Convolutional Networks 2: Training, deep convolutional networks

Hakan Bilen

Machine Learning Practical — MLP Lecture 8
30 October / 6 November 2018
Q1. How can we increase the receptive field area of a conv layer?
Q1. How can we increase the receptive field area of a conv layer?
Q2. Can we do it without increasing kernel size?
Input arguments for convolution function

```python
class torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=1, groups=1, bias=True)
```

Applies a 2D convolution over an input signal composed of several input planes.

- `in_channels`
- `out_channels`
- `kernel_size`
- `stride`
- `padding`
- `bias`
Input arguments for convolution function

```python
class torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=1, groups=1,
bias=True) [source]
```

Applies a 2D convolution over an input signal composed of several input planes.

- `in_channels`
- `out_channels`
- `kernel_size`
- `stride`
- `padding`
- `bias`
- `dilation?`
- `groups?`
Dilated convolutions

Increased receptive field by inflating the kernel by inserting $D - 1$ spaces between the kernel elements.
Dilated convolutions

Increased receptive field by inflating the kernel by inserting $D - 1$ spaces between the kernel elements.
Dilated convolutions

Increased receptive field by **inflating** the kernel by inserting $D - 1$ spaces between the kernel elements.
Increased receptive field by **inflating** the kernel by inserting $D - 1$ spaces between the kernel elements.

Why to increase receptive field size?
Dilated convolutions

Increased receptive field by **inflating** the kernel by inserting $D - 1$ spaces between the kernel elements.


(Convolutional) filter groups

Feature map $H^{L-1}$  Conv kernels  Feature map $H^L$
(Convolutional) filter groups

- $G$ is the number of groups
- Reduces the number of convolutional filters (or parameters)
- Regularisation effect
Convolution and cross-correlation

- We can write the feature map hidden unit equation (Index-0):

\[ h_{i,j} = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} l(m + i, n + j) W(m, n) \]

\[ h = W \otimes l \]

\( \otimes \) is a cross-correlation
Convolution and cross-correlation

- We can write the feature map hidden unit equation (Index-0):

\[
h_{i,j} = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} I(m+i, n+j)W(m, n)
\]

\[
h = W \otimes I
\]

\( \otimes \) is a cross-correlation.

- In signal processing a 2D convolution is written as

\[
h_{i,j} = (V \ast I) = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} I(m, n)V(i-m, j-n)
\]

\[
h_{i,j} = (I \ast V) = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} I(i-m, j-n)V(m, n)
\]

- If we “flip” (reflect horizontally and vertically) \( W \) (cross-correlation) then we obtain \( V \) (convolution).
Training Convolutional Networks

Forward pass

\[ h^l = f^l(h^{l-1}, \{w^l, b^l\}) \]

\[ h^{l-1} \rightarrow h^l \]

\[ \{w^l, b^l\} \]
Training Convolutional Networks

Forward pass

\[ h^l = f^l(h^{l-1}, \{w^l, b^l\}) \]

\[ h^{l-1} \rightarrow h^l \]

\{w^l, b^l\}

Backward pass

\[ \frac{\partial E}{\partial h^{l-1}} = \frac{\partial E}{\partial h^l} \frac{\partial h^l}{\partial h^{l-1}} \]

\[ \frac{\partial E}{\partial w^l} = \frac{\partial E}{\partial h^l} \frac{\partial h^l}{\partial w^l} \]
Example

$h_1^{l-1} = w_{11}^l h_1^{l-1} + w_{12}^l h_{12}^{l-1} + w_{21}^l h_{21}^{l-1} + w_{22}^l h_{22}^{l-1} + b$

$h_{12}^l = w_{12}^l h_{12}^{l-1} + w_{12}^l h_{13}^{l-1} + w_{21}^l h_{21}^{l-1} + w_{22}^l h_{23}^{l-1} + b$

$h_{21}^l = w_{11}^l h_{21}^{l-1} + w_{12}^l h_{22}^{l-1} + w_{21}^l h_{31}^{l-1} + w_{22}^l h_{32}^{l-1} + b$

