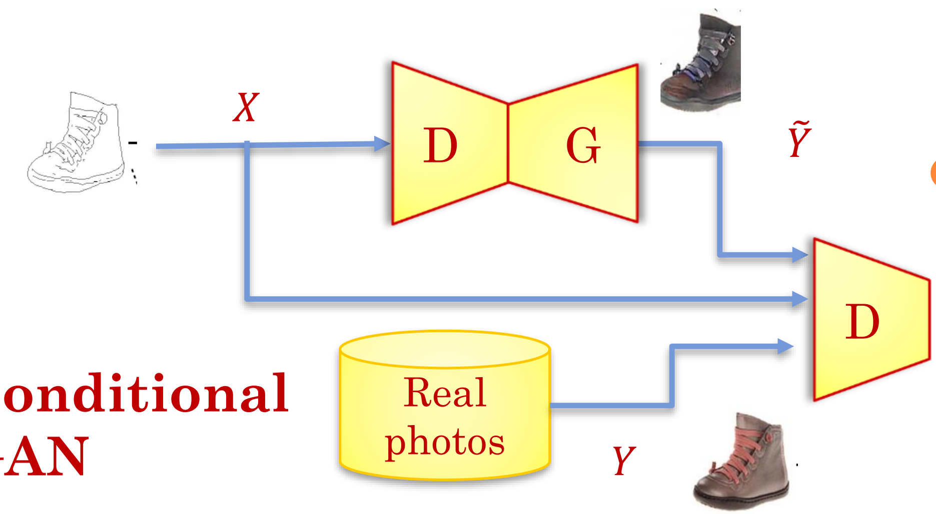


- Encoder output：mean $\boldsymbol{\mu}$ and stddev $\boldsymbol{\sigma}$
  - $z_i = \mu_i + n_i\sigma_i,\ n_i \sim N(0,1)$
  - record n_i, update μ_i and σ_i
- Add uncertainty to G： due to n_i

