Natural Language Processing

第三周 卷积神经网络

庞彦

yanpang@gzhu.edu.cn

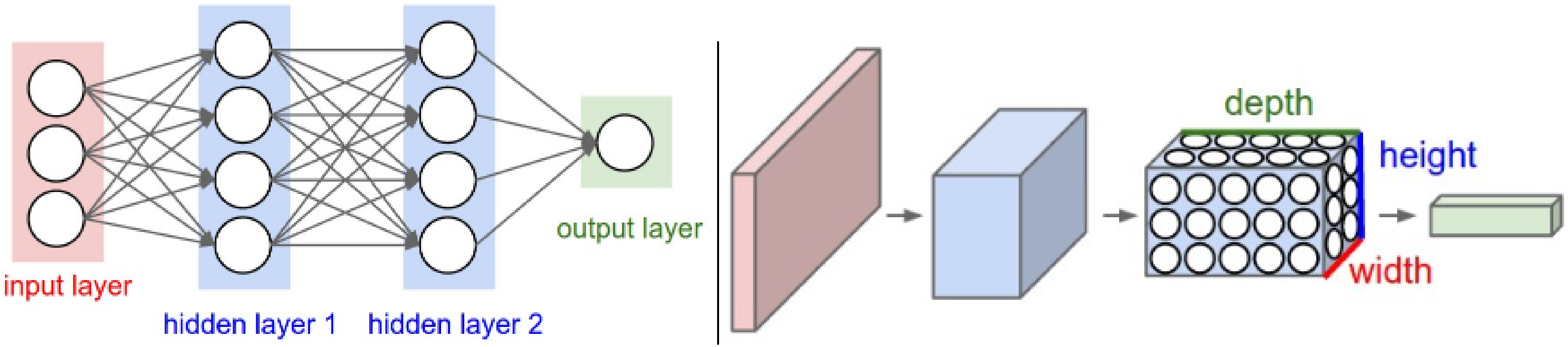
Overview

CONTENTS

01 Convolutional Neural Networks
卷积神经网络

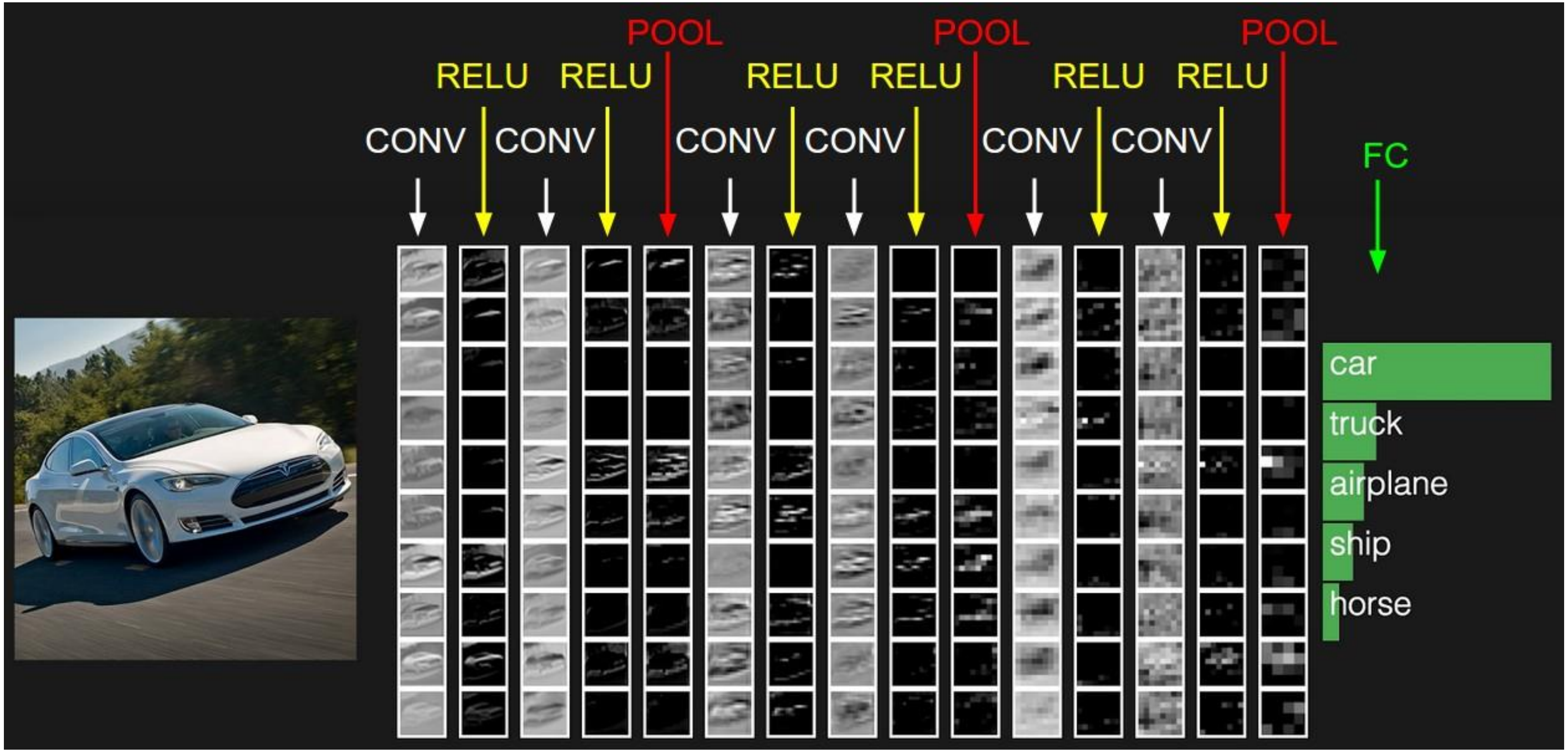
Spring 2023

Convolutional Neural Networks

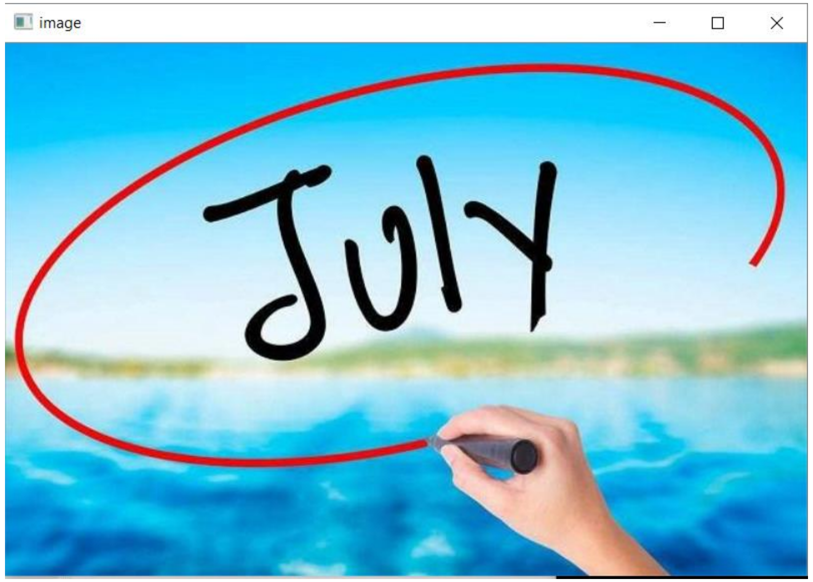


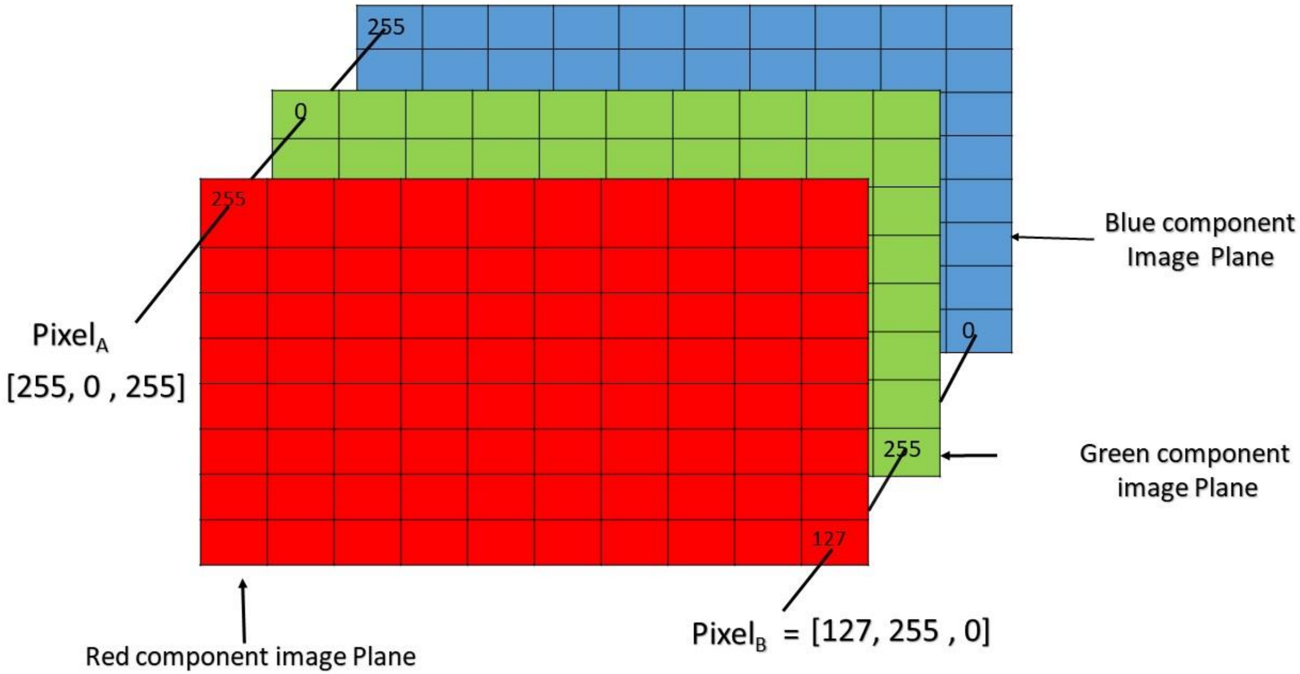
Left: A regular 3-layer Neural Network. Right: A ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

