

基于卷积神经网络的文本分析

内容

- 卷积的概念与运算
- TextCNN
- 基于卷积神经网络的图像分类

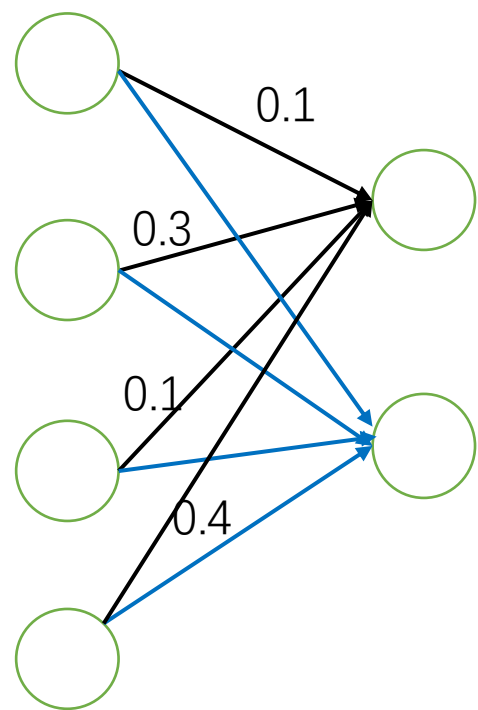
卷积神经网络

二维卷积：在两个维度进行卷积，比如空间信息。图像数据一般用二维卷积。

一维卷积：在一个维度进行卷积，比如时间信息。文本数据一般用一维卷积。

一维卷积(1D convolution)

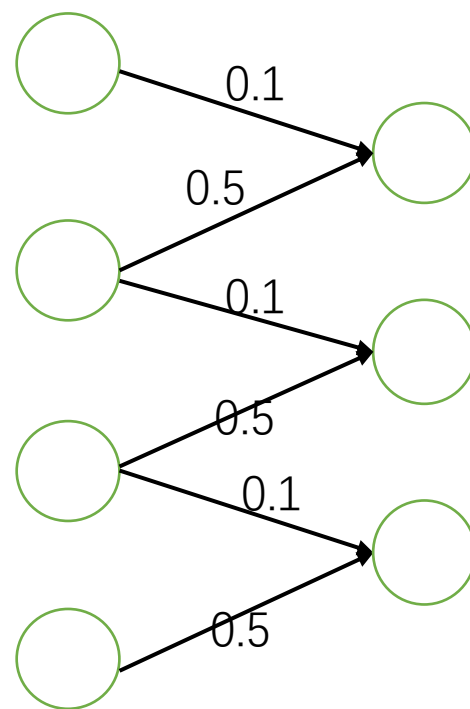
- 全连接，每个隐藏层节点与所有前一层的节点都有一组独立的权重： $n_l \times n_{l-1}$ 个，参数太多。



局部性
每个节点只与其窗口内的输入有关



共享权重



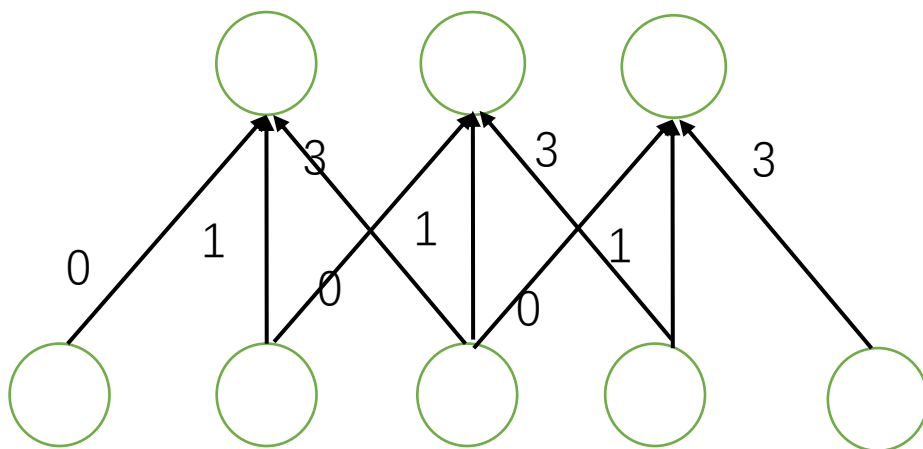
权重数目下降到多少？

权重从8个降到2个。

局部性使得连接降到6个，共享性降到2个。

一维卷积(1D convolution)

- 卷积核 (convolution kernel)：局部性、共享性，在信号处理中又叫滤波器(filter)。



卷积核大小代表什么？

感受野 (Receptive field)