GAN / cGAN



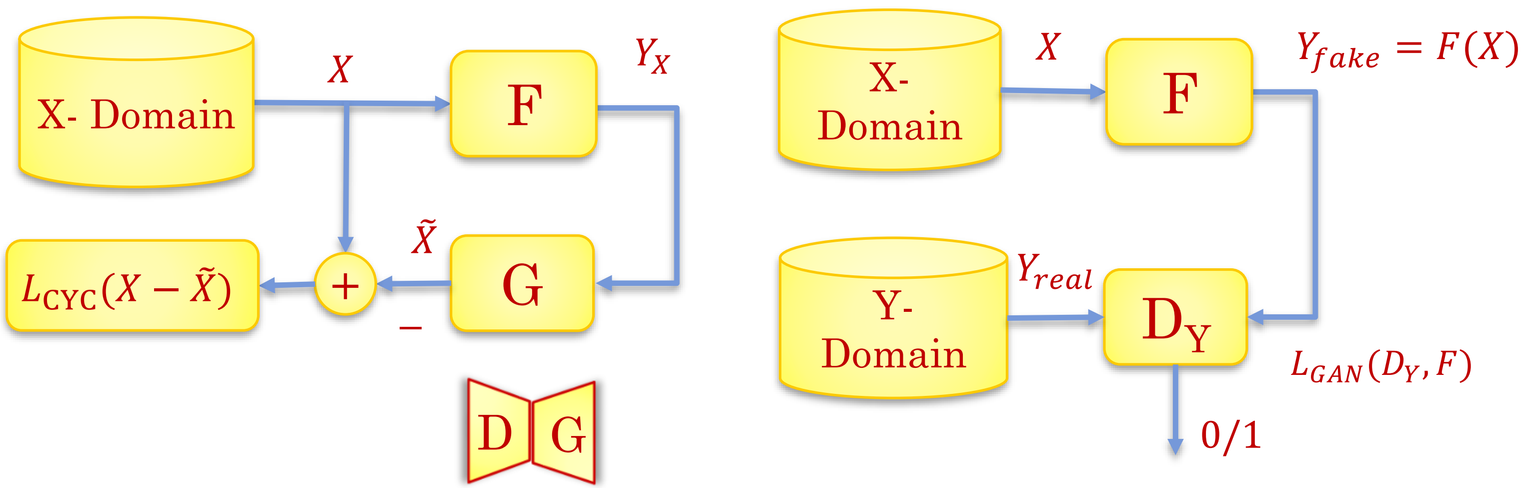
GAN



Conditional GAN

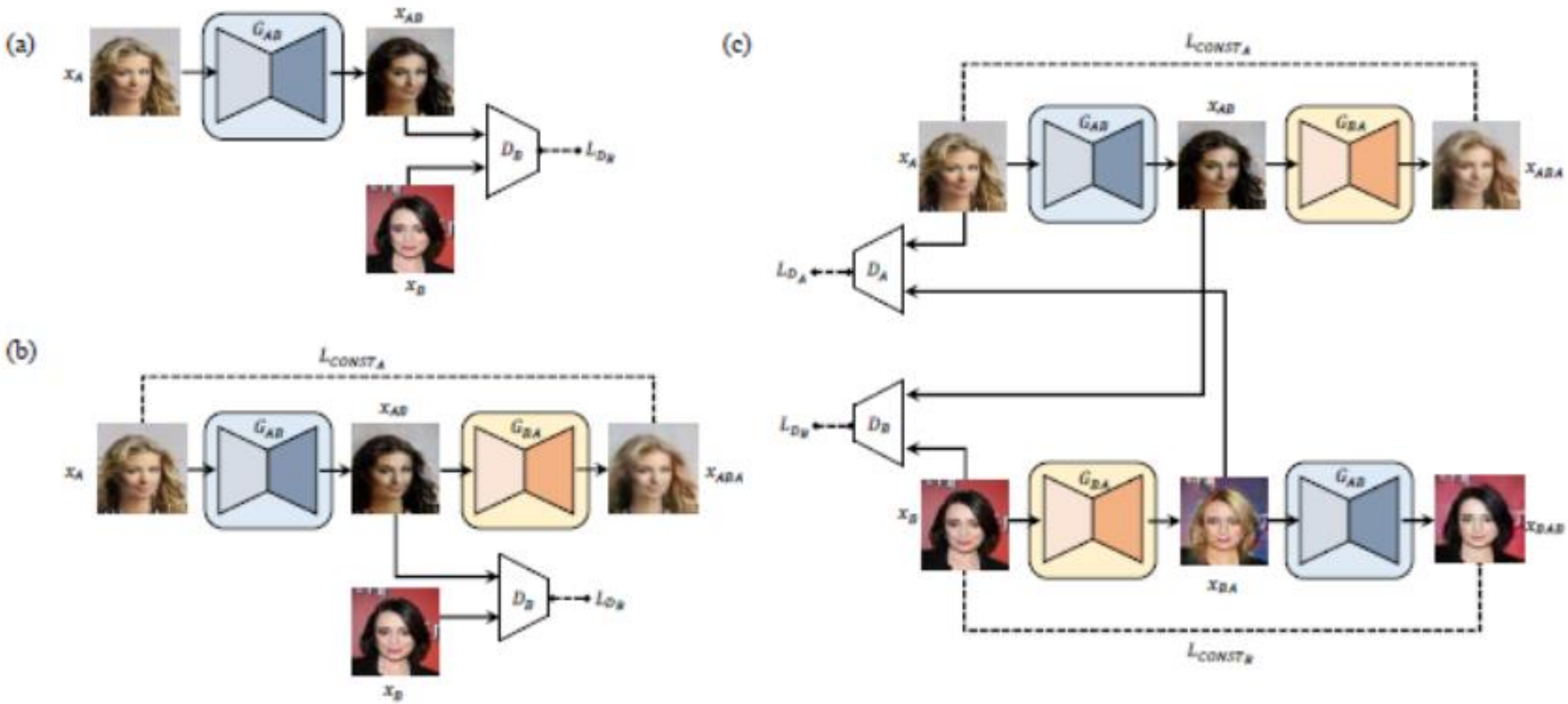
- GAN
  - Do not need paired data,
  - $X = \{X_i\}, Y = \{Y_j\}$
  - Not easy to converge well
  - 可加入L_1 loss if paired data available
- cGAN (conditional)
  - Need paired data
  - T=$\{(X_i, Y_i)\}$
  - Could add L_1 loss

