

基礎教程 | 卷積神經網絡 (CNN) 是什麼？

極市平台 昨天

以下文章來源於SimpleAI，作者Beyond

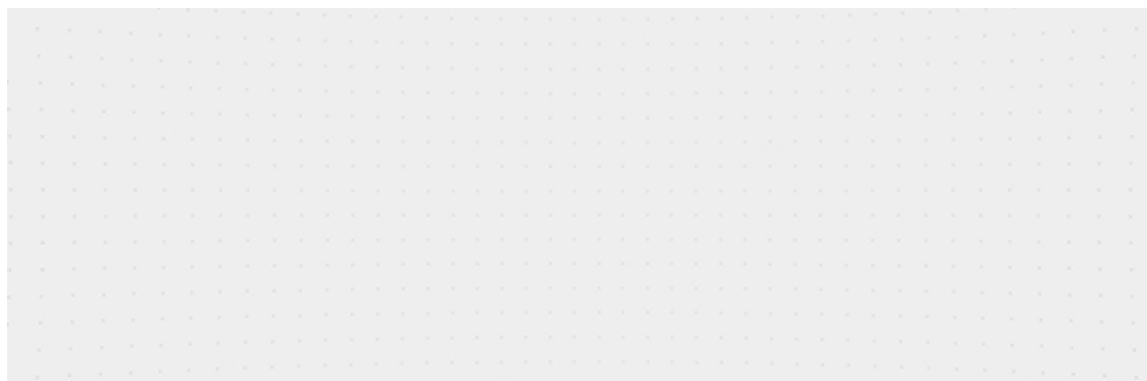


SimpleAI

人工智能、機器學習、深度學習還是遙不可及？來這裡看看吧~



↑點擊[藍字](#) 關注極市平台



作者 | 郭必揚

來源 | SimpleAI

編輯 | 極市平台

極市導讀

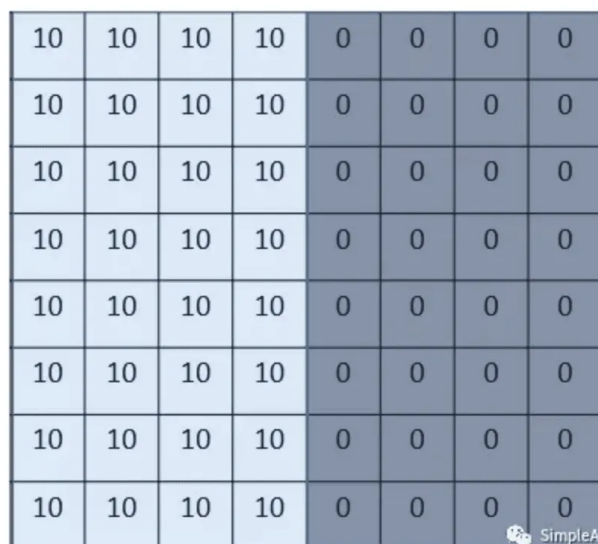
本文詳細講解了CNN各個基礎知識。 >>加入極市CV技術交流群，走在計算機視覺的最前沿

從今天起，正式開始講解卷積神經網絡。這是一種曾經讓我無論如何也無法弄明白的東西，主要是名字就太“高級”了，網上的各種各樣的文章來介紹“什麼是卷積”尤為讓人受不了。聽了吳恩達的網課之後，豁然開朗，終於搞明白了這個東西是什麼和為什麼。我這裡大概會用6~7篇文章來講解CNN並實現一些有趣的應用。看完之後大家應該可以自己動手做一些自己喜歡的事兒了。

一、引子——邊界檢測

我們來看一個最簡單的例子：“邊界檢測 (edge detection)”，假設我們有這樣的一張圖片，大小 8×8 ：

10	10	10	10	0	0	0	0
10	10	10	10	0	0	0	0
10	10	10	10	0	0	0	0
10	10	10	10	0	0	0	0
10	10	10	10	0	0	0	0
10	10	10	10	0	0	0	0
10	10	10	10	0	0	0	0
10	10	10	10	0	0	0	0



