



北京师范大学 珠海校区

BEIJING NORMAL UNIVERSITY AT ZHUHAI

# Pytorch入门

马静

github.com/MadelineMa/Recommender-System

github.com/MadelineMa/Recommender-System





**PyTorch更有利于研究人员、爱好者、小规模项目等快速搞出原型。而TensorFlow更适合大规模部署，特别是需要跨平台和嵌入式部署时。**

- TensorFlow可以看成是一个嵌入Python的编程语言。

TensorFlow代码会被Python编译成一张图，然后由TensorFlow执行引擎运行。TensorFlow有一些额外的概念需要学习，例如图、变量作用域（variable scoping）、占位符等，上手时间长。

- PyTorch中简单的图结构更容易理解，更重要的是，还更容易调试。

创建和运行计算图可能是两个框架最不同的地方。在PyTorch中，图结构是动态的，这意味着图在运行时构建。而在TensorFlow中，图结构是静态的，这意味着图先被“编译”然后再运行。

- 部署、序列化、设备管理等方面，tensorflow更胜一筹。

[PyTorch还是TensorFlow? 这有一份新手深度学习框架选择指南 - 知乎 \(zhihu.com\)](#)



北京師範大學 珠海校区

BEIJING NORMAL UNIVERSITY AT ZHUHAI

# CONTENT

- 01 安装
- 02 tf.keras使用
- 03 tf.estimator介绍
- 04 补充知识

We can read of things that happened  
5,000 years ago in the Near East,  
where people first learned to write.  
But there are some parts of the word  
where even now people cannot write.





北京师范大学 珠海校区  
BEIJING NORMAL UNIVERSITY AT ZHUHAI



# 01 pytorch安装



- 安装环境建立

[安装PyTorch详细过程\\_MCYZSF的博客-CSDN博客\\_pytorch 安装](#)

新建Python虚拟环境

```
(base) C:\Users\10845>conda create -n pytorch python=3.7
```

激活并进入环境

```
(base) C:\Users\10845>activate pytorch  
(pytorch) C:\Users\10845>
```

Pip list没有torch,  
需要安装

```
(pytorch) C:\Users\10845>pip list  
Package      Version  
-----  
certifi      2020.12.5  
pip          21.0.1  
setuptools   52.0.0.post20210125  
wheel        0.36.2  
wincertstore 0.2
```



- 安装前驱动更新

进入命令符号窗口，输入nvidia-smi,查看当前驱动的版本号，观察Driver Version的值是否大于400，如果小于请更新显卡驱动。(可参考原帖进行驱动更新操作)

```
C:\Users\10845>nvidia-smi
Sat May 08 16:30:56 2021

+-----+
| NVIDIA-SMI 442.50      | Driver Version: 442.50      | CUDA Version: 10.2      |
+-----+-----+
| GPU  Name            | TCC/WDDM | Bus-Id      | Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf    | Pwr:Usage/Cap |          | Memory-Usage | GPU-Util  Compute M. |
+-----+-----+-----+-----+-----+-----+
|    0  GeForce GTX 1050 | WDDM     | 00000000:01:00.0 Off |         |      0%      Default |
| N/A   49C    P8      | N/A /  N/A |          | 75MiB / 4096MiB |              |
+-----+-----+-----+-----+-----+-----+

+-----+
| Processes:                                | GPU Memory |
| GPU       PID    Type    Process name                     | Usage      |
+-----+-----+-----+-----+-----+-----+
| No running processes found |             |
+-----+-----+-----+-----+-----+-----+

https://blog.csdn.net/MCYZSF
```



- 安装torch

进入官网: <https://pytorch.org/>

复制这一段操作指令

## START LOCALLY

Select your preferences and run the install command. Stable represents the most currently tested and supported version of PyTorch. This should be suitable for many users. Preview is available if you want the latest, not fully tested and supported, 1.9 builds that are generated nightly. Please ensure that you have **met the prerequisites below (e.g., numpy)**, depending on your package manager. Anaconda is our recommended package manager since it installs all dependencies. You can also [install previous versions of PyTorch](#). Note that LibTorch is only available for C++.