$h_{22}^l = w_{11}^l h_{22}^{l-1} + w_{12}^l h_{23}^{l-1} + w_{21}^l h_{32}^{l-1} + w_{22}^l h_{33}^{l-1} + b$
Gradients of $E$ w.r.t $W^l$
Let's calculate the parameter updates ($\frac{\partial E}{\partial W^l}$)

$$
\frac{\partial E}{\partial w^l_{11}} = \frac{\partial E}{\partial h^l_{11}} \frac{\partial h^l_{11}}{\partial w^l_{11}} + \frac{\partial E}{\partial h^l_{12}} \frac{\partial h^l_{12}}{\partial w^l_{11}} + \frac{\partial E}{\partial h^l_{21}} \frac{\partial h^l_{21}}{\partial w^l_{11}} + \frac{\partial E}{\partial h^l_{22}} \frac{\partial h^l_{22}}{\partial w^l_{11}}
$$
Let’s calculate the parameter updates ($\frac{\partial E}{\partial W_l}$)

$$\frac{\partial E}{\partial w^l_{11}} = \frac{\partial E}{\partial h^{l-1}_{11}} h^{l-1}_{11} + \frac{\partial E}{\partial h^{l-1}_{12}} h^{l-1}_{12} + \frac{\partial E}{\partial h^{l-1}_{21}} h^{l-1}_{21} + \frac{\partial E}{\partial h^{l-1}_{22}} h^{l-1}_{22}$$
Example

\[ h_{11}^l = w_{11}^l h_{11}^{l-1} + w_{12}^l h_{12}^{l-1} + w_{21}^l h_{21}^{l-1} + w_{22}^l h_{22}^{l-1} + b \]
\[ h_{12}^l = w_{11}^l h_{12}^{l-1} + w_{12}^l h_{13}^{l-1} + w_{21}^l h_{22}^{l-1} + w_{22}^l h_{23}^{l-1} + b \]
\[ h_{21}^l = w_{11}^l h_{21}^{l-1} + w_{12}^l h_{22}^{l-1} + w_{21}^l h_{31}^{l-1} + w_{22}^l h_{32}^{l-1} + b \]
\[ h_{22}^l = w_{11}^l h_{22}^{l-1} + w_{12}^l h_{23}^{l-1} + w_{21}^l h_{32}^{l-1} + w_{22}^l h_{33}^{l-1} + b \]

Let’s calculate the parameter updates \( \frac{\partial E}{\partial W^l} \)

\[ \frac{\partial E}{\partial w_{12}^l} = \frac{\partial E}{\partial h_{11}^l} h_{12}^{l-1} + \frac{\partial E}{\partial h_{12}^l} h_{13}^{l-1} + \frac{\partial E}{\partial h_{21}^l} h_{22}^{l-1} + \frac{\partial E}{\partial h_{22}^l} h_{23}^{l-1} \]
Let's calculate the parameter updates \( \frac{\partial E}{\partial W^l} \)

\[
\frac{\partial E}{\partial w_{2,1}^l} = \frac{\partial E}{\partial h_{11}^l} h_{21}^{l-1} + \frac{\partial E}{\partial h_{12}^l} h_{22}^{l-1} + \frac{\partial E}{\partial h_{21}^l} h_{31}^{l-1} + \frac{\partial E}{\partial h_{22}^l} h_{32}^{l-1}
\]
Example

\[ h_{11}^l = w_{11}^l h_{11}^{l-1} + w_{12}^l h_{12}^{l-1} + w_{21}^l h_{21}^{l-1} + w_{22}^l h_{22}^{l-1} + b \]
\[ h_{12}^l = w_{11}^l h_{12}^{l-1} + w_{12}^l h_{13}^{l-1} + w_{21}^l h_{22}^{l-1} + w_{22}^l h_{23}^{l-1} + b \]
\[ h_{21}^l = w_{11}^l h_{21}^{l-1} + w_{12}^l h_{22}^{l-1} + w_{21}^l h_{31}^{l-1} + w_{22}^l h_{32}^{l-1} + b \]
\[ h_{22}^l = w_{11}^l h_{22}^{l-1} + w_{12}^l h_{23}^{l-1} + w_{21}^l h_{32}^{l-1} + w_{22}^l h_{33}^{l-1} + b \]