圖片中的數字代表該位置的像素值，我們知道，像素值越大，顏色越亮，所以為了示意，我們把右邊小像素的地方畫成深色。圖的中間兩個顏色的分界線就是我們要檢測的邊界。

怎麼檢測這個邊界呢？我們可以設計這樣的一個**濾波器 (filter, 也稱為kernel)**，大小 3×3 ：

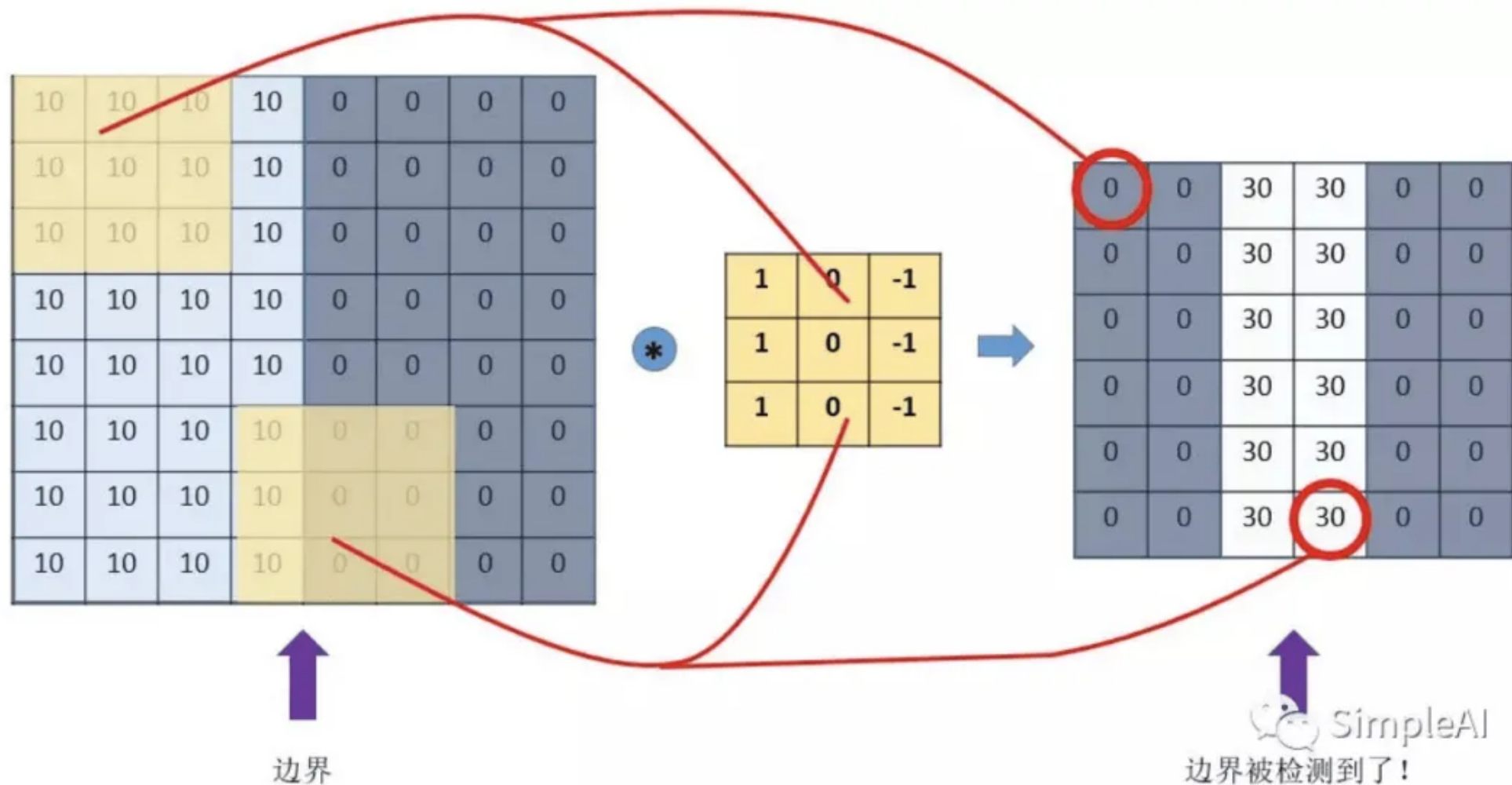
1	0	-1
1	0	-1
1	0	-1

然后，我们用这个filter，往我们的图片上“盖”，覆盖一块跟filter一样大的区域之后，对应元素相乘，然后求和。计算一个区域之后，就向其他区域挪动，接着计算，直到把原图片的每一个角落都覆盖到了为止。这个过程就是“卷积”。

（我们不用管卷积在数学上到底是指什么运算，我们只用知道在CNN中是怎么计算的。）

这里的“挪动”，就涉及到一个步长了，假如我们的步长是1，那么覆盖了一个地方之后，就挪一格，容易知道，总共可以覆盖 6×6 个不同的区域。

那么，我们将这 6×6 个区域的卷积结果，拼成一个矩阵：



诶?! 发现了什么?

这个图片，中间颜色浅，两边颜色深，这说明咱们的原图片中间的边界，在这里被反映出来了!