- 局部性：每个隐藏层节点只与在卷积核窗口内的前一层节点相关；
- 共享性：所有隐藏层节点共享同一个卷积核

上面两个性质使得权重数降为： k 个， k 为卷积核大小，远小于输入节点个数 n_{l-1} 。

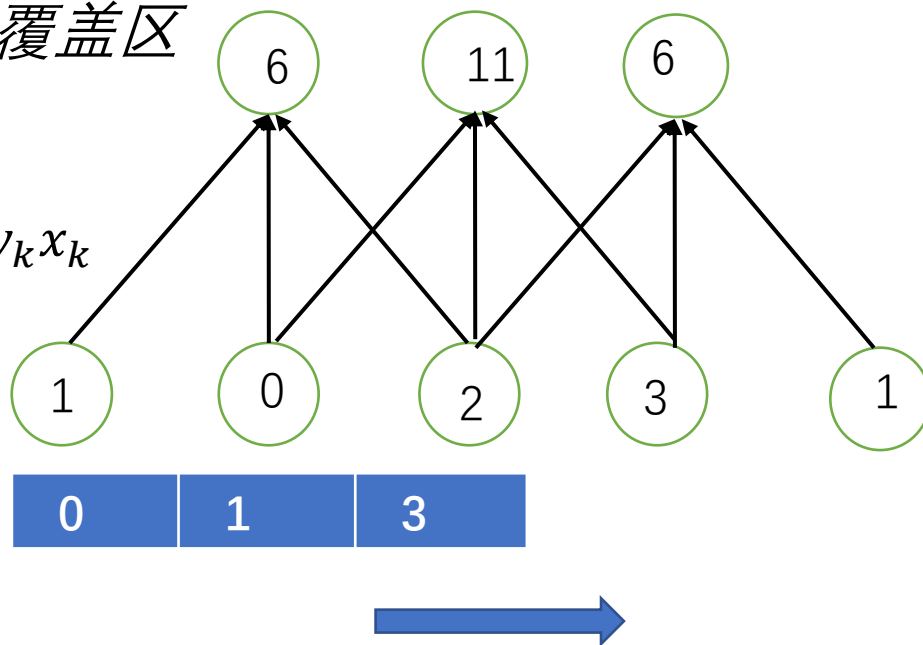
一维卷积

具体操作

- 以步长stride滑动，卷积核 w_l (长度为 k 的向量)与窗口内的节点加权求和再加一个偏置 b_l 。

卷积核覆盖区域内

$$\sum_{k=1}^K w_k x_k$$



卷积层的节点个数怎么确定？

旁边窗口内的节点数不到窗口长度怎么办？




一维卷积

- 卷积层节点个数不是随便指定，而是根据以下计算得到：

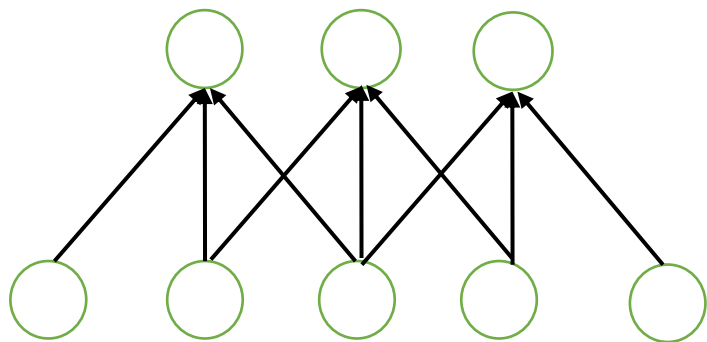
$$\frac{n - k + 2p}{s} + 1$$

其中 n 表示输入维度（前一层节点个数）， k 为卷积核的大小， s 表示步长， p 为填补0的个数。

- 假设前一层节点个数为 n 在步长 $s=1$ 时有

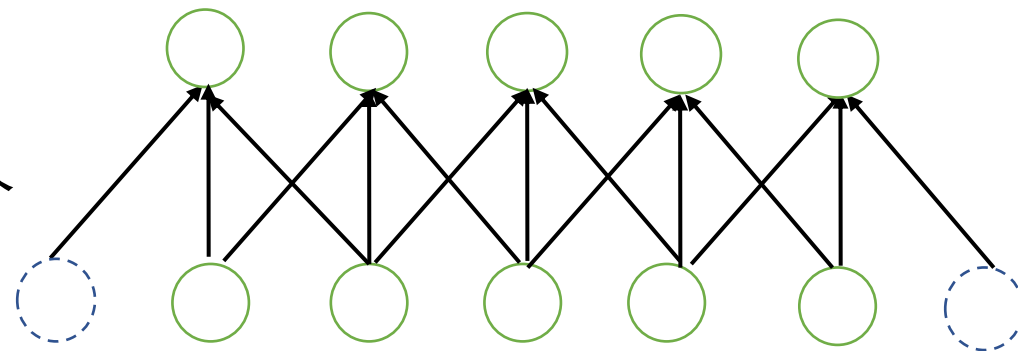
1. 不进行补零(padding)，卷积后输出长度为 $n - k + 1$ 。  变窄
2. 两端进行 $p = k - 1$ 个补零，卷积后输出长度 $n + k - 1$ 。  变宽
3. 与其输入层节点个数相同，两端需要补零 $p = (k - 1)/2$ 个。  等宽

一维卷积



变窄

两端补零
 $p = (k - 1)/2$ 个



等宽

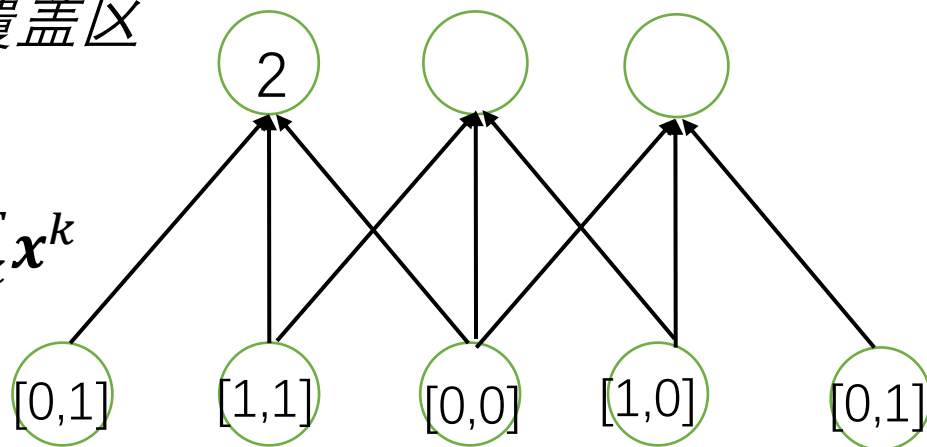
早期文献中默认窄卷积，现在默认等宽卷积。

输入是向量的一维卷积

卷积核覆盖区
域内

$$\sum_{k=1}^K$$

$$\mathbf{w}_k^T \mathbf{x}^k$$



[1,0] [2, 0] [0, 3]



多个卷
积核

覆盖区域内所有对应元素相
乘后的积相加;

一次卷积操作输出一个值;

