Intro to Machine Learning

Lecture 7: Convolutional Neural Networks

Shen Shen
March 22, 2024

(videos edited from 3b1b; some slides adapted from Phillip Isola and Kaiming He)
Outline

- Recap (fully-connected net)
- Motivation and big picture ideas of CNN
- Convolution operation
  - 1d and 2d convolution mechanics
  - interpretation:
    - local connectivity
    - weight sharing
  - 3d tensors
- Max pooling
  - Larger window
- Typical architecture and summary
A (feed-forward) neural network is

\[
\sum f_i(\cdot)
\]

input
learnable weights
linear combo
activations
layer

\[
\sum f_i(\cdot)
\]

\[
\sum f_i(\cdot)
\]

\[
\sum f_i(\cdot)
\]

\[
\sum f_i(\cdot)
\]

\[
\sum f_i(\cdot)
\]
Recap: Backpropagation

Forward propagation to obtain the output (model's guess)

\[ X = A^0 \rightarrow W^1_0 \rightarrow Z^1 \rightarrow f^1 \rightarrow A^1 \rightarrow W^2_0 \rightarrow Z^2 \rightarrow f^2 \rightarrow A^2 \rightarrow \ldots \rightarrow A^{L-1} \rightarrow W^L_0 \rightarrow Z^L \rightarrow f^L \rightarrow A^L \rightarrow \text{Loss} \]

\[ \frac{\partial \text{loss}}{\partial Z^1} \rightarrow \frac{\partial \text{loss}}{\partial A^1} \rightarrow \frac{\partial \text{loss}}{\partial Z^2} \rightarrow \frac{\partial \text{loss}}{\partial A^2} \rightarrow \frac{\partial \text{loss}}{\partial A^{L-1}} \rightarrow \frac{\partial \text{loss}}{\partial Z^L} \rightarrow \frac{\partial \text{loss}}{\partial A^L} \]

Backpropagation to obtain gradients with respect to the loss
Outline

- Recap (fully-connected net)
- Motivation and big picture ideas of CNN
- Convolution operation
  - 1d and 2d convolution mechanics
  - interpretation:
    - local connectivity
    - weight sharing
  - 3d tensors
- Max pooling
  - Larger window
- Typical architecture and summary
1. Why do we need a special network for images?
2. Why is CNN (the) special network for images?
Why do we need a special net for images?
784 weights per neuron
784 × 16 weights  16 biases
Q: Why do we need a specialized network?

A: fully-connected nets don't scale well to (interesting) images
Why do we think $9$ is $9$?
Why do we think any of

9 99999999999999999999

is a

9?
• Visual hierarchy

layering would help take care of that
- Visual hierarchy

- Spatial locality

- Translational invariance
CNN cleverly exploits

- Visual hierarchy
- Spatial locality
- Translational invariance

via

- layering (with nonlinear activations)
- convolution
- pooling

to handle images efficiently
cleverly exploits via to handle efficiently
Outline

- Recap (fully-connected net)
- Motivation and big picture ideas of CNN
  - Convolution operation
    - 1d and 2d convolution mechanics
    - interpretation:
      - local connectivity
      - weight sharing
    - 3d tensors
  - Max pooling
    - Larger window
  - Typical architecture and summary
Convolutional layer might sound foreign, but...

<table>
<thead>
<tr>
<th>Layer</th>
<th>Forward-pass {do}</th>
<th>Back-prop {learn}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully-connected</td>
<td>Dot-product</td>
<td>Neurons</td>
</tr>
<tr>
<td>Convolutional</td>
<td>Convolution</td>
<td>Filters (kernels)</td>
</tr>
</tbody>
</table>
input image

\[
\begin{array}{ccccc}
0 & 1 & 0 & 1 & 1 \\
\end{array}
\]

filter

\[
\begin{array}{cc}
-1 & 1 \\
\end{array}
\]

\[(0 \times -1) + (1 \times 1) = 1\]

output image

\[
\begin{array}{c}
1 \\
\end{array}
\]
input image

| 0 | 1 | 0 | 1 | 1 |

filter

| -1 | 1 |

\[(1 \times -1) + (0 \times 1) = -1\]