PyTorch Build	Stable (1.8.1)		Preview (Nightly)	
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python		C++ / Java	
Compute Platform	CUDA 10.2	CUDA 11.1	ROCm 4.0 (beta)	CPU
Run this Command:	<code>conda install pytorch torchvision torchaudio cudatoolkit=10.2 -c pytorch</code>			

```
(pytorch) C:\Users\10845>conda install pytorch torchvision torchaudio cudatoolkit=10.2 -c pytorch
```



- 检验安装

```
(pytorch) C:\Users\10845>pip list
Package                Version
-----
certifi                 2020.12.5
mkl-fft                 1.3.0
mkl-random              1.2.1
mkl-service             2.3.0
numpy                   1.20.1
olefile                 0.46
Pillow                  8.2.0
pip                     21.0.1
setuptools              52.0.0.post20210125
six                     1.15.0
torch                   1.8.1
torchaudio              0.8.1
torchvision             0.9.1
typing-extensions       3.7.4.3
wheel                   0.36.2
wincertstore            0.2.2
```

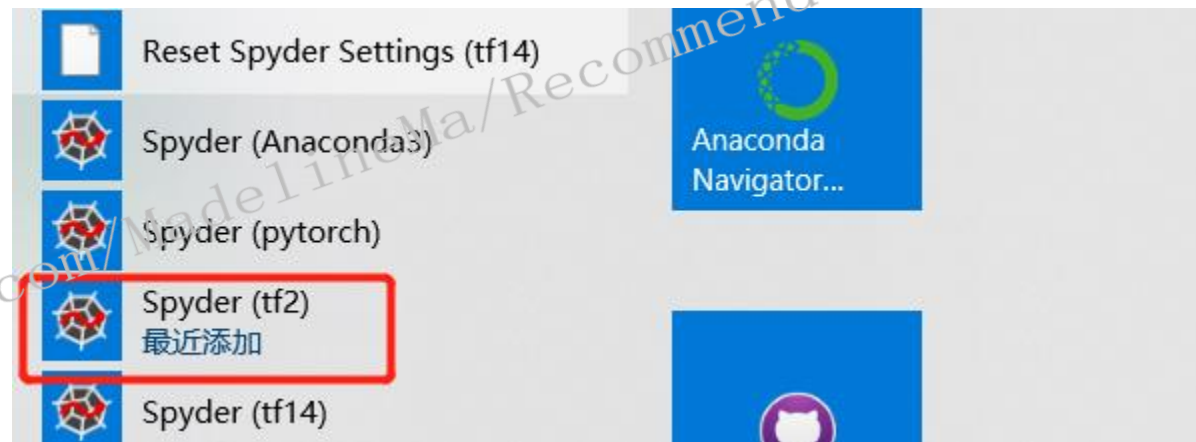
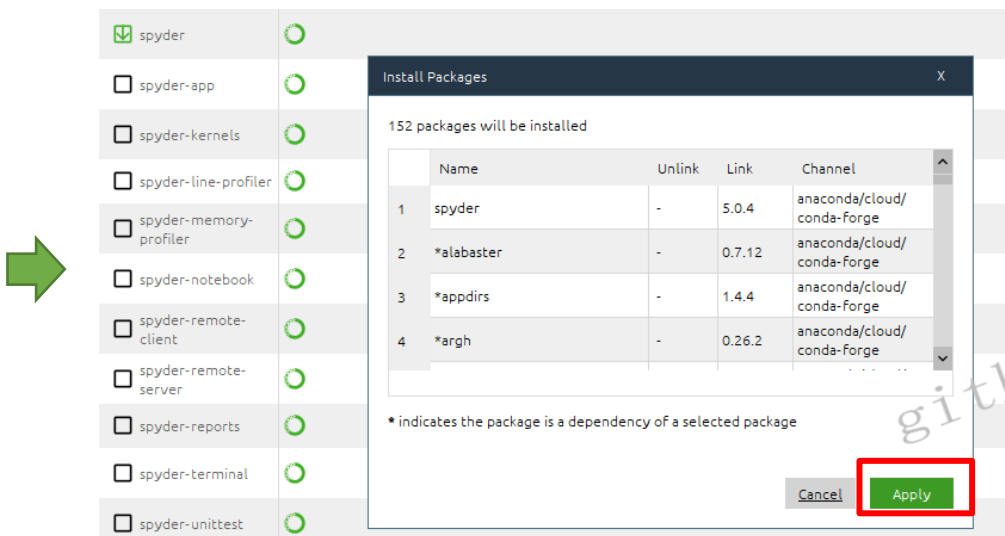
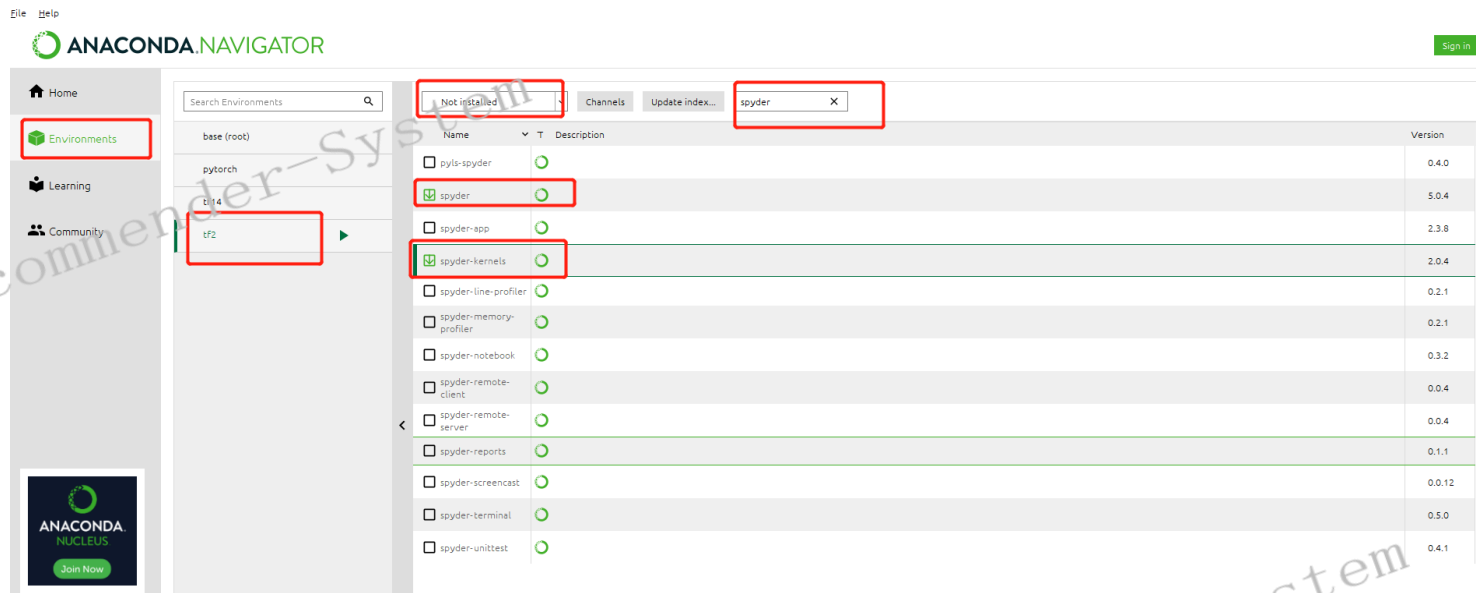
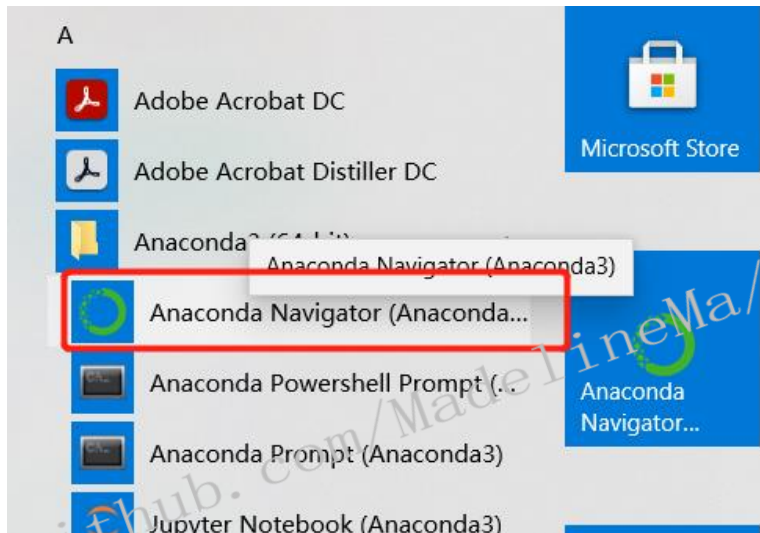
```
(pytorch) C:\Users\10845>python
Python 3.7.10 (default, Feb 26 2021, 13:06:18) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> import torch
>>> torch.cuda.is_available()
True
>>>
```