从上面这个例子中，我们发现，**我们可以通过设计特定的filter，让它去跟图片做卷积，就可以识别出图片中的某些特征，比如边界。**

上面的例子是检测垂直边界，我们也可以设计出检测水平边界的，只用把刚刚的filter旋转90°即可。对于其他的特征，理论上只要我们经过精细的设计，总是可以设计出合适的filter的。

我们的CNN (convolutional neural network) ，主要就是通过一个个的filter，不断地提取特征，从局部的特征到总体的特征，从而进行图像识别等功能。

那么问题来了，我们怎么可能去设计这么多各种各样的filter呀？首先，我们都不一定清楚对于一大推图片，我们需要识别哪些特征，其次，就算知道了有哪些特征，想真的去设计出对应的filter，恐怕也并非易事，要知道，特征的数量可能是成千上万的。

其实学过神经网络之后，我们就知道，**这些filter，根本就不用我们去设计**，每个filter中的各个数字，不就是参数吗，我们可以通过大量的数据，来**让机器自己去“学习”这些参数**嘛。这，就是CNN的原理。

二、CNN的基本概念

1.padding 填白

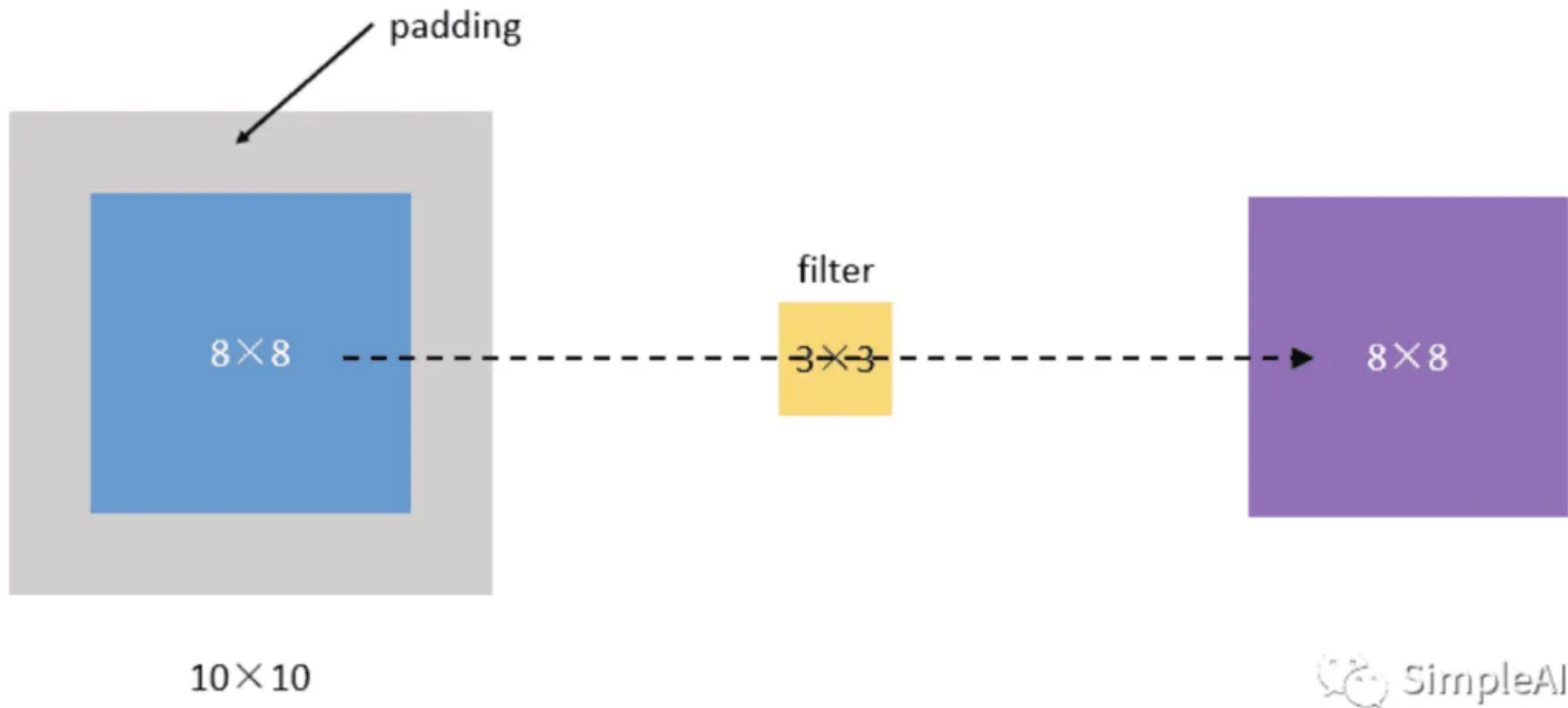
从上面的引子中，我们可以知道，原图像在经过filter卷积之后，变小了，从(8,8)变成了(6,6)。假设我们再卷一次，那大小就变成了(4,4)了。

