Machine Learning

Lecture 13
Generative Adversarial Network (GAN)

Chen-Kuo Chiang (江 振 國)
ckchiang@cs.ccu.edu.tw
中正大學 資訊工程學系

Generative Adversarial Network

Generative Models

Training samples

生成新樣貌

Model samples

Training samples

自然影像生成

Generative Adversarial Network

- First introduced by Ian Goodfellowet al. in 2014
- GANs have been used to generate images, videos, poems, and conversation 生成影像視訊

Generator:一個生成

隨機產生沒有看過結果

- Generates candidates/images (from a probability distribution)
- It's objective is to 'fool' the discriminator by producing novel synthesized instances that appear to come from the true data

綜合

欺騙

Discriminator: 一個對抗 檢察與辨別生成模型所產生資料

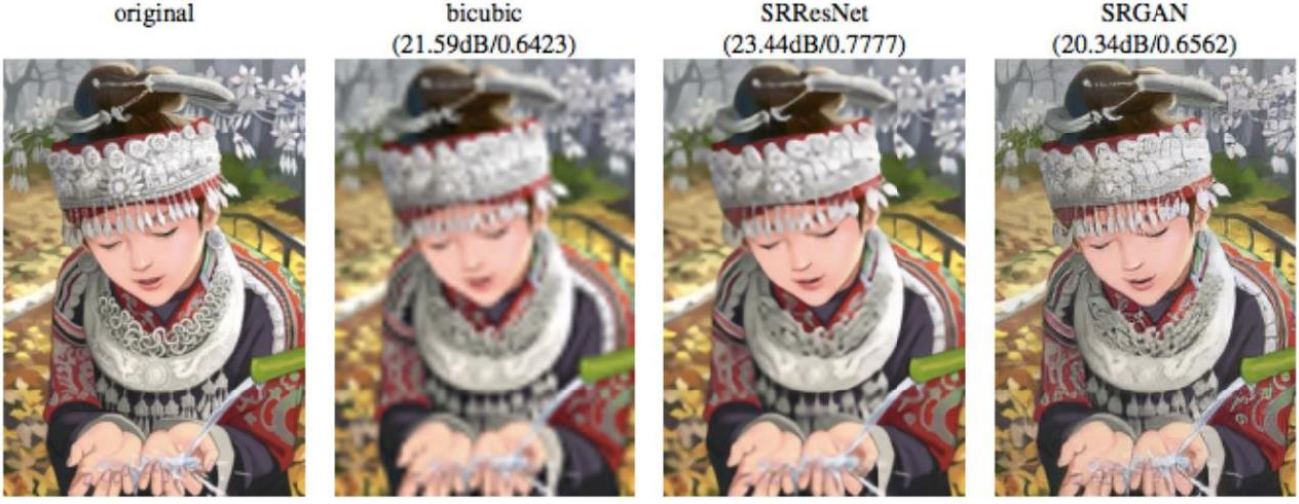
- Evaluates the generated images to see if they come from the true data or not

Backpropagation applied to both networks: 本來訓練一個

- Generator to produce better images
- Discriminator to be more skilled at evaluating generated images

Applications of GAN

- **Image generation tasks** 低解析轉換為高解析
  - Example: single-image super-resolution



Text-to-Image Translation 文字描述生成影像

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

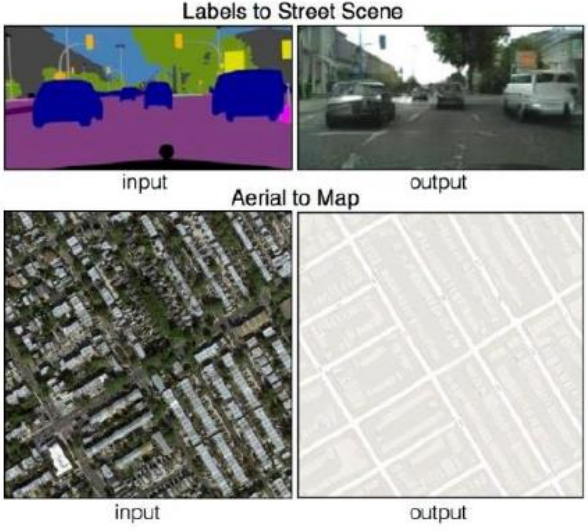
this magnificent fellow is almost all black with a red crest, and white cheek patch.

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

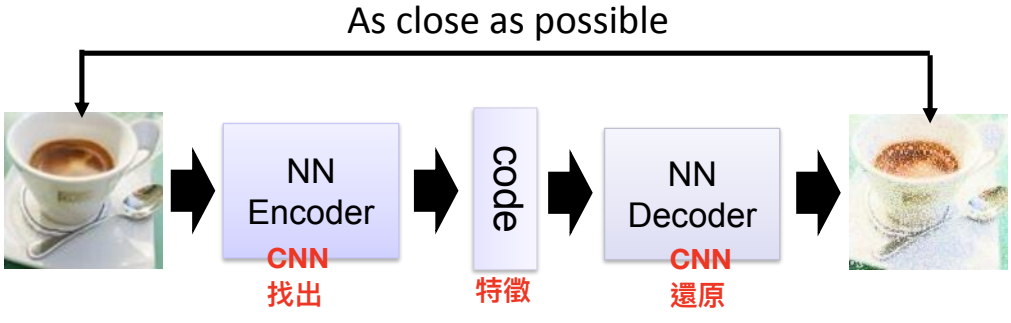
this white and yellow flower have thin white petals and a round yellow stamen

Image-to-Image Translation

把影像分塊 在做物件偵測



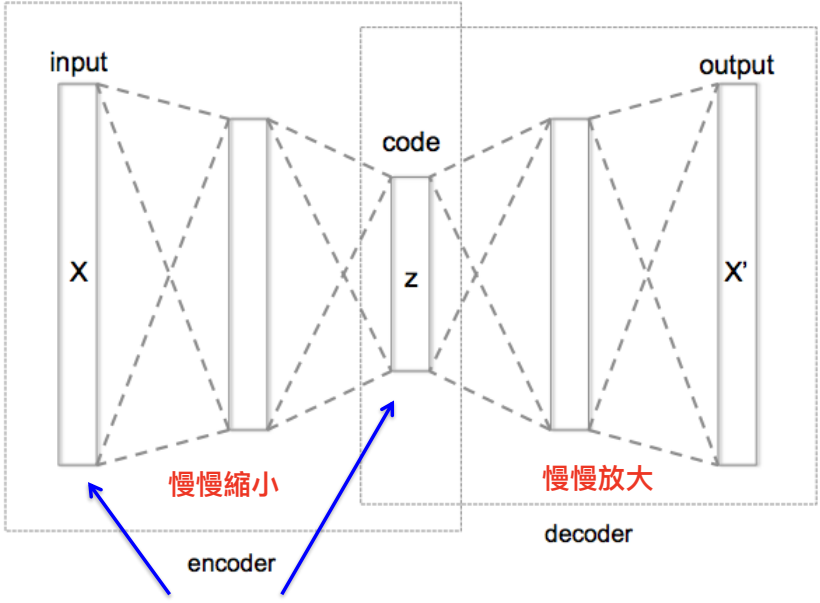
Motivation - Autoencoder



Randomly generate a vector as code

隨機產生的影像產生的結果為何？

Autoencoder with 3 fully connected layers

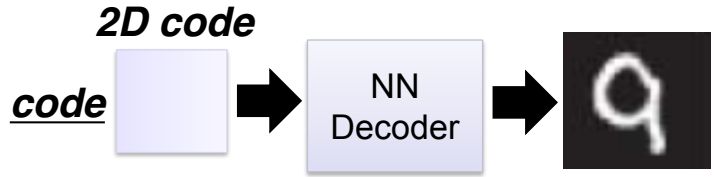


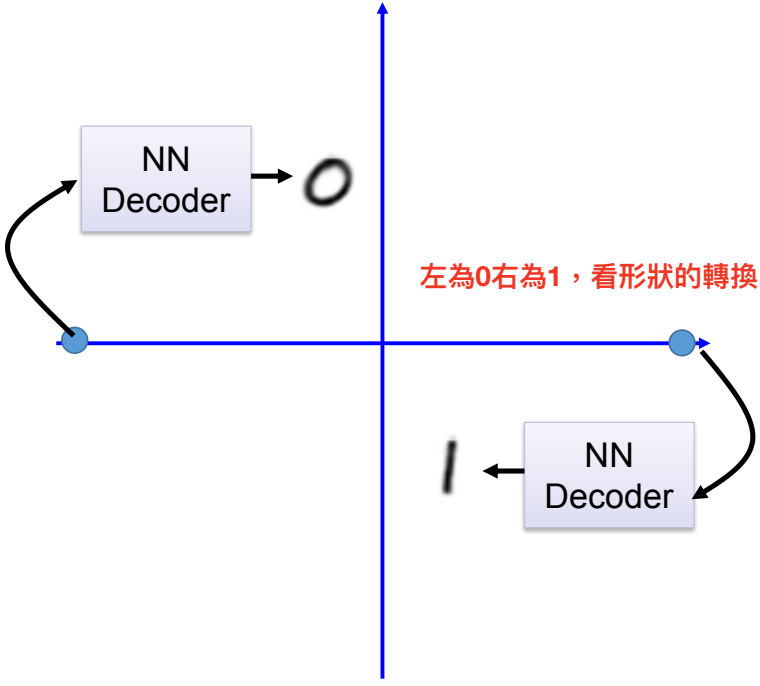
Training: model.fit(X,X')

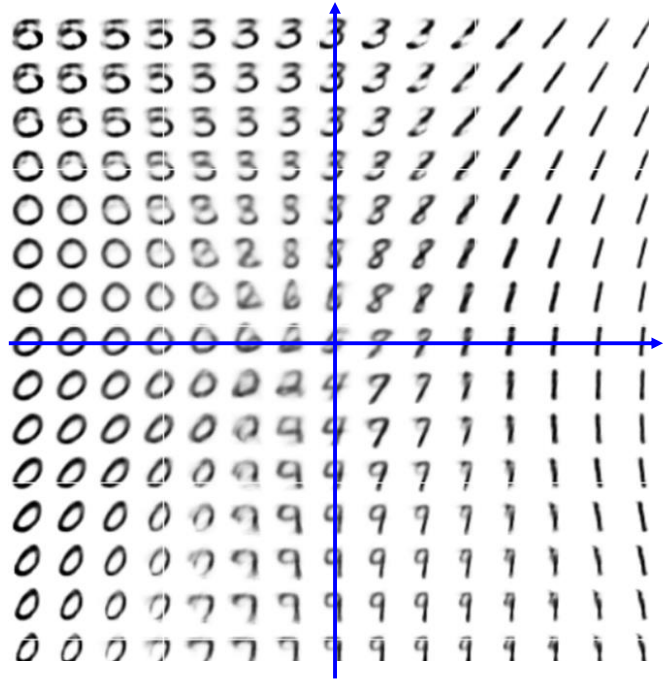
Cost function: $\Sigma_{k=1..N} (x_k - x'_k)^2$