# 新环境的spyder界面

同tensorflow





```
pip install pandas
```

```
pip install scikit-learn
```

```
...
```

[github.com/MadelineMa/Recommender-System](https://github.com/MadelineMa/Recommender-System)

[github.com/MadelineMa/Recommender-System](https://github.com/MadelineMa/Recommender-System)

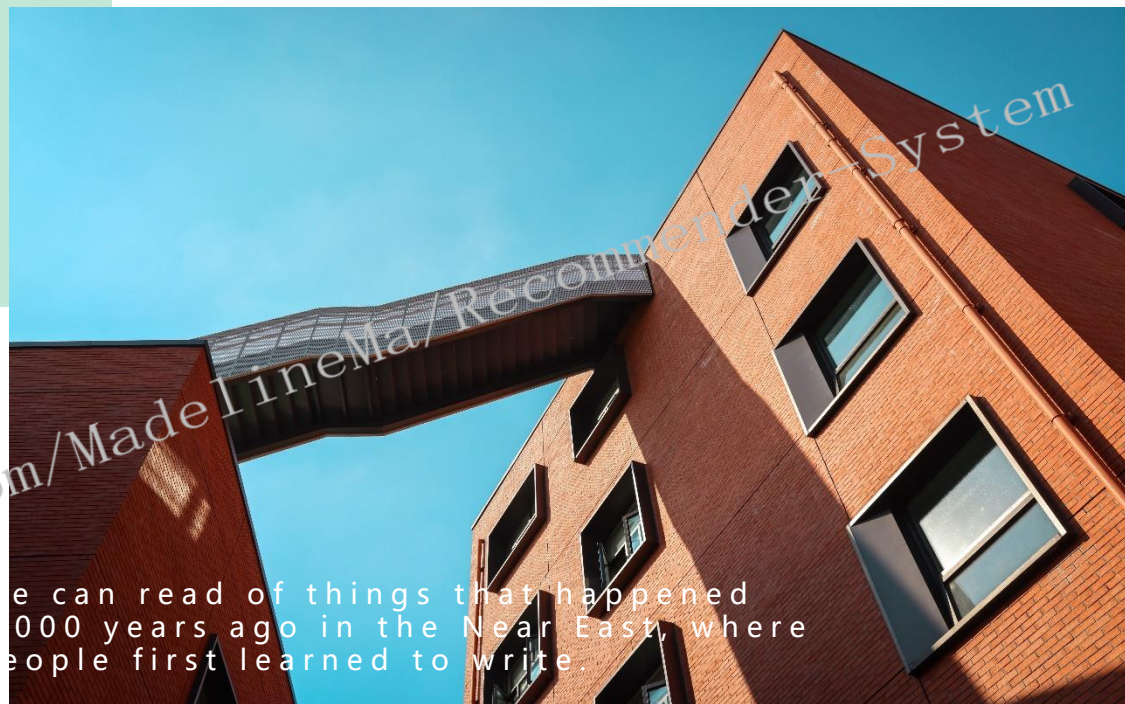




北京师范大学 珠海校区  
BEIJING NORMAL UNIVERSITY AT ZHUHAI

02

torch使用



we can read of things that happened  
1000 years ago in the Near East, where  
people first learned to write.



1. 张量了解，张量与numpy的关系：

[pytorch-handbook/1\\_tensor\\_tutorial.ipynb at master · zergtant/pytorch-handbook \(github.com\)](https://github.com/zergtant/pytorch-handbook/blob/master/1_tensor_tutorial.ipynb)

- `.to('cuda')`将张量放入GPU进行计算，加快计算速度

2. 梯度的理解，梯度的基本设置了解：

[pytorch-handbook/2\\_autograd\\_tutorial.ipynb at master · zergtant/pytorch-handbook \(github.com\)](https://github.com/zergtant/pytorch-handbook/blob/master/2_autograd_tutorial.ipynb)

```
In [10]: x = torch.randn(3, requires_grad=True)
         y = x * 2
         while y.data.norm() < 1000:
             y = y * 2

         print(y)
```

```
tensor([ 293.4463,  50.6356, 1031.2501], grad_fn=<MulBackward0>)
```

在这个情形中，`y`不再是个标量。`torch.autograd`无法直接计算出完整的雅可比行列，但是如果我们只想要vector-Jacobian product，只需将向量作为参数传入`backward`：

```
In [11]: gradients = torch.tensor([0.1, 1.0, 0.0001], dtype=torch.float)
         y.backward(gradients)

         print(x.grad)
```

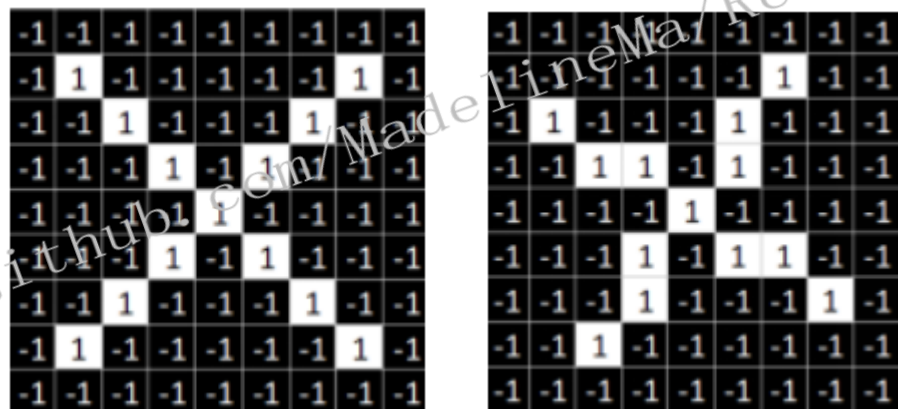
```
tensor([5.1200e+01, 5.1200e+02, 5.1200e-02])
```