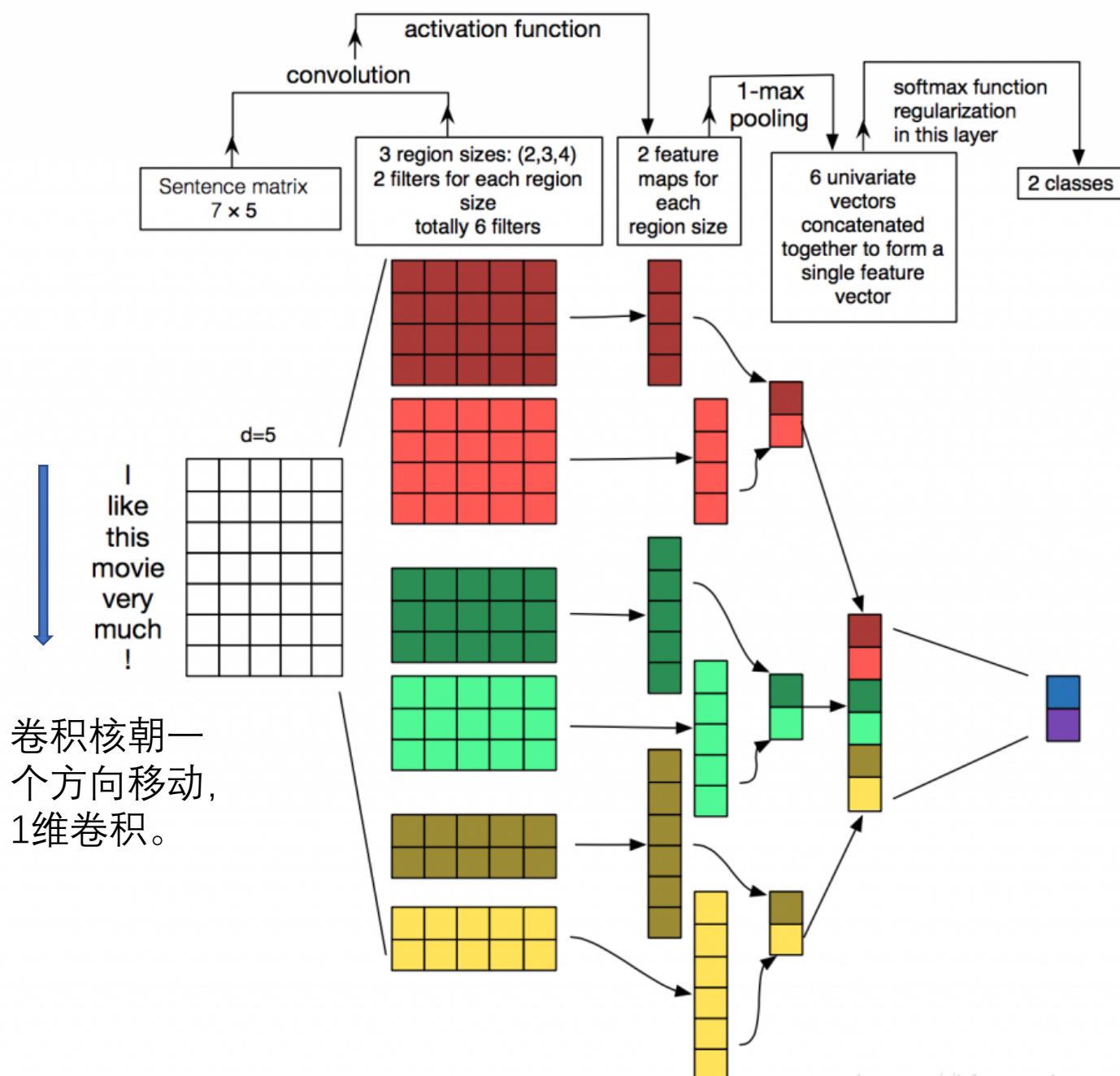
0	1	0	1	0
1	1	0	0	1

1	2	0
0	0	3



汇聚层 (pooling layer)

- 卷积层减少了连接数但是没有显著减少输出节点的个数，后面接分类器时输入维度太高。
- 汇聚层：对每个卷积核对应的输出进行下采样来减少输出节点个数--特征维度，减少参数量。
- 主要包括
 - 最大汇聚Max-Pooling：对给定窗口长度内的所有特征用其最大值代替。
 - 平均汇聚Mean-Pooling：对给定窗口长度内的所有特征用其平均值代替。



用了不同高度的卷积核，目的是得到不同尺度的信息。这里卷积核的宽度与词嵌入维度一样。

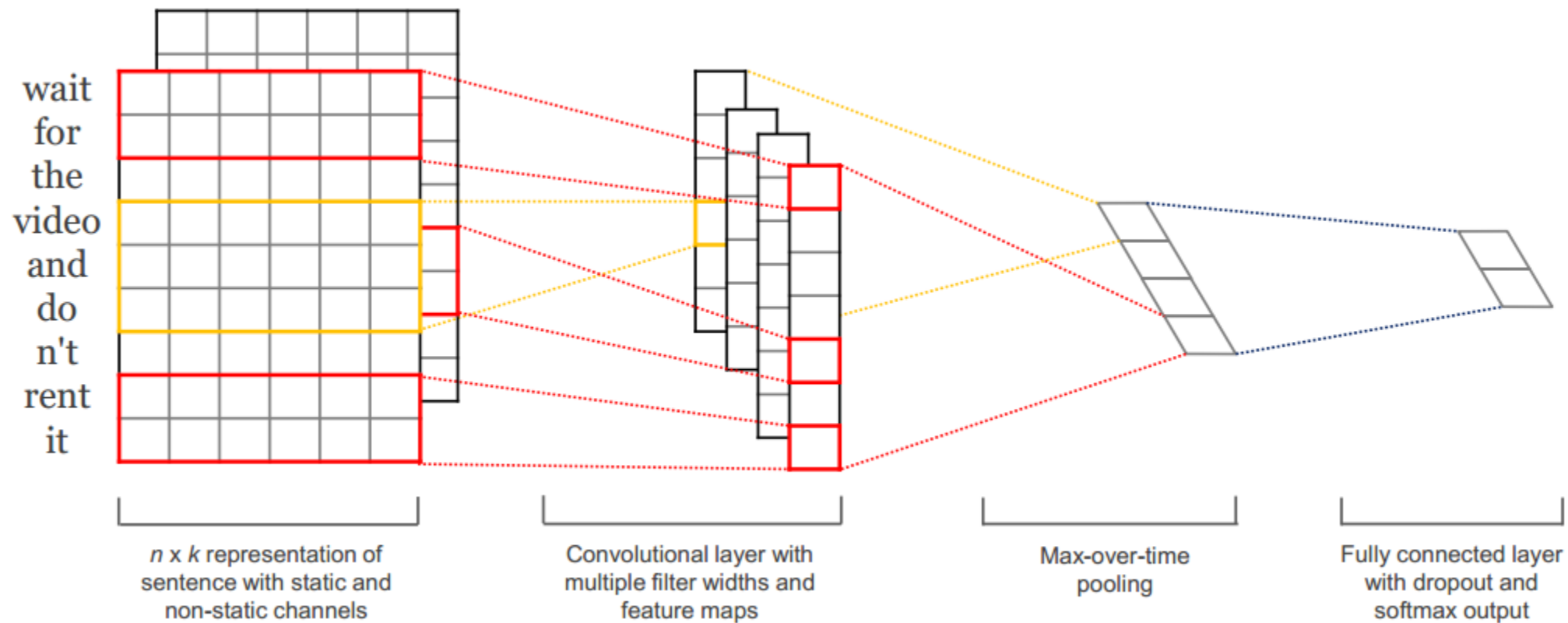
卷积层： stride=1，每个 kernel_size 用2个filter。

由于卷积核大小不同，卷积输出向量长度也不同。

采用1-max pooling： 对一个向量只输出其最大值，与长度无关。

猜猜K-Max Pooling是什么操作？

TextCNN <https://www.aclweb.org/anthology/S15-2079/>



双通道模型结构示意图

tf.keras实现



```
def build_model(self):  
    # 模型架构搭建  
    idx_input = tf.keras.layers.Input((self.config.max_seq_len,))  
    input_embedding = tf.keras.layers.Embedding(len(self.preprocessor.token2idx),  
                                                self.config.embedding_dim,  
                                                input_length=self.config.max_seq_len,  
                                                mask_zero=True)(idx_input)  
  
    convs = []  
    for kernel_size in [3, 4, 5]:  
        c = tf.keras.layers.Conv1D(128, kernel_size, activation='relu')(input_embedding)  
        c = tf.keras.layers.GlobalMaxPooling1D()(c)  
        convs.append(c)  
    fea_cnn = tf.keras.layers.Concatenate()(convs)  
    fea_cnn_dropout = tf.keras.layers.Dropout(rate=0.4)(fea_cnn)  
  
    fea_dense = tf.keras.layers.Dense(128, activation='relu')(fea_cnn_dropout)  
    output = tf.keras.layers.Dense(2, activation='softmax')(fea_dense)  
  
    model = tf.keras.Model(inputs=idx_input, outputs=output)  
    model.compile(loss='sparse_categorical_crossentropy',  
                  optimizer='adam',  
                  metrics=['accuracy'])  
  
    model.summary()  
  
    self.model = model
```

