Deep Computer Vision for Manufacturing

Aurélien Géron

May 23rd, 2018
Strata Data Conference, London
wheelchair basketball: 0.829
basketball: 0.114
streetball: 0.020
• Machine Vision
• Introduction to Convolutional Neural Networks
• A Tour of Deep Computer Vision
• Challenges in Manufacturing
Machine Vision
• Sorting
• Sorting
• Inspection
• Sorting
• Inspection
• Analytics
• Sorting
• Inspection
• Analytics
• Robot Guidance
• Sorting
• Inspection
• Analytics
• Robot Guidance
• Security
• Sorting
• Inspection
• Analytics
• Robot Guidance
• Security
• More!
Machine Vision > Filters

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

The image shows a 3x3 filter:

```
-1  0  1
-2  0  2
-1  0  1
```

This filter is applied to an image, resulting in the following transformation:

The flipped version of the original image is shown on the right side of the grid.
### Filters

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

The diagram on the right illustrates how the filter is applied to an image. The filter is moved pixel by pixel across the image, and the output is computed for each position. The resulting pixel values are determined by the convolution of the filter with the image pixels.
Machine Vision > Filters

```
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |
```

[Diagram showing a filter operation on an image]
Machine Vision > Filters

-1 0 1
-2 0 2
-1 0 1
### Machine Vision > Filters

#### 2D Convolutional Filters

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

![Convolution Example](image-url)
### Machine Vision Filters

#### Convolution Kernel

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

The convolution kernel is applied to the image to demonstrate the effect. The resulting output is shown on the right. The kernel is sliding over the image pixel by pixel, applying the values as specified in the kernel matrix to the corresponding image pixels, resulting in the transformed image.
Machine Vision > Filters

\[
\begin{array}{ccc}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1 \\
\end{array}
\]
Machine Vision > Filters

-1 0 1
-2 0 2
-1 0 1
Machine Vision > Filters

Convolution (cross-correlation)
Machine Vision > Filters > Zero Padding

\[
\begin{array}{ccc}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1 \\
\end{array}
\]
Machine Vision > Filters > Zero Padding

The diagram illustrates the concept of zero padding in the context of image processing. The table shows the values for zero padding, which are applied to the edges of an image to extend its boundaries. The padding values are applied to the edges of the image to ensure that the filter can be applied without losing information at the borders. This technique is often used in convolutional neural networks to avoid the need for explicit padding and to simplify the implementation of padding in algorithms.
### Machine Vision > Filters > Zero Padding

#### Table: Zero Padding

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

#### Diagram: Zero Padding

The diagram on the left represents an image filtered with a 3x3 zero padding filter. The numbers in the table correspond to the filter coefficients applied to the image. The right diagram shows the result of applying the filter to the image, with each pixel updated according to the filter equation:

\[ y_{i,j} = \sum_{m=-2}^{1} \sum_{n=-2}^{1} x_{m,n} f_{i-m,j-n} \]

where \( y_{i,j} \) is the value of the output image at position \( (i,j) \), \( x_{m,n} \) are the pixel values of the input image, and \( f_{i-m,j-n} \) are the filter coefficients from the table.

The result is a padded version of the original image, with the edges extended according to the zero padding method.
Machine Vision > Filters

Feature Map 1

Feature Map 2

Vertical filter

Horizontal filter

Input
Machine Vision > Blob Discovery

- label 1
- label 0
- background
Introduction to ConvNets
CNNs > Visual Cortex > Receptive Fields
CNNs > Convolutional Layers

Convolutional layer 2

Convolutional layer 1

Input layer
f_h = 3

f_w = 3

Zero padding
CNNs > Stride

\[ s_h = 2 \]

\[ s_w = 2 \]
Color Channels
CNNs > ConvNet Architecture

Input → Convolution → Pooling → Convolution → Pooling → Fully connected
CNNs > ConvNet Architecture

Input → Convolution → Pooling → Convolution → Pooling → Fully connected → Tree
CNNs > ConvNet Architecture

Input → Convolution → Pooling → Convolution → Pooling → Fully connected

Tree

Temple
CNNs > ConvNet Architecture

Input  Convolution  Pooling  Convolution  Pooling  Fully connected
A Tour of Deep Computer Vision
Deep Computer Vision > Classification

Sheep
Deep Computer Vision > ResNet Architecture

- Softmax
- Fully Connected: 1000 units
- Avg Pool: 1024, 7x7 + 1(V)
- Deep!
- Max Pool: 64, 3x3 + 2(S)
- Convolution: 64, 7x7 + 2(S)
- Input
- Convolution: 128, 3x3 + 1(S)
- Convolution: 128, 3x3 + 1(S)
- Convolution: 128, 3x3 + 2(S)
- Convolution: 64, 3x3 + 1(S)
- Convolution: 64, 3x3 + 1(S)
- Convolution: 64, 3x3 + 1(S)
- Convolution: 64, 3x3 + 1(S)
- Residual Unit
- ReLU
- Batch Norm
- BN + ReLU
- Skip connection
Sheep
Deep Computer Vision > OverFeat

- Input
- Convolution 64, 7×7 + 2(S)
- Max Pool 64, 3×3 + 2(S)
- Avg Pool 1024, 7×7 + 1(V)
- Deep!
- Fully Connected 1000 units
- Fully Connected 4 units
- Softmax
Object Detection

Sheep 1

Sheep 2

Sheep 3
Deep Computer Vision > YOLO
Deep Computer Vision > YOLO
Deep Computer Vision > YOLO
Deep Computer Vision > YOLO
Object Detection

starring

YOLOv3
Semantic Segmentation

- Sheep
- Grass
- Road
Deep Computer Vision > Instance Segmentation

Sheep 1
Sheep 2
Sheep 3
Grass
Road
Source: “Mask R-CNN”, by Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick, https://arxiv.org/abs/1703.06870 (Figure 1).
Mask R-CNN
- Object Detection
- Segmentation
Deep Computer Vision > Anomaly Detection

Input → Convolution → Pooling → Convolution → Pooling → Fully connected → One Class SVM
Deep Computer Vision > Anomaly Detection

Input → Convolution → Pooling → Convolution → Pooling → Fully connected → One Class SVM
Challenges
Challenges > Framing the Problem
Challenges > Framing the Problem
Challenges > Framing the Problem

Predictions

- 17% confidence
- 83% confidence
- Inclusion: 98%
- Fracture: 1%
- ...
Challenges > Framing the Problem

- **Left view**
- **Right view**
- **Top view**

**Classifier**

**Predictions**
- 17% confidence (Inclusion 98%, Fracture 1%)
- 83% confidence
Challenges > Training Set

- Training Set
  - Inclusion
  - Fracture

- Trainer

- Model
Challenges > Training Set
Challenges > Model Rot

Precision

Time
Challenges > High Volume, Low Latency
Challenges > Model Interpretation
Questions?
Media Credits