Features on each layer



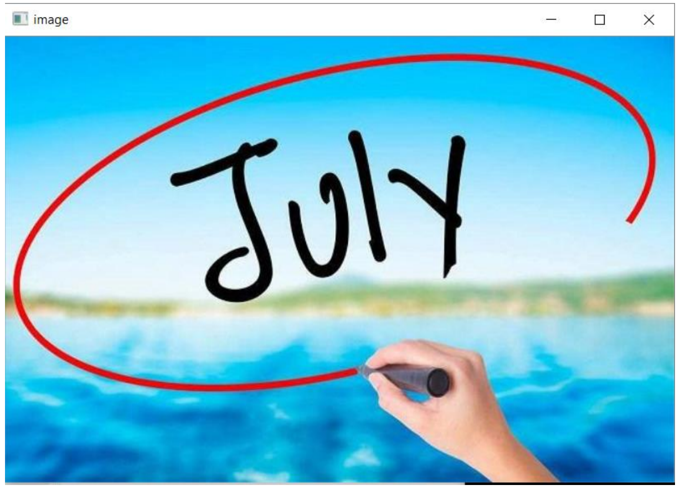
Color Image: 3 channels



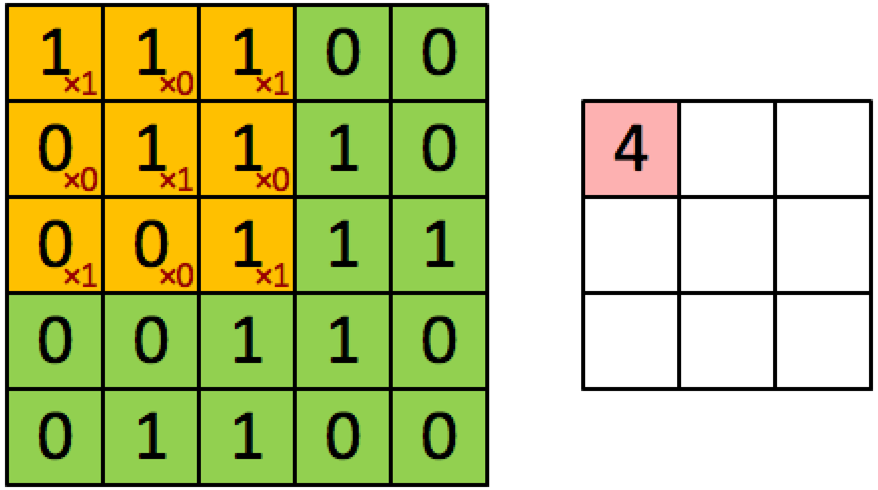


Pixel of an RGB image are formed from the corresponding pixel of the three component images

Color Image: 3 channels



Convolution Layer: The Kernel



Image

Convolved Feature

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Convolution Layer: The Kernel

Filter 1

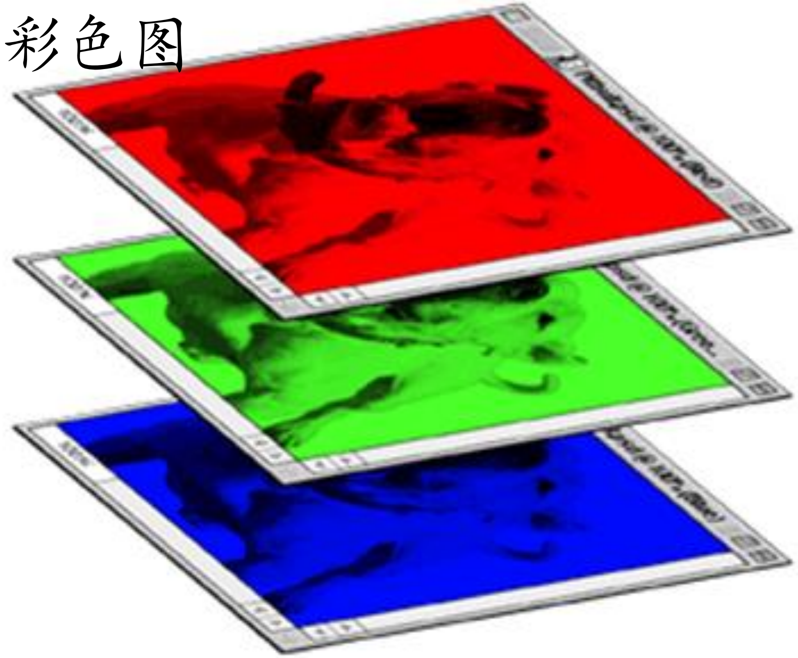
1	-1	-1
-1	1	-1
-1	-1	1

Filter 2

-1	1	-1
-1	1	-1
-1	1	-1

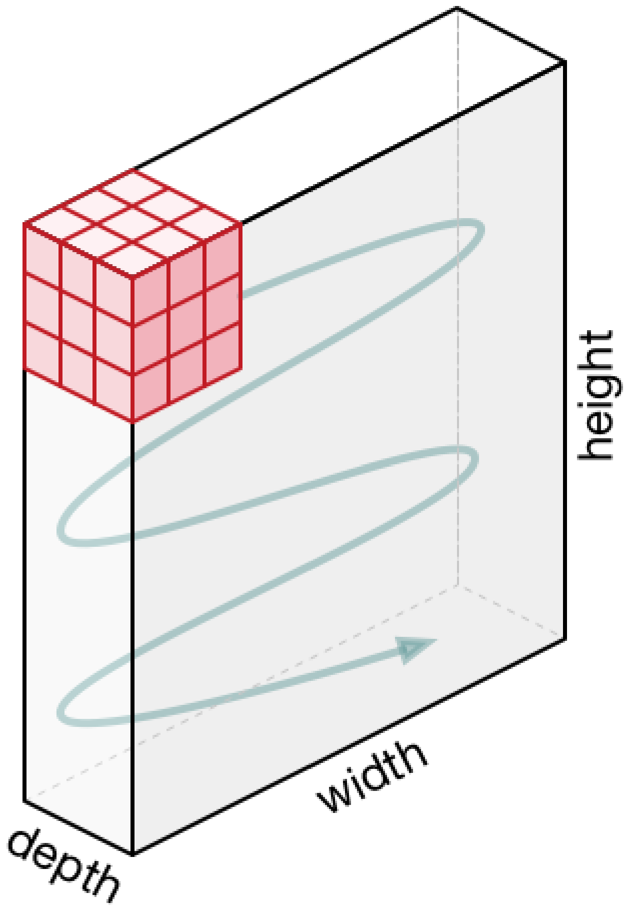
Colorful image

彩色图



1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

Convolution Layer: The Kernel



0	0	0	0	0	0	...
0	156	155	156	158	158	...
0	153	154	157	159	159	...
0	149	151	155	158	159	...
0	146	146	149	153	158	...
0	145	143	143	148	158	...
...	...	...	...	...	...	...

Input Channel #1 (Red)

0	0	0	0	0	0	...
0	167	166	167	169	169	...
0	164	165	168	170	170	...
0	160	162	166	169	170	...
0	156	156	159	163	168	...
0	155	153	153	158	168	...
...	...	...	...	...	...	...

Input Channel #2 (Green)

0	0	0	0	0	0	...
0	163	162	163	165	165	...
0	160	161	164	166	166	...
0	156	158	162	165	166	...
0	155	155	158	162	167	...
0	154	152	152	157	167	...
...	...	...	...	...	...	...

Input Channel #3 (Blue)

-1	-1	1
0	1	-1
0	1	1

Kernel Channel #1

1	0	0
1	-1	-1
1	0	-1

Kernel Channel #2

0	1	1
0	1	0
1	-1	1

Kernel Channel #3

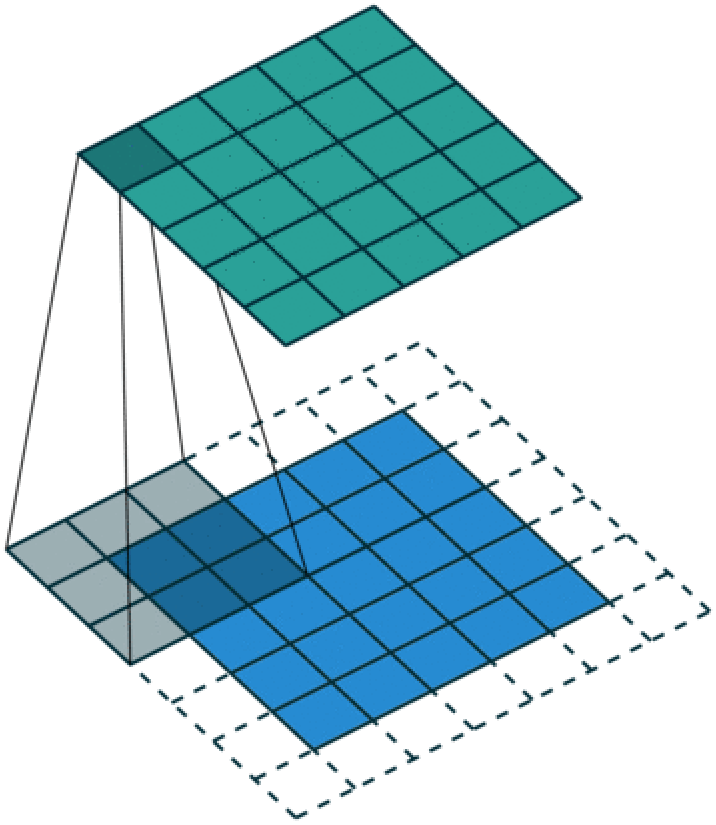
$$308 \quad + \quad -498 \quad + \quad 164 \quad +1 = -25$$