Cycle GAN



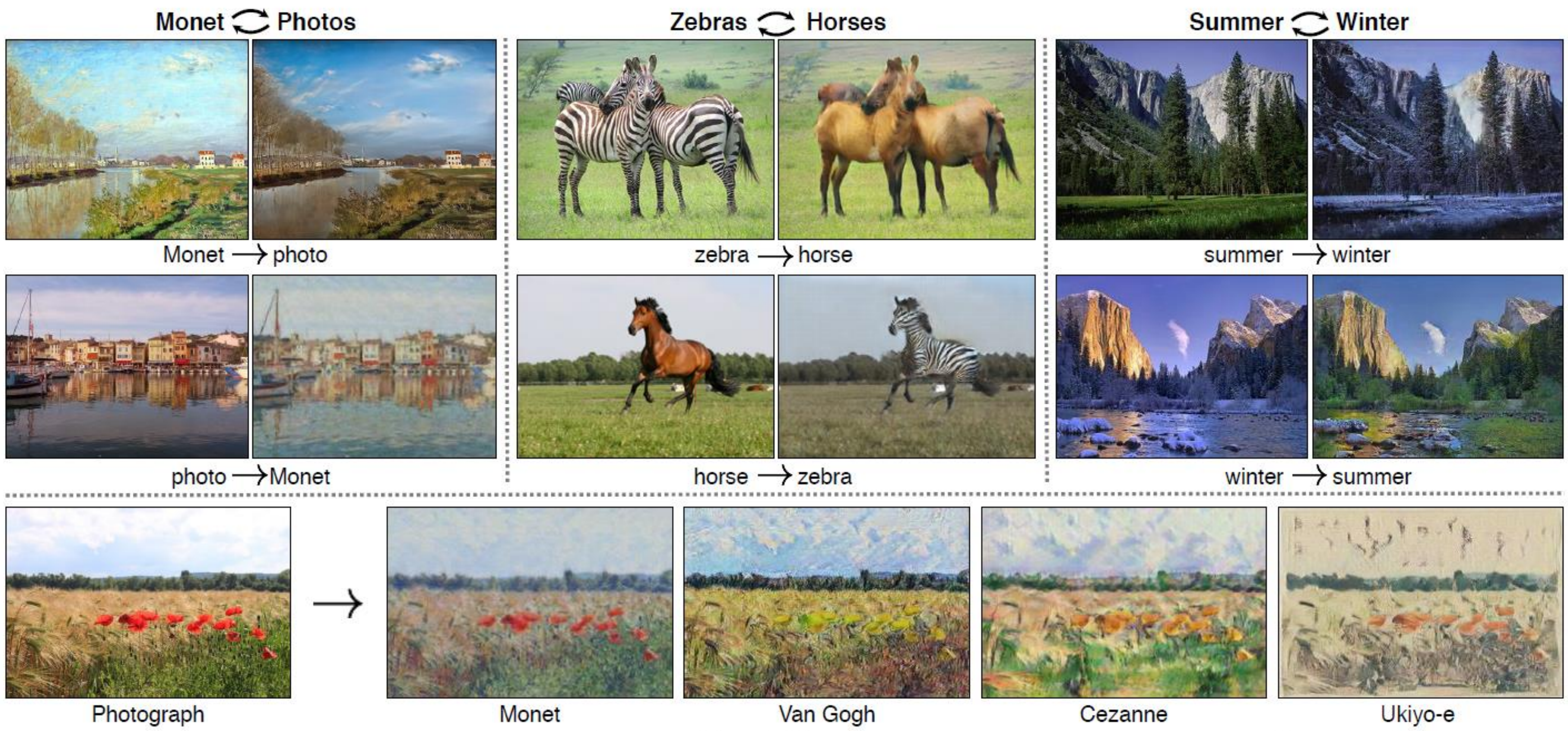
- X-domain和Y-domain: are not required to **be paired**
- F for X → Y, G for Y → X
  - 2 cycle losses: $L_{CYC}(X,\tilde{X})$ and $L_{CYC}(Y,\tilde{Y})$
- Transformed as *fake* data, Original as *real* data
  - 2 GAN losses: $L_{GAN}(D_X,G)$ and $L_{GAN}(D_Y,F)$
- Opt. for multiple networks (F, G, D_X, D_Y) with multiple objectives.

