Convolutional Neural Networks (CNN)

Dr. Josif Grabocka

ISMLL, University of Hildesheim

Convolutional Neural Networks

The Convolution Operation

- Generally speaking, convolution is an operation on two functions:

$$s(t) = \int x(a)\, w(t-a) da$$

- Often denoted with an asterisk:

$$s(t) = (x * w)\,(t)$$

- Example:
  - $x(t)$: a noisy measure the position of a spaceship
  - $w(a)$: relevance of a measurement with age a (Note: $\int w(a) da = 1$)
  - Given a sequence of noisy measurements $x(t), x(t-1), \ldots, x(t-\infty)$, what is the relevance-corrected position $s(t)$?

Convolutions in Deep Learning

- Terminology: $x(t)$: **Input**, $w(a)$: **Kernel/Filter**, $s(t)$: **Feature Map**
- Assuming two-dimensional images I and kernels K:

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

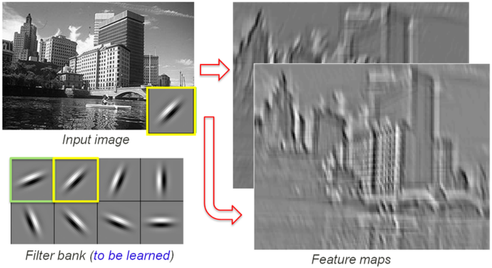
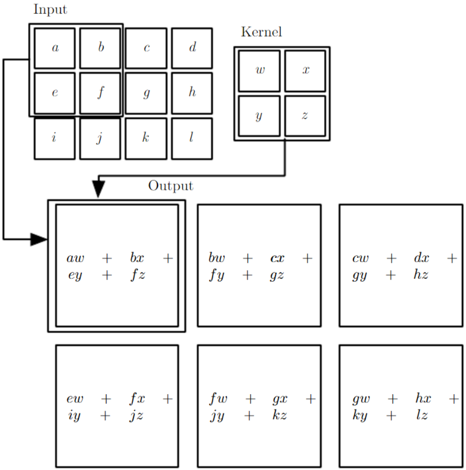


Figure 1: I top left, K bottom-left (green, yellow), S right; Credits: S. Lazebnik

2-D convolution (Source: Goodfellow et al., 2016)



Note: Kernel is **shared** and applied only in **valid** regions

Sparse Interactions/Connectivity

- Small kernels detect meaningful features (e.g. edges)
- Reduces memory footprint and computations
- m inputs, n outputs, kernel of size k reduce the activation complexity from $\mathcal{O}(n \times m)$ to $\mathcal{O}(n \times k)$, for $k << m$

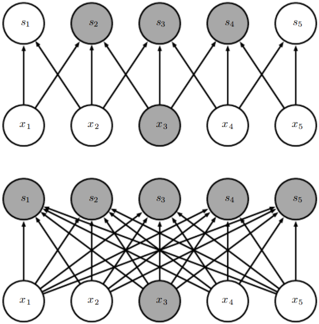


Figure 2: Sparse connectivity; the influence of x_3? (Source: Goodfellow et al., 2016)

Multi-layered Sparse Connectivity

- Stacked convolutions interact with larger portions of the input
- Capture interactions through sparse connections

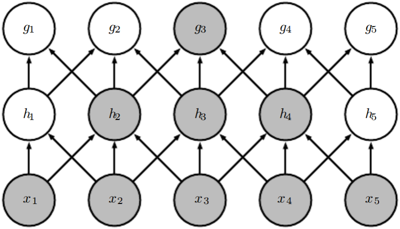


Figure 3: Stacked convolution (Source: Goodfellow et al., 2016)

- Illustration above:
  - 2-layer convolution: 26 ops. and 6 weights (assume per-layer kernel)
  - 2-layer full-nn: 50 ops. and 50 weights

Parameter Sharing or Tied-Weights

- In a convolutional setup weights are **tied/shared**:

