**Notebook credit**: Based on the original D2L notebook here.

# From Fully-Connected Layers to Convolutions

我们迄今为止讨论过的模型对于 tabular data 是合适的选择。所谓 tabular data, 是指数据由 exmaples 作为行, 与 features 作为列组成。 我们很自然会预料到，我们所寻找的 patterns 可能涉及 features 之间的 interaction，但我们并没有 assume any structure *a priori* concerning how the features interact.

有时，我们确实缺乏知识来指导构建更 craftier 的 architectures. 在这种情况下, an MLP may be the best that we can do. 然而，对于 high-dimensional perceptual data 来说，这种 structure-less networks 可能会变得越来越笨重。

例如，让我们回到区分猫和狗的例子。假设我们在数据收集方面做了充分的工作，收集了一百万像素照片的注释数据集。这意味着网络的每个输入都有一百万个维度。根据我们对全连接层参数化成本的讨论，即使把 hidden dimensions 减少到 1000 个, 也需要 $10^6 \times 10^3 = 10^9$ 个参数的全连接层。除非我们有大量的 GPU、分布式优化的天赋和超乎寻常的耐心, 否则 learning the parameters of this network 是不可行的。

然而，如今人类和计算机都能很好地区分猫和狗，这似乎与上述直觉相矛盾. 这是因为自然图像呈现出丰富的 structures, 使得人类和机器学习模型都可以利用这些结构. Convolutional neural networks (CNNs) 是机器学习利用 natural images 中的一些 known structure 的一种 creative way.

## Invariance(不变量): translation invariance(平移不变性) 与locality(局部性) principle

想象一下, 你想要检测图像中的一个物体: 无论我们使用什么方法来识别物体, 我们似乎都不应该过分关注物体在图像中所处的具体位置. 猪通常不会飞, 飞机通常不会游. 尽管如此, 如果有一只猪出现在图像的顶部, 我们仍然可以识别出来。CNN 将 *spatial invariance* (空间不变性) 这一理念进行 systematize, 利用它以更少的参数学习有用的 representations (表征).

现在, 我们可以通过 enumerate 一些 desiderata 来将这些直觉更加具体化, 以指导我们设计适合计算机视觉的神经网络架构 neural network architecture:

1. 在 earliest layers 中, 我们的应该对 the same patch 做出类似的响应, 无论它出现在图像的哪个位置. 这一原则称为 *translation invariance(平移不变性)*.
2. network 的 earliest layers 应该聚焦于 local regions, 而不考虑 contents of the image in distant regions. This is the *locality(局部性)* principle. 最终这些 local representations 应当可以被 aggregated to make predictions at the whole image level.

Let us see how this translates into mathematics.

## Constraining the MLP(多层感知器Multi-Layer Perceptron)

MLP 也就是我们之前一直在做的: 把一层层的 neutrons 连接在一起, 每个 neutron 都是对前一层的所有

我们可以考虑一个以二维图像 $\mathbf{X}$ 的形式作为 inputs, 以及它们的 immediate hidden representations 2d tensors $\mathbf{H}$ 构成的 MLP. 其中 $\mathbf{X}$ and $\mathbf{H}$ 大小相同.

Let that sink in. We now conceive of not only the inputs but also the hidden representations as possessing spatial structure.

我们现在使用 $[\mathbf{X}]_{i,j}$ and $[\mathbf{H}]_{i,j}$ 来分别表示 input image 中 $(i, j)$ 位置处的 pixel 以及它的 hidden representation.

因此，为了让每个 hidden unit 接收来自每个 input pixel 的 input, 我们将从使用weight matrix 转为用 **4th-order weight tensors** W 来表示我们的参数.

用 $\mathbf{U}$ 来表示 biases, 我们可以把这个 fully-connected layer 表达为:

$$[\mathbf{H}]_{i,j} = [\mathbf{U}]_{i,j} + \sum_k \sum_l [\mathsf{W}]_{i,j,k,l} [\mathbf{X}]_{k,l}$$

这是我们熟悉的格式. $[\mathsf{W}]_{i,j,k,l}$ 表示的是从 $[\mathbf{X}]_{k,l}$ 在 $[\mathbf{H}]_{i,j}$ 中的权重. 遍历所有 $k$ 和 $l$, 我们就得到了 $[\mathbf{H}]_{i,j}$ 的值.