Let’s calculate the parameter updates \( \left( \frac{\partial E}{\partial W^l} \right) \)

\[ \frac{\partial E}{\partial w_{2,2}^l} = \frac{\partial E}{\partial h_{11}^l} h_{22}^{l-1} + \frac{\partial E}{\partial h_{12}^l} h_{33}^{l-1} + \frac{\partial E}{\partial h_{21}^l} h_{32}^{l-1} + \frac{\partial E}{\partial h_{22}^l} h_{33}^{l-1} \]
Gradients of $E$ w.r.t $W^l$

\[
\frac{\partial E}{\partial w^l_{1,1}} = \frac{\partial E}{\partial h^l_{11}} h^{l-1}_{11} + \frac{\partial E}{\partial h^l_{12}} h^{l-1}_{12} + \frac{\partial E}{\partial h^l_{21}} h^{l-1}_{21} + \frac{\partial E}{\partial h^l_{22}} h^{l-1}_{22}
\]

\[
\frac{\partial E}{\partial w^l_{12}} = \frac{\partial E}{\partial h^l_{11}} h^{l-1}_{12} + \frac{\partial E}{\partial h^l_{12}} h^{l-1}_{13} + \frac{\partial E}{\partial h^l_{21}} h^{l-1}_{22} + \frac{\partial E}{\partial h^l_{22}} h^{l-1}_{23}
\]

\[
\frac{\partial E}{\partial w^l_{21}} = \frac{\partial E}{\partial h^l_{11}} h^{l-1}_{21} + \frac{\partial E}{\partial h^l_{12}} h^{l-1}_{22} + \frac{\partial E}{\partial h^l_{21}} h^{l-1}_{31} + \frac{\partial E}{\partial h^l_{22}} h^{l-1}_{32}
\]

\[
\frac{\partial E}{\partial w^l_{22}} = \frac{\partial E}{\partial h^l_{11}} h^{l-1}_{22} + \frac{\partial E}{\partial h^l_{12}} h^{l-1}_{33} + \frac{\partial E}{\partial h^l_{21}} h^{l-1}_{32} + \frac{\partial E}{\partial h^l_{22}} h^{l-1}_{33}
\]

Given $H^l \in \mathcal{R}^{M^l \times N^l}$

\[
\frac{\partial E}{\partial w^l_{r,s}} = \sum_{m=1}^{M^l} \sum_{n=1}^{N^l} \frac{\partial E}{\partial h^l_{m,n}} h^{l-1}_{r+m-1,s+n-1}
\]
Gradients of $E$ w.r.t $W^l$

\[
\frac{\partial E}{\partial w_{r,s}^l} = \sum_{m=1}^{M^l} \sum_{n=1}^{N^l} \frac{\partial E}{\partial h_{m,n}^l} h_{r+m-1,s+n-1}^{l-1}
\]

\[
\begin{array}{ccc}
  h_{11}^{l-1} & h_{12}^{l-1} & h_{13}^{l-1} \\
  h_{21}^{l-1} & h_{22}^{l-1} & h_{23}^{l-1} \\
  h_{31}^{l-1} & h_{32}^{l-1} & h_{33}^{l-1} \\
\end{array}
\quad
\begin{array}{cc}
  \frac{\partial E}{\partial h_{11}^l} & \frac{\partial E}{\partial h_{12}^l} \\
  \frac{\partial E}{\partial h_{21}^l} & \frac{\partial E}{\partial h_{22}^l} \\
\end{array}
= \begin{array}{cc}
  \frac{\partial E}{\partial w_{11}^l} & \frac{\partial E}{\partial w_{12}^l} \\
  \frac{\partial E}{\partial w_{21}^l} & \frac{\partial E}{\partial w_{22}^l} \\
\end{array}
\]