Cycle GAN - example



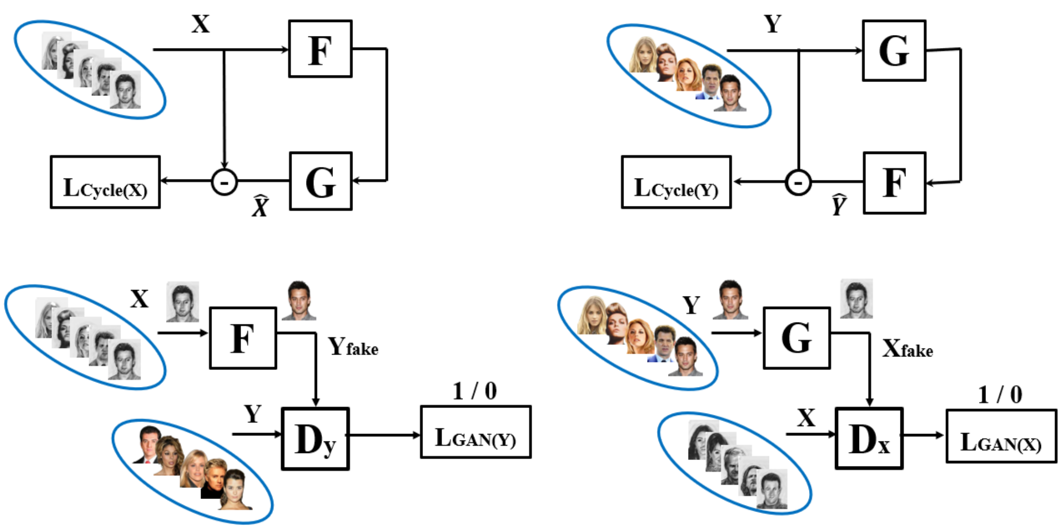
- Cited from *Learning to Discover Cross-Domain Relations with Generative Adversarial Networks*

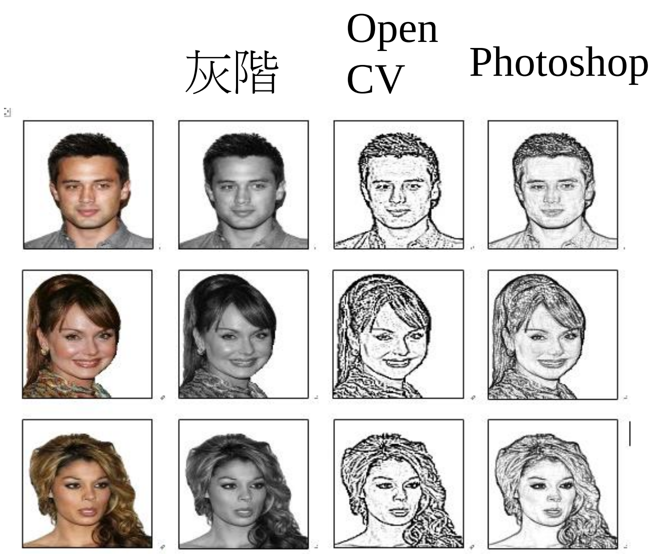
Cycle GAN - example



- Cited from *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*

EXAMPLES:



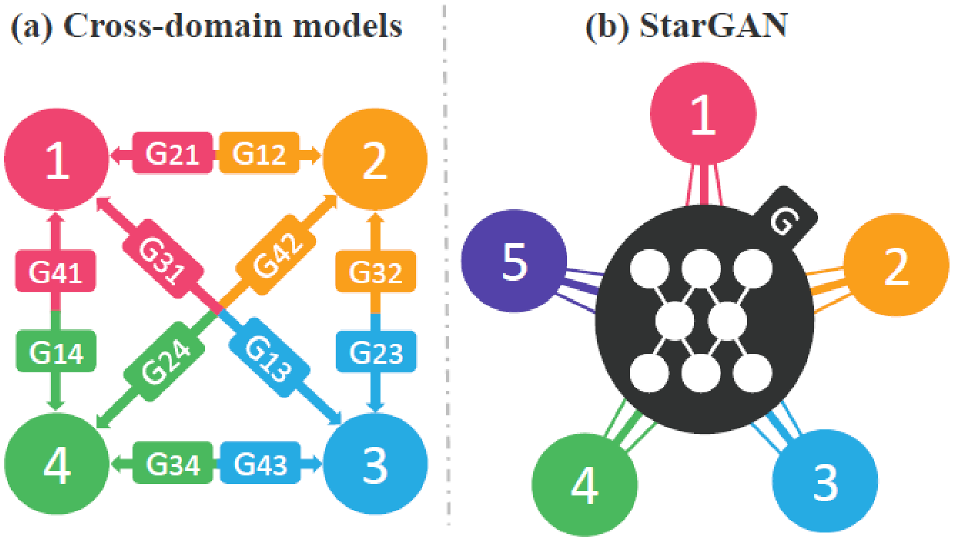


- Cited from Daiva's master thesis

Discussions on Cycle-GAN

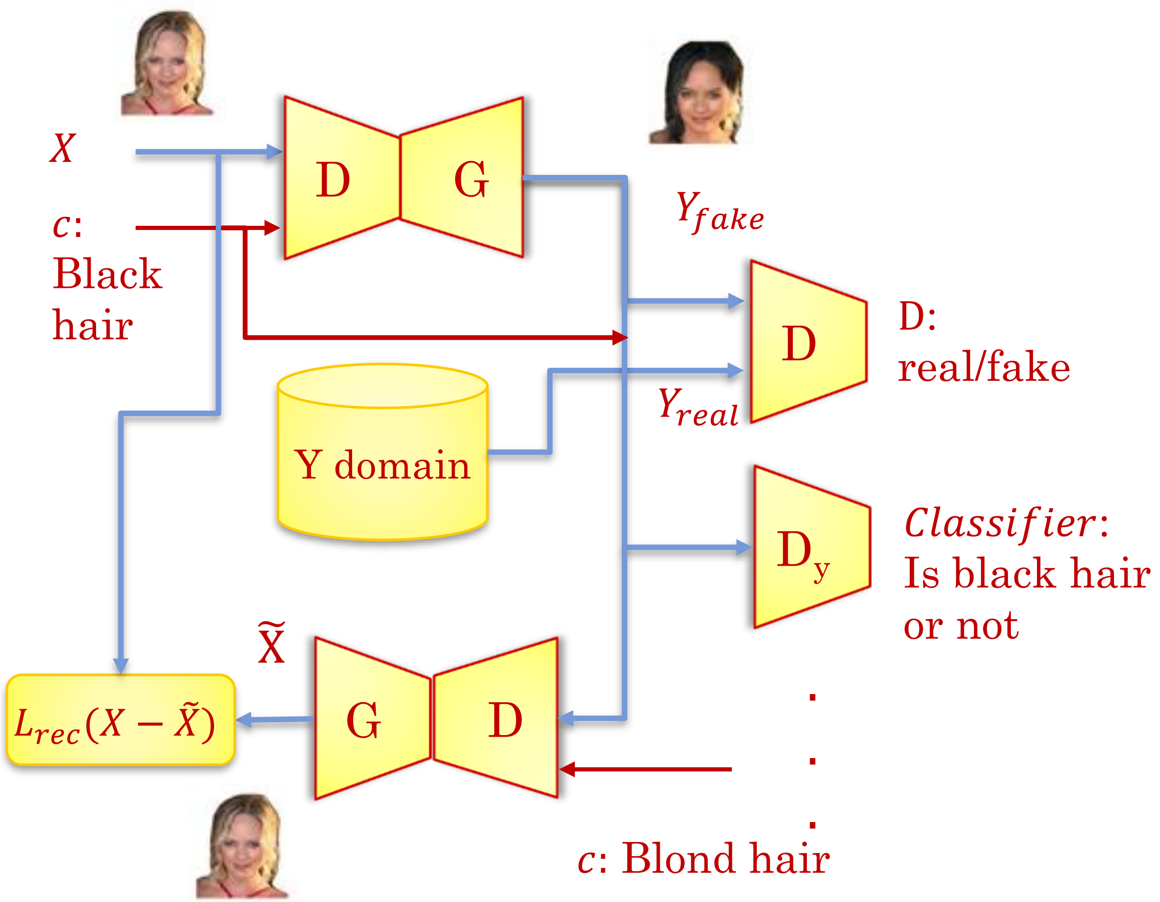
- To train the transformer instead of the generator
  - Domain transformation
  - black hair to blond hair, horse to zebra
- Without requiring pair data
  - Compare with transformer (requiring pair data)
- Complicated and time consuming
  - Joint optimization of multiple networks with multiple objectives.
  - Reconstruction loss may help to improve the quality (peek the ground truth)
  - U-net or residual net used to accelerate the convergence
  - Inconvenient for transforming among multiple attributes