这样有啥问题呢？

主要有两个问题：

- 每次卷积，图像都缩小，这样卷不了几次就没了；
- 相比于图片中间的点，图片边缘的点在卷积中被计算的次数很少。这样的话，边缘的信息就易于丢失。

为了解决这个问题，我们可以采用padding的方法。我们每次卷积前，先给图片周围都补一圈空白，让卷积之后图片跟原来一样大，同时，原来的边缘也被计算了更多次。



比如，我们把(8,8)的图片给补成(10,10)，那么经过(3,3)的filter之后，就是(8,8)，没有变。

我们把上面这种“让卷积之后的大小不变”的padding方式，称为 **“Same”** 方式，把不经过任何填白的，称为 **“Valid”**方式。这个是我们在使用一些框架的时候，需要设置的超参数。

2.stride 步长

前面我们所介绍的卷积，都是默认步长是1，但实际上，我们可以设置步长为其他的值。

比如，对于(8,8)的输入，我们用(3,3)的filter，

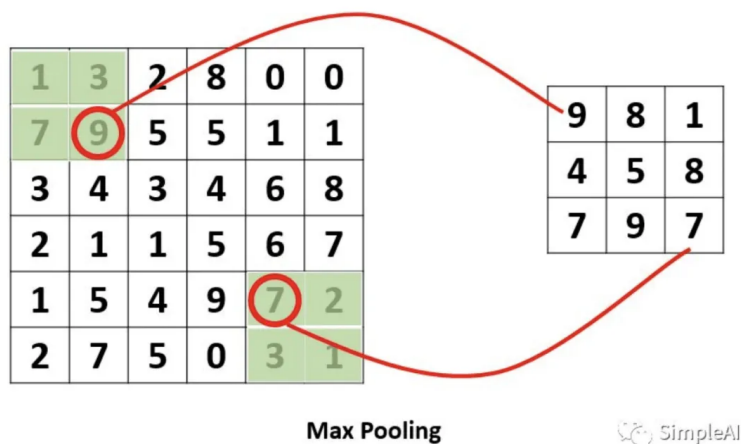
如果stride=1，则输出为(6,6)；

如果stride=2，则输出为(3,3)；(这里例子举得不太好，除不断就向下取整)

3.pooling 池化

这个pooling，是为了提取一定区域的主要特征，并减少参数数量，防止模型过拟合。

比如下面的MaxPooling，采用了一个 2×2 的窗口，并取stride=2：



除了MaxPooling,还有AveragePooling，顾名思义就是取那个区域的平均值。

4.对多通道 (channels) 图片的卷积 (重要！)

这个需要单独提一下。彩色图像，一般都是RGB三个通道（channel）的，因此输入数据的维度一般有三个：**（长，宽，通道）**。

比如一个 28×28 的RGB图片，维度就是 $(28, 28, 3)$ 。

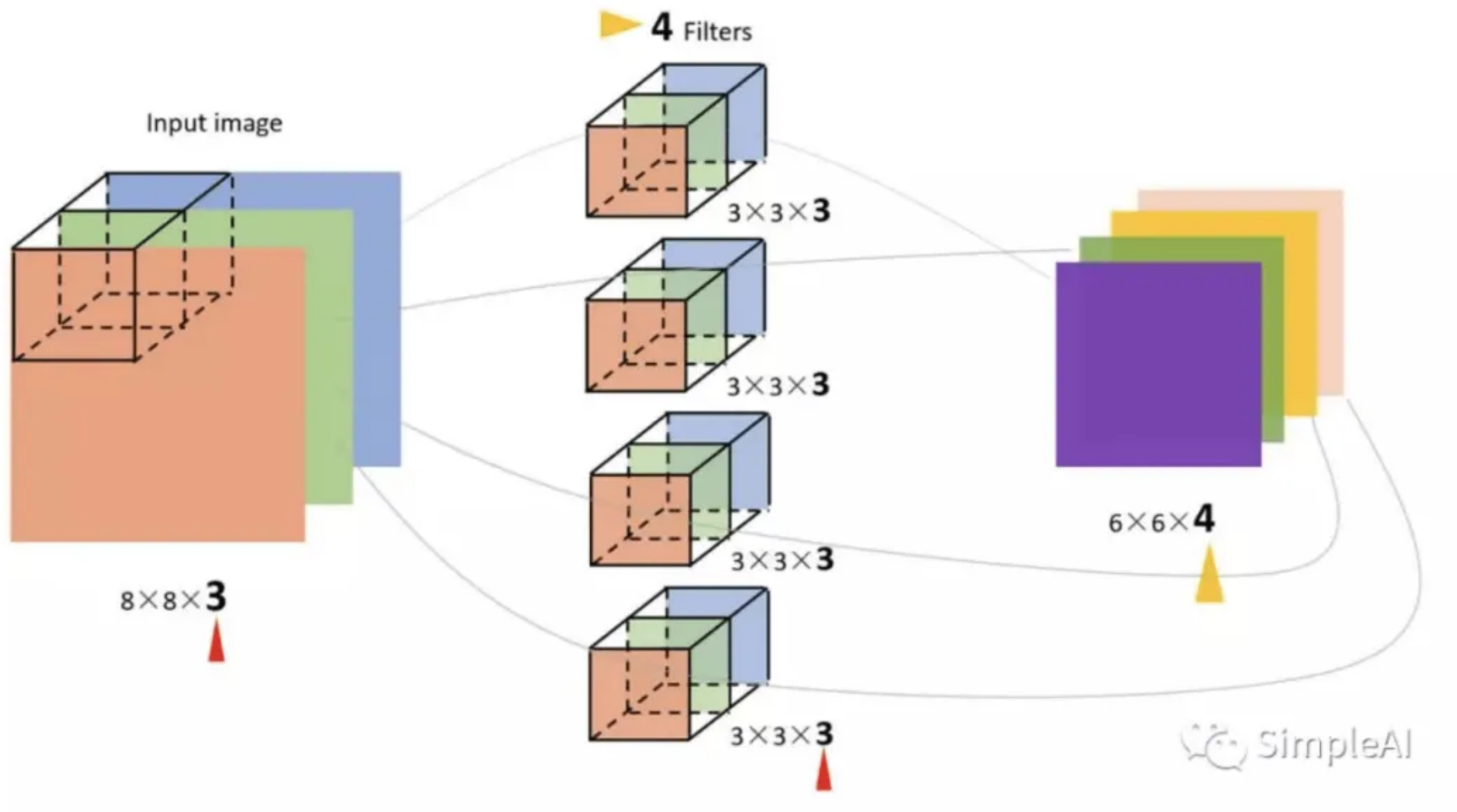
前面的引子中，输入图片是2维的 $(8, 8)$ ，filter是 $(3, 3)$ ，输出也是2维的 $(6, 6)$ 。