但是现在我们换一种思路: 我们对于每个 $(i, j)$ 对, re-index the subscripts $(k, l)$ such that $k = i + a$ and $l = j + b$, 得到一对 $(a, b)$, 其中 $a = k - i, b = l - j$.

于是我们就把 $[\mathsf{W}]_{i,j,k,l}$ 转成了 $[\mathsf{W}]_{i,j,i+a,j+b}$, 我们把它重新写成 $[\mathsf{V}]_{i,j,a,b} = [\mathsf{W}]_{i,j,i+a,j+b}$.

于是表达式变为:

$$\begin{aligned}[\mathbf{H}]_{i,j} &= [\mathbf{U}]_{i,j} + \sum_k \sum_l [\mathsf{W}]_{i,j,k,l} [\mathbf{X}]_{k,l} \\ &= [\mathbf{U}]_{i,j} + \sum_a \sum_b [\mathsf{V}]_{i,j,a,b} [\mathbf{X}]_{i+a,j+b}\end{aligned}$$

where the switch from W to V is entirely cosmetic for now since there is a one-to-one correspondence between coefficients in both fourth-order tensors.

### (1)Translation Invariance: 由于平移不变性, $[\mathsf{V}]_{i,j,a,b} = [\mathbf{V}]_{a,b}$, 无关于 $i, j$

Now let us invoke the first principle established above: translation invariance. This implies that a shift in the input $\mathbf{X}$ should simply lead to a shift in the hidden representation $\mathbf{H}$. This is only possible if V and $\mathbf{U}$ do not actually depend on $(i, j)$, i.e., we have $[\mathsf{V}]_{i,j,a,b} = [\mathbf{V}]_{a,b}$ and $\mathbf{U}$ is a constant, say $u$. As a result, we can simplify the definition for $\mathbf{H}$:

$$[\mathbf{H}]_{i,j} = u + \sum_a \sum_b [\mathbf{V}]_{a,b} [\mathbf{X}]_{i+a,j+b}.$$

translation Invariance 使得这个 layer 从 MLP (fully-connected) 变为了一个 *convolution*(严谨称呼其实是cross-correlation)

We are effectively weighting pixels at $(i + a, j + b)$ in the vicinity of location $(i, j)$ with coefficients $[\mathbf{V}]_{a,b}$ to obtain the value $[\mathbf{H}]_{i,j}$. Note that $[\mathbf{V}]_{a,b}$ needs many fewer coefficients than $[\mathsf{V}]_{i,j,a,b}$ since it no longer depends on the location within the image. We have made significant progress!

## (3) Locality: (a, b) 不需要遍历全 image, 而是在某个 Δ 范围内

Now let us invoke the second principle: locality. As motivated above, we believe that we should not have to look very far away from location $(i, j)$ in order to glean relevant information to assess what is going on at $[\mathbf{H}]_{i,j}$. This means that outside some range $|a| > \Delta$ or $|b| > \Delta$, we should set $[\mathbf{V}]_{a,b} = 0$. Equivalently, we can rewrite $[\mathbf{H}]_{i,j}$ as

$$[\mathbf{H}]_{i,j} = u + \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} [\mathbf{V}]_{a,b} [\mathbf{X}]_{i+a,j+b}.$$

实际意义就是: 如果我们要检索一个猫的图像, 那么我们大概在 25x25 的 image 中 设置一个 5x5 的 kernel 就可以了, 这样我们只需要 5x5 的 weight tensor (如果先忽略 channel 的话).

# ConvoLutional Layer 基本构建完成

$$[\mathbf{H}]_{i,j} = u + \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} [\mathbf{V}]_{a,b} [\mathbf{X}]_{i+a,j+b}.$$

于是到此, 我们基本就 build 完了我们的 *convolutional layer*.

CNN 就是一种包含 *convolutional layer* 的 special family of NNs.

这里的 $\mathbf{V}$ 被称为 *convolution kernel(卷积核)*, 也被称为一个 *filter(滤波器)*, 或者简单地说就是 the layer's *weights*.

当 kernel 较小时，与 fully-connected layer 相比, 差异会非常明显. 以前，我们可能需要数十亿个参数来表示图像处理网络中的单层，而现在，我们通常只需要几百个参数，而且不会改变输入或隐藏表示的维度。

参数大大幅度减少的代价是: 现在我们只有 translation invariant 的 features, 且我们的 layer 现在只处理了 local information. 所有的 learning 都是依赖于我们的 inductive bias 的. 如果 bias 和现实符合, 那么 When that bias agrees with reality, 那么我们就会得到有效的模型. 反之则无法. 比如说, 如果 images 并不是 translation invariant 的, 那么我们的模型甚至不会 fit our training data.