手动推导，为什么是这个结果



# 补： 图片分类基本知识

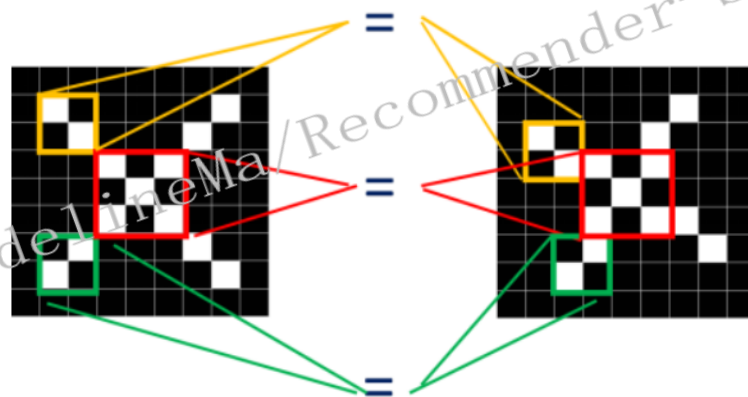
卷积神经网络： 将X和O的图片二分类



机器学会各种X的图片， 要经历卷积层， 池化层等

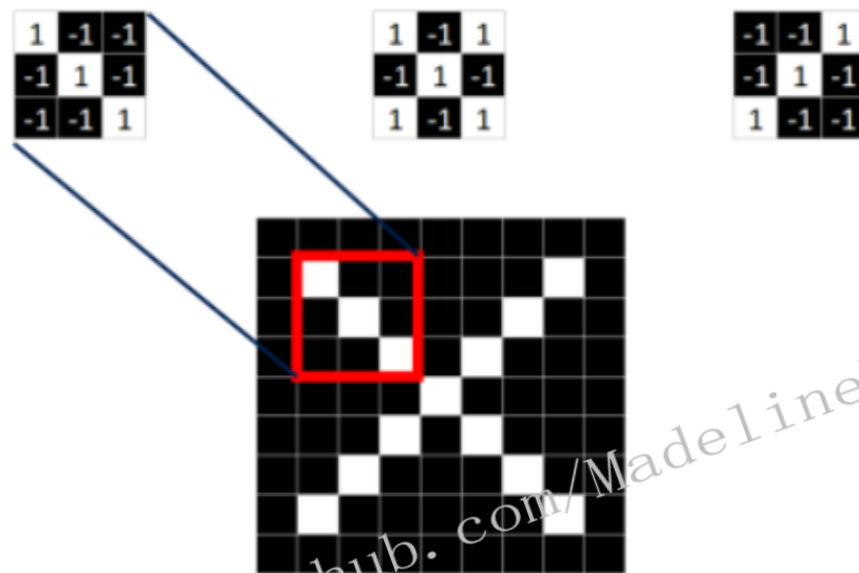
卷积层： 学会图片不同的feature

池化层： 缩小存储参数



# 补：图片分类基本知识

- 图片包含3个feature，只要用这三个feature便可定位到X的某个局部，feature在CNN中也被成为卷积核（filter），一般是3X3，或者5X5的大小。

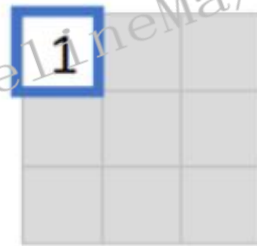
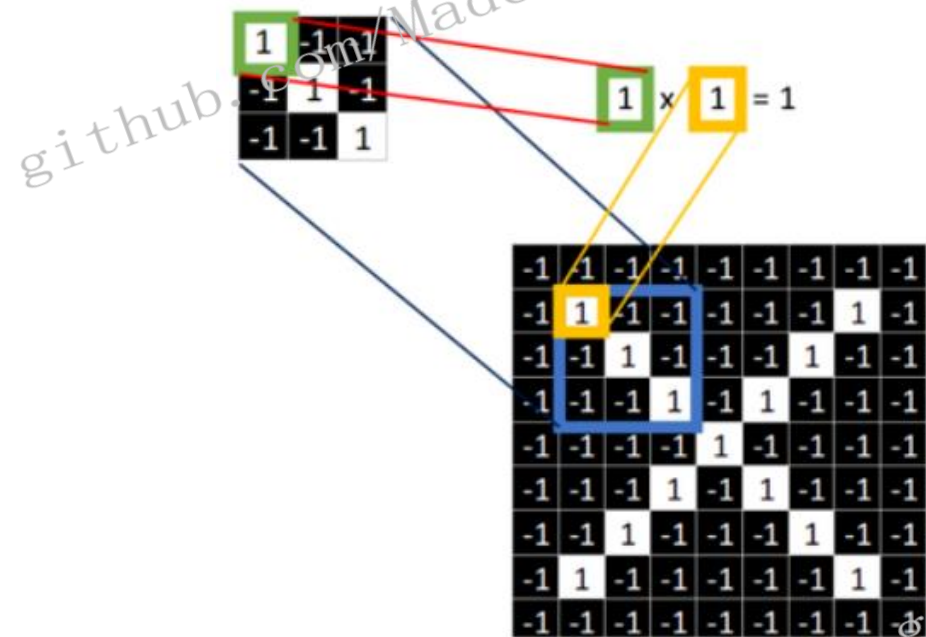




