CSC320 Tutorial Notes (Convolution)

Marcus Brubaker (mbrubake@cs)

March 17th, 2006

Convolution Examples

Working with a small (4x4) image

$$I(x,y) = \begin{bmatrix} 5 & 10 & 3 & 25 \\ 3 & 5 & 4 & 33 \\ 3 & 4 & 5 & 22 \\ 2 & 0 & 1 & 4 \end{bmatrix}$$

we will apply two 3x3 convolutions to this "image." They will be expressed in matrix notation as follows

$$T(i.j) = \begin{bmatrix} T(-1,-1) & T(-1,0) & T(-1,1) \\ T(0,-1) & T(0,0) & T(0,1) \\ T(1,-1) & T(1,0) & T(1,1) \end{bmatrix}$$

where it is assumed that $T(i,j) = 0$ where not specified in the matrix.

Sobel Mask

First a familiar operator, the Sobel Mask. Recall that it has a kernel which looks like this

$$S_x(i,j) = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Now, we apply this kernel to $I(x,y)$

$$(I \star S_x)(x,y) = \sum_{i=0}^{3} \sum_{j=0}^{3} I(i,j) S_x(x-i, y-j)$$

From here on out we will look at the specific instance of applying this at $(1,1)$.

Notice that this only references a submatrix of $I(x,y)$ (called the area of support for the kernel). We can put all of these entries into a vector and rewrite

this as a dot product. Specifically,

$$\begin{aligned}
\vec{I}_{1,1} &= \begin{bmatrix} I(0,0) & I(0,1) & I(0,2) & I(1,0) & I(1,1) & I(1,2) & I(2,0) & I(2,1) & I(2,2) \end{bmatrix} \\
&= \begin{bmatrix} 5 & 10 & 3 & 3 & 5 & 4 & 3 & 4 & 5 \end{bmatrix} \\
\vec{S_x} &= \begin{bmatrix} S(1,1) & S(1,0) & S(1,-1) & S(0,1) & S(0,0) & S(0,-1) & S(-1,1) & S(-1,0) & S(-1,-1) \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & -1 & 2 & 0 & -2 & 1 & 0 & -1 \end{bmatrix}
\end{aligned}$$

then

$$\begin{aligned}
(I \star S_x)(1,1) &= \vec{I}_{1,1} \cdot \vec{S_x} \\
&= 5 + 0 - 3 + 6 + 0 - 8 + 3 + 0 - 5 \\
&= -8
\end{aligned}$$

which is what we were doing earlier for derivative estimation.

Gaussian Mask

Another example is to use the Gaussian smoothing mask

$$G_\sigma(i,j) = \frac{1}{2\pi\sigma^2} e^{-\frac{i^2+j^2}{2\sigma^2}}$$

which, when normalized, gives the following mask,

$$\bar{G}_\sigma(i,j) = \alpha \begin{bmatrix} e^{-\frac{1}{\sigma^2}} & e^{-\frac{1}{2\sigma^2}} & e^{-\frac{1}{\sigma^2}} \\ e^{-\frac{1}{2\sigma^2}} & e^{0} & e^{-\frac{1}{2\sigma^2}} \\ e^{-\frac{1}{\sigma^2}} & e^{-\frac{1}{2\sigma^2}} & e^{-\frac{1}{\sigma^2}} \end{bmatrix}$$

where

$$\alpha = \frac{1}{2\pi\sigma^2} \sum_{i,j} G_\sigma(i,j)$$

A discrete approximation of this looks something like this

$$\hat{G}(i,j) = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

which we can apply to the image similarly

$$\begin{aligned}
(I \star \hat{G})(1,1) &= \frac{1}{16}(5 + 20 + 3 + 6 + 20 + 8 + 3 + 8 + 5) \\
&= \frac{78}{16} \\
&= 4.875
\end{aligned}$$

Separability

Some kernels have a special property which allows them to be computed more efficiently called separability. Specifically, the mask can be expressed in the following form

$$T(i,j) = t_1(i)t_2(j)$$

which is equivalent to saying that in matrix form you can express the mask as the following

$$\begin{aligned} T &= t_1 t_2^T \\ &= \begin{bmatrix} t_1(-h) \\ \vdots \\ t_1(h) \end{bmatrix} \begin{bmatrix} t_2(-w) & \cdots & t_2(w) \end{bmatrix} \end{aligned}$$

The equivalence of these can be seen by multiplying out the vectors to get

$$T = \begin{bmatrix} t_1(-h)t_2(-w) & \cdots & t_1(-h)t_2(w) \\ \vdots & \ddots & \vdots \\ t_1(h)t_2(-w) & \cdots & t_1(h)t_2(w) \end{bmatrix}$$

Separable 2D filters can then be viewed as the successive application of two 1D filters which changes the computation from quadratic to linear time in the size of the support.

While few (common) kernels exhibit this property fortunately one of the more important ones does, the (isotropic) Gaussian smoothing kernel. To see this we start with the formula for the Gaussian

$$\begin{aligned} G_\sigma(i,j) &= \frac{1}{2\pi\sigma^2} e^{-\frac{i^2+j^2}{2\sigma^2}} \\ &= \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{i^2}{2\sigma^2}} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{j^2}{2\sigma^2}} \\ &= g_1(i)g_2(j) \end{aligned}$$

Linear Interpolation

This is interpolation example #2 from Wednesdays notes. As in the example, assume a 1D image $I(x)$ and a kernel and some kernel $L(\delta)$ then the interpolation (as a convolution) is

$$\begin{aligned} (I \star L)(x) &= \sum_{k=0}^{M-1} I(k)L(x-k) \\ &= I(0)L(x-0) + I(1)L(x-1) + I(2)L(x-2) + \cdots + I(M-1)L(x-(M-1)) \end{aligned}$$

There is an error in the slides here. The correct kernel is

$$L(d) = \begin{cases} 1-|d| & |d| \leq 1 \\ 0 & |d| > 0 \end{cases}$$

which looks like a triangle.

Lets look at some particular values of this interpolation. Specifically, lets look at $(I \star L)(5+\delta)$ for $0 < \delta \leq 1$

$$\begin{aligned}
(I \star L)(5+\delta) &= I(0)L(5+\delta) + \cdots + I(3)L(2+\delta) + I(4)L(1+\delta) + I(5)L(\delta) + I(6)L(\delta-1) + I(7)L(\delta \\
&= 0 + I(5)L(\delta) + I(6)L(\delta-1) + 0 \\
&= I(5)(1-|\delta|) + I(6)(1-|1-\delta|) \\
&= I(5)(1-\delta) + I(6)\delta
\end{aligned}$$

Now, what happens at some specific points.

$$\begin{aligned}
(I \star L)(5) &= I(5)(1-0) + I(6)(0) \\
&= I(5) \\
(I \star L)(6) &= (I \star L)(5+1) \\
&= I(5)(1-1) + I(6)(1) \\
&= I(6)
\end{aligned}$$

so at the exact points the approximation is equal to the original data at those points. What happens in between

$$\begin{aligned}
(I \star L)(5.5) &= I(5)(1-0.5) + I(6)(0.5) \\
&= 0.5I(5) + 0.5I(6) \\
&= \frac{1}{2}(I(5) + I(6))
\end{aligned}$$

In other words, the value of the interpolation (convolution) is a linear interpolation of the two neighboring points. And thus a piecewise linear interpolation of the image.

(At this point draw some points on the board and the connecting piecewise linear interpolation that this generates to make things more concrete.)