$H^{l-1}$

$\partial E / \partial H^l$

$\partial E / \partial W^l$
Gradients of $E$ w.r.t $H^{l-1}$

**Forward pass**

$$h^l = f^l(h^{l-1}, \{w^l, b^l\})$$

**Backward pass**

$$\frac{\partial E}{\partial h^{l-1}} = \frac{\partial E}{\partial h^l} \cdot \frac{\partial h^l}{\partial h^{l-1}}$$

$$\frac{\partial E}{\partial w^l} = \frac{\partial E}{\partial h^l} \cdot \frac{\partial h^l}{\partial w^l}$$
Gradients of $E$ w.r.t $H^{l-1}$

\[
\begin{array}{ccc}
 h_{11}^{l-1} & h_{12}^{l-1} & h_{13}^{l-1} \\
 h_{21}^{l-1} & h_{22}^{l-1} & h_{23}^{l-1} \\
 h_{31}^{l-1} & h_{32}^{l-1} & h_{33}^{l-1} \\
\end{array}
\]

$H^{l-1}$

\[
\begin{array}{cc}
 w_{11}^{l} & w_{12}^{l} \\
 w_{21}^{l} & w_{22}^{l} \\
\end{array}
\]

$W^{l}$

\[
\begin{array}{cc}
 h_{11}^{l} & h_{12}^{l} \\
 h_{21}^{l} & h_{22}^{l} \\
\end{array}
\]

$H^{l}$
Gradients of $E$ w.r.t $H^{l-1}$

Imagine inverting the receptive field!
Gradients of $E$ w.r.t $h^{l-1}$

\[ h_{11}^{l} = w_{11}^{l} h_{11}^{l-1} + w_{12}^{l} h_{12}^{l-1} + w_{21}^{l} h_{21}^{l-1} + w_{22}^{l} h_{22}^{l-1} + b \]
\[ h_{12}^{l} = w_{11}^{l} h_{12}^{l-1} + w_{12}^{l} h_{13}^{l-1} + w_{21}^{l} h_{22}^{l-1} + w_{22}^{l} h_{23}^{l-1} + b \]
\[ h_{21}^{l} = w_{11}^{l} h_{21}^{l-1} + w_{12}^{l} h_{22}^{l-1} + w_{21}^{l} h_{31}^{l-1} + w_{22}^{l} h_{32}^{l-1} + b \]
\[ h_{22}^{l} = w_{11}^{l} h_{22}^{l-1} + w_{12}^{l} h_{23}^{l-1} + w_{21}^{l} h_{32}^{l-1} + w_{22}^{l} h_{33}^{l-1} + b \]

Let’s calculate the gradients of loss function ($E$) with respect to previous layer ($H^{l-1}$)

\[
\frac{\partial E}{\partial h_{11}^{l-1}} = \frac{\partial E}{\partial h_{11}^{l}} \frac{\partial h_{11}^{l}}{\partial h_{11}^{l-1}} + \frac{\partial E}{\partial h_{12}^{l}} \frac{\partial h_{12}^{l}}{\partial h_{11}^{l-1}} + \frac{\partial E}{\partial h_{21}^{l}} \frac{\partial h_{21}^{l}}{\partial h_{11}^{l-1}} + \frac{\partial E}{\partial h_{22}^{l}} \frac{\partial h_{22}^{l}}{\partial h_{11}^{l-1}}
\]
Gradients of $E$ w.r.t $h^{l-1}$

$$h_{11}^l = w_{11}^l h_{11}^{l-1} + w_{12}^l h_{12}^{l-1} + w_{21}^l h_{21}^{l-1} + w_{22}^l h_{22}^{l-1} + b$$

$$h_{12}^l = w_{11}^l h_{12}^{l-1} + w_{12}^l h_{13}^{l-1} + w_{21}^l h_{22}^{l-1} + w_{22}^l h_{23}^{l-1} + b$$

$$h_{21}^l = w_{11}^l h_{21}^{l-1} + w_{12}^l h_{22}^{l-1} + w_{21}^l h_{31}^{l-1} + w_{22}^l h_{32}^{l-1} + b$$

$$h_{22}^l = w_{11}^l h_{22}^{l-1} + w_{12}^l h_{23}^{l-1} + w_{21}^l h_{32}^{l-1} + w_{22}^l h_{33}^{l-1} + b$$