距離要越小越好

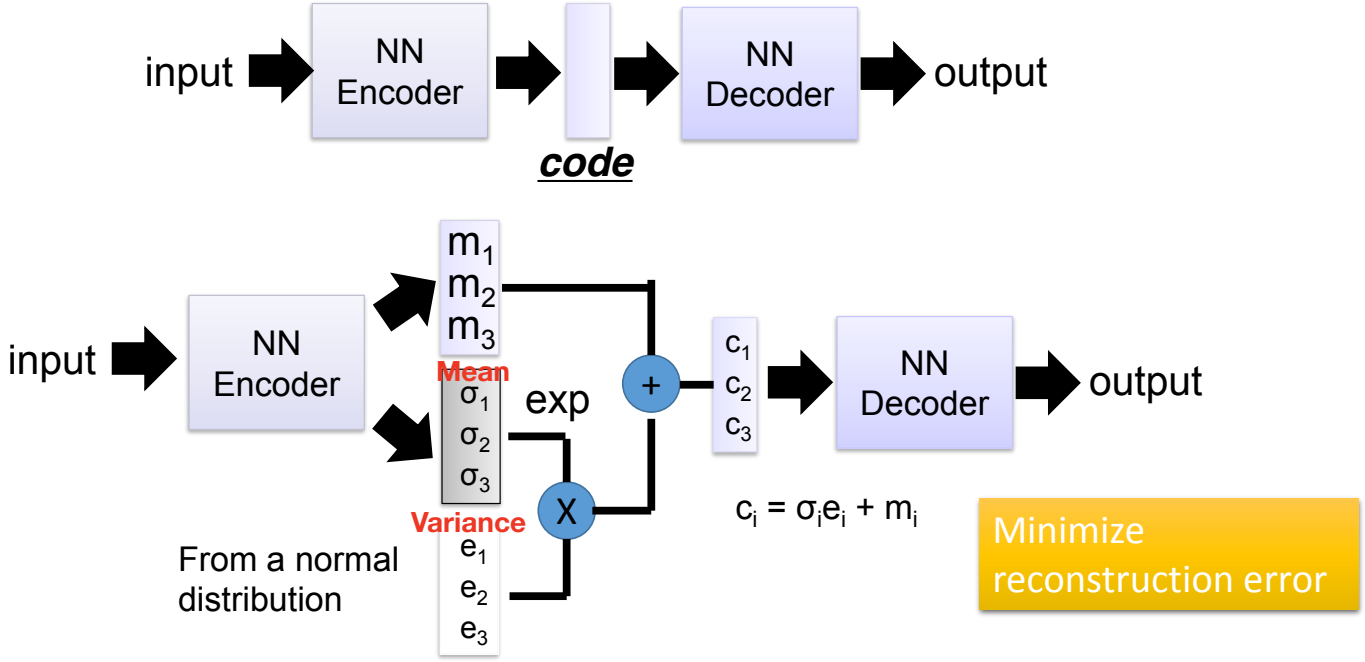
Auto-encoder







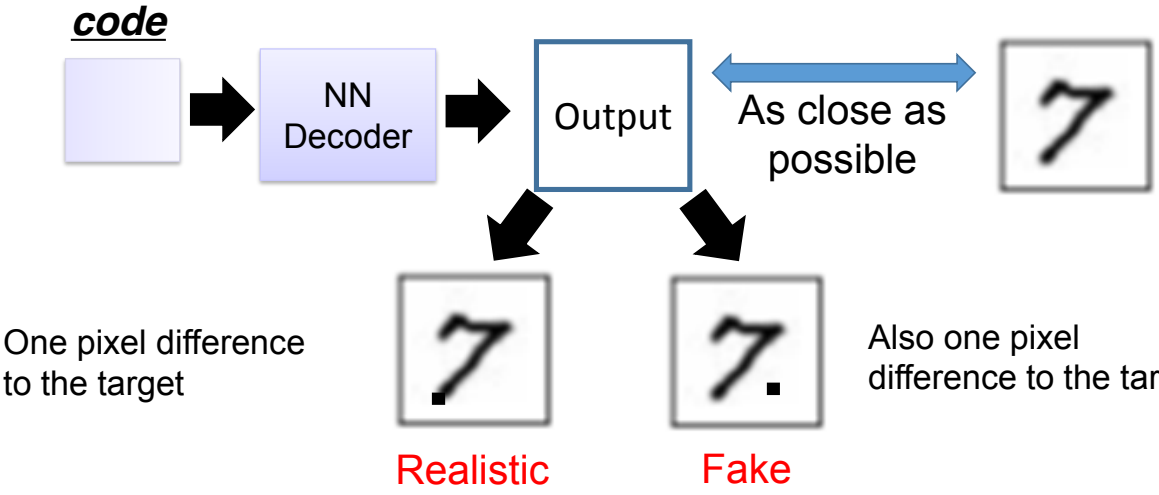
Variational Autoencoder



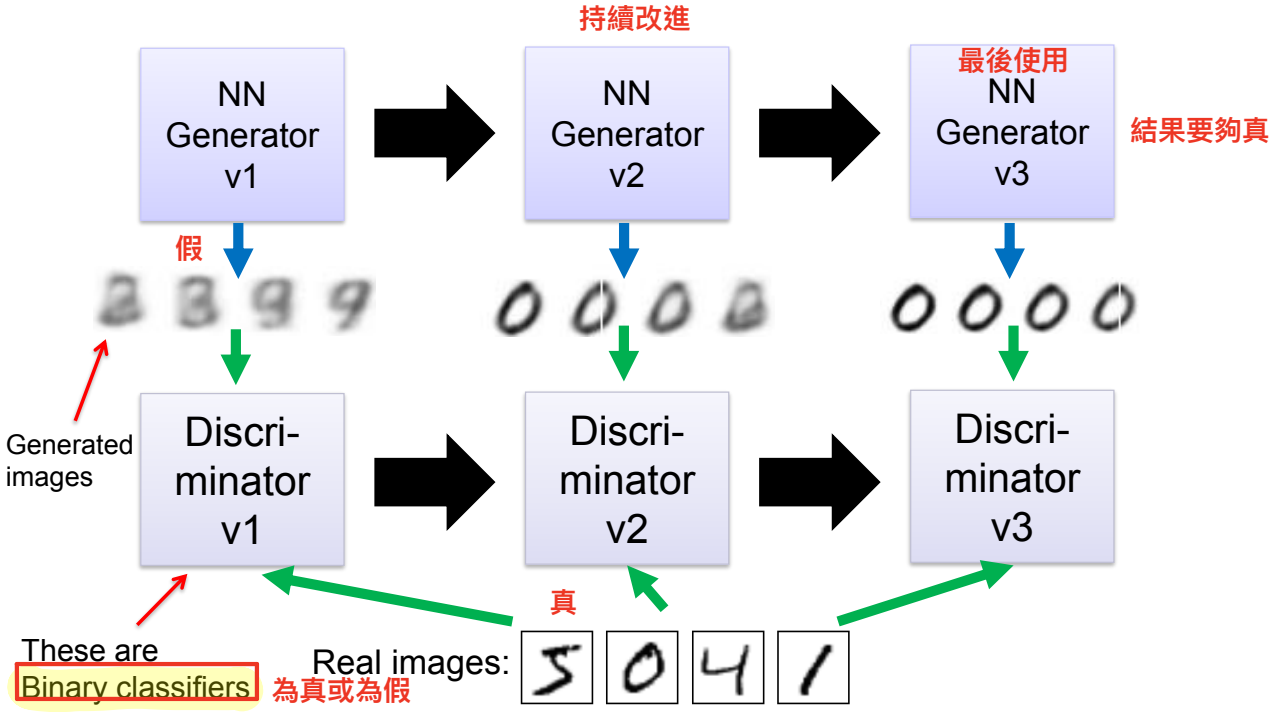
Auto-Encoding Variational Bayes, https://arxiv.org/abs/1312.6114

Problems of VAE

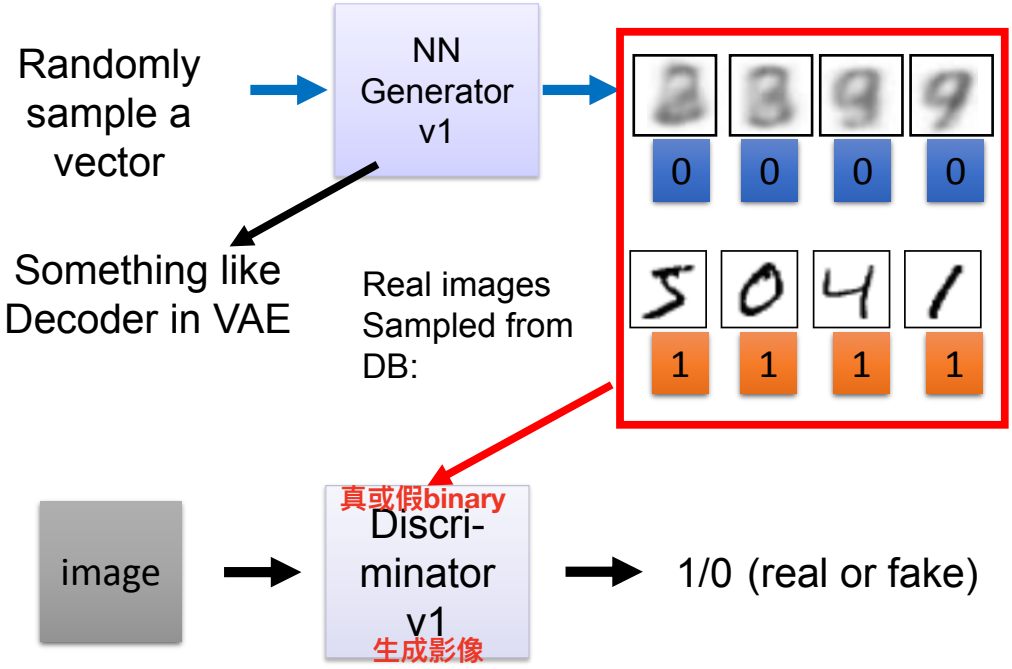
- It does not really try to simulate real images