这里二分类输出用了两个节点，激活用softmax。也可以用1个节点的sigmoid。

但不要2个节点再用sigmoid，这样就不对。

tensorflow实现

```
network = input_data(shape=[None, 100], name='input')
network = tflearn.embedding(network, input_dim=10000, output_dim=128)
branch1 = conv_1d(network, 128, 3, padding='valid', activation='relu', regularizer="L2")
branch2 = conv_1d(network, 128, 4, padding='valid', activation='relu', regularizer="L2")
branch3 = conv_1d(network, 128, 5, padding='valid', activation='relu', regularizer="L2")
network = merge([branch1, branch2, branch3], mode='concat', axis=1)
network = tf.expand_dims(network, 2)
network = global_max_pool(network)
network = dropout(network, 0.5)
network = fully_connected(network, 2, activation='softmax')
```

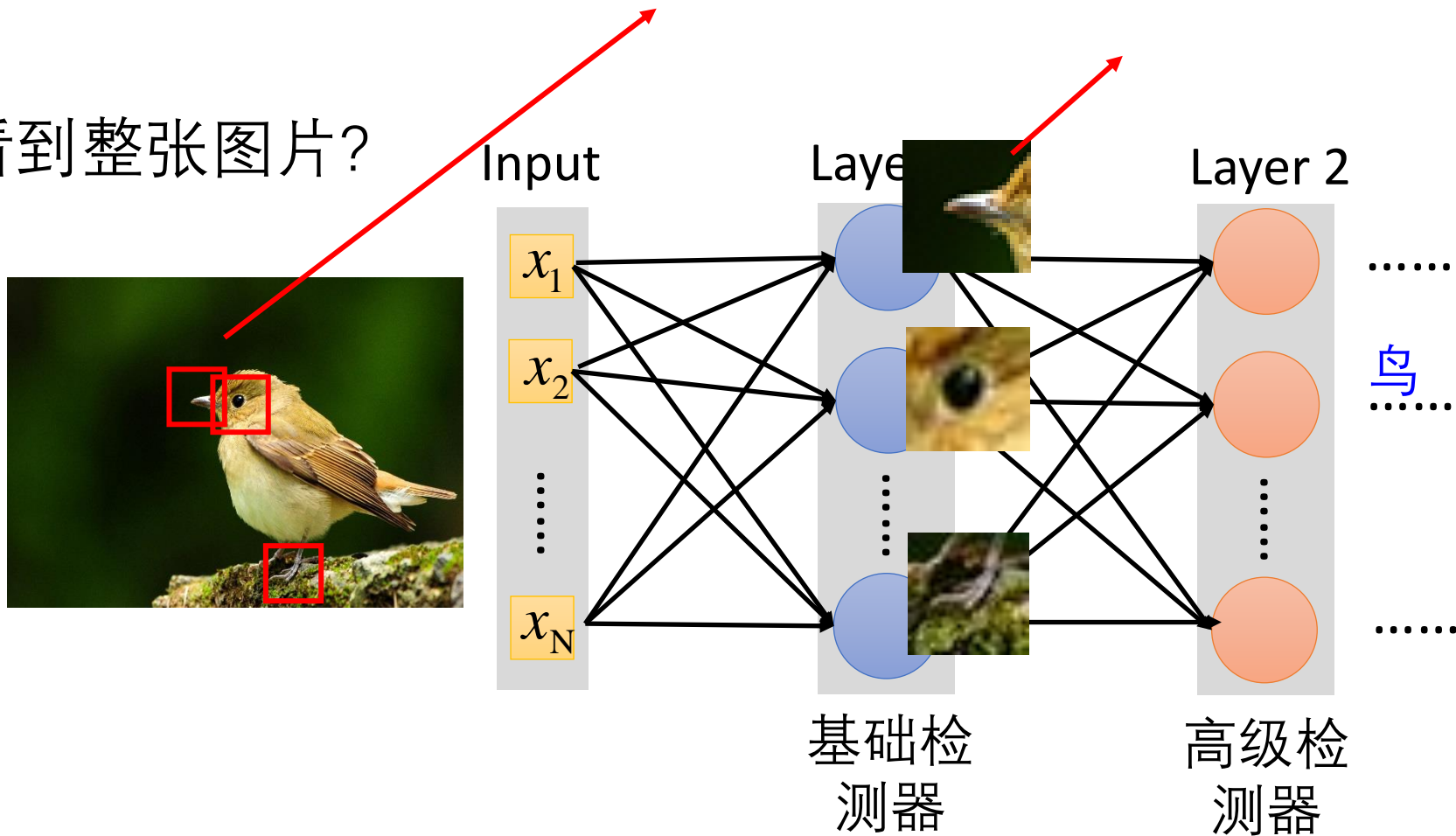
文本数据的通道

- 把不同类的词向量表征（例如word2vec和GloVe）看做是独立的通道。
- 把不同语言版本的同一句话看作是一个通道。

模式识别

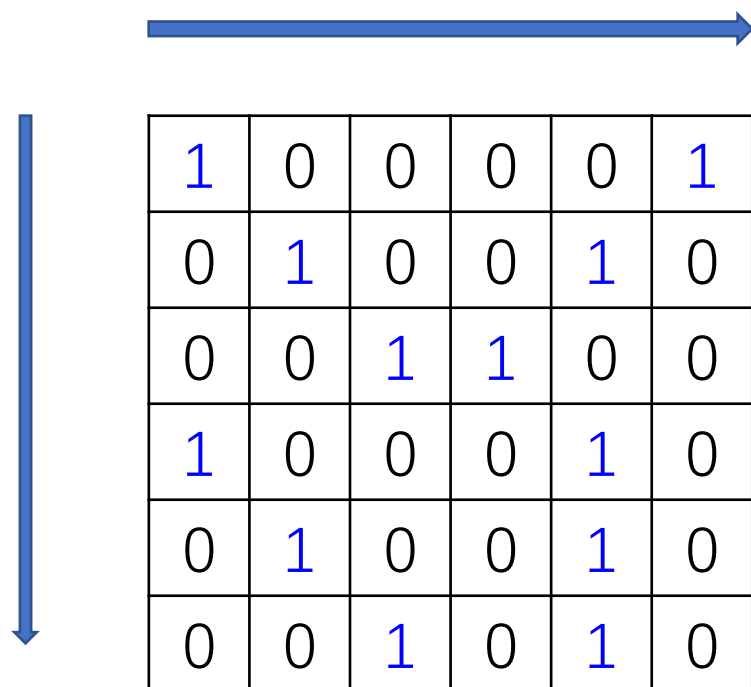
需要看到整张图片?