Let’s calculate the gradients of loss function ($E$) with respect to previous layer ($H^{l-1}$)

$$\frac{\partial E}{\partial h_1^{l-1}} = \frac{\partial E}{\partial h_{11}^l} \frac{\partial h_{11}^l}{\partial h_1^{l-1}} + \frac{\partial E}{\partial h_{12}^l} \frac{\partial h_{12}^l}{\partial h_1^{l-1}} + \frac{\partial E}{\partial h_{21}^l} \frac{\partial h_{21}^l}{\partial h_1^{l-1}} + \frac{\partial E}{\partial h_{22}^l} \frac{\partial h_{22}^l}{\partial h_1^{l-1}}$$

$$\frac{\partial E}{\partial h_2^{l-1}} = \frac{\partial E}{\partial h_{11}^l} w_{11}^l + \frac{\partial E}{\partial h_{12}^l} w_{12}^l + \frac{\partial E}{\partial h_{21}^l} w_{21}^l + \frac{\partial E}{\partial h_{22}^l} w_{22}^l$$
Gradients of $E$ w.r.t $h^{l-1}$

$$
\begin{align*}
    h_{11}^l &= w_{11}^l h_{11}^{l-1} + w_{12}^l h_{12}^{l-1} + w_{21}^l h_{21}^{l-1} + w_{22}^l h_{22}^{l-1} + b \\
    h_{12}^l &= w_{11}^l h_{12}^{l-1} + w_{12}^l h_{13}^{l-1} + w_{21}^l h_{22}^{l-1} + w_{22}^l h_{23}^{l-1} + b \\
    h_{21}^l &= w_{11}^l h_{21}^{l-1} + w_{12}^l h_{22}^{l-1} + w_{21}^l h_{31}^{l-1} + w_{22}^l h_{32}^{l-1} + b \\
    h_{22}^l &= w_{11}^l h_{22}^{l-1} + w_{12}^l h_{23}^{l-1} + w_{21}^l h_{32}^{l-1} + w_{22}^l h_{33}^{l-1} + b
\end{align*}
$$

Let’s calculate the gradients of loss function ($E$) with respect to previous layer ($H^{l-1}$)

$$
\frac{\partial E}{\partial h_{22}^{l-1}} = \frac{\partial E}{\partial h_{11}^l} \frac{\partial h_{11}^l}{\partial h_{22}^{l-1}} + \frac{\partial E}{\partial h_{12}^l} \frac{\partial h_{12}^l}{\partial h_{22}^{l-1}} + \frac{\partial E}{\partial h_{21}^l} \frac{\partial h_{21}^l}{\partial h_{22}^{l-1}} + \frac{\partial E}{\partial h_{22}^l} \frac{\partial h_{22}^l}{\partial h_{22}^{l-1}}
$$
Gradients of $E$ w.r.t $h^{l-1}$

\[
\begin{align*}
    h_{11}^l &= w_{11}^l h_{11}^{l-1} + w_{12}^l h_{12}^{l-1} + w_{21}^l h_{21}^{l-1} + w_{22}^l h_{22}^{l-1} + b \\
    h_{12}^l &= w_{11}^l h_{12}^{l-1} + w_{12}^l h_{13}^{l-1} + w_{21}^l h_{22}^{l-1} + w_{22}^l h_{23}^{l-1} + b \\
    h_{21}^l &= w_{11}^l h_{21}^{l-1} + w_{12}^l h_{22}^{l-1} + w_{21}^l h_{31}^{l-1} + w_{22}^l h_{32}^{l-1} + b \\
    h_{22}^l &= w_{11}^l h_{22}^{l-1} + w_{12}^l h_{23}^{l-1} + w_{21}^l h_{32}^{l-1} + w_{22}^l h_{33}^{l-1} + b
\end{align*}
\]