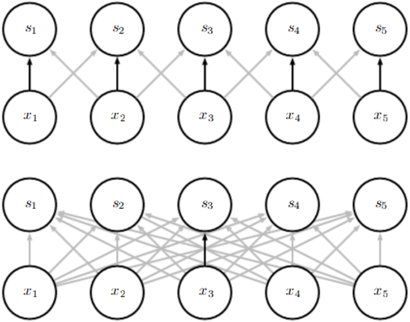


Figure 4: Black arrow represents the same weight (Source: Goodfellow et al., 2016)

- Achieves a translation-invariant capturing of patterns

Stacked Convolutions with Nonlinear Activations

- Input image: $V^{(0)} \in \mathbb{R}^{I^{(0)} \times X \times Y}$
- Kernels: $K^{(1)} \in \mathbb{R}^{I^{(1)} \times C \times M^{(1)} \times N^{(1)}}, \ldots, K^{(\ell)} \in \mathbb{R}^{I^{(\ell)} \times I^{(\ell-1)} \times M^{(\ell)} \times N^{(\ell)}}$
- Feature Maps: $V^{(1)} \in \mathbb{R}^{I^{(1)} \times X \times Y}, \ldots, V^{(\ell)} \in \mathbb{R}^{I^{(\ell)} \times X \times Y}$

$$
\begin{aligned}
Z^{(1)}_{i,x,y} &= \sum_{c=1}^{I^{(0)}} \sum_{m=1}^{M^{(1)}} \sum_{n=1}^{N^{(1)}} V^{(0)}_{c,x+m-1,y+n-1} K^{(1)}_{i,c,m,n} \\
V^{(1)}_{i,x,y} &= f\left(Z^{(1)}_{i,x,y}\right), \quad \text{e.g. } f(x) = \max(0, x) \\
&\vdots \\
Z^{(\ell)}_{i,x,y} &= \sum_{c=1}^{I^{(\ell-1)}} \sum_{m=1}^{M^{(\ell)}} \sum_{n=1}^{N^{(\ell)}} V^{(\ell-1)}_{c,x+m-1,y+n-1} K^{(\ell)}_{i,c,m,n} \\
V^{(\ell)}_{i,x,y} &= f\left(Z^{(\ell)}_{i,x,y}\right)
\end{aligned}
$$

Nonlinear Activation of Feature Maps

$$V_{i,x,y}^{(\ell)} = \max\left(0, \sum_{c=1}^{I^{(\ell-1)}} \sum_{m=1}^{M^{(\ell)}} \sum_{n=1}^{N^{(\ell)}} V_{c,x+m-1,y+n-1}^{(\ell-1)} K_{i,c,m,n}^{(\ell)}\right)$$

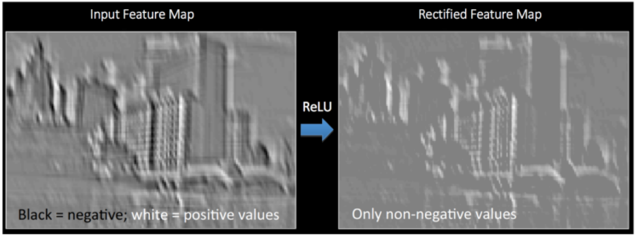


Figure 5: Rectified Feature Map, Credits: Rob Fergus

Strided Convolutions

- Convolve with every s-th position in each dimension:

$$Z^{(\ell)}_{i,x,y} = \sum_{c=1}^{I^{(\ell-1)}} \sum_{m=1}^{M^{(\ell)}} \sum_{n=1}^{N^{(\ell)}} V^{(\ell-1)}_{c,(x-1)s+m,(y-1)s+n} K^{(\ell)}_{i,c,m,n}$$

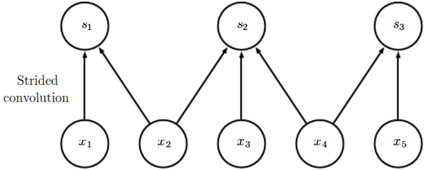


Figure 6: Strided convolutions $s = 2$, (Source: Goodfellow et al., 2016)