# 补： 图片分类基本知识

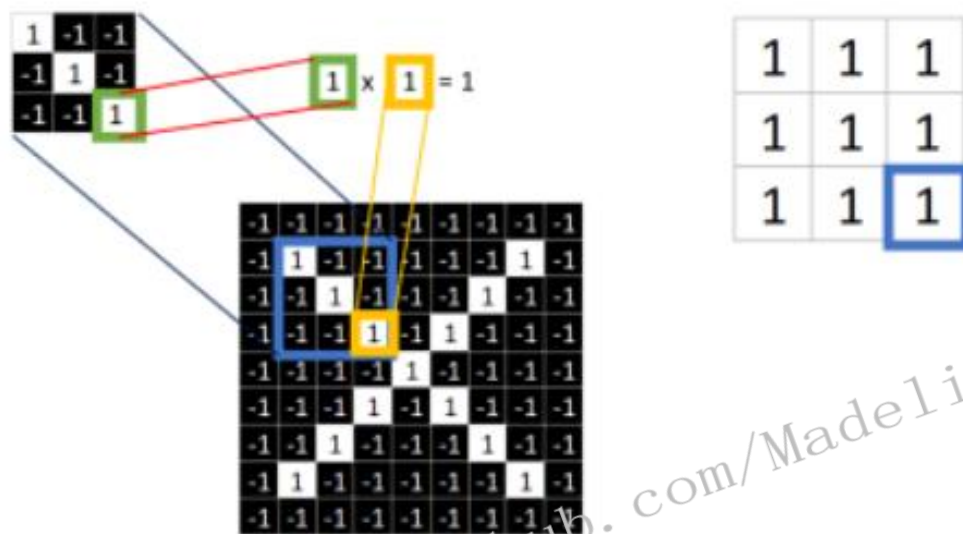
## 运算方法： 对应相乘

取 feature 里的 (1, 1) 元素值，再取图像上蓝色框内的 (1, 1) 元素值，二者相乘等于1。把这个结果1填入新的图中。



# 补： 图片分类基本知识

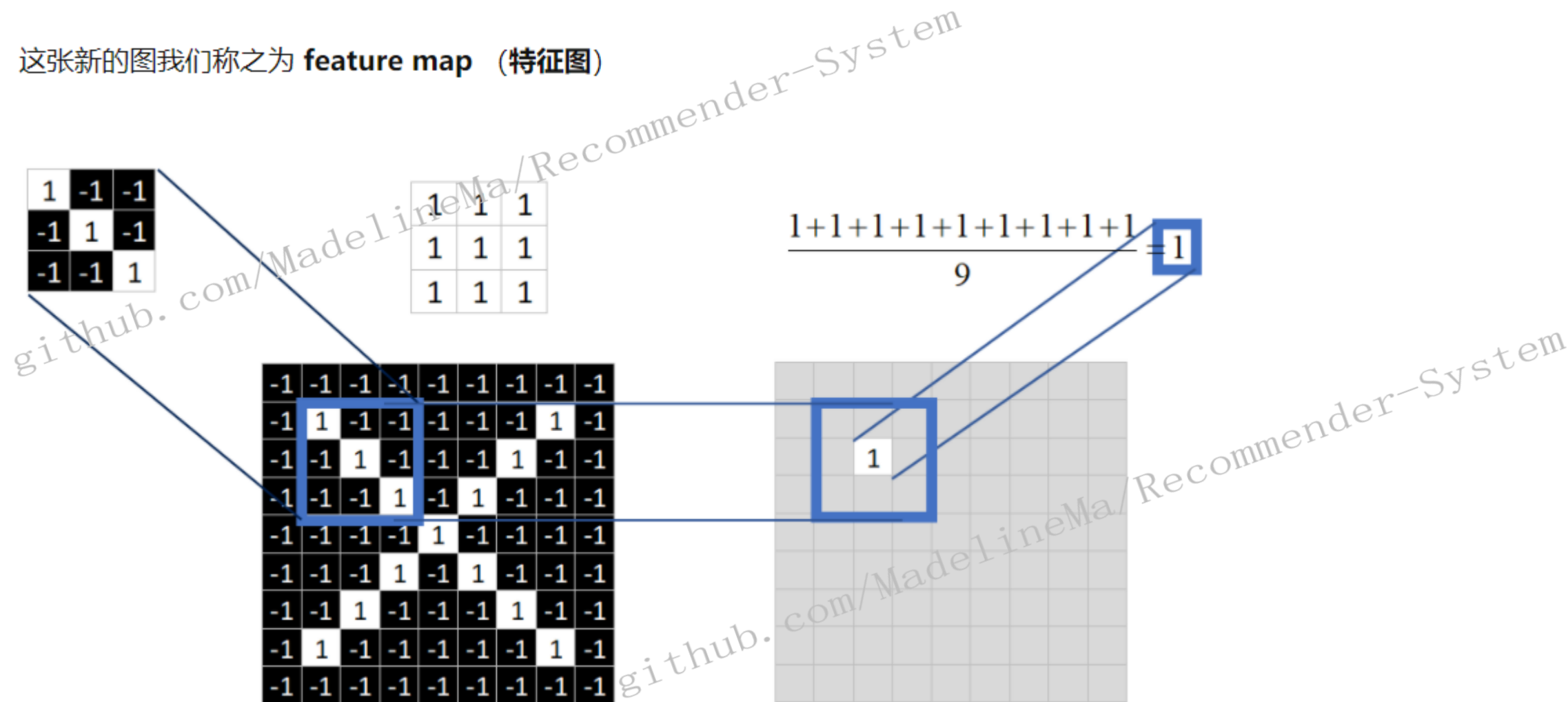
一个窗口的卷积核计算原始结果，一般会对这个结果取平均  
9个都计算完了就会变成这样。





# 补： 图片分类基本知识

这张新的图我们称之为 **feature map** (特征图)



# 补： 图片分类基本知识

- 窗口滑动后，得到新的结果图

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

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