Let's calculate the gradients of loss function ($E$) with respect to previous layer ($H^{l-1}$)

\[
\frac{\partial E}{\partial h_{22}^{l-1}} = \frac{\partial E}{\partial h_{11}^l} \frac{\partial h_{11}^l}{\partial h_{22}^{l-1}} + \frac{\partial E}{\partial h_{12}^l} \frac{\partial h_{12}^l}{\partial h_{22}^{l-1}} + \frac{\partial E}{\partial h_{21}^l} \frac{\partial h_{21}^l}{\partial h_{22}^{l-1}} + \frac{\partial E}{\partial h_{22}^l} \frac{\partial h_{22}^l}{\partial h_{22}^{l-1}}
\]

\[
\frac{\partial E}{\partial h_{22}^{l-1}} = \frac{\partial E}{\partial h_{11}^l} w_{22}^l + \frac{\partial E}{\partial h_{12}^l} w_{21}^l + \frac{\partial E}{\partial h_{21}^l} w_{12}^l + \frac{\partial E}{\partial h_{22}^l} w_{11}^l
\]
Gradients of $E$ w.r.t $H^{l-1}$

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$\frac{\partial E}{\partial h_{11}^l}$</td>
<td>$\frac{\partial E}{\partial h_{12}^l}$</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>$\frac{\partial E}{\partial h_{21}^l}$</td>
<td>$\frac{\partial E}{\partial h_{22}^l}$</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Padded $\frac{\partial E}{\partial H^l}$

<table>
<thead>
<tr>
<th></th>
<th>$w_{22}^l$</th>
<th>$w_{21}^l$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_{12}^l$</td>
<td>$w_{11}^l$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Rotated $W^l$

\[ \begin{array}{ccc}
\frac{\partial E}{\partial h_{11}^{l-1}} & \frac{\partial E}{\partial h_{12}^{l-1}} & \frac{\partial E}{\partial h_{13}^{l-1}} \\
\frac{\partial E}{\partial h_{21}^{l-1}} & \frac{\partial E}{\partial h_{22}^{l-1}} & \frac{\partial E}{\partial h_{23}^{l-1}} \\
\frac{\partial E}{\partial h_{31}^{l-1}} & \frac{\partial E}{\partial h_{32}^{l-1}} & \frac{\partial E}{\partial h_{33}^{l-1}} \\
\end{array} \]

Imagine inverting the receptive field!
Backpropagation for pooling

- Max function: $m = \max(a, b)$

$$
\frac{\partial m}{\partial a} = \begin{cases} 
1 & \text{if } a > b, \\
0 & \text{else.}
\end{cases}
\quad
\frac{\partial m}{\partial b} = \begin{cases} 
1 & \text{if } b > a, \\
0 & \text{else.}
\end{cases}
$$
Backpropagation for pooling

- Max function: $m = \max(a, b)$

$$\frac{\partial m}{\partial a} = \begin{cases} 1 \text{ if } a > b, \\ 0 \text{ else.} \end{cases} \quad \frac{\partial m}{\partial b} = \begin{cases} 1 \text{ if } b > a, \\ 0 \text{ else.} \end{cases}$$
Backpropagation for pooling

- **Max function:** \( m = \max(a, b) \)

\[
\frac{\partial m}{\partial a} = \begin{cases} 
1 & \text{if } a > b, \\
0 & \text{else.}
\end{cases}
\]

\[
\frac{\partial m}{\partial b} = \begin{cases} 
1 & \text{if } b > a, \\
0 & \text{else.}
\end{cases}
\]
Implementing fully-connected networks

Example at a time:

- Input vector
- Weight matrix
- Output vector

MLP Lecture 8 / 30 October / 6 November 2018
Implementing fully-connected networks

Minibatch:

- Input vector (minibatch)
- Weight matrix
- Output vector (minibatch)
Implementing fully-connected networks

Minibatch:

input vector (minibatch)

weight matrix

output vector (minibatch)

input dimension x minibatch: Represent each layer as a 2-dimension matrix, where each row corresponds to a training example, and the number of minibatch examples is the number of rows
Implementing Convolutional Networks