Zero Padding - Avoid Size Shrinking

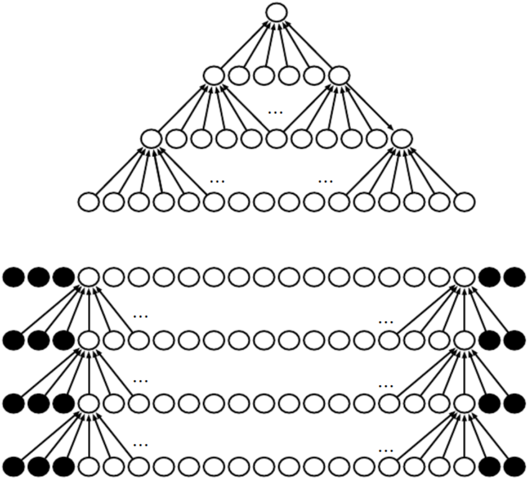


Figure 7: Top: No padding, Bottom: padding (Source: Goodfellow et al., 2016)

Pooling

- A convolutional network has three stages:
  1. **Convolutions** (in parallel) for multiple kernels
  2. **Nonlinear activations** of the convolutions (ReLU)
  3. **Pooling** (summary statistics)

- Reduces the dimensionality of the latent representation
- Ensure invariance to small translations of the input

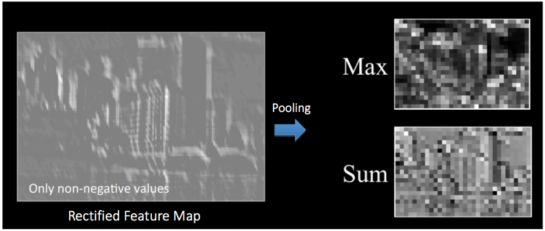


Figure 8: Max and Avg Pooling, Credits: Rob Fergus

Pooling - Translation Invariance Illustration

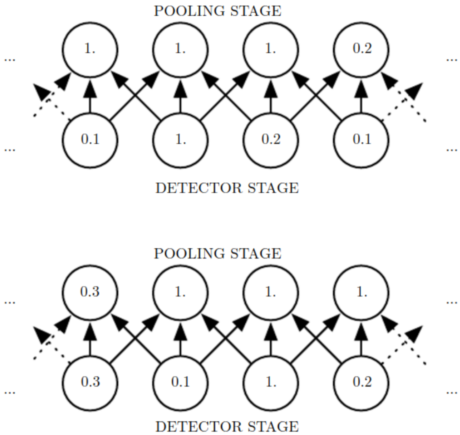


Figure 9: Shifting the input one pixel to the right has smaller effect on the pooling layer, compared to the detector layer (conv+nonlinearity). (Source: Goodfellow et al., 2016)

Pooling - Down-sampling and Strides

Pooling (**max** or **avg**) squared regions of size $\phi \times \phi$ with a stride s:

$$V_{i,x,y}^{(\ell,Pooled)} := \underset{\substack{\tilde{x}\in\{(x-1)s+1,\ldots,(x-1)s+\phi\} \\ \tilde{y}\in\{(y-1)s+1,\ldots,(y-1)s+\phi\}}}{\text{Pooling}^{(\ell)}} V_{i,\tilde{x},\tilde{y}}^{(\ell)}$$

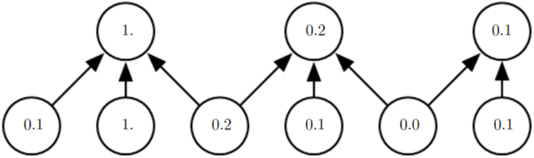
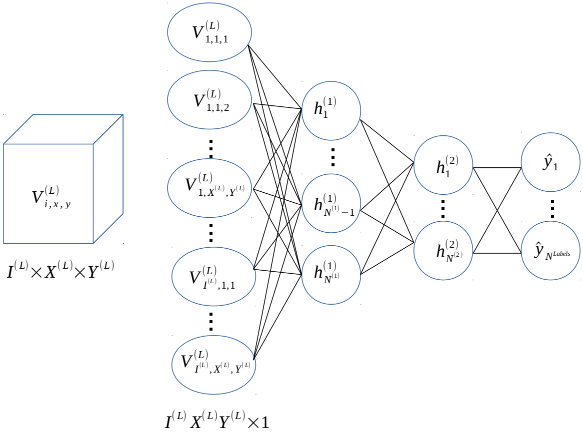


Figure 10: Max-pooling with $s = 2$, $\phi = 3$ (Source: Goodfellow et al., 2016)

Note: For simplicity, we assume $s = \phi$ in the following slides!