如果输入图片是三维的呢（即增多了一个channels），比如是 $(8, 8, 3)$ ，这个时候，我们的filter的维度就要变成 $(3, 3, 3)$ 了，它的**最后一维要跟输入的channel维度一致**。

这个时候的卷积，**是三个channel的所有元素对应相乘后求和**，也就是之前是9个乘积的和，现在是27个乘积的和。因此，输出的维度并不会变化。还是 $(6, 6)$ 。

但是，一般情况下，我们会**使用多了filters同时卷积**，比如，如果我们同时使用4个filter的话，那么**输出的维度则会变为 $(6, 6, 4)$** 。

我特地画了下面这个图，来展示上面的过程：

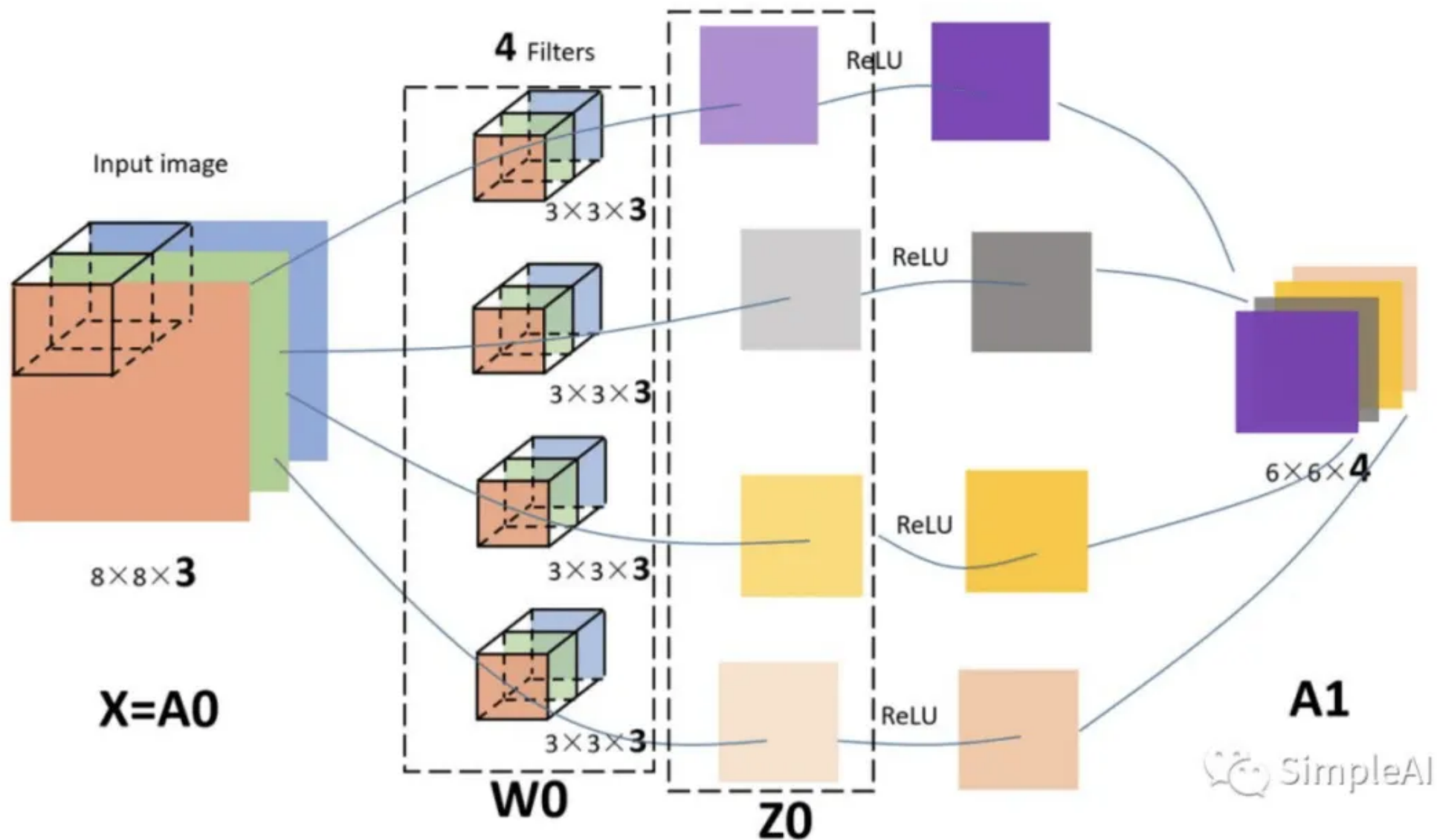


图中的输入图像是 $(8, 8, 3)$ ，filter有4个，大小均为 $(3, 3, 3)$ ，得到的输出为 $(6, 6, 4)$ 。我觉得这个图已经画的很清晰了，而且给出了3和4这两个关键数字是怎么来的，所以我不啰嗦了（这个图画了我起码40分钟）。

其实，如果套用我们前面学过的神经网络的符号来看待CNN的话，

- 我们的输入图片就是X, $\text{shape}=(8,8,3)$;
- 4个filters其实就是第一层神经网络的参数 $W1$, $\text{shape}=(3,3,3,4)$,这个4是指有4个filters;
- 我们的输出, 就是 $Z1$, $\text{shape}=(6,6,4)$;