Bias = 1

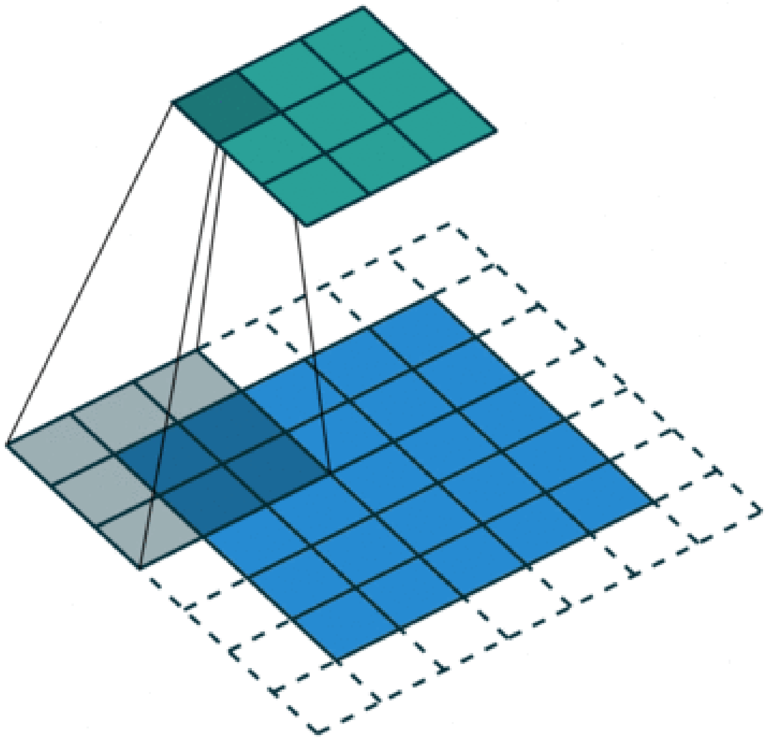
Output

-25				...
				...
				...
				...
...	...	...	...	...

