9/8/15

①

# BACKPROP in CNNs

→ Since design is modular we just have to think about B-PASS in conv layers!

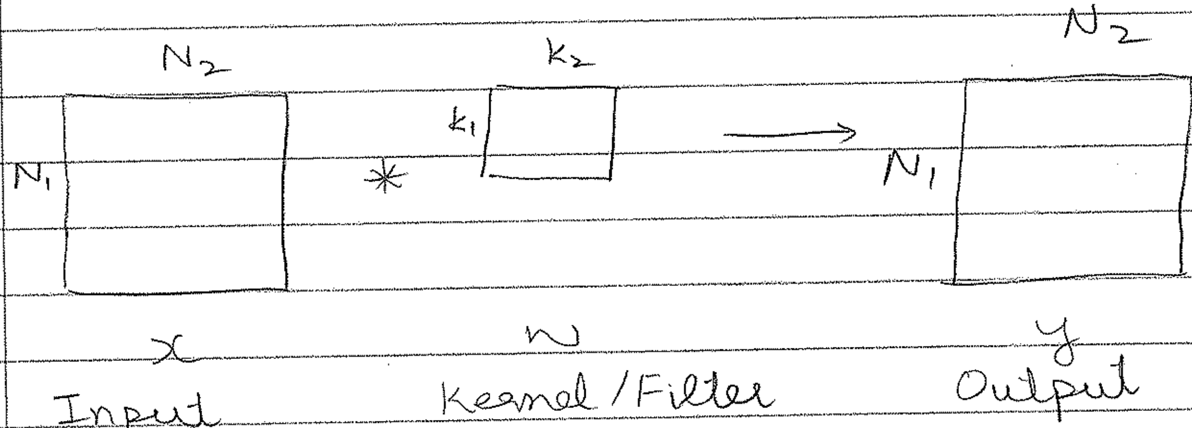
① CNN-B-PASS

→ This is going to be a notational nightmare

→ So let's simplify thing a bit

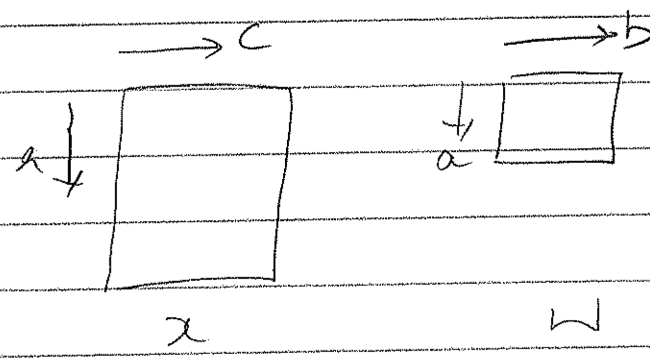
→ Say $C_1 = 1$ $C_2 = 1$ [We can always generalize. It's just an extra sum]

→ And let's not overload $h$. And let's drop super-scripts.




[Assume padding so same size for simplicity]

Notation: for iterators



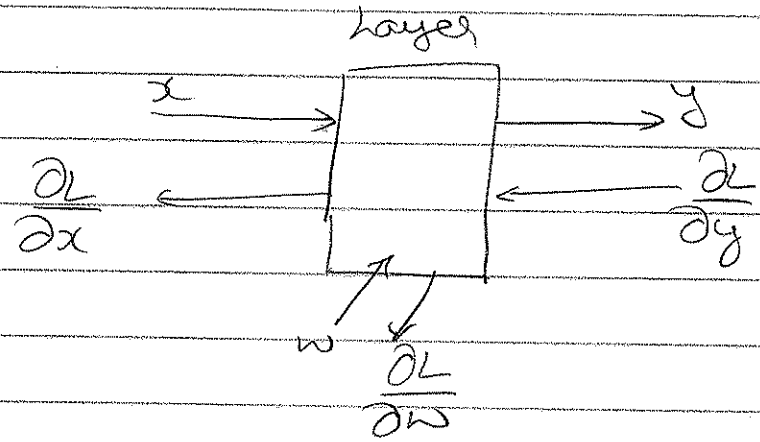
Recall: $$y[r,c] = \sum_{a=0}^{k_1-1} \sum_{b=0}^{k_2-1} x[r+a,\, c+b]\, w[a,b]$$

$$= \underbrace{\overrightarrow{xvec}^T}_{\substack{[\ \ ]_{1\times k_1 k_2} \\ \text{Cropped Image}}} \ \underbrace{\overrightarrow{wvec}}_{[\ \ ]_{k_1 k_2 \times 1}}$$