- 后面其实还应该有一个激活函数, 比如relu, 经过激活后, $Z1$ 变为 $A1$, $\text{shape}=(6,6,4)$;

所以, 在前面的图中, 我加一个激活函数, 给对应的部分标上符号, 就是这样的:



【个人觉得，这么好的图不收藏，真的是可惜了】

三、CNN的结构组成

上面我们已经知道了卷积 (convolution)、池化 (pooling) 以及填白 (padding) 是怎么进行的，接下来我们就来看看CNN的整体结构，它包含了3种层 (layer)：

1. Convolutional layer (卷积层—CONV)

由滤波器filters和激活函数构成。

一般要设置的超参数包括filters的数量、大小、步长，以及padding是“valid”还是“same”。当然，还包括选择什么激活函数。

2. Pooling layer (池化层—POOL)

这里里面没有参数需要我们学习，因为这里里面的参数都是我们设置好了，要么是Maxpooling，要么是Averagepooling。

需要指定的超参数，包括是Max还是average，窗口大小以及步长。

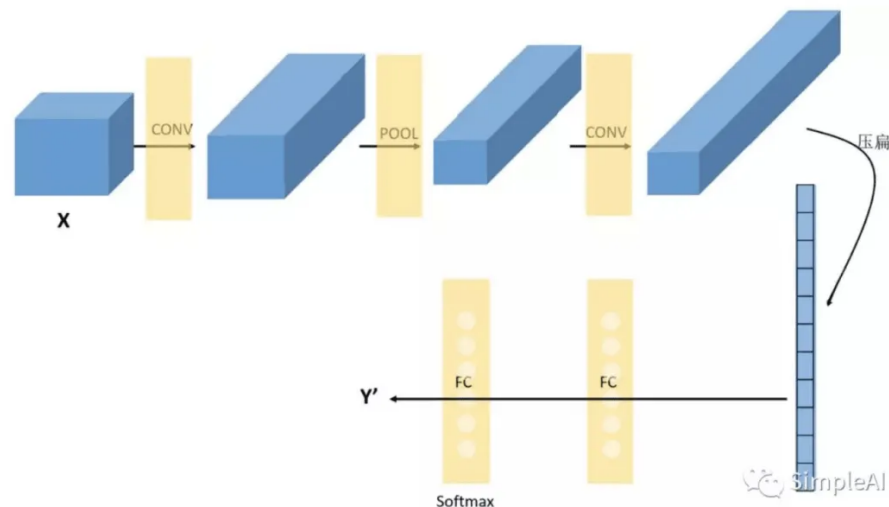
通常，我们使用的比较多的是Maxpooling,而且一般取大小为(2,2)步长为2的filter，这样，经过pooling之后，输入的长宽都会缩小2倍，channels不变。