Convolution Layer: Padding and Stride

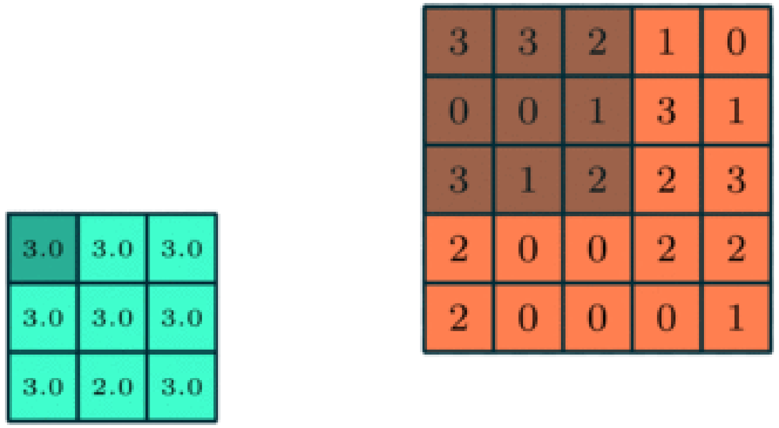


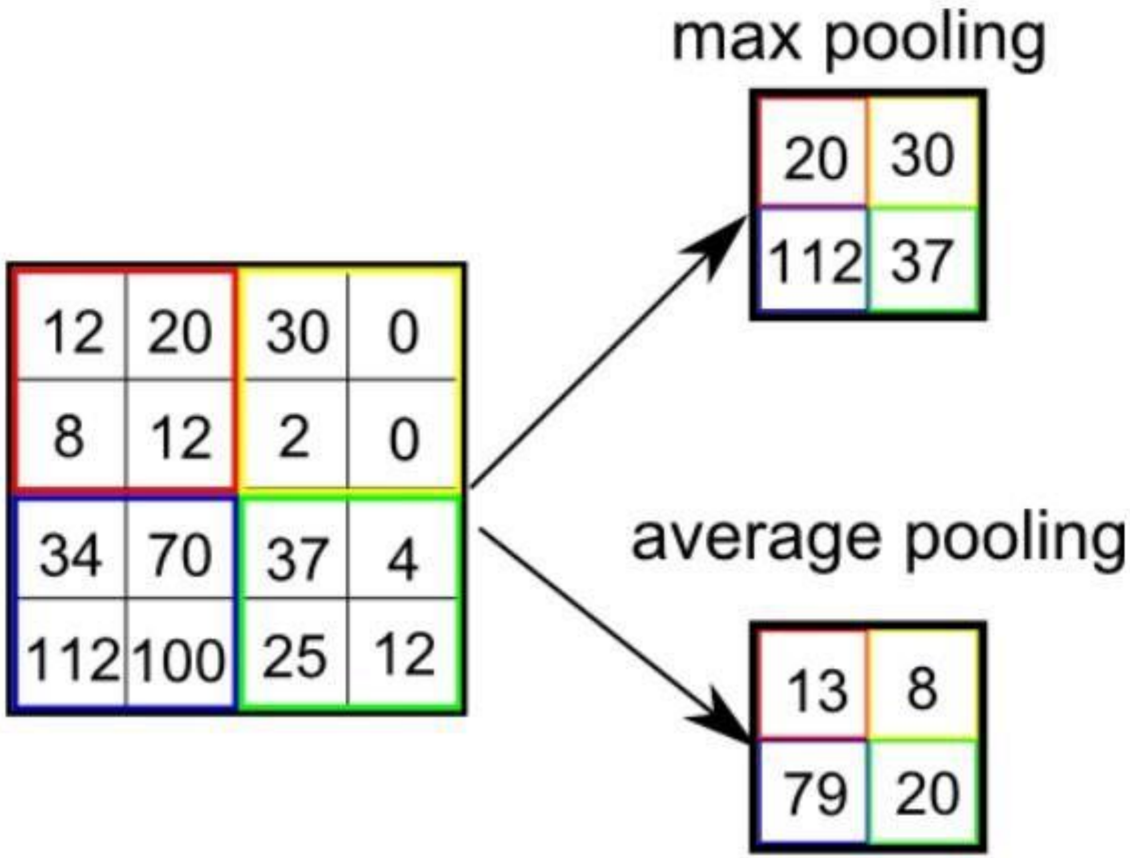
padding = 1
stride = 1



padding = 1
stride = 2

Pooling Layer

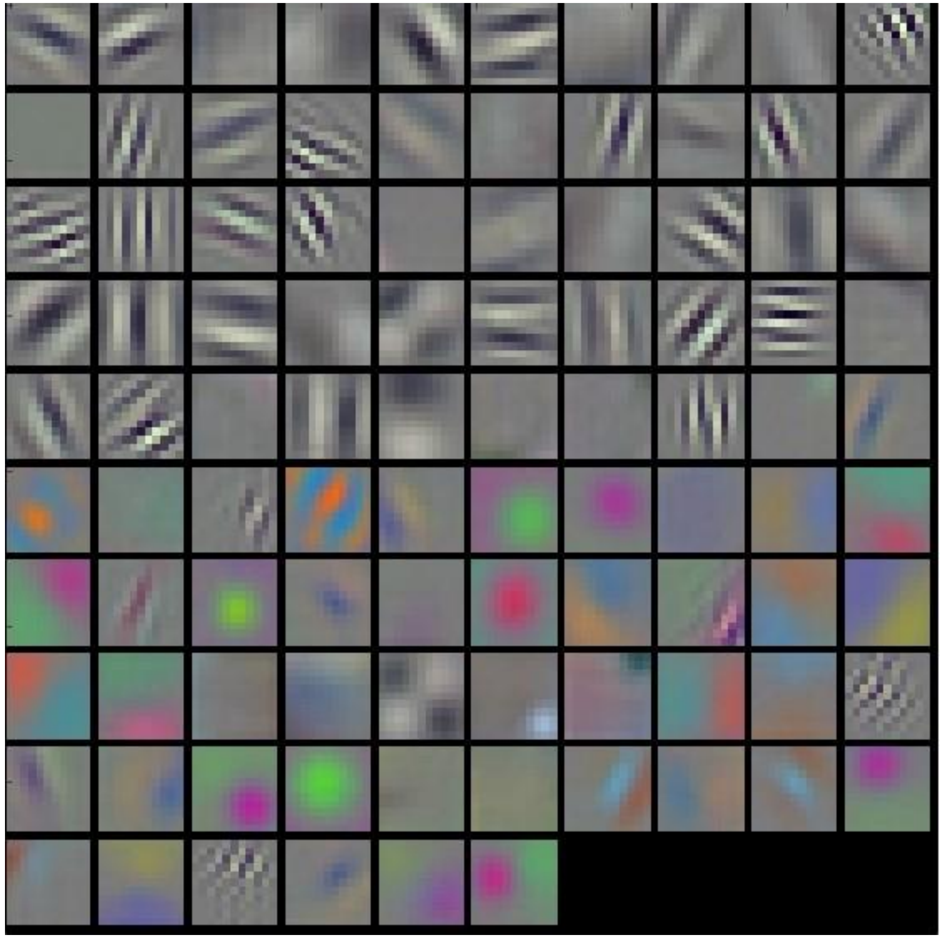




Features

Typical-looking filters on the trained first layer

预训练模型 首层特征图

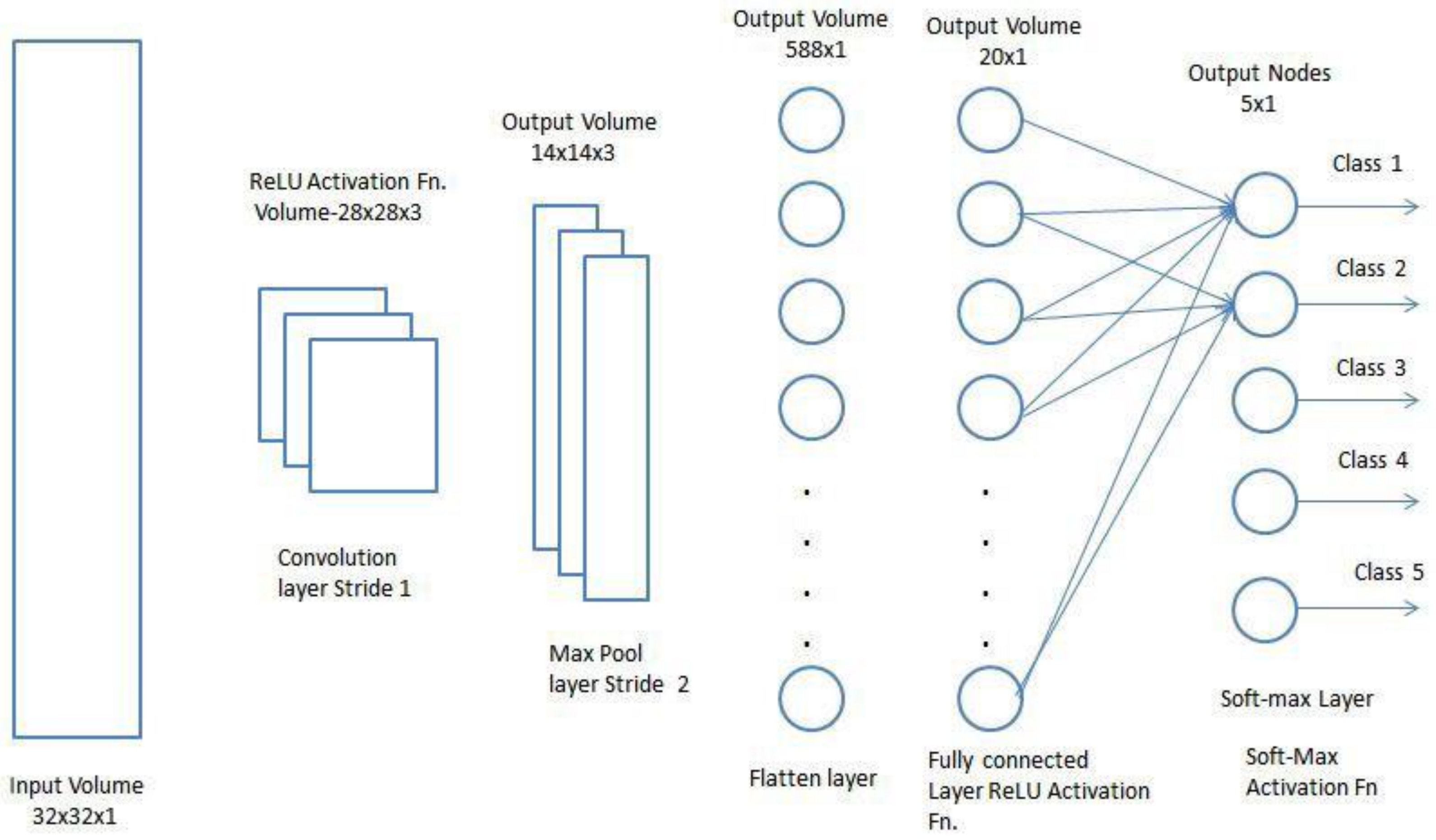


11 x 11
(AlexNet)

Features

Maximally activating images for some POOL5 (5th pool layer) neurons of an AlexNet.

Fully Connected Layer (FC)