Reshaping and Fully Connected Layers



Remember $\frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial V^{(L)}_{i,x,y}} = \sum_i \frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial h^{(1)}_i} \frac{\partial h^{(1)}_i}{\partial V^{(L)}_{i,x,y}}$

CNN - Forward Prediction

Algorithm 1: Convolutional Neural Network

$$\{\text{Convolutional steps: } L^{\text{Conv}} \times \{\text{Convolution, Nonlinear, Pooling}\}\}$$

for $\ell = 1, \ldots, L^{\text{Conv}}$ **do**

$$Z^{(\ell)}_{i,x,y} := \sum_{c=1}^{I^{(\ell-1)}} \sum_{m=1}^{M^{(\ell)}} \sum_{n=1}^{N^{(\ell)}} V^{((\ell-1),Pool)}_{c,(x-1)s^{(\ell,\text{Conv})}+m,(y-1)s^{(\ell,\text{Conv})}+n} \; K^{(\ell)}_{i,c,m,n}$$

$$V^{(\ell)}_{i,x,y} := f^{(\ell)}(Z^{(\ell)}_{i,x,y})$$

$$V^{(\ell,Pool)}_{i,x,y} := \underset{\substack{\tilde{x} \in \left\{(x-1)s^{(\ell,\text{Pool})}+1,\ldots,(x-1)s^{(\ell,\text{Pool})}+\phi^{(\ell)}\right\} \\ \tilde{y} \in \left\{(y-1)s^{(\ell,\text{Pool})}+1,\ldots,(y-1)s^{(\ell,\text{Pool})}+\phi^{(\ell)}\right\}}}{\text{Pooling}^{(\ell)}} V^{(\ell)}_{i,\tilde{x},\tilde{y}}$$

$$\{\text{Fully connected layers}\}$$

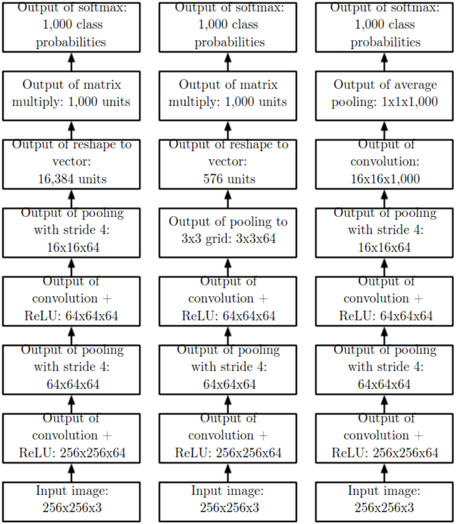
$$h^{(0)} := \left[V^{(L^{\text{Conv}})}_{1,1,1}, \ldots, V^{(L^{\text{Conv}})}_{I^{(L)},X^{(L)},Y^{(L)}} \right]$$

for $\ell = 1, \ldots, L^{\text{Full}}$ **do**

$$h^{(\ell)}_{:} = f^{(\ell)}(W^{(\ell)}_{:,:} h^{(\ell-1)}_{:} + b^{(\ell)}_{:})$$

return $\hat{\mathcal{Y}} := h^{L^{\text{Full}}}_{:}$

Example Architectures (Source: Goodfellow et al., 2016)



Gradients

- Convolutions and pooling are computational graph nodes
- Apply the standard back-propagation for computational graphs
- Remember a convolution:

$$Z^{(\ell)}_{i,x,y} = \sum_{c=1}^{I^{(\ell-1)}} \sum_{m=1}^{M^{(\ell)}} \sum_{n=1}^{N^{(\ell)}} V^{(\ell-1)}_{c,(x-1)s+m,(y-1)s+n} K^{(\ell)}_{i,c,m,n}$$