3. Fully Connected layer (全连接层—FC)

这个前面没有讲，是因为这个就是我们最熟悉的家伙，**就是我们之前学的神经网络中的那种最普通的层，就是一排神经元**。因为这一层是每一个单元都和前一层的每一个单元相连接，所以称之为“全连接”。

这里要指定的超参数，无非就是神经元的数量，以及激活函数。

接下来，我们随便看一个CNN的模样，来获取对CNN的一些感性认识：



上面这个CNN是我随便拍脑门想的一个。它的结构可以用：

$X \rightarrow \text{CONV}(\text{relu}) \rightarrow \text{MAXPOOL} \rightarrow \text{CONV}(\text{relu}) \rightarrow \text{FC}(\text{relu}) \rightarrow \text{FC}(\text{softmax}) \rightarrow Y$ 来表示。

这里需要说明的是，在经过数次卷积和池化之后，我们 **最后会先将多维的数据进行“扁平化”**，也就是把 **(height,width,channel)** 的数据压缩成长度为 **height × width × channel** 的一维数组，然后再与 **FC层**连接，**这之后就跟普通的神经网络无异了**。

可以从图中看到，随着网络的深入，我们的图像（严格来说中间的那些不能叫图像了，但是为了方便，还是这样说吧）越来越小，但是channels却越来越大了。在图中的表示就是长方体面对我们的面积越来越小，但是长度却越来越长了。

四、卷积神经网络 VS. 传统神经网络

其实现在回过头来看，CNN跟我们之前学习的神经网络，也没有很大的差别。

传统的神经网络，其实就是多个FC层叠加起来。

CNN，无非就是把FC改成了CONV和POOL，就是把传统的由一个个神经元组成的layer，变成了由filters组成的layer。

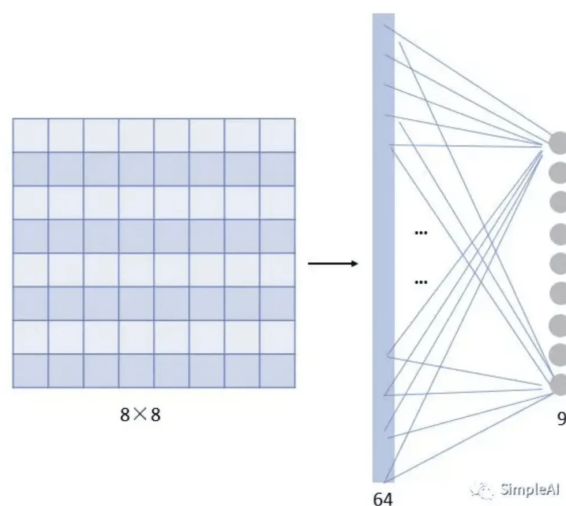
那么，为什么要这样变？有什么好处？

具体说来有两点：

1. 参数共享机制 (parameters sharing)