StarGAN



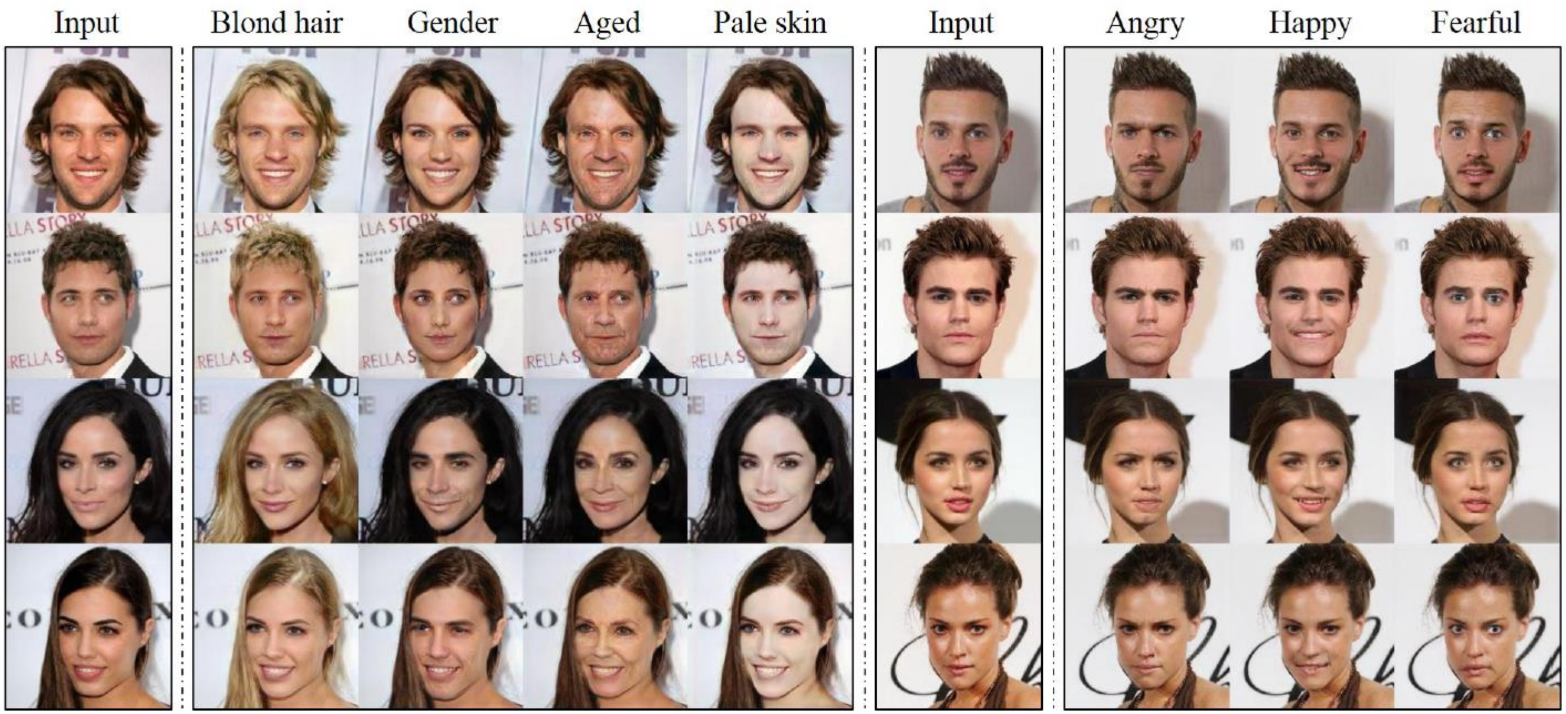
- If using CycleGAN
  - Multiple transformer
  - A lot of computations
  - Not flexible

StarGAN



X
c: Black hair
D
G
Y_{fake}
D
D: real/fake
Y domain
Y_{real}
D_y
$Classifier$: Is black hair or not
$\tilde{X}$
$L_{rec}(X - \tilde{X})$
G
D
c: Blond hair

StarGAN Example



- Cited from *StarGAN: Unified Generative Adversarial Networks for Multi-domain Image-to-Image Translation*