Gradual and Step-wise Generation



GAN – Learn a Discriminator



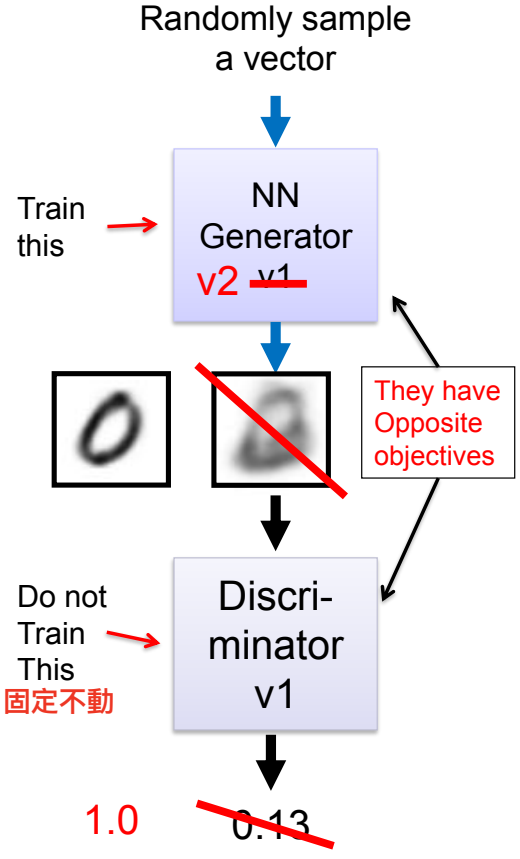
GAN – Learn a Generator

Updating the parameters of generator

The output be classified as “real” (as close to 1 as possible)

Generator + Discriminator = a network

Using gradient descent to update the parameters in the generator, but fix the discriminator 生成改進在衡量



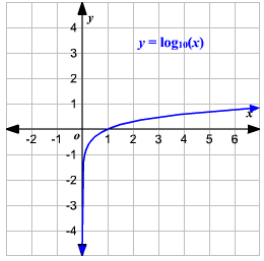
Generative Adversarial Networks

- The generator and discriminator networks compete:
  - Discriminator network trains to classify real vs. generated images.
    - Tries to maximize probability of real images, minimize probability of sampled images. 生成與照片做二元分類問題
    - A standard supervised learning problem.
  - Generator network adjust parameters so samples fool the discriminator. 生成隨機影像改進
    - It never sees real data. 完全沒有看過正確影像
    - Trains using the gradient of the discriminator network.
      - Backpropagated through the network so samples look more like real images.

Generative Model of Natural Image

- Notation of Generative Model:
  - The distribution of real data : $x \sim p_{data}(x)$
  - Sample from $p_{data}(x)$: X
  - The distribution of generated data : $z \sim p(\mathrm{z})$
  - Sample from $P(z)$: Z **Dataset生成的資料**
  - Generator Network : $G(z)$ **生成器**
  - Discriminator Network : $D(x)$ **判斷**
  - Real data X : $x \sim P_{data}$
- Unsupervised representation learning
  - Transfer learned representation to discriminative tasks, retrieval, clustering, etc.
  - Semi-supervised learning: very little labeled data, regularization, etc.
- Understand data; Density estimation ...

Objective Function



- Minimax value function: 最大化或最小化

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

最大化式子

Discriminator pushes up

Generator pushes down

Discriminator's ability to recognize data as being real

真實資料丟入值要為1

Discriminator's ability to recognize generator samples as being fake

0~1的範圍值

Objective Function - Generator

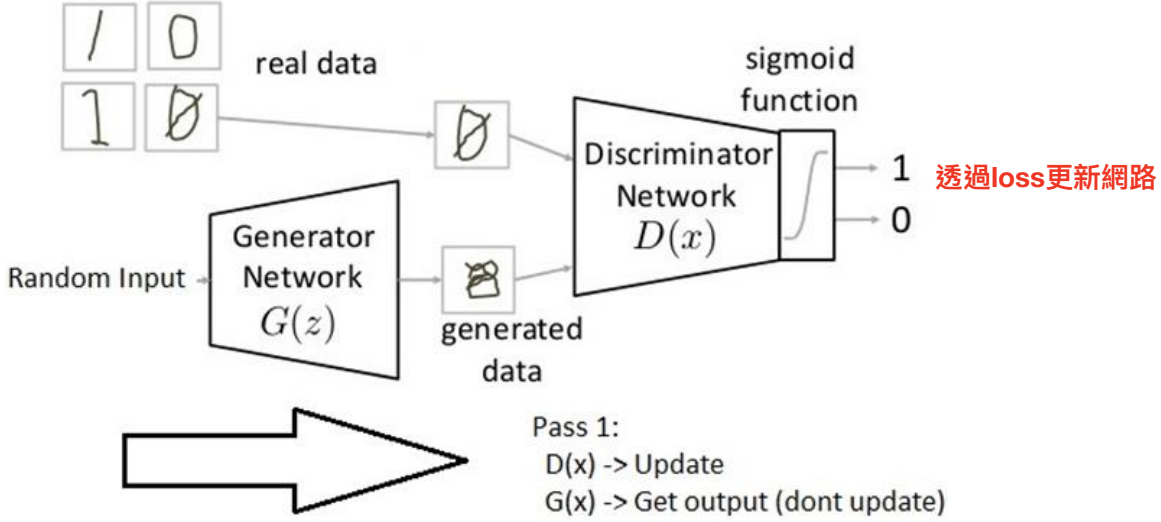
- While the previous objective function is theoretically elegant, in practice the gradient for *G* vanishes before reaching best solution.
- Use same objective for *D* but change objective for *G* to:

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D(G(\boldsymbol{z}))$$

- Sometimes better if instead of using one minibatch at a time to compute gradient and do batch normalization, we also have a fixed subset of training set, and use combination of fixed subset and current minibatch.

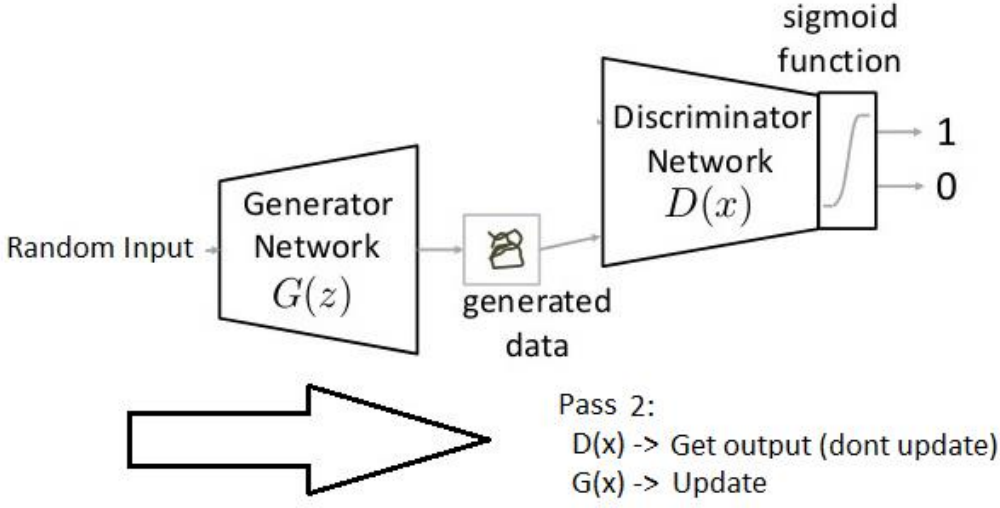
Training GAN (1)

- **Pass 1**: Train discriminator and freeze generator.



Training GAN (2)

- **Pass 2**: Train generator and freeze discriminator



Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do** K個步驟

此步驟做k次 • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.

  • Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$. 判斷一定要是準的

  • Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for 只有一次

  • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.

  • Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

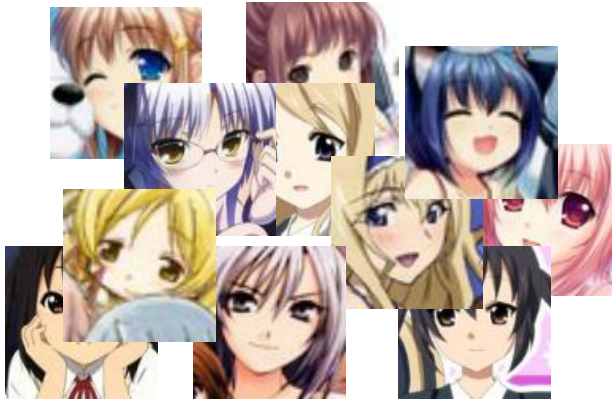
end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Steps to train a GAN

- **Step 1: Define the problem.** Do you want to generate fake images or fake text. Here you should completely define the problem and collect data for it.
- **Step 2: Define architecture of GAN.** Define how your GAN should look like. Should both your generator and discriminator be multi layer perceptrons, or convolutional neural networks? This step will depend on what problem you are trying to solve.
- **Step 3: Train Discriminator on real data for *n* epochs.** Get the data you want to generate fake on and train the discriminator to correctly predict them as real. Here value *n* can be any natural number between 1 and infinity.
- **Step 4: Generate fake data from Generator and train Discriminator on fake data.** Get generated data and let the discriminator correctly predict them as fake.
- **Step 5: Train Generator with the output of Discriminator.** Now when the discriminator is trained, you can get its predictions and use it as an objective for training the generator. Train the generator to fool the discriminator.
- **Step 6: Repeat step 3 to step 5 for a few epochs.**
- **Step 7: Check if the fake data manually if it seems legit. If it seems appropriate, stop training, else go to step 3.**

Generating 2nd element figures

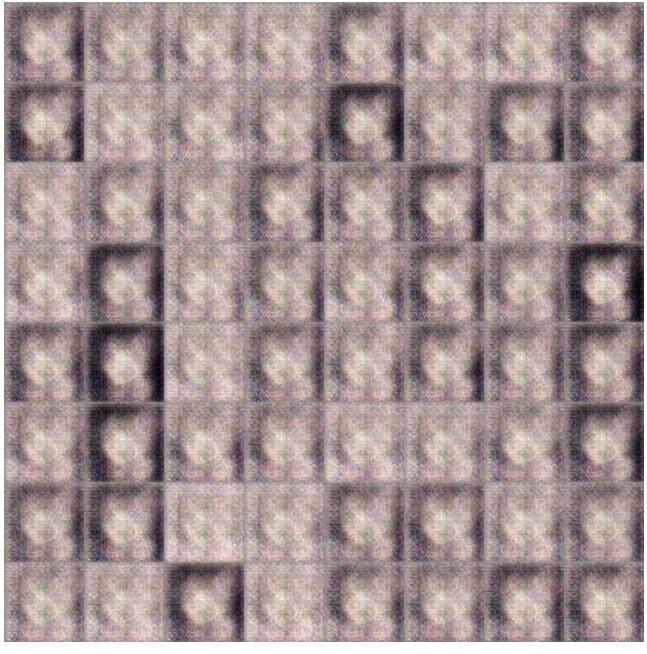


You can use the following to start a project (but this is in Chinese):

Source of images: https://zhuanlan.zhihu.com/p/24767059
From Dr. HY Lee's notes.