More Applications: Deep Style

https://deepdreamgenerator.com/

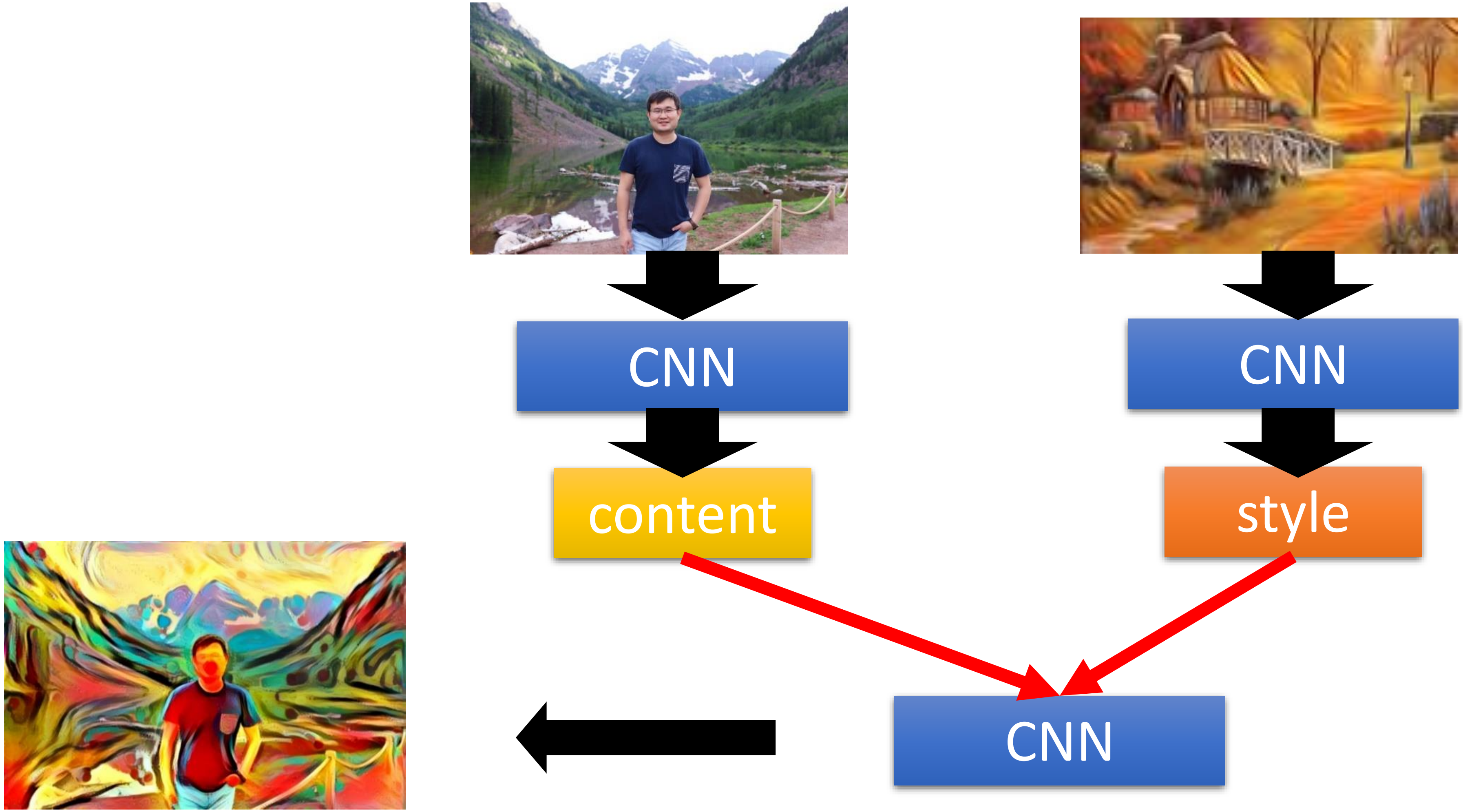
More Applications: Deep Dream

+

=

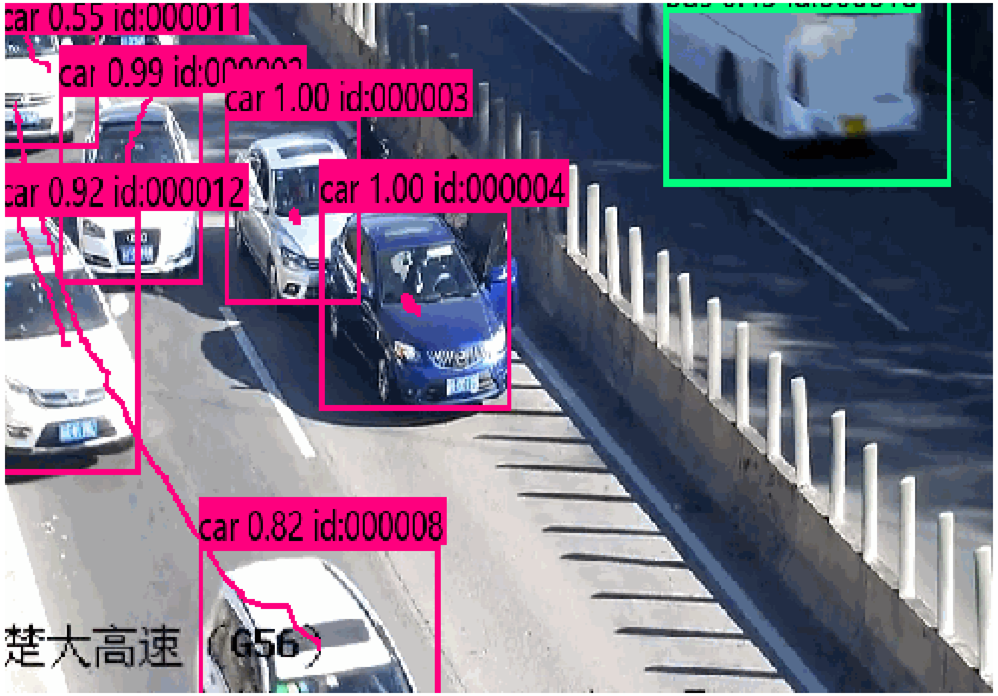
https://deepdreamgenerator.com/

Given a photo, make its style like famous paintings



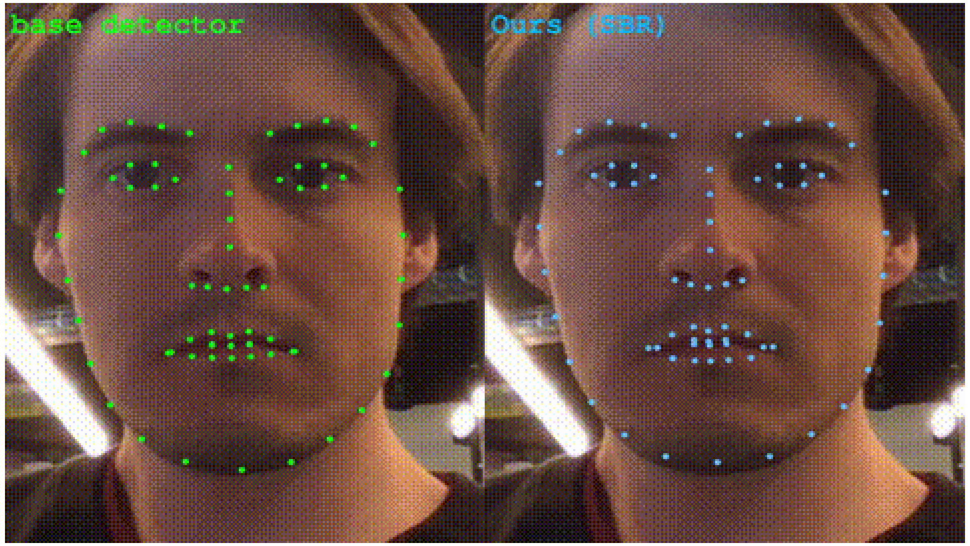
Deep Learning for Computer Vision

Object Recognition



Object Detection

Deep Learning for Computer Vision

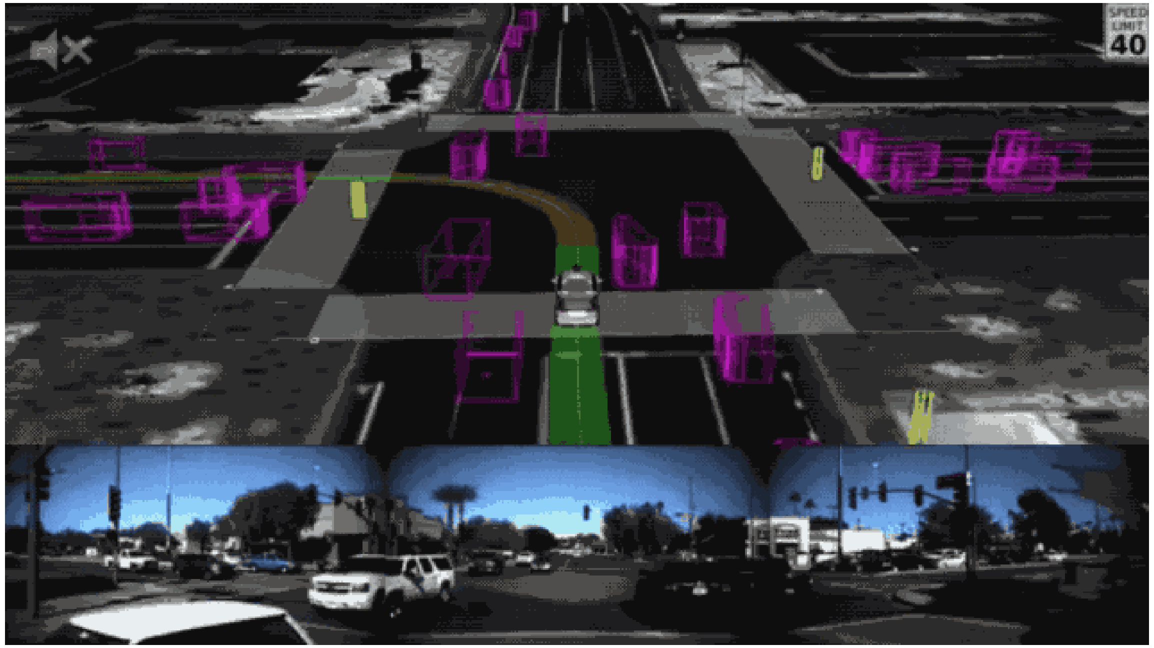


Face Detection

Skelton Detection

Deep Learning for Computer Vision

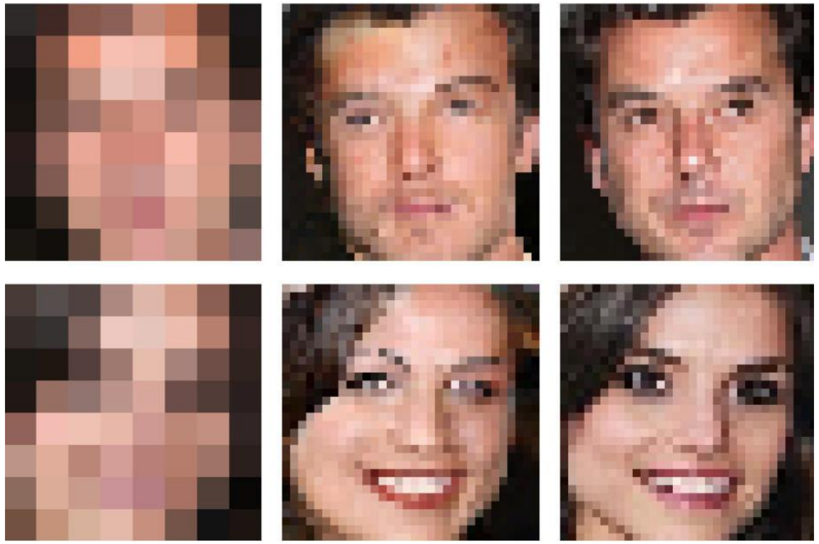
Video Generation



Unmanned Vehicle

Deep Learning for Computer Vision

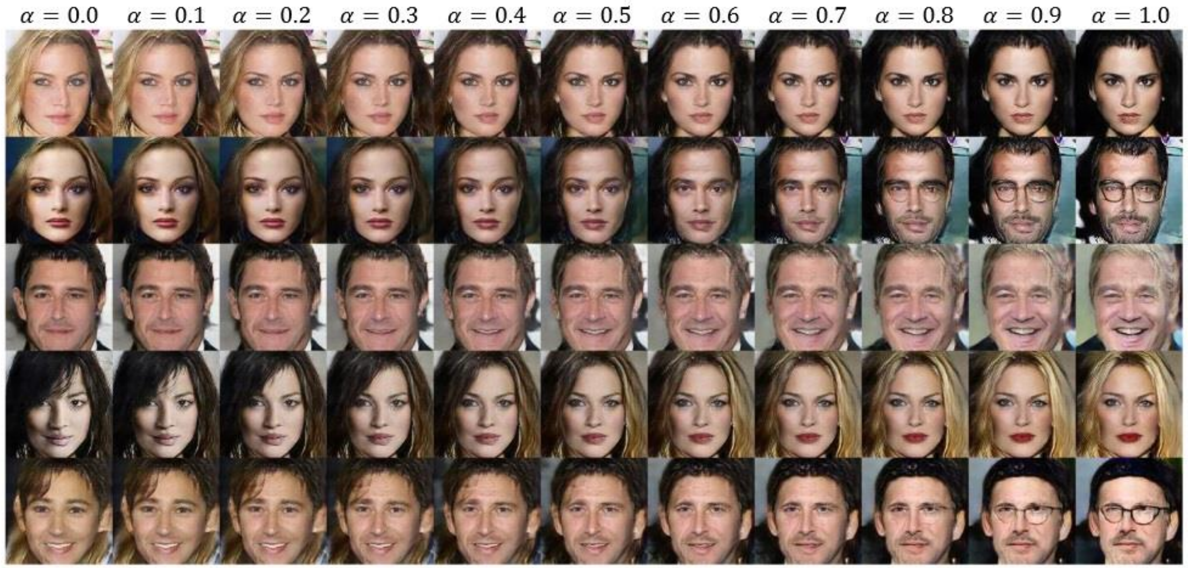
Deep Learning for Computer Vision



Pixel Recursive Super Resolution

Image Translation

Deep Learning for Computer Vision



Generative Adversarial Networks

Cycle GAN

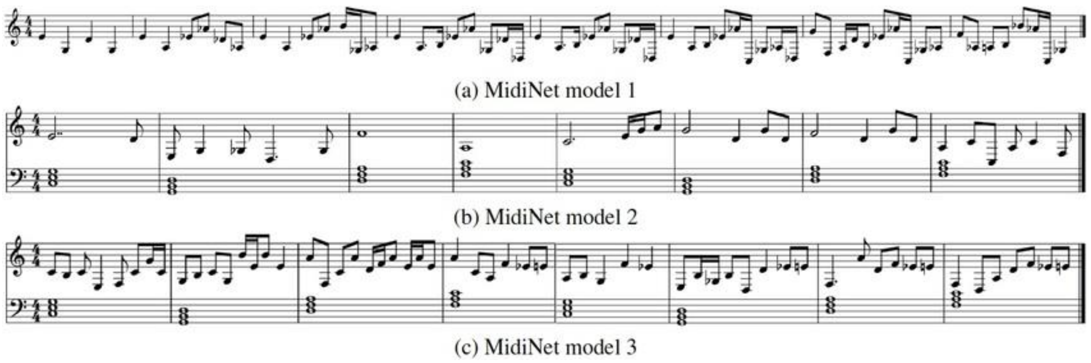
02 Generative Adversarial Networks

生成对抗网络

Generative Adversarial Networks

Game development and animation production are expensive and hire many production artists for relatively routine tasks.