## 什么是一个 ConVolution

Before going further, we should briefly review why the above operation is called a convolution. In mathematics, the *convolution* between two functions, say $f, g : \mathbb{R}^d \to \mathbb{R}$ is defined as

$$(f * g)(\mathbf{x}) = \int f(\mathbf{z}) g(\mathbf{x} - \mathbf{z}) d\mathbf{z}.$$

That is, we measure the overlap between $f$ and $g$ when one function is "flipped" and shifted by $\mathbf{x}$. Whenever we have discrete objects, the integral turns into a sum. For instance, for vectors from the set of square summable infinite dimensional vectors with index running over $\mathbb{Z}$ we obtain the following definition:

$$(f * g)(i) = \sum_a f(a) g(i - a).$$

For two-dimensional tensors, we have a corresponding sum with indices $(a, b)$ for $f$ and $(i - a, j - b)$ for $g$, respectively:

$$(f * g)(i, j) = \sum_a \sum_b f(a, b) g(i - a, j - b).$$

This looks similar to what we had in our convolution layer, with one major difference. Rather than using $(i + a, j + b)$, we are using the difference instead. Our original definition in more properly describes a *cross-correlation*. We will come back to this in the following section.

## Channels: 考虑到 RGB 三个色道, kernel V 中需要再加一对 channel 与 channel 的坐标对应, 于是变为 4d tensor.

There is just one problem with this approach. So far, we blissfully ignored that images consist of 3 channels: red, green, and blue. In reality, images are not two-dimensional objects but rather third-order tensors, characterized by a height, width, and channel, e.g., with shape $1024 \times 1024 \times 3$ pixels. While the first two of these axes concern spatial relationships, the third can be regarded as

assigning a multidimensional representation to each pixel location. We thus index $\mathsf{X}$ as $[\mathsf{X}]_{i,j,k}$. The convolutional filter has to adapt accordingly. Instead of $[\mathbf{V}]_{a,b}$, we now have $[\mathsf{V}]_{a,b,c}$.

Moreover, just as our input consists of a third-order tensor, it turns out to be a good idea to similarly formulate our hidden representations as third-order tensors $\mathsf{H}$. In other words, rather than just having a single hidden representation corresponding to each spatial location, we want an entire vector of hidden representations corresponding to each spatial location. We could think of the hidden representations as comprising a number of two-dimensional grids stacked on top of each other. As in the inputs, these are sometimes called *channels*. They are also sometimes called *feature maps*, as each provides a spatialized set of learned features to the subsequent layer. Intuitively, you might imagine that at lower layers that are closer to inputs, some channels could become specialized to recognize edges while others could recognize textures.

To support multiple channels in both inputs ($\mathsf{X}$) and hidden representations ($\mathsf{H}$), we can add a fourth coordinate to $\mathsf{V}$: $[\mathsf{V}]_{a,b,c,d}$. Putting everything together we have:

$$[\mathsf{H}]_{i,j,d} = \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} \sum_{c} [\mathsf{V}]_{a,b,c,d} [\mathsf{X}]_{i+a,j+b,c},$$

where $d$ indexes the output channels in the hidden representations $\mathsf{H}$. The subsequent convolutional layer will go on to take a third-order tensor, $\mathsf{H}$, as the input. Being more general, this is the definition of a convolutional layer for multiple channels, where $\mathsf{V}$ is a kernel or filter of the layer.

There are still many operations that we need to address. For instance, we need to figure out how to combine all the hidden representations to a single output. We also need to decide how to compute things efficiently, how to combine multiple layers, appropriate activation functions, and how to make reasonable design choices to yield networks that are effective in practice. We turn to these issues in later lectures.

## Summary

- Translation invariance in images implies that all patches of an image will be treated in the same manner.
- Locality means that only a small neighborhood of pixels will be used to compute the corresponding hidden representations.
- In image processing, convolutional layers typically require many fewer parameters than fully-connected layers.
- CNNS are a special family of neural networks that contain convolutional layers.
- Channels on input and output allow our model to capture multiple aspects of an image at each spatial location.