DCGAN: https://github.com/carpedm20/DCGAN-tensorflow

GAN – generating 2nd element figures



100 rounds

This is fast, I think you can use your CPU

GAN – generating 2nd element figures

1000 rounds

GAN – generating 2nd element figures

2000 rounds

GAN – generating 2nd element figures

5000 rounds

GAN – generating 2nd element figures

10,000 rounds

GAN – generating 2nd element figures

20,000 rounds

GAN – generating 2nd element figures

50,000 rounds

Challenges with GANs

難辨別生成時影像需要包含的東西

- **Problem with Counting**: GANs fail to differentiate how many of a particular object should occur at a location. As we can see below, it gives more number of eyes in the head than naturally present.





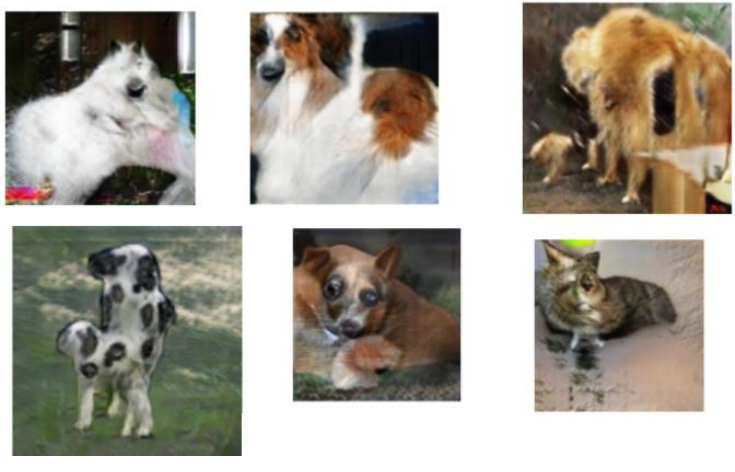

Challenges with GANs

較難應用在3D圖形

- **Problems with Perspective**: GANs fail to adapt to 3D objects. It doesn't understand perspective, i.e.difference between frontview and backview. As we can see below, it gives flat (2D) representation of 3D objects.

Challenges with GANs 難顧及整個架構輪廓，如身體頭四隻腳

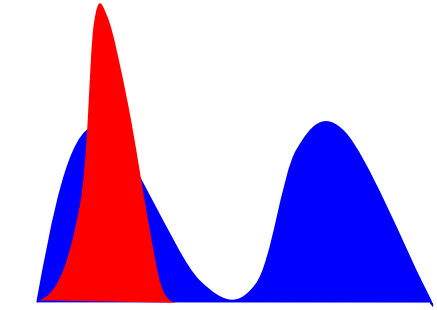
- **Problems with Global Structures:** Same as the problem with perspective, GANs do not understand a holistic structure. For example, a cow standing on its hind legs and simultaneously on all four legs. That is definitely not possible in real life!



Mode Collapse

2D平面較佳

Converge to same faces



Sometimes, this is hard to tell since
one sees only what's generated, but not what's missed.

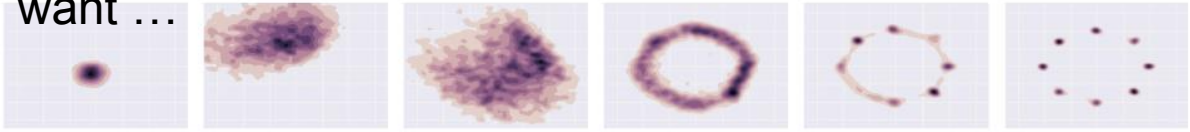
Mode Collapse Example

8 Gaussian distributions:

生成完變的特徵差不多

What we want …

生成8群



In reality …

只變成一種狀態

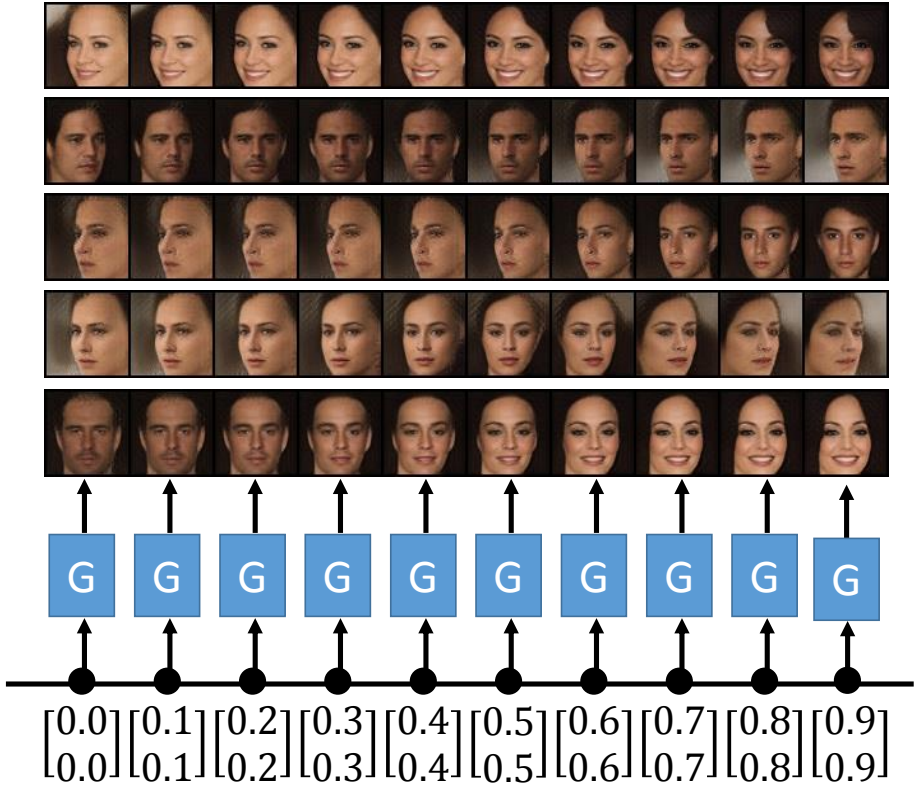
Deep Convolutional Generative Adversarial Network (DCGAN)

- Use deep CNN for generator and discriminator instead of MLP.
  - Replace any pooling layers with strided convolution.
  - Use batchnorm in both the generator and the discriminator.
  - Remove fully connected hidden layers for deeper architectures.
  - Uses Tanh for the output (and sigmod).
  - Use LeakyReLU in the discriminator and ReLU in the generator.
- Use the trained discriminators for image classification tasks.

Image Generation Results

Figure : Generated bedrooms on LSUN

Face Generation

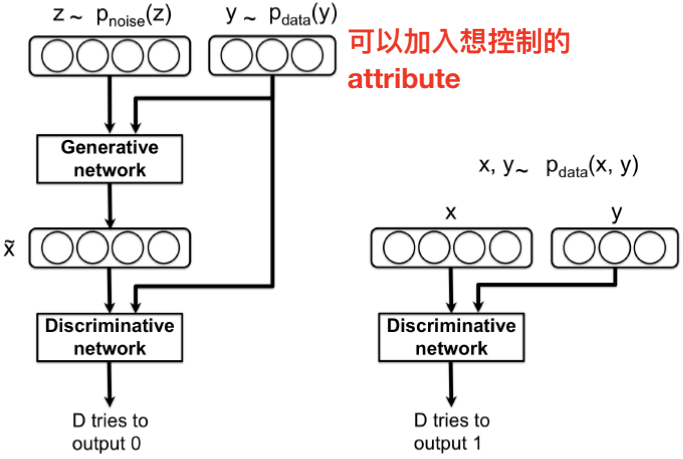


Conditional Generative Adversarial Nets

- Condition generation on additional info y (e.g. class label)
- Objective function:

$$\min_G \max_D \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})|\boldsymbol{y}))]$$