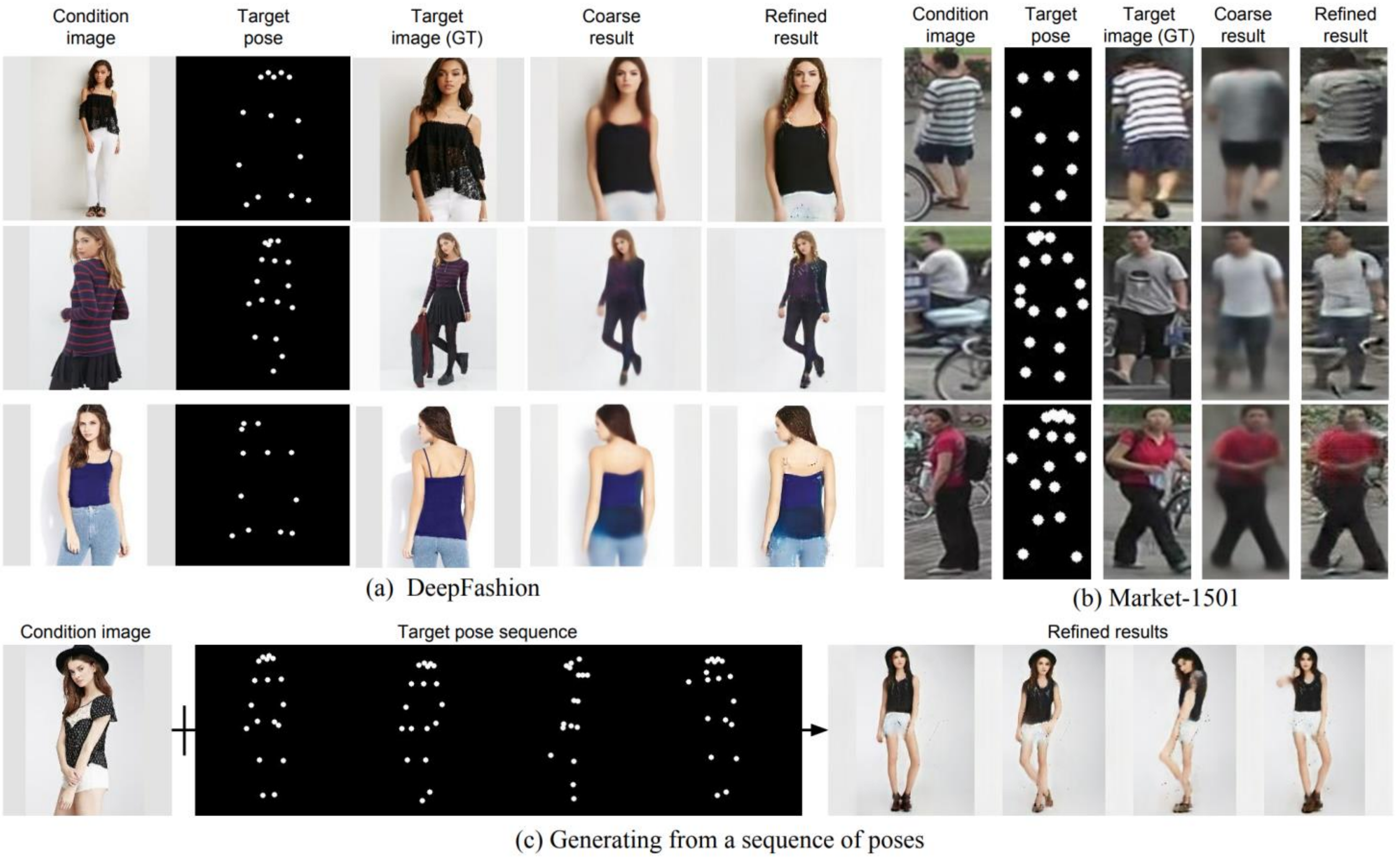
GAN can auto-generate and colorize Anime characters. GAN可以自动生成彩色卡通人物。



Pose Guided Person Image Generation

With an additional input of the pose, we can transform an image into different poses.

输入一个姿势，我们可以将图像中的人物变换出不同的姿势。

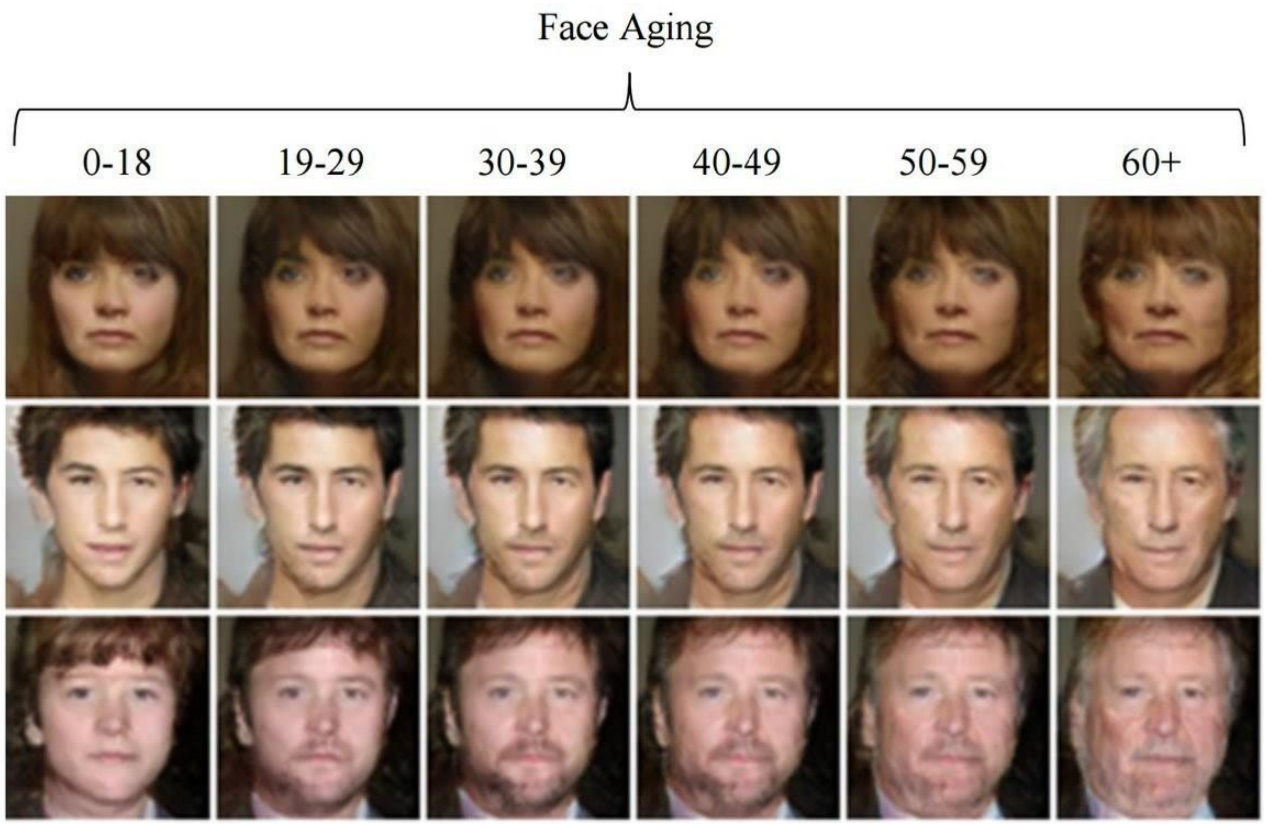


CycleGAN

Cross-domain transfer GANs will be likely the first batch of commercial applications. These GANs transform images from one domain (say real scenery) to another domain (Monet paintings or Van Gogh).

跨场景迁移图像风格

Cycle GAN



PixelDTGAN

Suggesting merchandise based on celebrity pictures has been popular for fashion blogger and e-commerce. 给电商生成不同风格的衣服图片。

PixelDTGAN creates clothing images and styles from an image. 从一张图像上生成不同风格的衣服图像

A source image.

Possible target images.

StackGAN

Text to image is one of the earlier application of domain-transfer GAN. We input a sentence and generate multiple images fitting the description.

文字（输入）生成图像（输出）。

DiscoGAN

DiscoGAN provides matching style: many potential applications. 风格配对

DiscoGAN learns cross domain relationship without labels or pairing. For example, it successfully transfers style (or patterns) from one domain (handbag) to another (shoe). 无需人工标注

(b) Handbag images (input) & **Generated** shoe images (output)

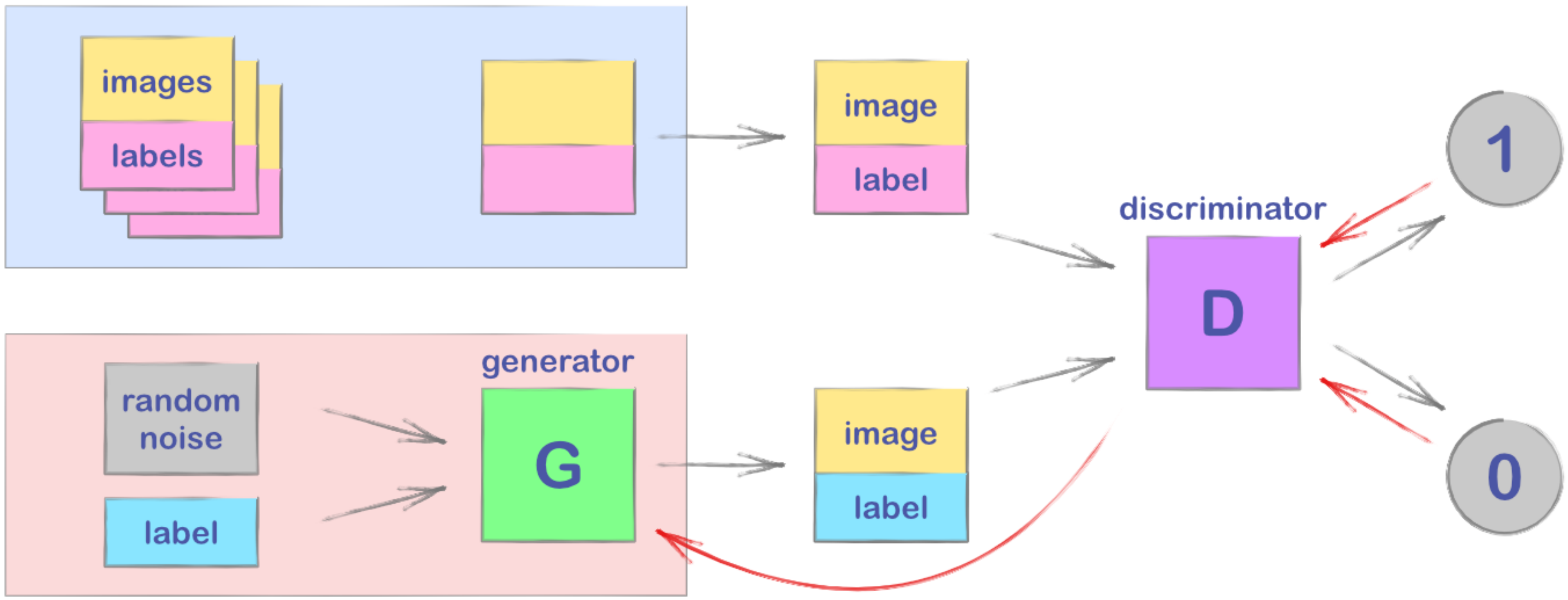
Generative Adversarial Networks

GANs have various applications on commercial market. 对抗生成网络在商业市场中无处不在。

Welcome to GANs’ world. 欢迎来到对抗生成的世界。

Figure 1: Class-conditional samples generated by our model.

