## The convolutional operation

- Convolutional operation can be seen as a summation/integral of reweighting.
- Cross-correlation

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m, j+n) \cdot K(m,n)$$

## Motivation

- Sparse interactions

Here "sparse" means that some connections lost in neural networks, for deep networks, the scale of input's effect shrinks, but output remains affected. Cause only some direct connections failed.

- Parameter sharing

A kernel can be used all the spaces within one image.

- Equivariance to transition.

The output moves the same scale with the input moves.

## Pooling

- Three steps of convolutional neural networks
  - Convolutional Operational stage
  - Detector stage (often some non-linear functions used here ⇒ "Relu")
  - Pooling stage
- Pooling as a way of down-sampling, under assumption that small changes will not affect the output. (down-sampling: De-Noising)
- Pooling can also be regarded a way of adjusting the output dimension in hidden layer.
- Pooling (max pooling) can be seen as max-out operation, stable to the variant.

## Convolution and Pooling as an Infinitely Strong Prior

- The weights are partly equal to each other.
- The input and output have property of equivariant.
- Some weights are zero ( the connection is sparse )

## Variants of the Basic Convolution Function

- Convolution with input with channels.

$$Z_{i,j,k} = \sum_{l,m,n} V_{l,j+m-1,k+n-1} K_{i,l,m,n}$$

$Z_{i,j,k}$: output channels; $V_{l,j+m-1,k+n-1}$: input elements; $K_{i,l,m,n}$: kernel elements; $i$ is the kernel index

- Convolution with strides

$$Z_{i,j,k} = \sum_{l,m,n} \left[ V_{l,(j-1)\times s+m,(k-1)\times s+n} K_{i,l,m,n} \right]$$

Here $s$ is the stride size.

- Convolution with paddings ⇒ Adding zeros out the edge of input / feature map

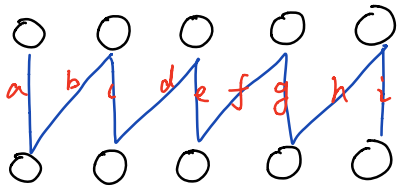
Valid convolution

Each layer contains same elements → Same convolution

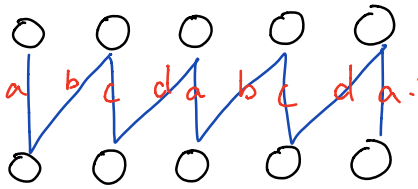
where the best model often locates

Each pixel equally operated → Full convolution

- Unshared convolution / Locally connected layer




where $a \neq b \neq c \neq d \neq e \neq f \neq g \neq h \neq i$




Here neighbour pixels are not shared with weights, but we use the weights kindly.

- Computing Gradients of Convolutional Neural Networks.

$$g(G,V,s)_{i,j,k,l} = \frac{\partial}{\partial K_{i,j,k,l}} J(V,K) = \sum_{m,n} G_{i,m,n} V_{j,(m-1)\times s+k,(n-1)\times s+l}$$

$g(G,V,s)_{i,j,k,l}$: gradient for one element in conv.; $V$: Input; chain rule.

**Structured Outputs**

Some discussion of the outputs of CNN: one single judgement, a class distribution over feature map or combining with RNN for continuously refinement on the convolutions.

**Data Types:** Different dimensions with channels corresponding to different datas.

**Random or Unsupervised features**: Unsupervised pretraining for regularization and computation cost saving.

**Neuroscientific Basis for convolutional networks**