生成時可以下condition



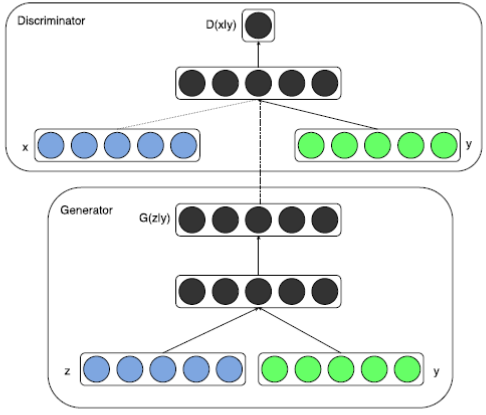


Figure : Model structure for conditional GAN

Figure : Conditional GAN with MLP

Results on CIFAR-100 Generated 讓生成結果不太一樣

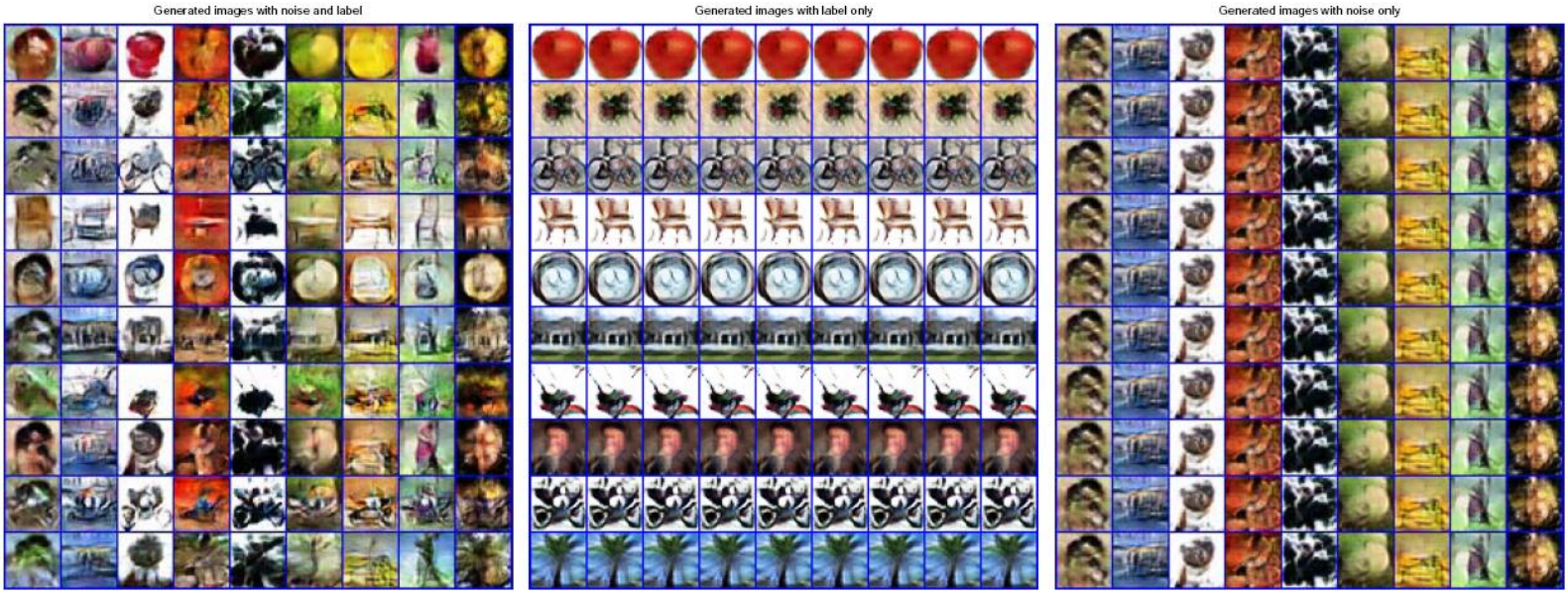


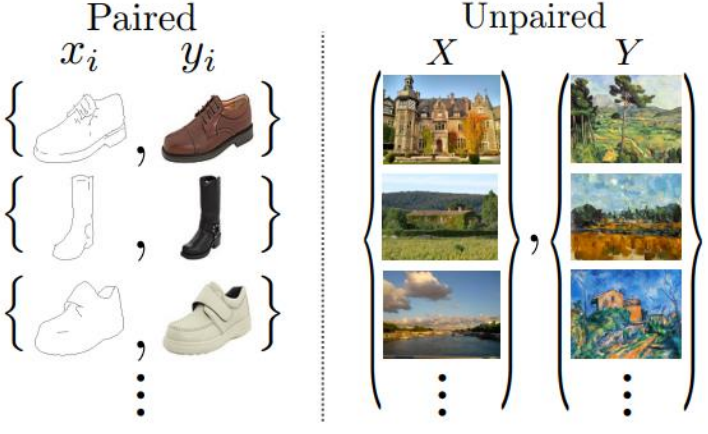
Figure : (Left) Each row has the same label y and each column has the same noisy vector z (Middel) Set noisy vector as zero (Right) Set label vector as zero

Cycle-Consistent Adversarial Networks (**CycleGAN**)

如果只想做兩個領域的圖生成

- Image-to-image translation is a class of vision and graphics problems where the goal is to learn the mapping between an input image and an output image using a training set of aligned image pairs.
- However, for many tasks, paired training data will not be available.

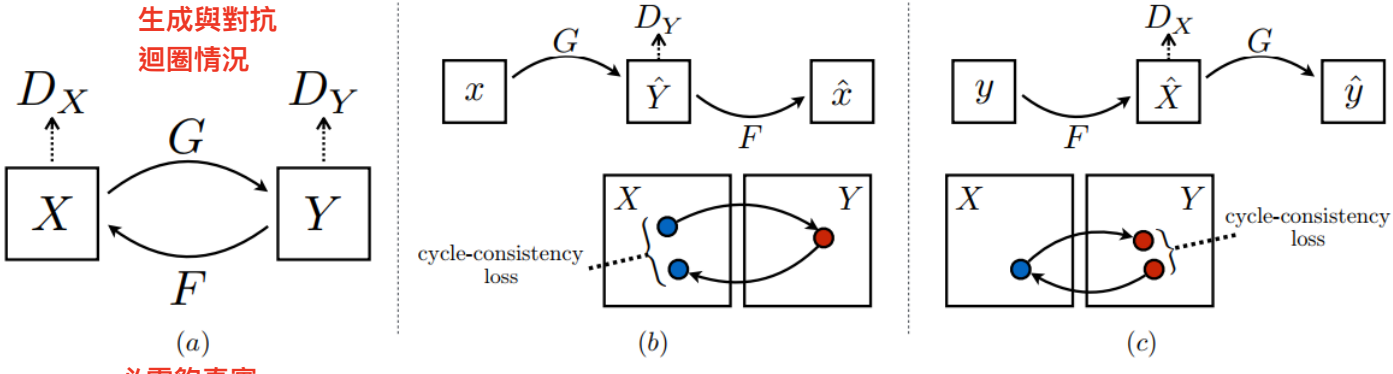
必需為成隊影像



The Problem of Adversarial Loss

- Adversarial training can, in theory, learn mappings G and F that produce outputs identically distributed as target domains Y and X respectively.
- However, with large enough capacity, a network can map the same set of input images to any random permutation of images in the target domain, where any of the learned mappings can **induce an output distribution that matches the target distribution**.
- Thus, adversarial losses alone **cannot guarantee that the learned function can map an individual input x_i to a desired output y_i** .

Cycle Consistency Loss



Cycle Consistency Loss

- For each image x from domain X, the image translation cycle should be able to bring x back to the original image,
  i.e. **x → G(x) → F(G(x)) ≈ x**. 轉過去與轉回來要夠像
  We call this forward cycle consistency
- Similarly, for each image y from domain Y , G and F should also satisfy backward cycle consistency:
  i.e. **y → F(y) → G(F(y)) ≈ y**.

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

Full Objective

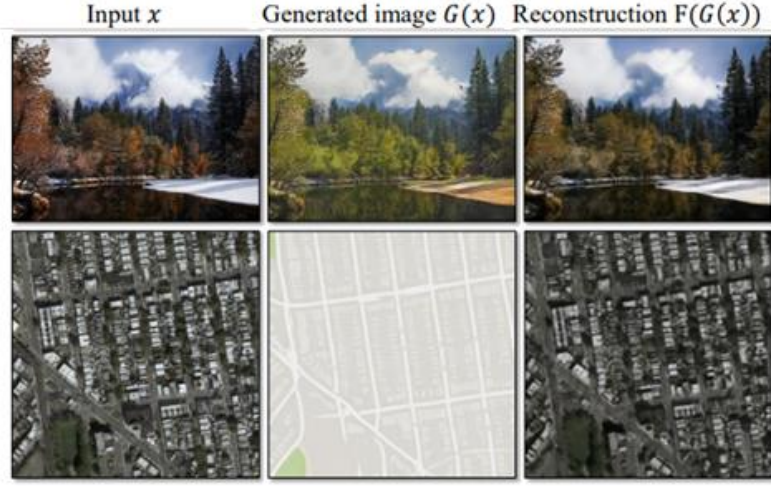
- Full objective is:

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) =& \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda \mathcal{L}_{\text{cyc}}(G, F), \qquad (3)\end{aligned}$$

- Aim to solve:

$$G^*, F^* = \arg\min_{G,F} \max_{D_x, D_Y} \mathcal{L}(G, F, D_X, D_Y). \qquad (4)$$