Basic Ideas of GAN



Generator

We will control what to generate latter. → Conditional Generation 条件生成

Image Generation图像生成

$$\begin{bmatrix} 0.3 \\ -0.1 \\ \vdots \\ -0.7 \end{bmatrix} \begin{bmatrix} 0.1 \\ -0.1 \\ \vdots \\ 0.7 \end{bmatrix} \begin{bmatrix} -0.3 \\ 0.1 \\ \vdots \\ 0.9 \end{bmatrix} \rightarrow \boxed{\text{NN Generator}} \rightarrow$$

In a specific range

Sentence Generation句子生成

$$\begin{bmatrix} 0.3 \\ 0.5 \\ \vdots \\ -0.6 \end{bmatrix} \begin{bmatrix} 0.2 \\ -0.3 \\ \vdots \\ 0.2 \end{bmatrix} \begin{bmatrix} -0.3 \\ 0.2 \\ \vdots \\ 0.3 \end{bmatrix} \rightarrow \boxed{\text{NN Generator}} \rightarrow$$

What's up?

How's going?

Any question?

Generator

$$\begin{bmatrix} 0.1 \\ -3 \\ \vdots \\ 2.4 \\ 0.9 \end{bmatrix} \Rightarrow \text{Generator 生成器} \Rightarrow$$

Each dimension of input vector represents some characteristics.

$$\begin{bmatrix} \boxed{3} \\ -3 \\ \vdots \\ 2.4 \\ 0.9 \end{bmatrix} \Rightarrow \text{Generator 生成器} \Rightarrow$$

Longer hair

$$\begin{bmatrix} 0.1 \\ 2.1 \\ \vdots \\ \boxed{5.4} \\ 0.9 \end{bmatrix} \Rightarrow \text{Generator 生成器} \Rightarrow$$

blue hair

$$\begin{bmatrix} 0.1 \\ -3 \\ \vdots \\ 2.4 \\ \boxed{3.5} \end{bmatrix} \Rightarrow \text{Generator 生成器} \Rightarrow$$

Open mouth

Discriminator

Image
图像

→ Discriminator
辨别器

→ scalar

Larger value means real, 大为真
smaller value means fake. 小为假

→ D → 1.0

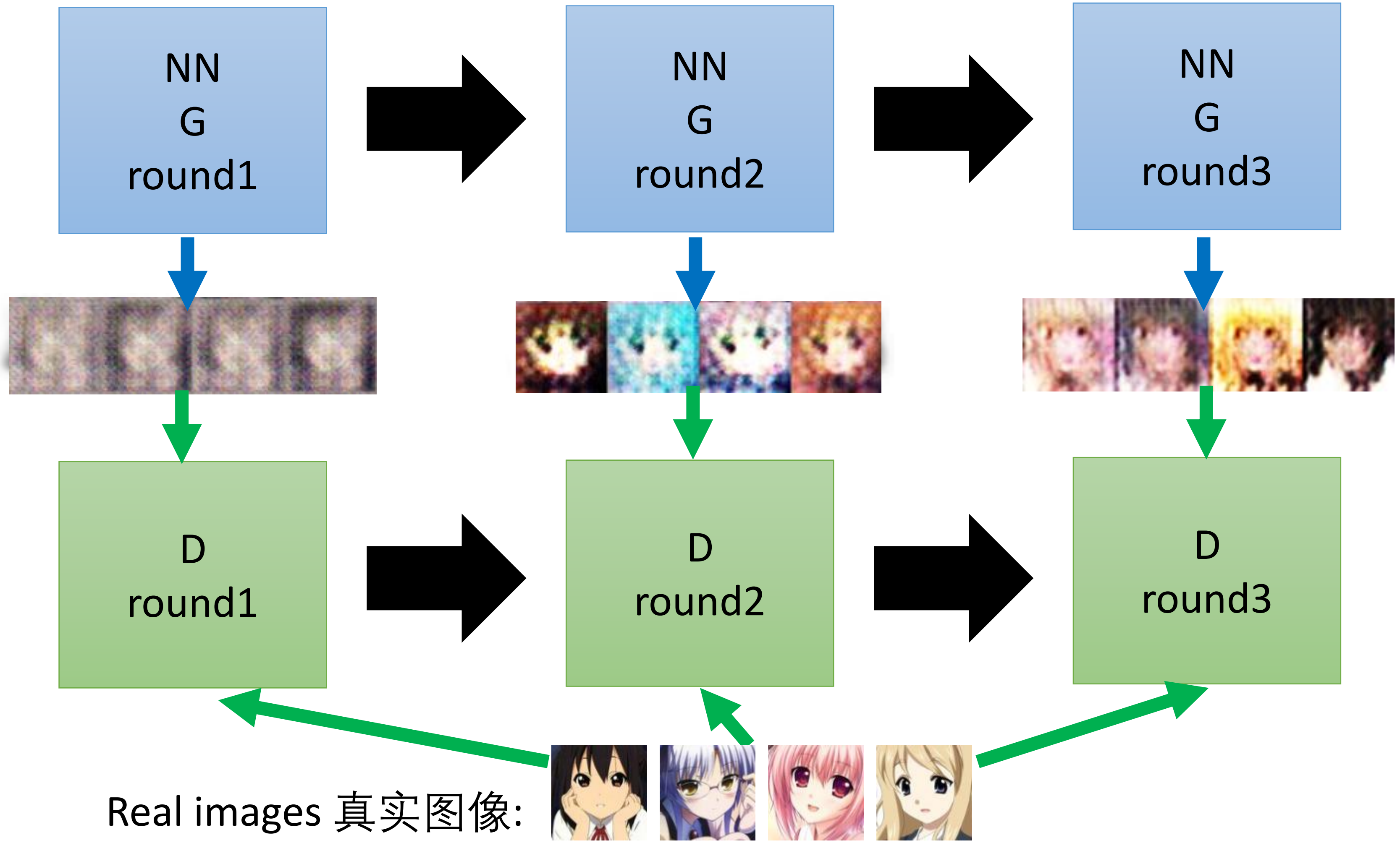
→ D → 1.0

→ D → 0.1

→ D → 0.1

Generator vs Discriminator

DiscoGAN

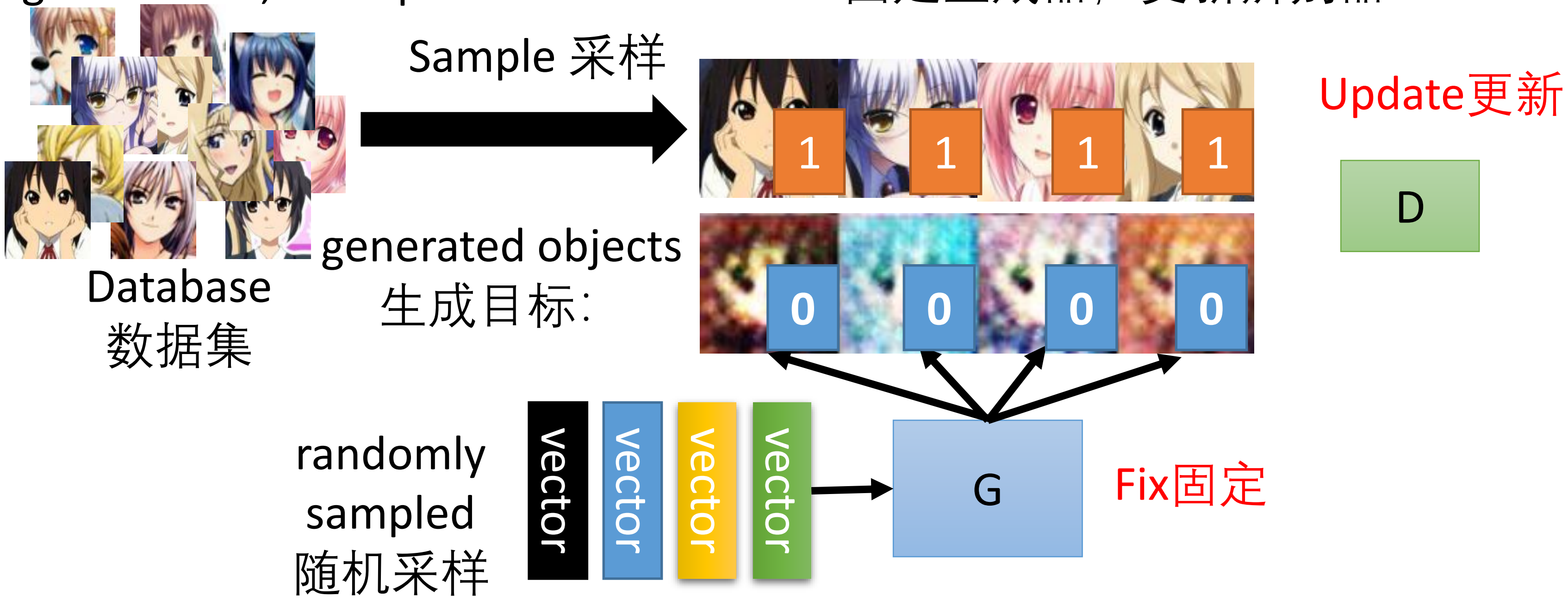


NN
G
round1
NN
G
round2
NN
G
round3
D
round1
D
round2
D
round3
Real images 真实图像:

Algorithm

- Initialize generator and discriminator 初始化: G D
- In each training iteration 对于训练中的每次迭代:

Step 1: Fix generator G, and update discriminator D 固定生成器， 更新辨别器



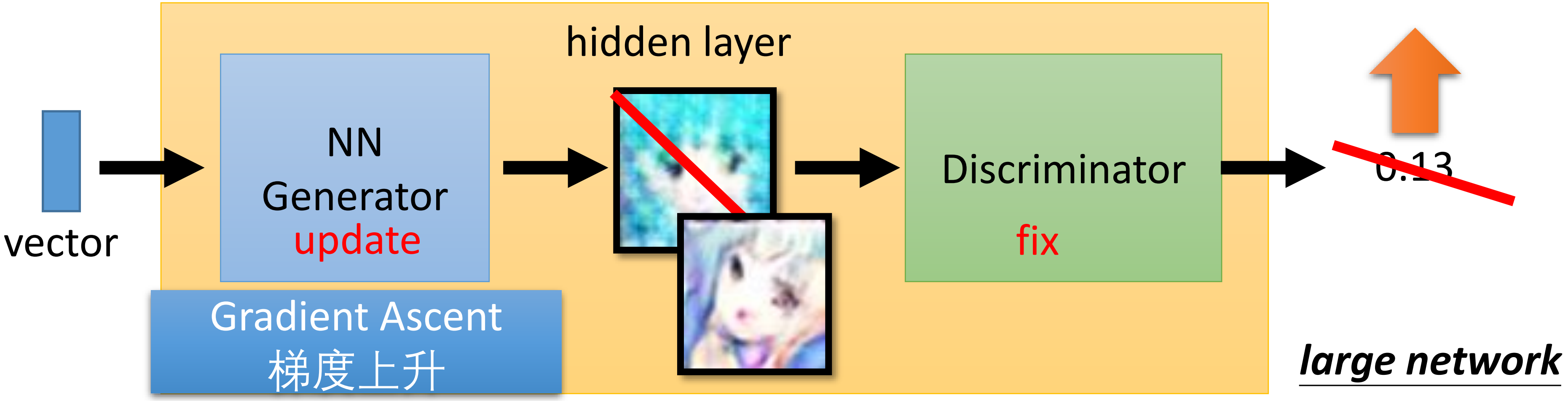
Discriminator learns to assign high scores to real objects and low scores to generated objects. 辨别器学习到如何对真实目标打高分， 对生成目标打低分。

Algorithm

- Initialize generator and discriminator　G　D
- In each training iteration:

Step 2: Fix discriminator D, and update generator G 固定辨别器，更新生成器。

Generator learns to “fool” the discriminator 生成器学习如何去耍弄辨别器



Algorithm

Initialize θ_d for D and θ_g for G

- In each training iteration:

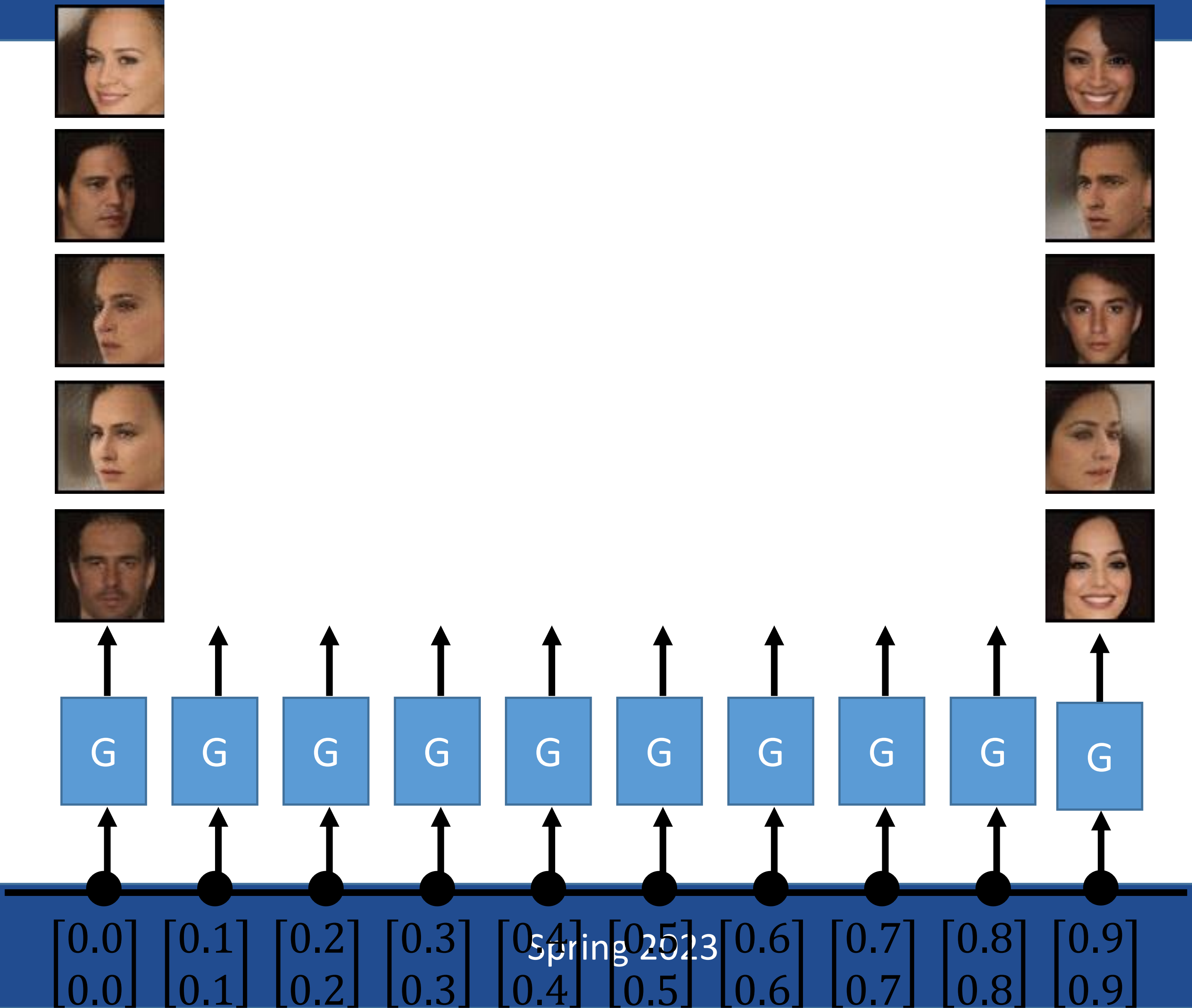
Learning D

- Sample m examples $\{x^1, x^2, \dots, x^m\}$ from database
- Sample m noise samples $\{z^1, z^2, \dots, z^m\}$ from a distribution
- Obtaining generated data $\{\tilde{x}^1, \tilde{x}^2, \dots, \tilde{x}^m\}$, $\tilde{x}^i = G(z^i)$
- Update discriminator parameters θ_d to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} log D(x^i) + \frac{1}{m}\sum_{i=1}^{m} log\left(1 - D(\tilde{x}^i)\right)$
  - $\theta_d \leftarrow \theta_d + \eta \nabla \tilde{V}(\theta_d)$
- Sample m noise samples$\{z^1, z^2, \dots, z^m\}$ from a distribution
- Update generator parameters θ_g to maximize

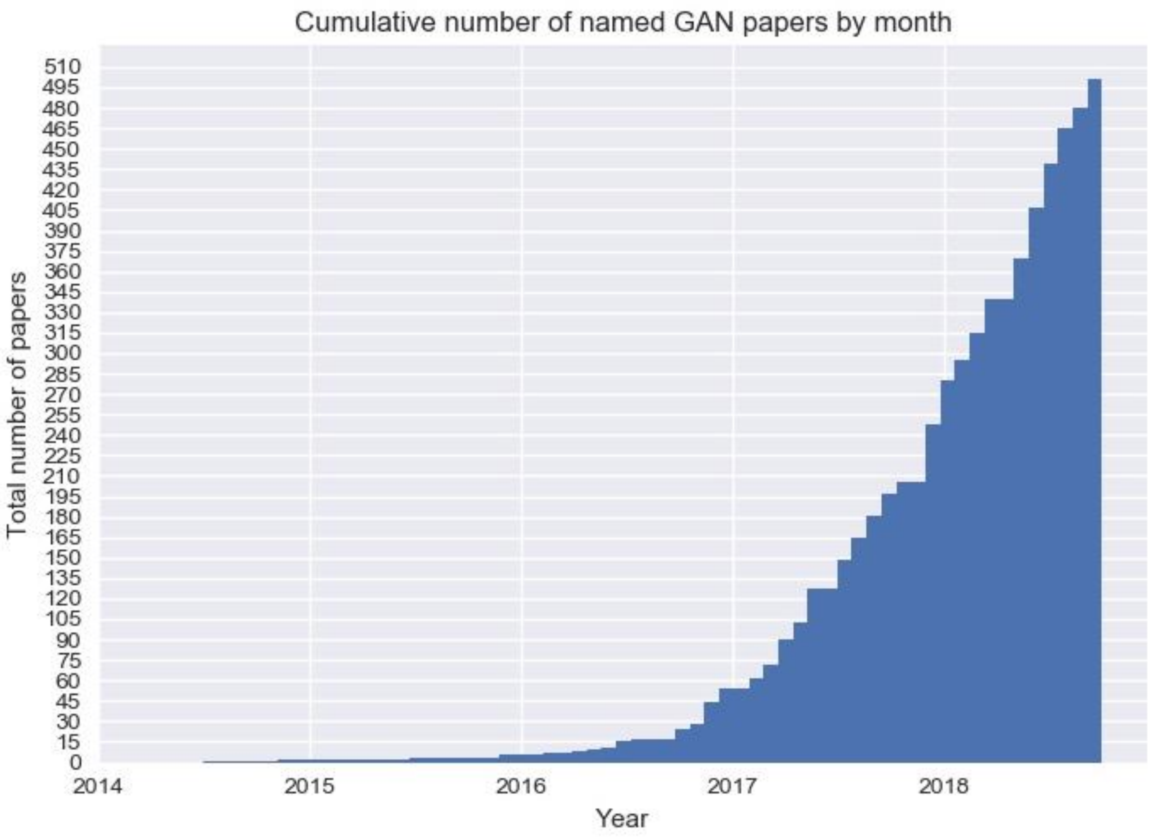
Learning G

  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} log\left(D\left(G(z^i)\right)\right)$
  - $\theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)$

GAN


G
G
G
G
G
G
G
G
G
G
[0.0 0.0]
[0.1 0.1]
[0.2 0.2]
[0.3 0.3]
[0.4 0.4]
[0.5 0.5]
[0.6 0.6]
[0.7 0.7]
[0.8 0.8]
[0.9 0.9]

Generative Adversarial Networks



Q&A