每个节点只需要看到很小的部分



二维卷积

卷积核在两个方向移动



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

6 x 6 image

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

卷积核1

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

卷积核 2

⋮

二维卷积

stride=1

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

6 x 6 image

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

卷积核 1



3	-1	-3	-1
-3			3
-3	-3	0	1
3	-2	-2	-1

Feature Map

二维卷积

stride=1

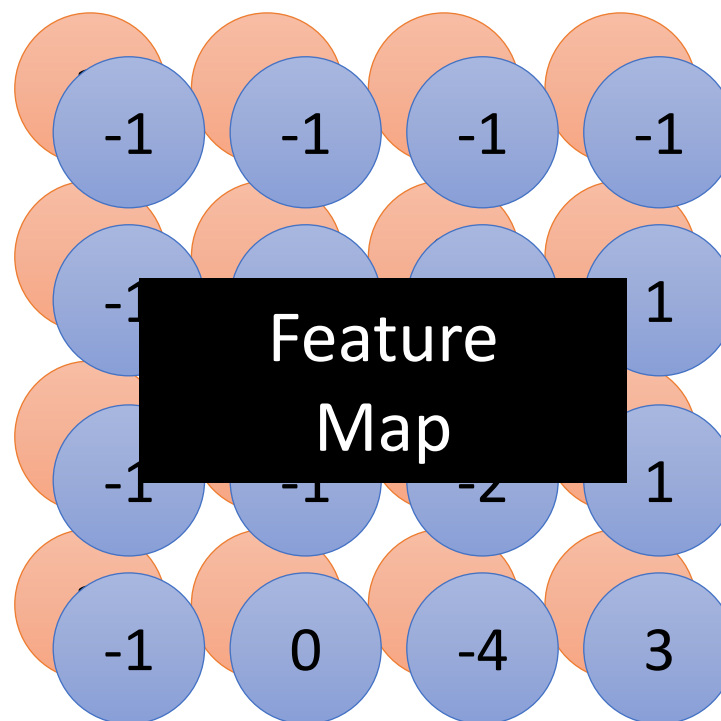
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

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

卷积核 2

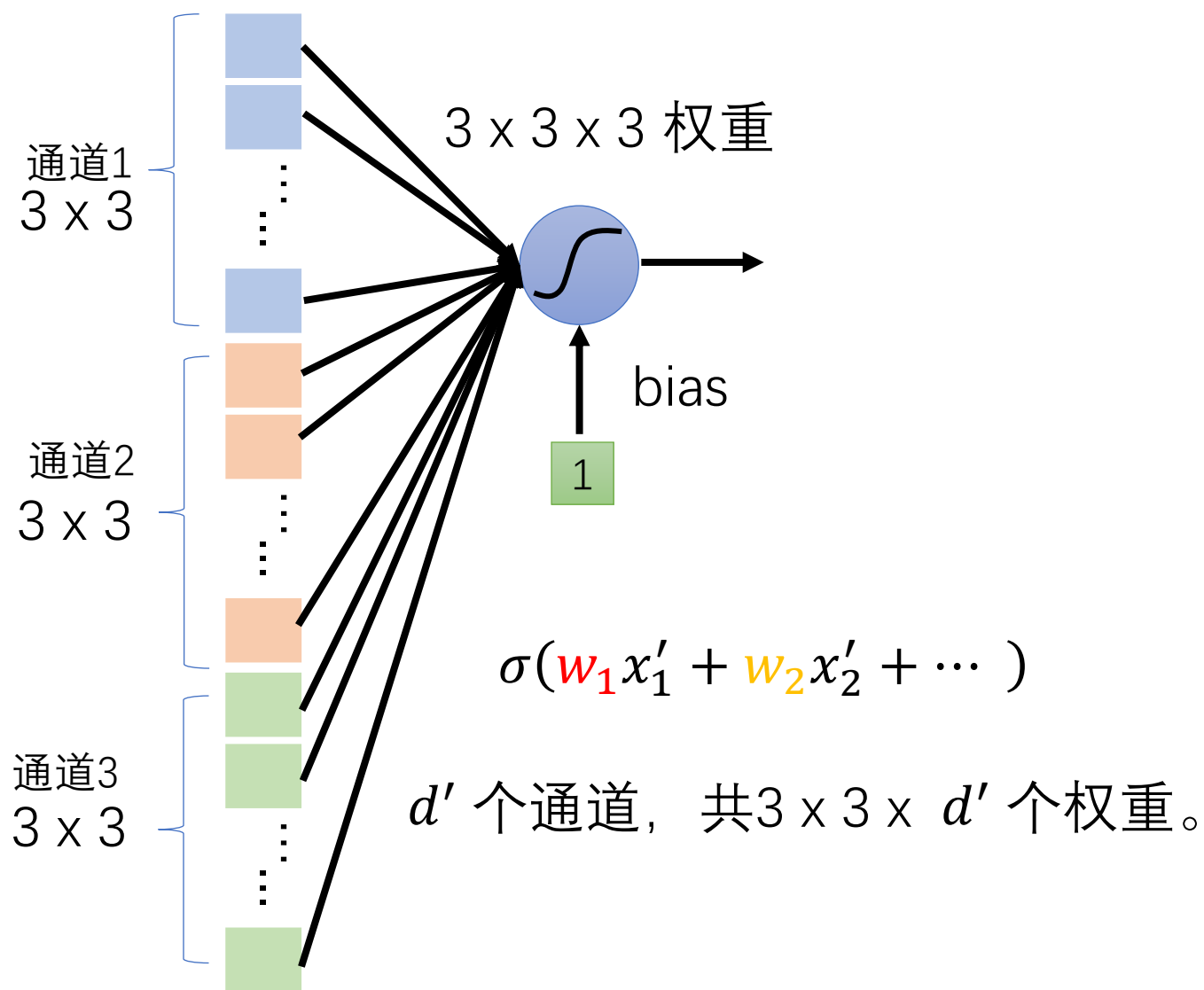
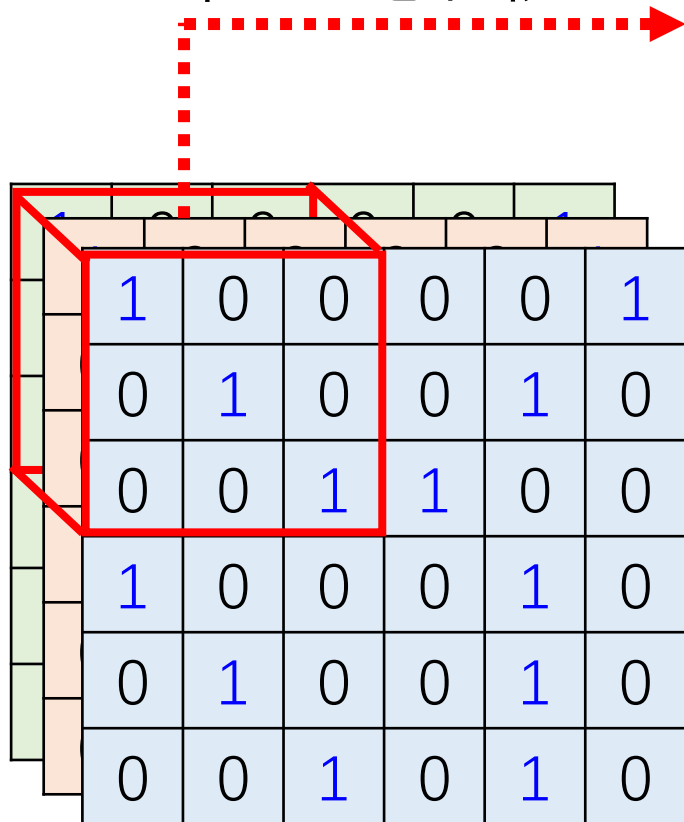
每个卷积核进行同样操作