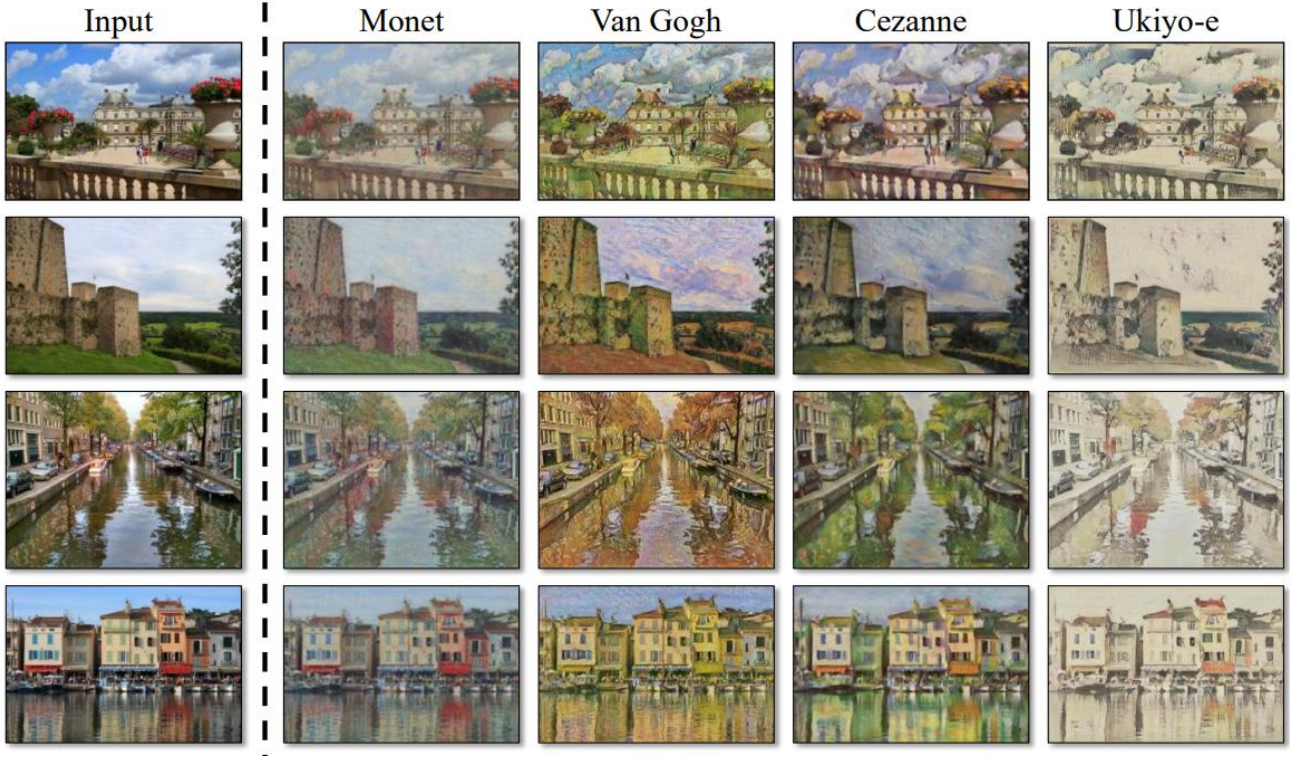
Cycle Consistency Loss



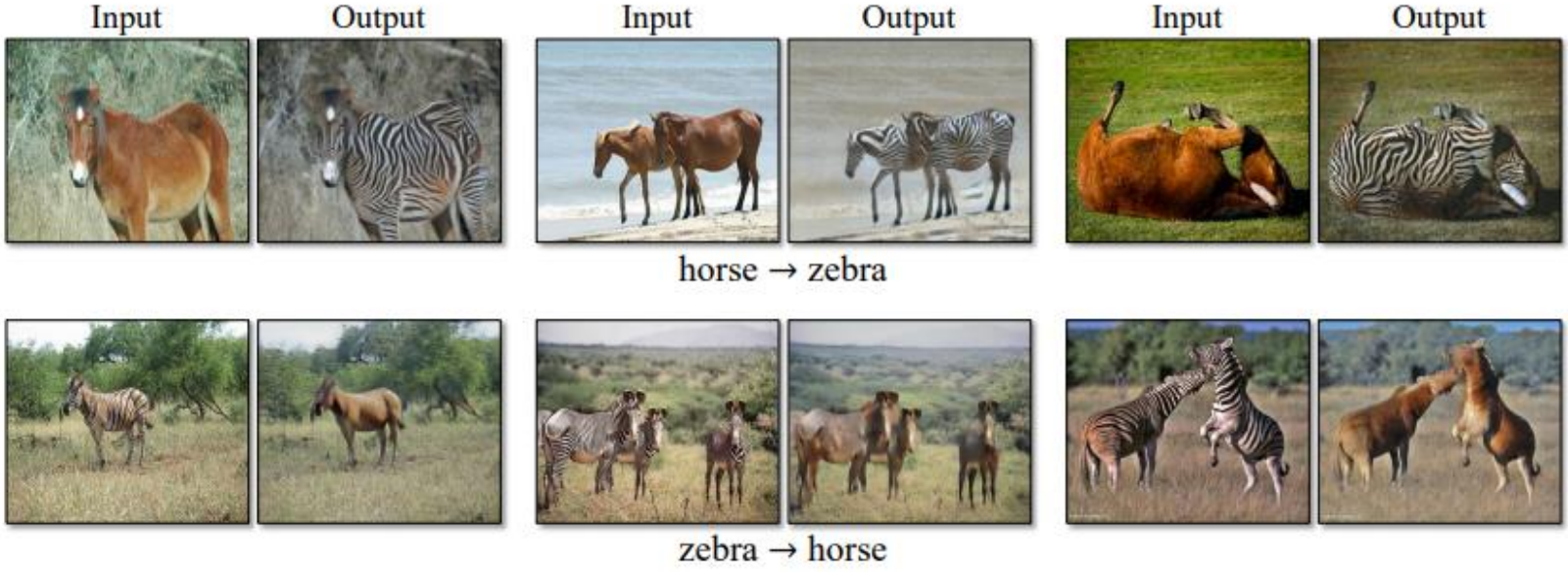
Applications

- Collection style transfer 季節轉換
- Object transfiguration
- Season transfer
- Photo generation from paintings
- Photo enhancement

Collection Style Transfer 畫風轉換



Object Transfiguration



Season Transfer

winter Yosemite → summer Yosemite

summer Yosemite → winter Yosemite

Photo Generation from Paintings

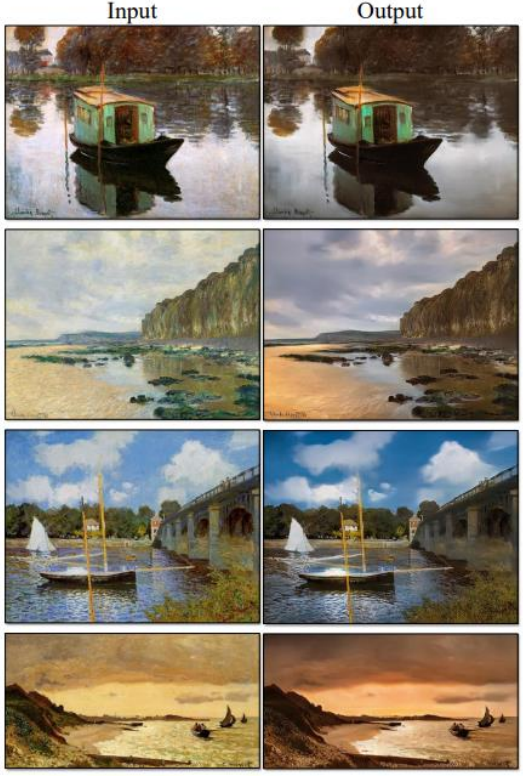
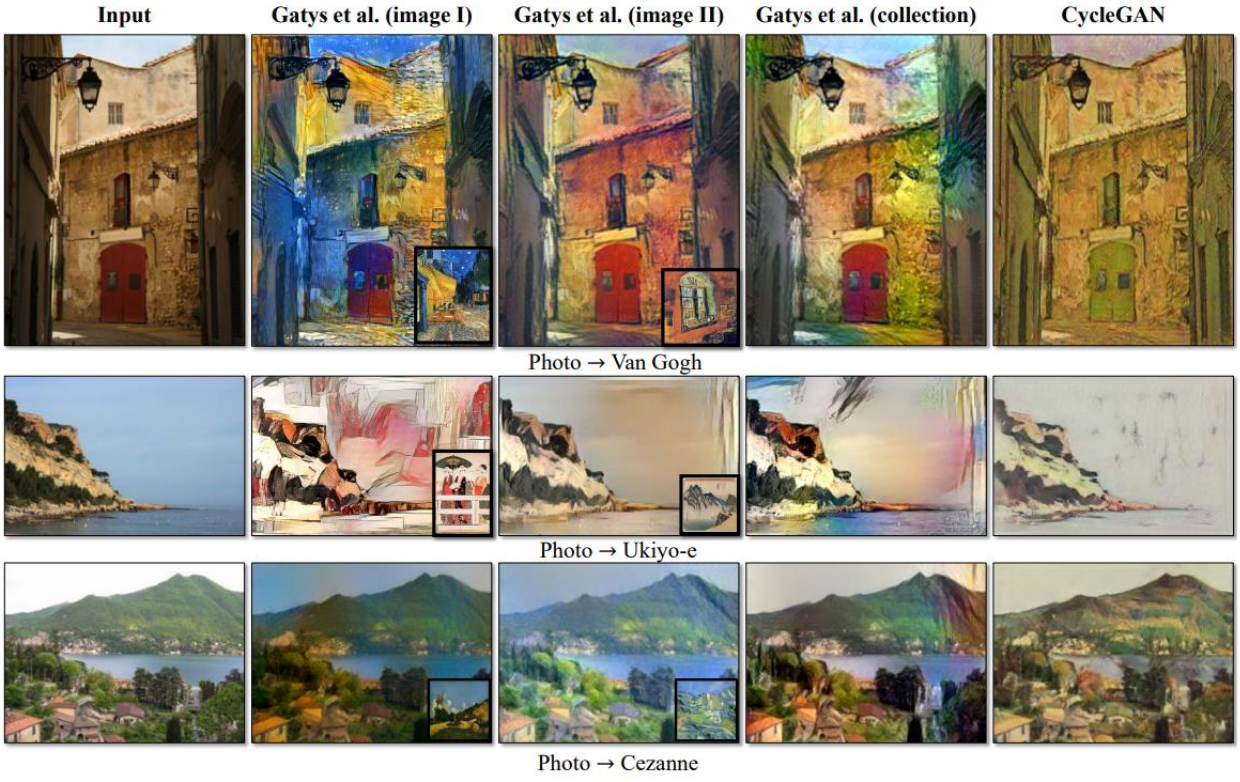


Photo Enhancement – Simulation of Depth Camera

產生景深照片

Figure 14: Photo enhancement: mapping from a set of iPhone snaps to professional DSLR photographs, the system often learns to produce shallow focus. Here we show some of the most successful results in our test set – average performance is considerably worse. Please see our website for more comprehensive and random examples.

Comparison with Gatys et al.



Failure Cases

蘋果轉橘子

沒有看過人在馬上，產生錯誤

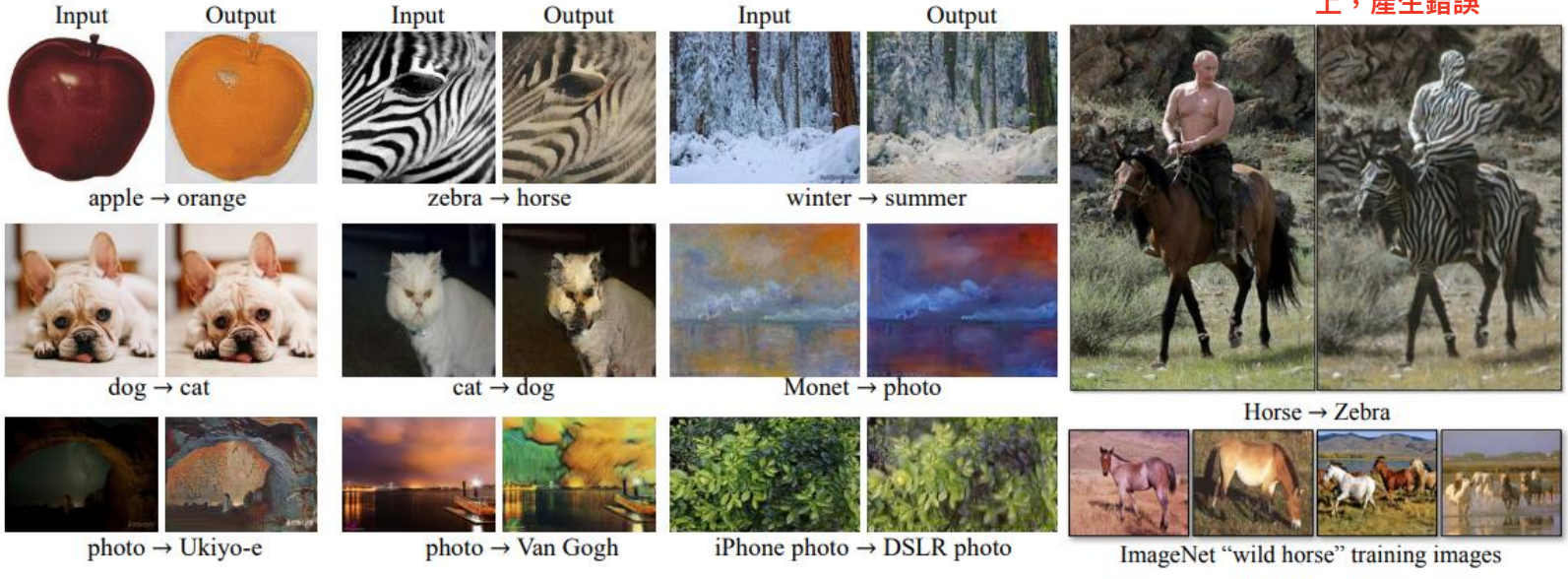


Figure 17: Typical failure cases of our method. Please see our website for more comprehensive results.