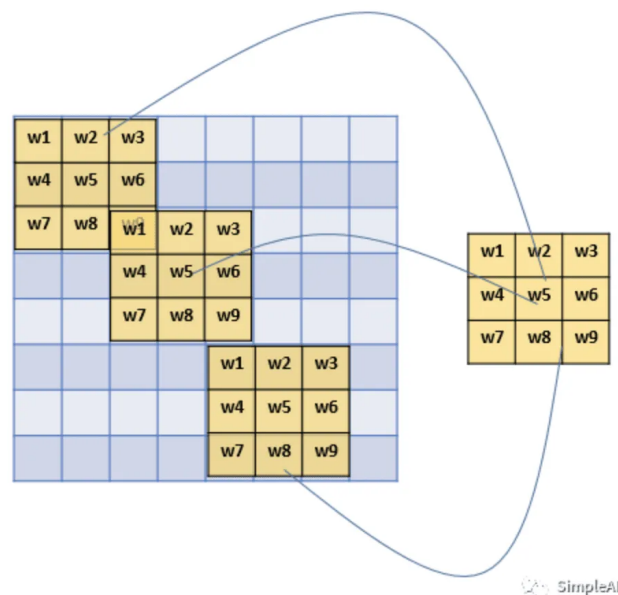
我们对比一下传统神经网络的层和由filters构成的CONV层：

假设我们的图像是 8×8 大小，也就是64个像素，假设我们用一个有9个单元的全连接层：



那这一层我们需要多少个参数呢？需要 $64 \times 9 = 576$ 个参数（先不考虑偏置项 b ）。因为每一个链接都需要一个权重 w 。

那我们看看 同样有9个单元的filter是怎么样的：



其实不用看就知道，有几个单元就几个参数，所以总共就9个参数！

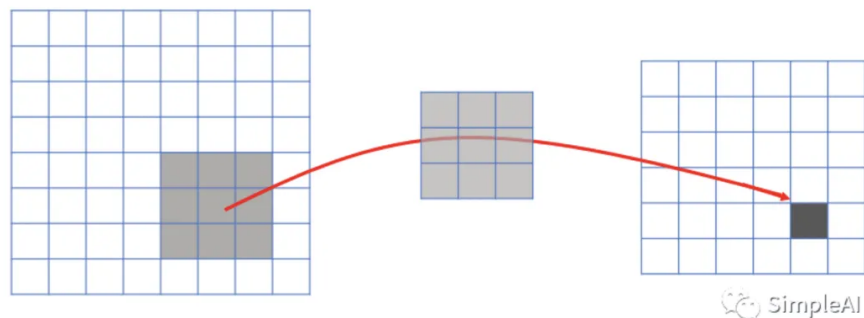
因为，对于不同的区域，我们都共享同一个filter，因此就共享这同一组参数。这也是有道理的，通过前面的讲解我们知道，filter是用来检测特征的，那一个特征一般情况下很可能在不止一个地方出现，比如“竖直边界”，就可能在一幅图中多出出现，那么 **我们共享同一个filter不仅是合理的，而且是应该这么做的。**

由此可见，参数共享机制，**让我们的网络的参数数量大大地减少**。这样，我们可以用较少的参数，训练出更加好的模型，典型的事半功倍，而且可以有效地 **避免过拟合**。

同样，由于filter的参数共享，即使图片进行了一定的平移操作，我们照样可以识别出特征，这叫做“**平移不变性**”。因此，模型就更加稳健了。

2.连接的稀疏性 (sparsity of connections)

由卷积的操作可知，输出图像中的任何一个单元，**只跟输入图像的一部分有关系**：



而传统神经网络中，由于都是全连接，所以输出的任何一个单元，都要受输入的所有的单元的影响。这样无形中会对图像的识别效果大打折扣。比较，每一个区域都有自己的专属特征，我们不希望它受到其他区域的影响。

正是由于上面这两大优势，使得CNN超越了传统的NN，开启了神经网络的新时代。

好了，今天的文章到此结束！今天是我画图最累的一次，不过也画的最有成就感的一次！没想到用PowerPoint也可以画出这么好看的图hhh，让我自己得意一下~~

如果觉得有用，就请分享到朋友圈吧！



极市平台

专注计算机视觉前沿资讯和技术干货，官网：www.cvmart.net

472篇原创内容



公众号

△点击卡片关注极市平台，获取[最新CV干货](#)

公众号后台回复“[长尾](#)”获取长尾特征学习资源~

极市干货

YOLO教程：[一文读懂YOLO V5 与 YOLO V4](#) | [大盘点](#) | [YOLO 系目标检测算法总览](#) | [全面解析YOLO V4网络结构](#)

实操教程：[PyTorch vs LibTorch：网络推理速度谁更快？](#) | [只用两行代码，我让Transformer推理加速了50倍](#) | [PyTorch AutoGrad C++层实现](#)

算法技巧 (trick)：[深度学习训练tricks总结（有实验支撑）](#) | [深度强化学习调参Tricks合集](#) | [长尾识别中的Tricks汇总（AAAI2021）](#)

最新CV竞赛：[2021 高通人工智能应用创新大赛](#) | [CVPR 2021](#) | [Short-video Face Parsing Challenge](#) | [3D人体目标检测与行为分析竞赛开赛，奖池7万+，数据集达16671张！](#)



CV技术社群邀请函



△长按添加极市小助手

添加极市小助手微信 (ID : **cvmart2**)

备注: **姓名-学校/公司-研究方向-城市** (如: 小极-北大-目标检测-深圳)

即可申请加入极市 **目标检测/图像分割/工业检测/人脸/医学影像/3D/SLAM/自动驾驶/超分辨率/姿态估计/ReID/GAN/图像增强/OCR/视频理解** 等技术交流群

每月大咖直播分享、真实项目需求对接、求职内推、算法竞赛、干货资讯汇总、与 **10000+** 来自港科大、北大、清华、中科院、CMU、腾讯、百度等名校名企视觉开发者互动交流~

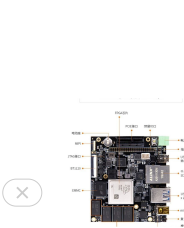
觉得有用麻烦给个在看啦~

[阅读原文](#)

喜欢此内容的人还喜欢

深度学习模型在FPGA上的部署

FPGA之家



使用计算机视觉来做异常检测

小白学视觉