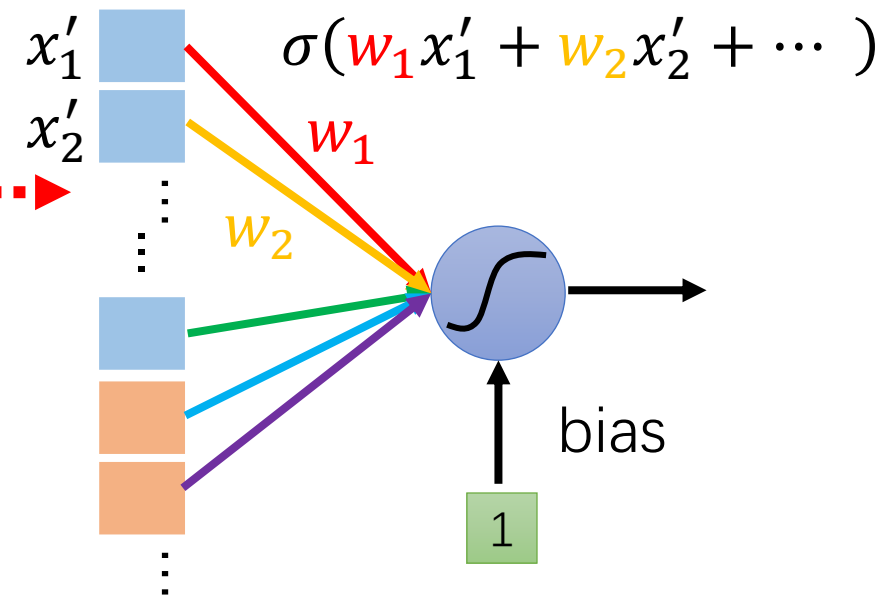
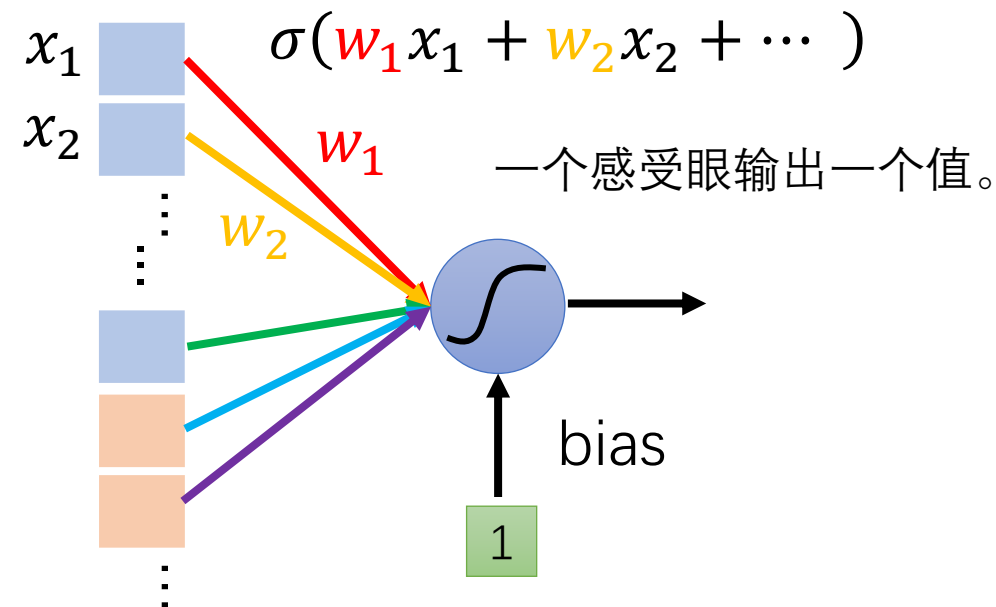
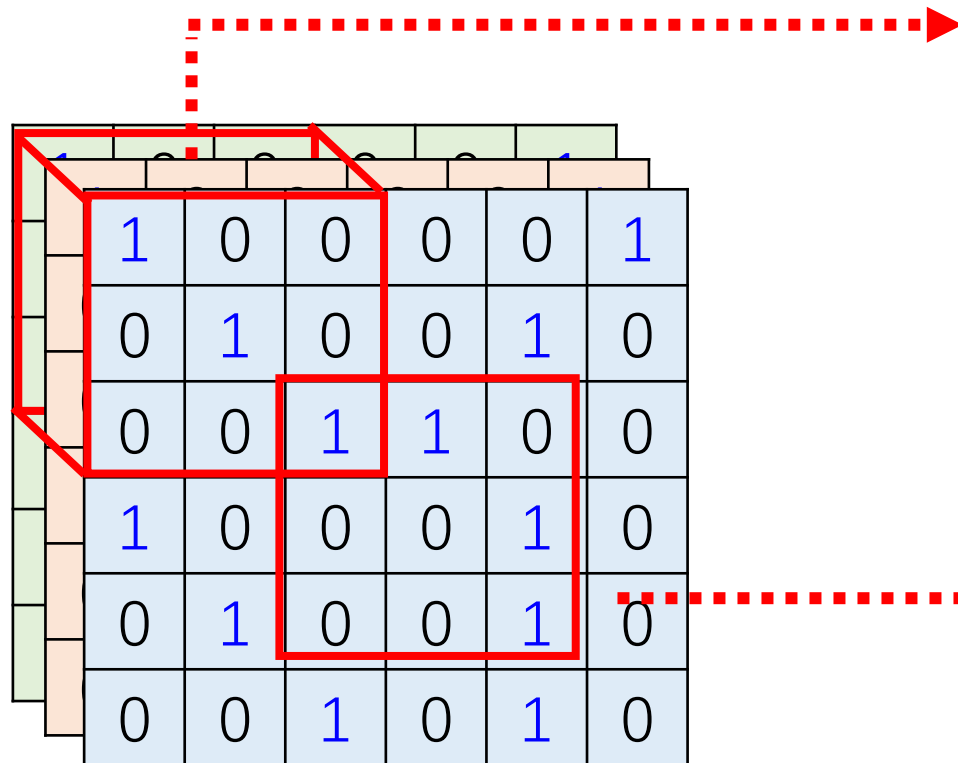
不一样的卷积核得到不同的 FeatureMap