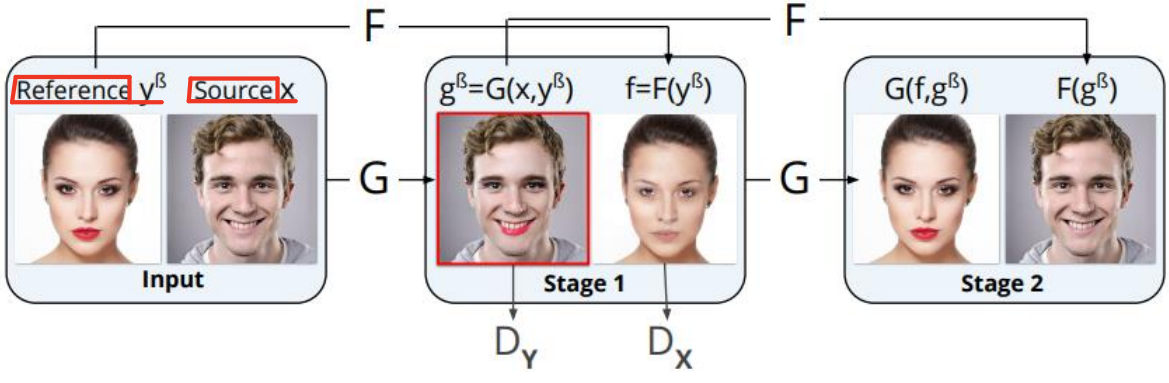
Limitations and Discussion

- On translation tasks that involve color and texture changes, like many of those reported above, the method often succeeds.
- Some failure cases are caused by the distribution characteristics of the training datasets. For example, the horse → zebra example (Figure 17, right) has got confused, because our model was trained on the wild horse and zebra synsets of ImageNet, which does not contain images of a person riding a horse or zebra.
- Handling more varied and extreme transformations, especially **geometric changes**, is an important problem for future work.

PairedCycleGAN : Makeup Transfer and Removal

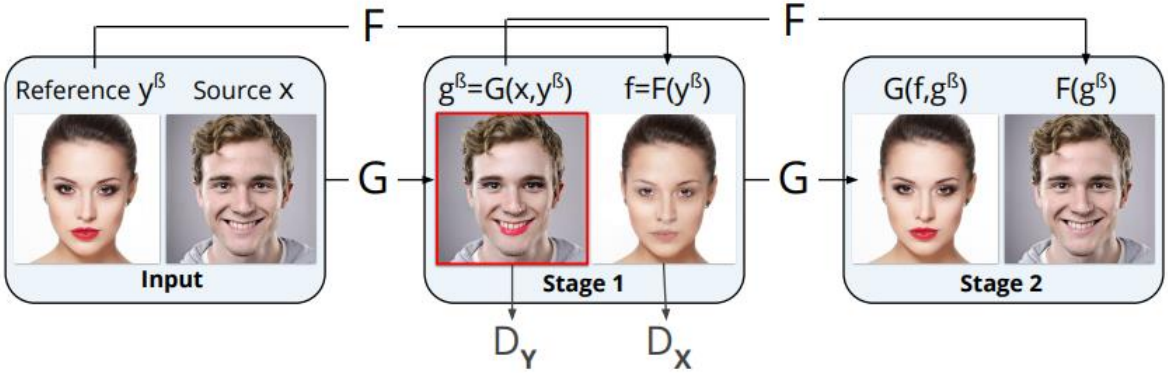
CycleGAN would not replicate a specific example makeup style.

Network Pipeline



G: markup transfer function
F: markup removal function
X: no-makeup image domain
Y: with-makeup image domain
β: specific makeup style

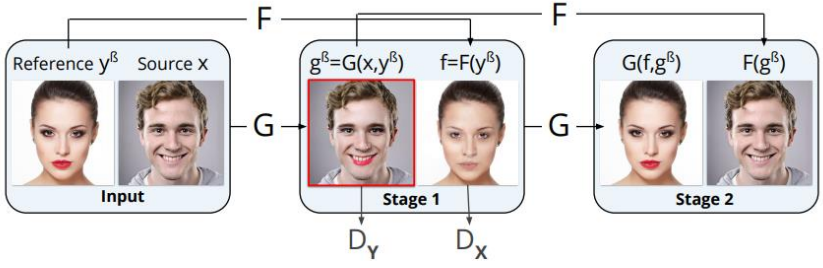
Total Loss



$$\min_{G,F} \max_{D_X,D_Y,D_S} L$$

$$L = \lambda_G L_G + \lambda_F L_F + L_I + L_S + \lambda_P L_P$$

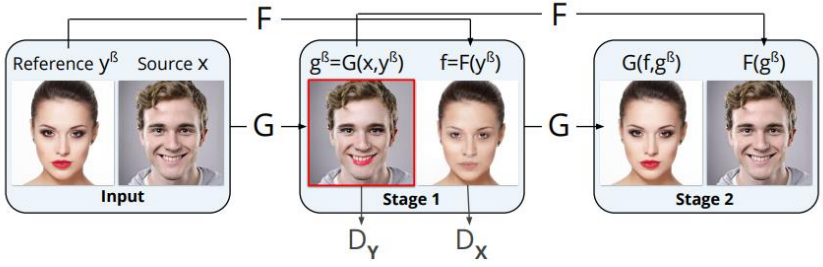
Adversarial Loss for G



$$L = \lambda_G L_G + \lambda_F L_F + L_I + L_S + \lambda_P L_P$$

$$L_G(G, D_Y) = \mathbb{E}_{y \sim \mathcal{P}_Y}[log D_Y(y)] + \mathbb{E}_{x \sim \mathcal{P}_X, y \sim \mathcal{P}_Y}[log(1 - D_Y(G(x, y)))] \quad (1)$$

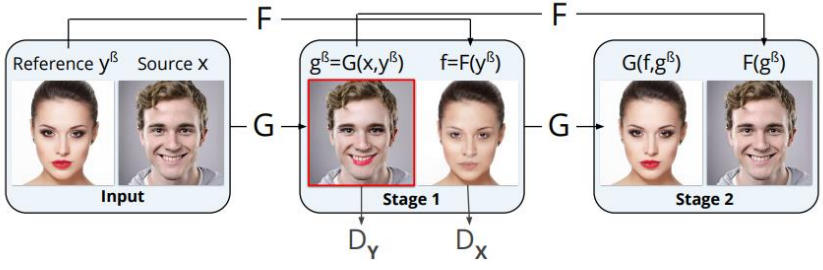
Adversarial Loss for F



$$L = \lambda_G L_G + \lambda_F L_F + L_I + L_S + \lambda_P L_P$$

$$L_F(F, D_X) = \mathbb{E}_{x \sim \mathcal{P}_X}[log D_X(x)] + \mathbb{E}_{y^\beta \sim \mathcal{P}_Y}[log(1 - D_X(F(y^\beta)))] \quad (2)$$

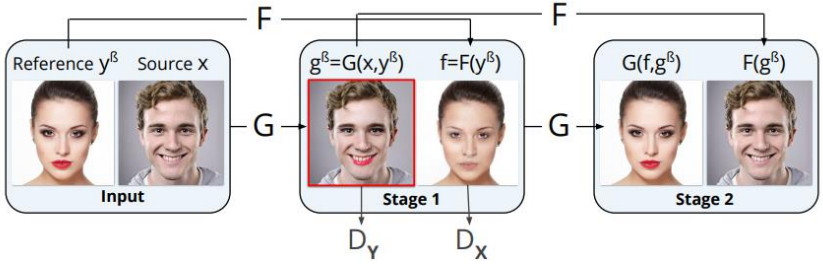
Identity Loss



$$L = \lambda_G L_G + \lambda_F L_F + L_I + L_S + \lambda_P L_P$$

$$L_I(G, F) = \mathbb{E}_{x \sim \mathcal{P}_X, y^\beta \sim \mathcal{P}_Y}[||F(G(x, y^\beta)) - x||_1] \quad (3)$$

Style Losses: L1 Reconstruction Loss

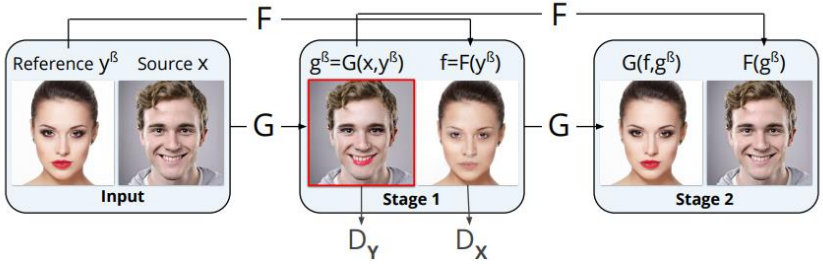


$$L = \lambda_G L_G + \lambda_F L_F + L_I + L_S + \lambda_P L_P$$

$$L_S(G, F) = \mathbb{E}_{x \sim \mathcal{P}_X, y^\beta \sim \mathcal{P}_Y} [||G(F(y^\beta), G(x, y^\beta)) - y^\beta||_1] \quad (4)$$

But it leads to blurry results incapable of transferring sharp edges (e.g. eyelashes and eyeliners)

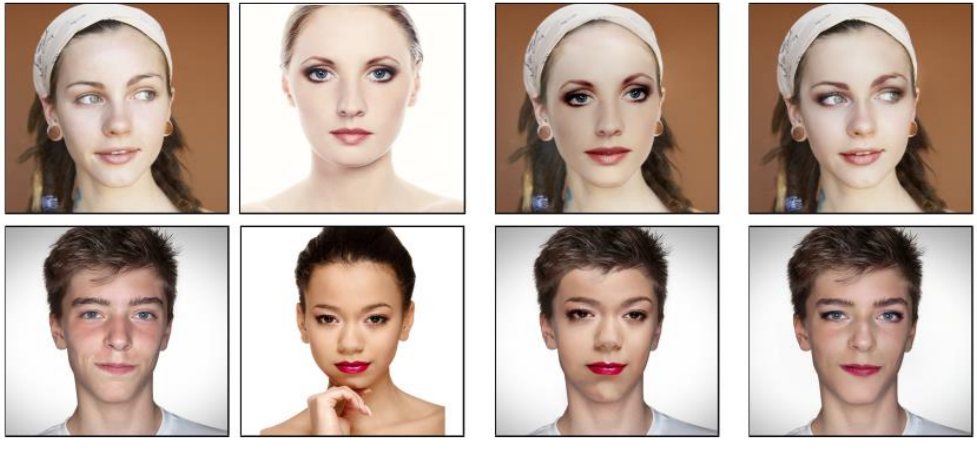
Style Losses: Style Discriminator Loss



$$L = \lambda_G L_G + \lambda_F L_F + L_I + L_S + \lambda_P L_P$$

$$L_P(G, D_S) = \mathbb{E}_{x \sim \mathcal{P}_X, y^{\beta} \sim \mathcal{P}_Y}[log D_S(y^{\beta}, W(x, y^{\beta}))] + \mathbb{E}_{x \sim \mathcal{P}_X, y^{\beta} \sim \mathcal{P}_Y}[log(1 - D_S(y^{\beta}, G(x, y^{\beta})))] \quad (5)$$