Note: sizes $|y| = N_1 \times N_2$

Thus $\left|\frac{\partial L}{\partial y}\right| = N_1 \times N_2$ $\quad\left|\frac{\partial L}{\partial x}\right| = N_1 \times N_2$

We will use $\frac{\partial L}{\partial y[r,c]}$ to access members of incomming gradients




Need to compute: (a) $\frac{\partial L}{\partial x}$ (b) $\frac{\partial L}{\partial w}$

(b) $\frac{\partial L}{\partial w}$ Let's do 1 pixel at a time

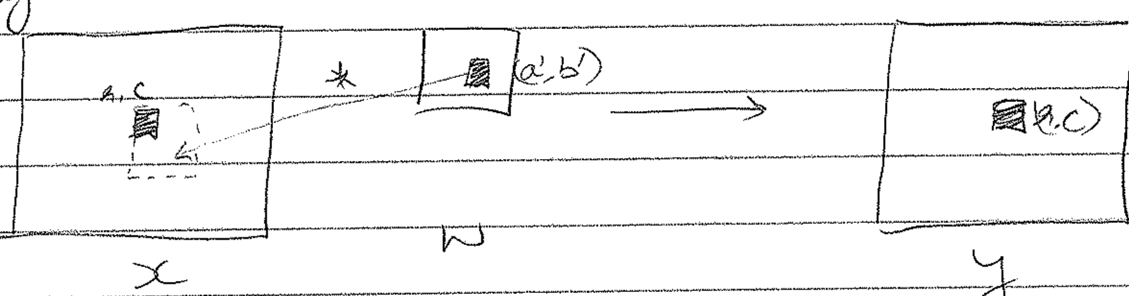
$$\underbrace{\frac{\partial L}{\partial w[a', b']}}$$

What does this weight affect? Everything!

via $\frac{\partial L}{\partial y[0,0]}, \frac{\partial L}{\partial y[0,1]} \cdots \frac{\partial L}{\partial y[N_1, N_2]}$

$$\frac{\partial L}{\partial w[a', b']} = \underbrace{\boxed{\sum_{r=0}^{N_1-1} \sum_{c=0}^{N_2-1}}}_{\text{all output pixels}} \underbrace{\frac{\partial L}{\partial y[r,c]}}_{\text{known}} \underbrace{\frac{\partial y[r,c]}{\partial w[a', b']}}_{\text{To compute}}$$

Visually:




$$\frac{\partial y[r,c]}{\partial w[a', b']} = x[r+a', c+b']$$

[can also prove analytically]

$$\Rightarrow \underbrace{\frac{\partial L}{\partial w[a', b']} = \sum_{r=0}^{N_1-1} \sum_{c=0}^{N_2-1} \frac{\partial L}{\partial y[r,c]} x[r+a', c+b']}$$

Looks like a convolution of 2 $(N_1 \times N_2)$ matrices!

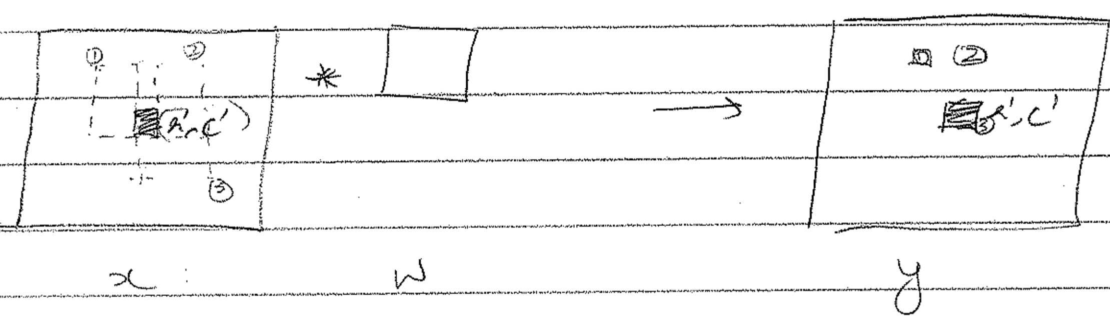
$$x * \frac{\partial L}{\partial y}$$

But clipped to $(k_1, k_2)$!