- Given $\frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial V^{(\ell)}_{i,x,y}}$, define $G^{(\ell)}_{i,x,y} := \frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial Z^{(\ell)}_{i,x,y}} = \frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial V^{(\ell)}_{i,x,y}} \left(f^{(\ell)\prime}(Z^{(\ell)}_{i,x,y}) \right)$
- Yielding:

$$\frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial K^{(\ell)}_{i,c,m,n}} = \sum_{x=1}^{X^{(\ell)}} \sum_{y=1}^{Y^{(\ell)}} G^{(\ell)}_{i,x,y} V^{(\ell-1)}_{c,(x-1)s+m,(y-1)s+n}$$

Gradients (II)

- We need the gradient w.r.t. $V^{(\ell-1)}_{:,:,:}$ to propagate the error down:

$$\frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial V^{(\ell-1)}_{i,x,y}} = \sum_{\substack{x',p \\ (x'-1)s+p=x}} \sum_{\substack{y',q \\ (y'-1)s+q=y}} \sum_{c=1}^{I^{(\ell)}} K^{(\ell)}_{c,i,p,q} G^{(\ell)}_{c,x',y'}$$

- Gradients of pooling are simpler, e.g. for max layer and $s = \phi$:

$$\frac{\partial V^{(\ell,Pooled)}_{i,x,y}}{\partial V^{(\ell)}_{i,\tilde{x},\tilde{y}}} = \begin{cases} 1 & \text{if } (\tilde{x},\tilde{y}) = \underset{\substack{\tilde{x}^*\in\{(x-1)s+1,\ldots,(x-1)s+\phi\} \\ \tilde{y}^*\in\{(y-1)s+1,\ldots,(y-1)s+\phi\}}}{\operatorname{argmax}} V^{(\ell)}_{i,\tilde{x}^*,\tilde{y}^*} \\ 0 & \text{else} \end{cases}$$

CNN - Back-Propagation

Algorithm 2: CNN's parameters gradients (Only Convolutional Layers)

$$\frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial V^{(L^{\text{Conv}},Pool)}_{i,x,y}} = \sum_i \frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial h^{(1)}_i} \frac{\partial h^{(1)}_i}{\partial V^{(L^{\text{Conv}},Pool)}_{i,x,y}}$$

for $\ell = L^{\text{Conv}}, \ldots, 1$ **do**

$$\frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial V^{(\ell)}_{i,\tilde{x},\tilde{y}}} = \frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial V^{(\ell,Pool)}_{i,x,y}} \times \begin{cases} 1 & (\tilde{x},\tilde{y}) = \underset{\substack{\tilde{x}^* \in \{(x-1)s+1,\ldots,(x-1)s+\phi\} \\ \tilde{y}^* \in \{(y-1)s+1,\ldots,(y-1)s+\phi\}}}{\text{argmax}} V^{(\ell)}_{i,\tilde{x}^*,\tilde{y}^*} \\ 0 & \text{else} \end{cases}$$

$$G^{(\ell)}_{i,\tilde{x},\tilde{y}} := \frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial V^{(\ell)}_{i,\tilde{x},\tilde{y}}} \left(f^{(\ell)\prime}(Z^{(\ell)}_{i,\tilde{x},\tilde{y}}) \right)$$

$$\frac{\partial \mathcal{L}(y,\hat{y})}{\partial K^{(\ell)}_{i,c,m,n}} = \sum_{\tilde{x}=1}^{X^{(\ell)}} \sum_{\tilde{y}=1}^{Y^{(\ell)}} G^{(\ell)}_{i,\tilde{x},\tilde{y}} V^{(\ell-1,Pool)}_{c,(\tilde{x}-1)s+m,(\tilde{y}-1)s+n}$$

$$\frac{\partial \mathcal{L}(\mathcal{Y},\hat{\mathcal{Y}})}{\partial V^{(\ell-1,Pool)}_{i,\tilde{x},\tilde{y}}} = \sum_{\substack{x',p \\ (x'-1)s+p=\tilde{x}}} \sum_{\substack{y',q \\ (y'-1)s+q=\tilde{y}}} \sum_{c=1}^{I^{(\ell)}} K^{(\ell)}_{c,i,p,q} G^{(\ell)}_{c,x',y'}$$

return $\frac{\partial \mathcal{L}(y,\hat{y})}{\partial K}$