Synthetic Groundtruth W



Source Reference Warping results Our results

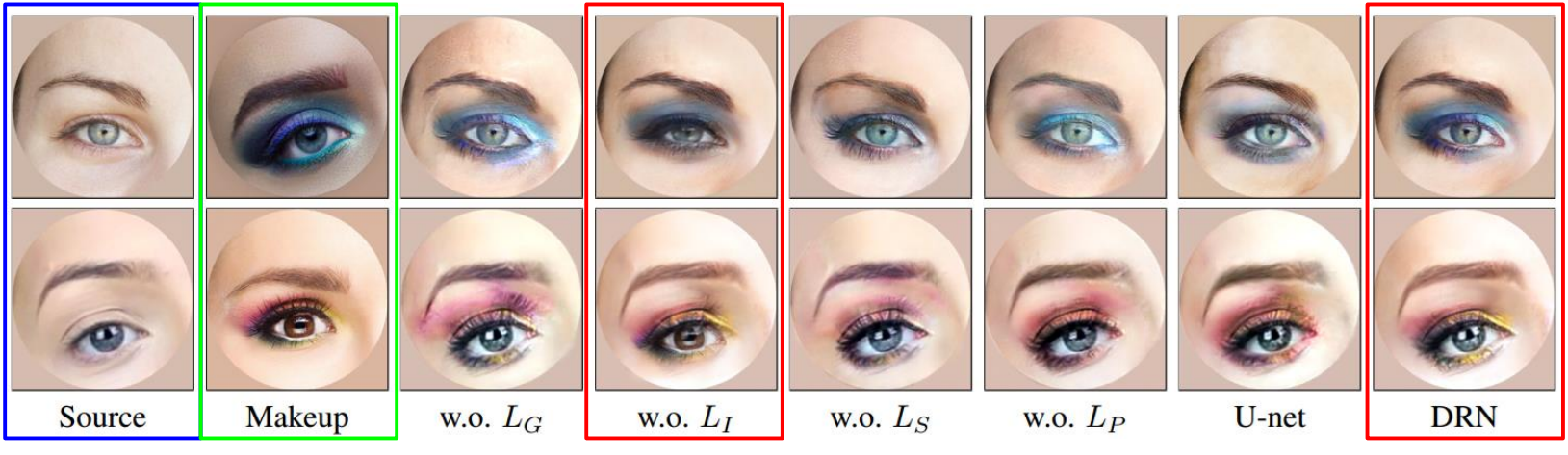
沒有化妝轉換為有化妝

Warp the reference makeup face to match the detected facial landmarks in the source face.

Network Architecture and Loss Analysis

Without loss for generator,
G could apply the eyeshadow anywhere around the eye.

Network Architecture and Loss Analysis

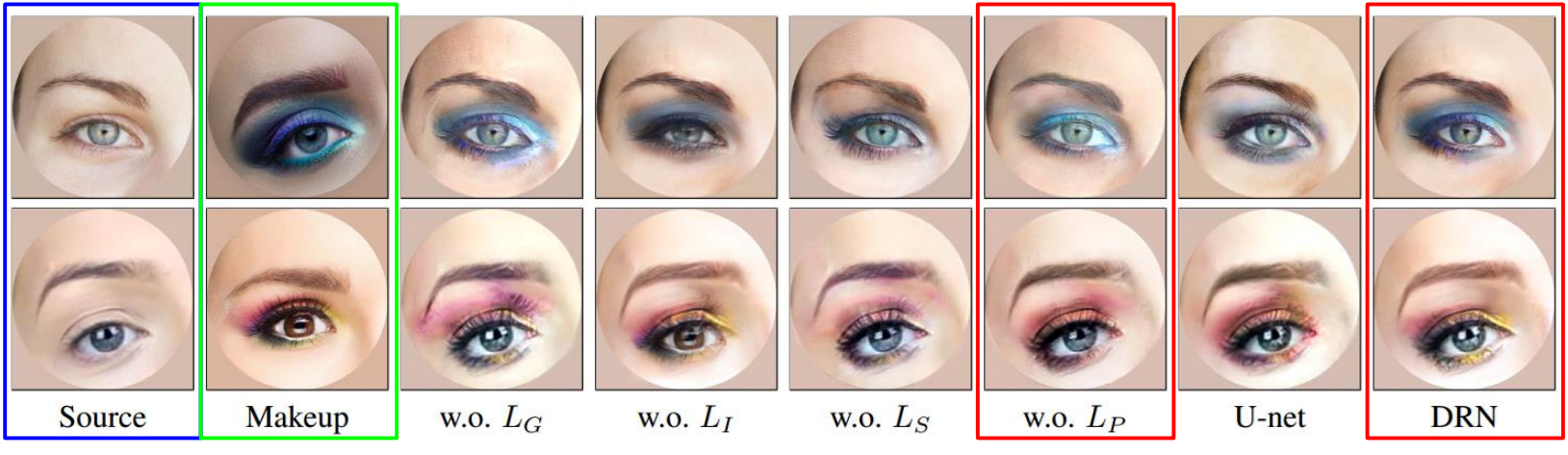


Without identity loss,
G could encodes the characteristics of eyes in both source and reference.

Network Architecture and Loss Analysis

Without style loss,
the results look less saturated and the makeup style is not fully transferred.

Network Architecture and Loss Analysis



Without style discriminator loss,
the results become more vivid, but some eyelashes get neglected.

Network Architecture and Loss Analysis

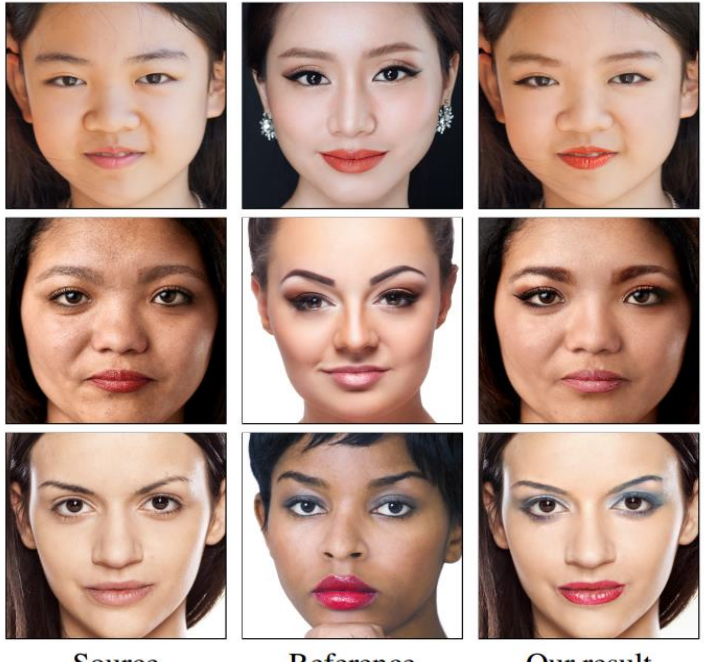
DRN > U-net

Results of the Lip and Eye Regions

The generated lips exhibit plausible colors and inherit the shiny appearance from the reference.

When the distance between eyes and eyebrows or the orientation of eyebrows are very different between the source and the reference, it can still synthesize plausible result.

Results on the Entire Face 上完妝與本人很像



Limitations: Makeup Style Unseen during Training

只有轉換有看過的部分

Makeup Transfer Comparison

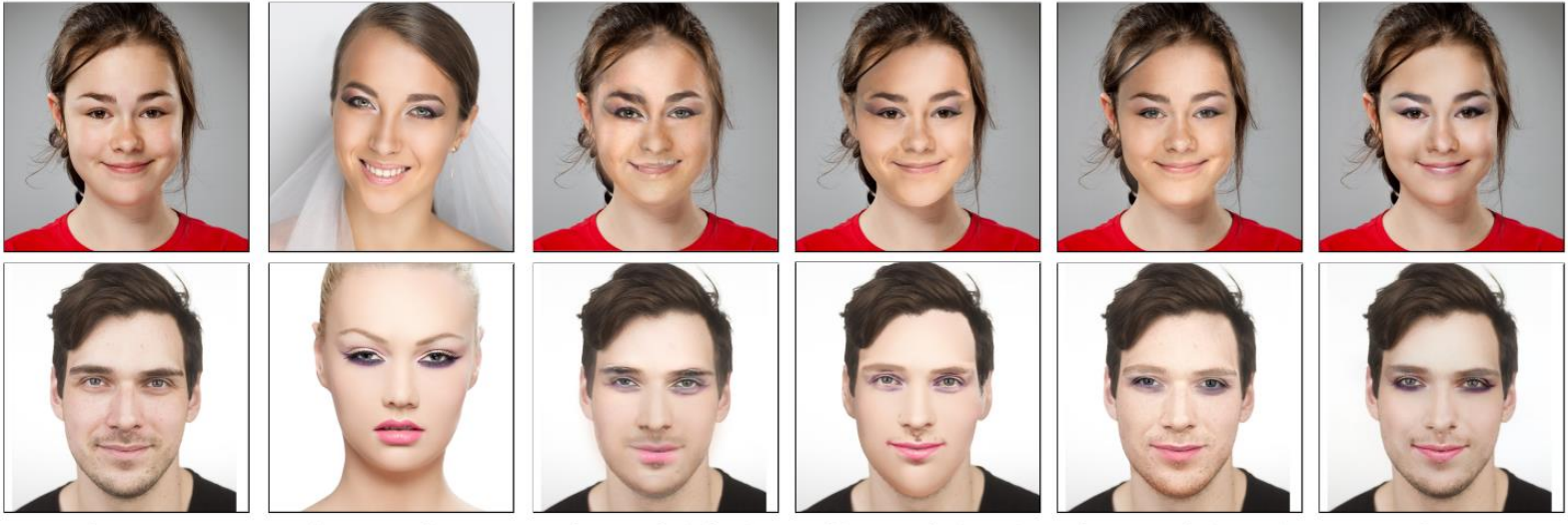


Figure 11: Makeup Transfer Results. We compare with makeup and portrait style transfer work [18, 5, 16].

Makeup Removal Comparison

Figure 12: Demakup Results. We compare with makeup removal work by Wang et al. *[24]. Our demakeup network F can remove the detected makeup to virtually recover the original face.*