Example at a time, single input image, single feature map:

input image  weight matrix (kernel)  feature map
Implementing Convolutional Networks

Example at a time, single input image, multiple feature map:

- Input image
- Weight matrices (kernels)
- Feature maps

\[ \begin{array}{c}
\text{input image} \\
\downarrow \\
\text{weight matrices (kernels)} \\
\downarrow \\
\text{feature maps}
\end{array} \]
Implementing Convolutional Networks

Example at a time, multiple input images, multiple feature map:

multiple input images

weight matrices (kernels)

feature maps
Minibatch, multiple input images, multiple feature map:

- Minibatch of multiple input images
- Weight matrices (kernels)
- Minibatch of feature maps

MLP Lecture 8 / 30 October / 6 November 2018
Implementing Convolutional Networks

- **Inputs / layer values:**
  - Each input image (and convolutional and pooling layer) is 2-dimensions \((x, y)\)
  - If we have multiple feature maps, then that is a third dimension
  - And the minibatch adds a fourth dimension
  - Thus we represent each input (layer values) using a 4-dimension *tensor* (array): 
    \((\text{minibatch-size, num-fmaps, } x, y)\)

- **Weight matrices (kernels):**
  - Each weight matrix used to scan across an image has 2 spatial dimensions \((x, y)\)
  - If there are multiple feature maps to be computed, then that is a third dimension
  - Multiple input feature maps adds a fourth dimension
  - Thus the weight matrices are also represented using a 4-dimension tensor: \((F_{\text{in}}, F_{\text{out}}, x, y)\)
Both forward and back prop thus involves multiplying 4D tensors. There are various ways to do this:

- Explicitly loop over the dimensions: this results in simpler code, but can be inefficient. Although using cython to compile the loops as C can speed things up.

- Serialisation: By replicating input patches and weight matrices, it is possible to convert the required 4D tensor multiplications into a large dot product. Requires careful manipulation of indices!

- Convolutions: use explicit convolution functions for forward and back prop, rotating for the backprop.
Deep convolutional networks
LeNet5 (LeCun et al, 1997)

- 2 convolutional layers \{C1, C3\} + non-linearity
- 2 average pooling \{S2, S4\}
- 2 fully connected hidden layer (no weight sharing) \{C5, F6\}
- Softmax classifier layer


- 5 convolutional layers + non-linearity (ReLU)
- 3 max pooling layers
- 2 fully connected hidden layer
- Softmax classifier layer

**Network Design**

**Key design choices:**
- 3x3 conv. kernels – very small
- conv. stride 1 – no loss of information

**Other details:**
- Rectification (ReLU) non-linearity
- 5 max-pool layers (x2 reduction)
- no normalisation
- 3 fully-connected (FC) layers
Simply stacking more layers?

http://arxiv.org/abs/1512.03385

CIFAR-10

56-layer net has higher training error and test error than 20-layer net!
Deep Residual Learning ("ResNets")


http://arxiv.org/abs/1512.03385

http://arxiv.org/abs/1512.03385
Deep Residual Learning (“ResNets”)


http://arxiv.org/abs/1512.03385
Deep Residual Learning ("ResNets")


http://arxiv.org/abs/1512.03385

A solution by construction:

- original layers: copied from a learned shallower model
- extra layers: set as identity
- at least the same training error
Deep Residual Learning ("ResNets")


http://arxiv.org/abs/1512.03385
Hierarchical Representations

Pixel $\rightarrow$ edge $\rightarrow$ texton $\rightarrow$ motif $\rightarrow$ part $\rightarrow$ object


Slide credits: Lecun & Ranzato
Convolutional networks include local receptive fields, weight sharing, and pooling leading

Backprop training can also be implemented as a “reverse” convolutional layer (with the weight matrix rotated)

Implement using 4D tensors:
- Inputs / Layer values: minibatch-size, number-fmaps, x, y
- Weights: $F_{in}, F_{out}, x, y$
- Arguments: stride, kernel size, dilation, filter groups

Reading:
Goodfellow et al, *Deep Learning* (ch 9)
http://www.deeplearningbook.org/contents/convnets.html