output image

| 1 | -1 |
\[
\begin{align*}
\text{input image} & \quad \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \end{bmatrix} \\
\text{filter} & \quad \begin{bmatrix} -1 & 1 \end{bmatrix} \\
\quad & \quad (0 \times -1) + (1 \times 1) = 1 \\
\text{output image} & \quad \begin{bmatrix} 1 & -1 & 1 \end{bmatrix}
\end{align*}
\]
<table>
<thead>
<tr>
<th>input image</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td></td>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1 * -1) + (1 * 1) = 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>output image</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
input image

\[
\begin{array}{cccc}
0 & 1 & -1 & 1 & 1 \\
\end{array}
\]

filter

\[
\begin{array}{cc}
-1 & 1 \\
\end{array}
\]

output image

\[
\begin{array}{cccc}
1 & -1 & 2 & 0 \\
\end{array}
\]

- 'look' locally
- parameter sharing
- "template" matching

\[
(0 \times -1) + (1 \times 1) = 1
\]

\[
(-1 \times -1) + (1 \times 1) = 2
\]
- 'look' locally

input image

| 0 | 1 | -1 | 1 | 1 |

filter

| -1 | 1 |

output image

| 1 | -1 | 2 | 0 |
- parameter sharing
- parameter sharing

```
1 1 0 1 1
```

convolve with
```
-1 1
```

or dot with
```
-1 0 0 0
1 1 -1 0
0 1 -1 0
0 0 1 -1
0 0 0 1
```

= 
```
1 -1 1 0
```
| 0 | 1 | 0 | 1 | 1 |

convolve with ? =

| 0 | 1 | 0 | 1 | 1 |
convolve with \[
\begin{array}{c}
1
\end{array}
\]

dot-product with \[
I_{5 \times 5}
\]
Convolution: a 2-D example

<table>
<thead>
<tr>
<th>input</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 0 0 0 1 1 0</td>
</tr>
<tr>
<td>0 1 1 1 1 1 1 0</td>
</tr>
<tr>
<td>0 1 1 1 1 1 1 0</td>
</tr>
<tr>
<td>0 1 1 1 1 1 1 0</td>
</tr>
<tr>
<td>0 1 1 1 1 1 1 0</td>
</tr>
<tr>
<td>0 0 1 1 1 0 0 0</td>
</tr>
<tr>
<td>0 0 1 1 1 0 0 0</td>
</tr>
<tr>
<td>0 0 1 1 1 0 0 0</td>
</tr>
<tr>
<td>0 0 0 0 0 0 0 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 1</td>
</tr>
<tr>
<td>0 0 0</td>
</tr>
<tr>
<td>-1 -2 -1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
**Convolution: a 2-D example**

**Input**

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>-2</td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Filter**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-2</td>
<td>-1</td>
</tr>
</tbody>
</table>

**Output**

-3
Convolution: a 2-D example

input

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

filter

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-2</td>
<td>-1</td>
</tr>
</tbody>
</table>

output

<table>
<thead>
<tr>
<th>-3</th>
<th>-4</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-3</td>
<td>-4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Convolution: a 2-D example**

**Input**
```
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 1 0
0 1 0 0 0 1 1 1 0
0 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 0
0 0 1 1 1 0 0 0 0
0 0 1 1 1 0 0 0 0
0 0 0 0 0 0 0 0 0
```

**Filter**
```
0 1 0 0 0
-1 -2 -1
```

**Output**
```
-3 -4 -4
1 2 1
0 0 0
-1 -2 -1
```
**Convolution: a 2-D example**

<table>
<thead>
<tr>
<th>input</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
</tr>
<tr>
<td>0 0 0</td>
</tr>
<tr>
<td>0 1 1</td>
</tr>
<tr>
<td>0 1 1</td>
</tr>
<tr>
<td>0 1 1</td>
</tr>
<tr>
<td>0 0 1</td>
</tr>
<tr>
<td>0 0 0</td>
</tr>
<tr>
<td>0 0 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 1</td>
</tr>
<tr>
<td>0 0 0</td>
</tr>
<tr>
<td>-1 -2 -1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3 -4 -4 -4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Convolution: a 2-D example

input

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

filter

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-2</td>
<td>-1</td>
</tr>
</tbody>
</table>

output

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-3</td>
<td>-4</td>
<td>-4</td>
<td>-4</td>
<td>-4</td>
<td>-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3</td>
<td>-4</td>
<td>-4</td>
<td>-3</td>
<td>-1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Convolution: padding