(a) $\frac{\partial L}{\partial x}$ Let's do 1 pixel at a time

$\frac{\partial L}{\partial x[r', c']}$

What does this pixel affect?



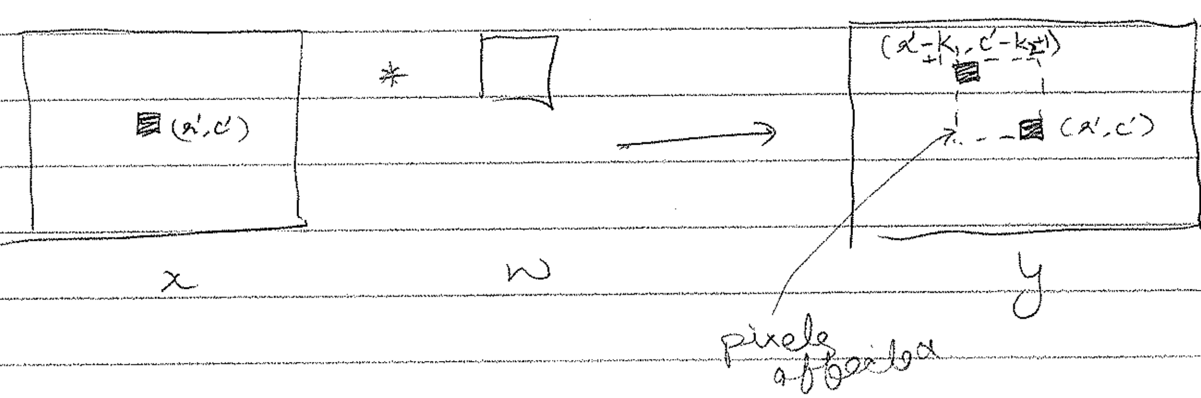

x[r, c] affect y in a neighbour hood around [r, c]

Neighbourhood size is determined by size of filter w.

In figure above, box 1 represents the "first" (left) box where x[r', c'] plays a role.

Box 3 represents the "last" box. Box 2 is in between.

So x[r', c'] affects the following output pixels



So let's apply chain rule

$$\frac{\partial L}{\partial x[r', c']} = \underbrace{\sum_{\substack{\text{pixels} \\ \text{affected} \\ p}}}_{} \frac{\partial L}{\partial y[p]} \frac{\partial y[p]}{\partial x[r', c']}$$

Let's make this a bit more formal.

$$\frac{\partial L}{\partial x[r', c']} = \sum_{a=0}^{k_1-1} \sum_{b=0}^{k_2-1} \underbrace{\frac{\partial L}{\partial y[r'-a, c'-b]}}_{\text{given}} \underbrace{\frac{\partial y[r'-a, c'-b]}{\partial x[r', c']}}_{\text{Let's compute/derive}}$$

→ Can 'derive' visually (like last time)
→ or analytically.

Let's try analytically this time:

$$y[r', c'] = \sum_{a'=0}^{k_1-1} \sum_{b'=0}^{k_2-1} x[r'+a', c'+b'] \, w[a', b']$$

[Same equation as a few pages ago, just written with 'primes']

Now $$y[r'-a, c'-b] = \sum_{a'=0}^{k_1-1} \sum_{b'=0}^{k_2-1} x[r'-a+a', c'-b+b'] \, w[a', b']$$

$$\frac{\partial y[r'-a, c'-b]}{\partial x[r', c']} = w[a, b]$$ [Why? ∵ $x[r', c']$ only appears once in the expression]

$$\Rightarrow \frac{\partial L}{\partial x[x', c']} = \underbrace{\sum_{a=0}^{k_1-1} \sum_{b=0}^{k_2-1} \frac{\partial L}{\partial y[x'-a, c'-b]} w[a,b]}$$

Very Nice! Almost like a 'convolution'
[More like cross-correlation]

Actually if we "flip" w about it's center horizontally & vertically to get $w^{flip}$

$$\boxed{\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y} * w^{flip}}$$

Super cool!

So F-PROP = Convolution

B-PROP = Convolution with flipped filter!