多通道卷积

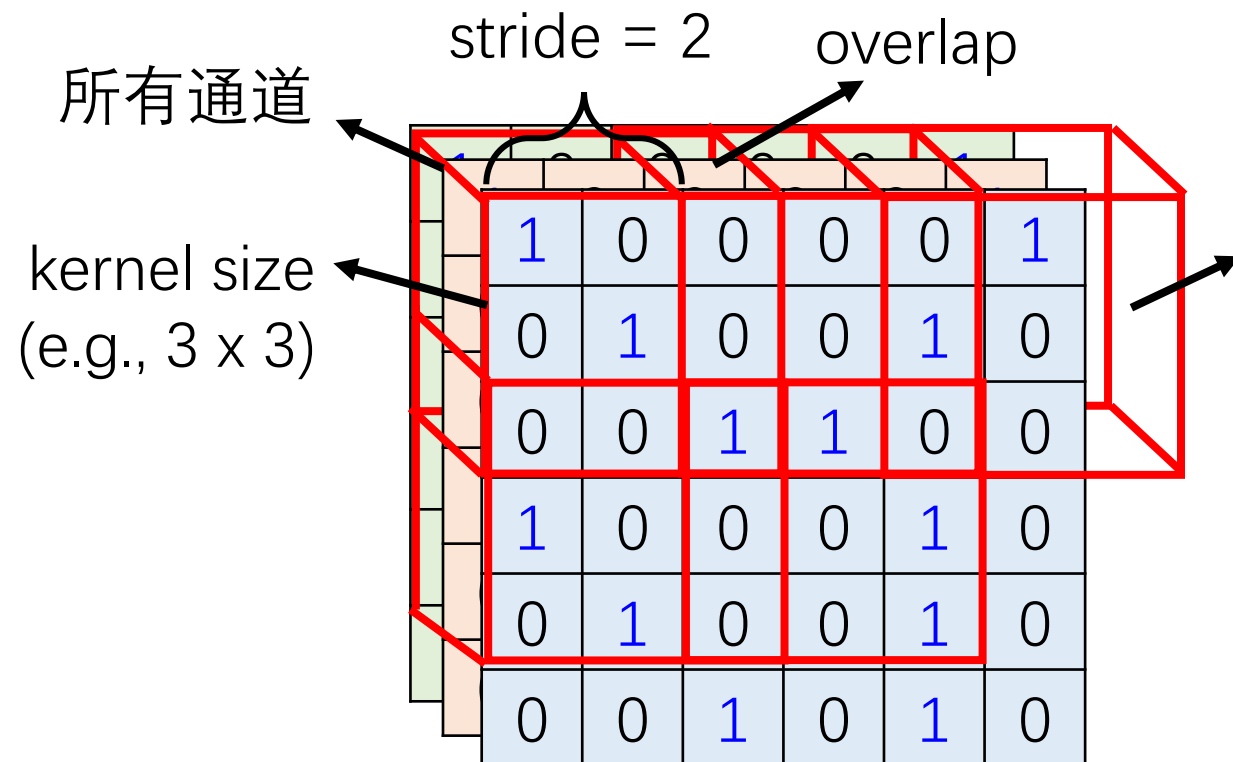
每个通道一个 3×3 卷积核



多通道卷积



典型设置



一个感受野内的所有通道的
二维卷积的值相加。

为了得到多个特征，用多个
卷积，比如64个。

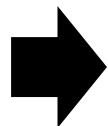
卷积核的大小表示为4
阶张量：

$$\mathbf{W} \in R^{u \times v \times d' \times d}$$

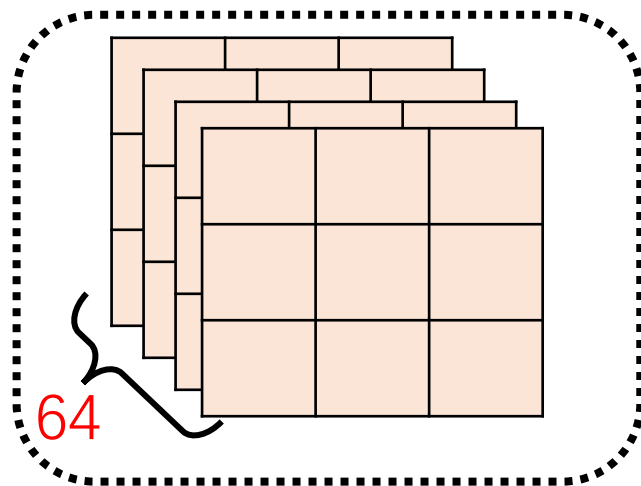
u 和 v 分别为宽和高

d' 为输入的通道数， d 为该卷积输出的通道数

第一个卷积层输出

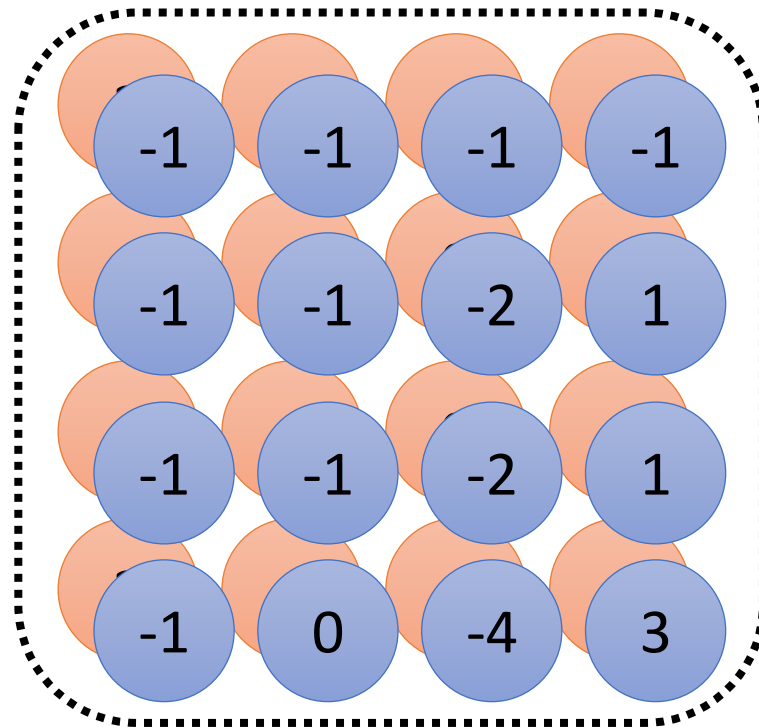


Convolution



Filter:

$3 \times 3 \times 3 \times 64$



“Image” with 64 channels

多个卷积层



64
filters

Convolution

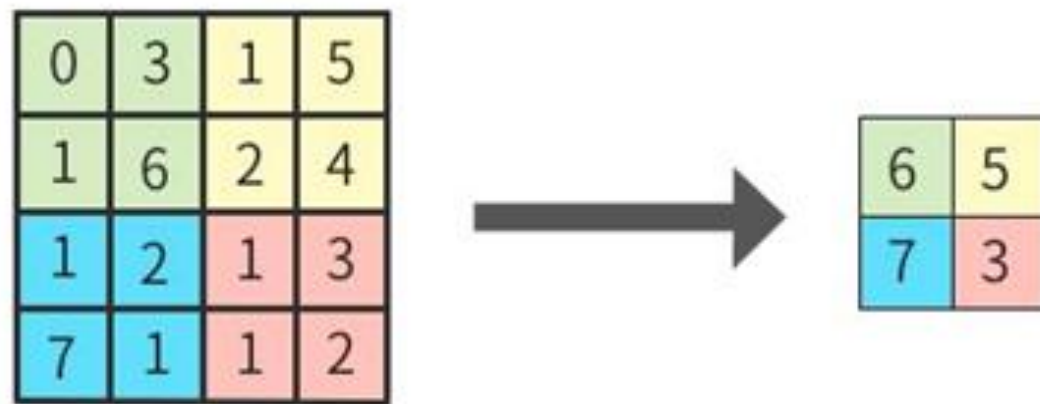
Convolution

⋮

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

-1	-1	-1	-1
-1	-1	-2	1
-1	-1	-2	1
-1	0	-4	3

池化层/汇聚层 (pooling layer)



Max-pooling

Pooling 与卷积同样，用一个较小的窗口在输入节点上滑动来实现。但是卷积层包含（多个）卷积核的参数要学习，而Pooling没有要学习的参数。Pooling层窗口之间一般不重叠。

窗口太大使得输出维度急剧减少，可能造成过多的信息损失。

下采样不改变内容信息

bird

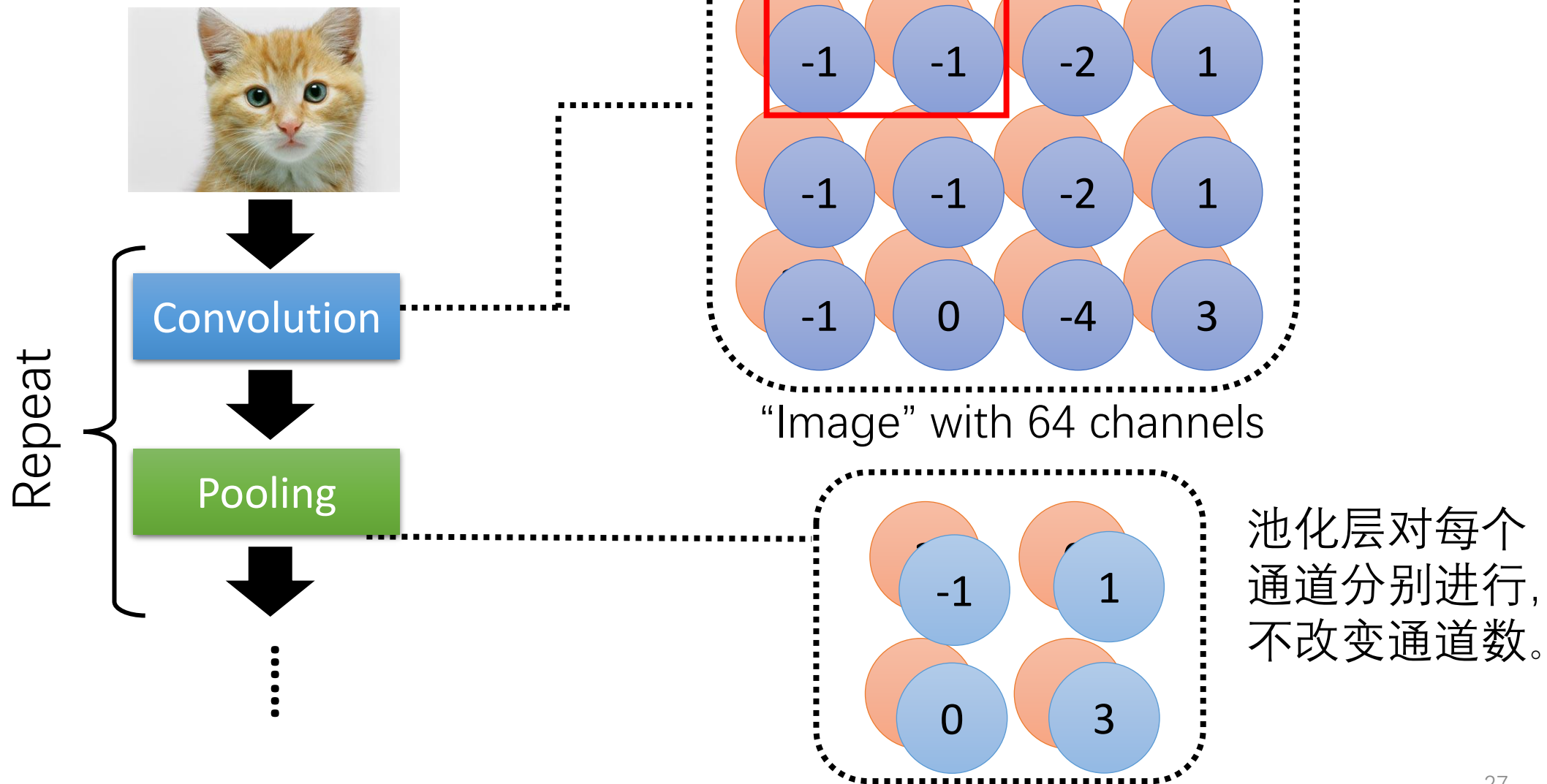


subsampling

bird



卷积+池化



整个网络