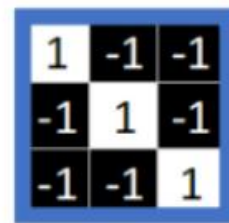
对图像运用该卷积核，  
产生的结果



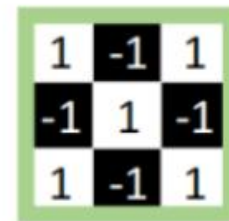
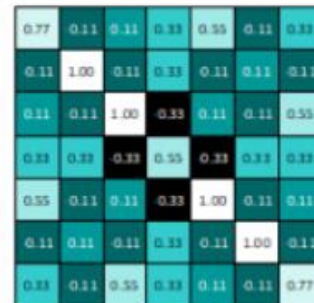
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

# 补： 图片分类基本知识

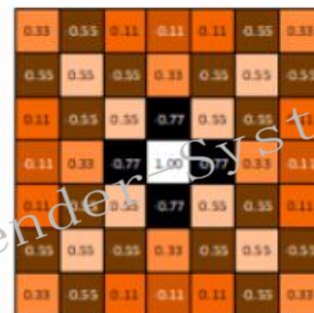
n个卷积核， 得到n张图



=



=



=





# 补： 图片分类基本知识

- 卷积核操作后会跟随一个激活函数，一般用ReLU:  $f(x)=\max(0,x)$

卷积后产生的特征图中的值，越靠近1表示与该特征越关联，越靠近-1表示越不关联，而我们进行特征提取时，为了使得数据更少，操作更方便，就直接舍弃掉那些不相关联的数据。