input: $H \times W = 8 \times 8$

output: $H \times W = 6 \times 6$
Convolution: padding

input: $H \times W = 8 \times 8$

output: $H \times W = 6 \times 6$
Convolution: padding

input: $H \times W = 8 \times 8$

output: $H \times W = 6 \times 6$
Convolution: padding

input: $H \times W = 8 \times 8$

output: $H \times W = 6 \times 6$
Convolution: padding

input: $H \times W = 8 \times 8$

output: $H \times W = 6 \times 6$
Convolution: padding

input: $H \times W = 8 \times 8$

output: $H \times W = 6 \times 6$
Convolution: padding

input: $H \times W = 8 \times 8$

output: $H \times W = 6 \times 6$
Convolution: padding

input: $8 \times 8$, + pad

output: $H \times W = 8 \times 8$
Convolution: padding

input: 8 × 8, + pad

output: H × W = 8 × 8
Convolution: padding

input: 8 × 8, + pad

output: H × W = 8 × 8
Convolution: padding

input: $8 \times 8$, + pad

output: $H \times W = 8 \times 8$
Convolutions: padding

input: $8 \times 8$, + pad

output: $H \times W = 8 \times 8$
Convolution: padding

input: $8 \times 8$, + pad

output: $H \times W = 8 \times 8$
Convolution: padding

input: \(8 \times 8, +\) pad

output: \(H \times W = 8 \times 8\)
Convolution: padding

input: $8 \times 8$, + pad

output: $H \times W = 8 \times 8$
Convolutions: padding

input: $8 \times 8$, + pad

output: $H \times W = 8 \times 8$
Convolution: padding

input: 8 × 8, + pad

output: H × W = 8 × 8
Convolution: stride

input

stride = 2

output

filter
Convolution: stride

stride = 2
Convolution: stride

input

stride = 2

filter

output
Convolution: stride

Input:

\[
\begin{array}{ccc}
\circ & \circ & \circ \\
\end{array}
\]

Filter:

\[
\begin{array}{ccc}
\otimes & \otimes & \otimes \\
\otimes & \otimes & \otimes \\
\end{array}
\]

Output:

\[
\begin{array}{ccc}
\circ & \circ & \circ \\
\end{array}
\]

Stride = 2
Convolution: stride

Input:

```
  o  o  o  o  o
```

Filter:

```
  o  o  o  o  o
```

Stride = 2

Output:

```
  o  o  o  o  o
  o  o  o  o  o
```

```
Convolution: stride

input

filter

output

stride = 2
Convolution: stride

input

output

filter
Convolution: stride

input: $H \times W = 8 \times 8$

stride = 2

filter

output: $H \times W = 4 \times 4$
Convolution: stride

Input: $H \times W = 8 \times 8$

Output: $H \times W = 4 \times 4$

Stride = 2
Outline

- Recap (fully-connected net)
- Motivation and big picture ideas of CNN
- Convolution operation
  - 1d and 2d convolution mechanics
  - interpretation:
    - local connectivity
    - weight sharing
  - 3d tensors
- Max pooling
  - Larger window
- Typical architecture and summary
A tender intro to tensor:

[Image credit: tensorflow]
- 3d tensor input, depth $d$
- 3d tensor filter, depth $d$
- 2d tensor (matrix) output
3d tensor input, depth $d$

$k$ 3d filters:
- each filter of depth $d$
- each filter makes a 2d tensor (matrix) output

total output 3d tensor, depth $k$
Outline

- Recap (fully-connected net)
- Motivation and big picture ideas of CNN
- Convolution operation
  - 1d and 2d convolution mechanics
  - interpretation:
    - local connectivity
    - weight sharing
  - 3d tensors
- Max pooling
  - Larger window
- Typical architecture and summary
sliding window (w. stride)

convolution

max pooling

sliding window (w. stride)
Pooling across spatial locations achieves stability w.r.t. small translations:
Pooling across spatial locations achieves stability w.r.t. small translations:

large response regardless of exact position of edge
Pooling across spatial locations achieves stability w.r.t. small translations:
Outline

- Recap (fully-connected net)
- Motivation and big picture ideas of CNN
- Convolution operation
  - 1d and 2d convolution mechanics
  - interpretation:
    - local connectivity
    - weight sharing
  - 3d tensors
- Max pooling
  - Larger window
- Typical architecture and summary
We'd love it for you to share some lecture feedback.

Thanks!