EDAN95
Applied Machine Learning
http://cs.lth.se/edan95/
Lecture 6: Convolutional Networks

Pierre Nugues

Lund University
Pierre.Nugues@cs.lth.se
http://cs.lth.se/pierre_nugues/

November 19, 2018

The Origins: The Convolution

The product of a function and a moving window, called the kernel.
Mathematical definition:

$$(f * g)(x) = \int_{-\infty}^{+\infty} f(x-t)g(t)\,\mathrm{d}t,$$

where f is the function and g, the convolution kernel
Notice that one of the function, here f, is reversed and shifted to guarantee commutativity
In the discrete case, we have:

$$(f * g)(i) = \sum_{j=-\infty}^{\infty} f(i-j)g(j)$$

It can be extended to two dimensions.

Convolution in Image Processing

Convolution is used extensively in pattern recognition to implement spatial filtering.

In image processing, f is an image and g a small window, most frequently (3, 3) or (5, 5),.

For a kernel of dimensions (M, N), normally odd numbers, we have:

$$(f * g)(x, y) = \sum_{i=-M/2}^{M/2} \sum_{j=-N/2}^{N/2} f(x-i, y-j) g(i, j),$$

where g is centered at 0.

Example of a Convolution

A blurring kernel, normally normalized by its sum (1/9):

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \mathbf{219} & 253 & 247 \\ 0 & 0 & 190 & 0 \\ 0 & 0 & \mathbf{0} & 93 \\ 0 & 0 & 221 & 253 \\ 136 & 212 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \mathbf{662} & & 0 \\ 0 & & & 0 \\ 0 & & \mathbf{757} & 0 \\ 0 & & & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Borders can either be padded or ignored. In the latter case, the kernel must always fit in the image and the output image has a reduced size.
(Complete the matrix...)

Spatial Filters

The kernels enable us to create filters, for instance a smoothing or sharpening kernel.
The Sobel operator is a popular edge detector. It corresponds to the gradient norm of the input image.
We compute the x and y derivatives using two kernels:

$$\mathbf{G}_x = \mathbf{I} * \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}; \mathbf{G}_y = \mathbf{I} * \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\mathbf{G} = \sqrt{\mathbf{G}_x^2 + \mathbf{G}_y^2}$$

We can also compute the gradient angle:

$$\tan\theta = \frac{\mathbf{G}_y}{\mathbf{G}_x}; \theta = \arctan\frac{\mathbf{G}_y}{\mathbf{G}_x}$$

Code Example

Jupyter Notebook

Generator

A construct to build sequences (iterators) with a minimal memory footprint
Compare:

```
# A list
a = [i for i in range(10000000)]
print(sys.getsizeof(a))
```

8 times the number of items.
And

```
# A generator
b = (i for i in range(10000000))
print(sys.getsizeof(b))
```

A very small size
But you can only traverse it once

Code Example

Jupyter Notebook

Building a Convolutional Neural Network (CNN)