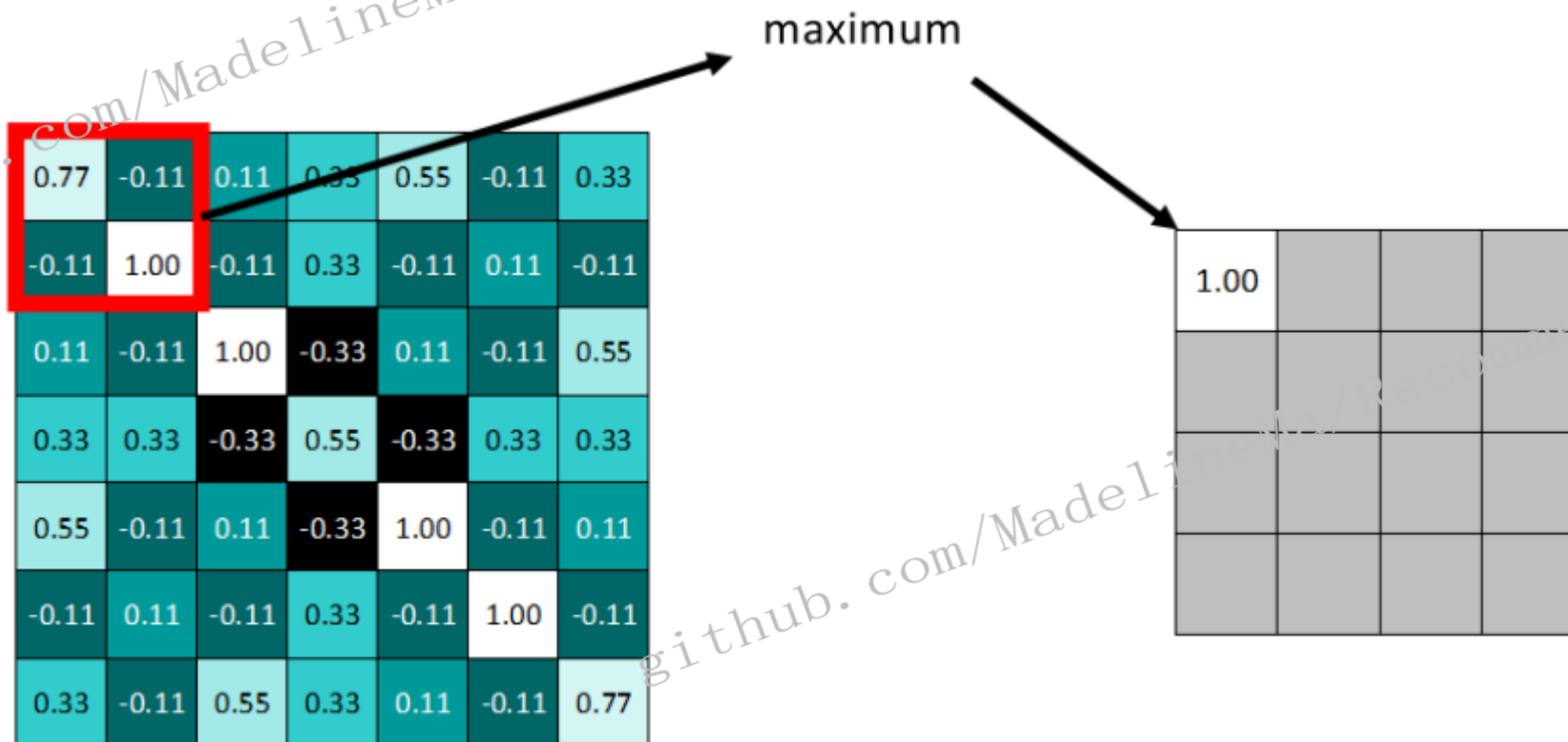
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77



0.77	0	0.11	0.33	0.55	0	0.33
0	1.00	0	0.33	0	0.11	0
0.11	0	1.00	0	0.11	0	0.55
0.33	0.33	0	0.55	0	0.33	0.33
0.55	0	0	0	1.00	0	0.11
0	0.11	0	0.33	0	1.00	0
0.33	0	0.55	0.33	0.11	0	0.77

# 补： 图片分类基本知识

Pooling: 拿最大池化举例： 选择池化尺寸为2x2， 因为选定一个2x2的窗口， 在其内选出最大值更新进新的feature map。



# 补： 图片分类基本知识

在常见的几种CNN中，这三层都是可以堆叠使用的，将前一层的输入作为后一层的输出。比如：







### 3. 一个分类器的例子，理解torch的net和forward思想

- [pytorch-handbook/4\\_cifar10\\_tutorial.ipynb at master · zergtant/pytorch-handbook \(github.com\)](https://github.com/zergtant/pytorch-handbook/blob/master/4_cifar10_tutorial.ipynb)
- `Git/exercises/pytorch_multiclassifier_image.py`
- 分清 epoch, batch, 优化位置
- `nn.Conv2d()`, `nn.MaxPool2d()`, `enumerate()`, `iter()`等陌生函数的意义，养成主动查找的习惯。

### 4. 自己写一个pytorch的rf分类器，解决蘑菇分类的问题.



## 参考书籍

Pytorch的中文手册:

<https://github.com/zergtant/pytorch-handbook>

最简单的CNN例子：

[卷积神经网络CNN完全指南终极版（一） - 知乎 \(zhihu.com\)](#)





北京師範大學 珠海校区  
BEIJING NORMAL UNIVERSITY AT ZHUHAI

THANKS

DESIGNED BY 2xh