CS568 Deep Learning, Spring 2020

Recitation 5: Convolution

April 2, 2020

Convolution

- Convolution is the linear operation to transform the local intensities of pixels.
- It takes image and mask and computes the dot product.
- The mask is also called convolution matrix, filter, kernel or template.
- Convolution plays a central role in image processing.

Convolution

It replaces each pixel by a linear combination of its neighbours.

x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
x_{21}	x_{22}	x_{23}	x_{24}	x_{25}
x_{31}	x_{32}	x_{33}	x_{34}	x_{35}
x_{41}	x_{42}	x_{43}	x_{44}	x_{45}
x_{51}	x_{52}	x_{53}	x_{54}	x_{55}

$\mathbf{I_{5\times5}}$

$*$

w_{11}	w_{12}	w_{13}
w_{21}	w_{22}	w_{23}
w_{31}	w_{32}	w_{33}

$\mathbf{M_{3\times3}}$

$$
\begin{aligned}
I_{2,2} = &(w_{11} \times x_{11}) + (w_{12} \times x_{12}) + (w_{13} \times x_{13}) + (w_{21} \times x_{21}) \\
&+ (w_{21} \times x_{21}) + (w_{22} \times x_{22}) + (w_{23} \times x_{23}) + (w_{31} \times x_{31}) \\
&+ (w_{32} \times x_{32}) + (w_{33} \times x_{33})
\end{aligned}
$$

$$
I_{2,2} = \sum_{i=1}^{3} \sum_{j=1}^{3} w_{ij} \times x_{ij}
$$

Convolution

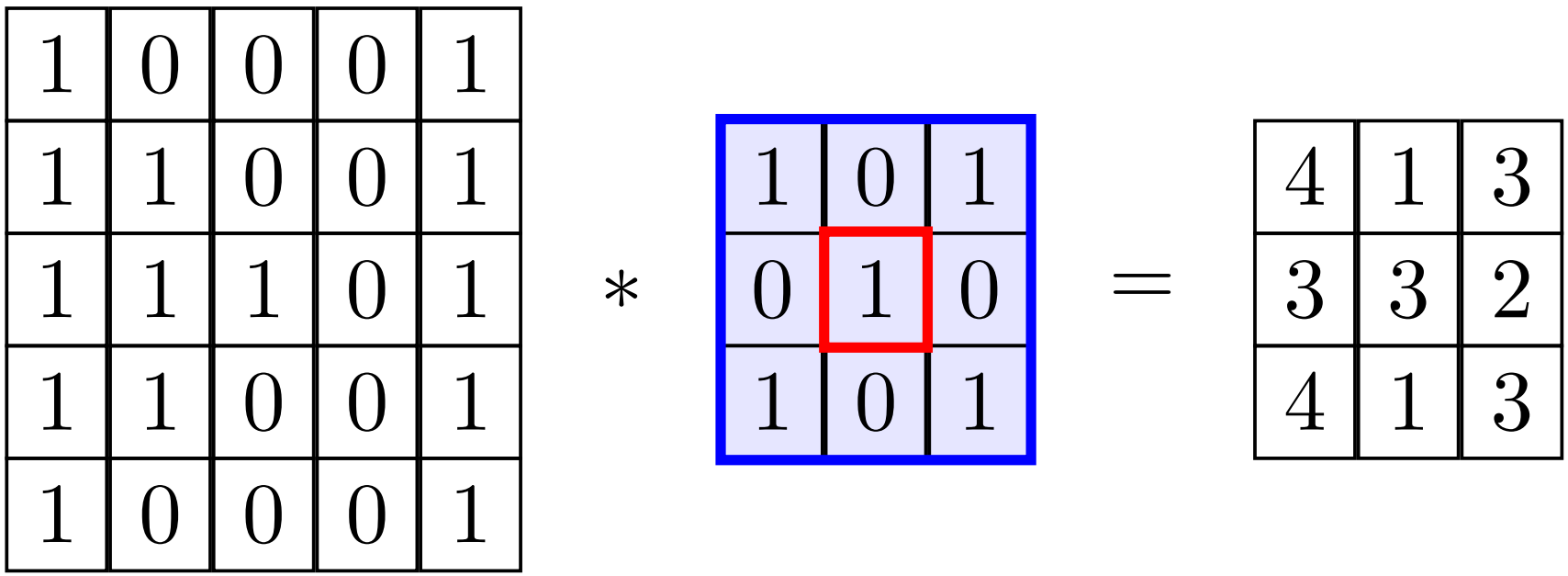
For image I and mask M, convolution is defined as

$$I_2[i,j] = \sum_k \sum_l I_1[k,l] M[i-k, j-l]$$

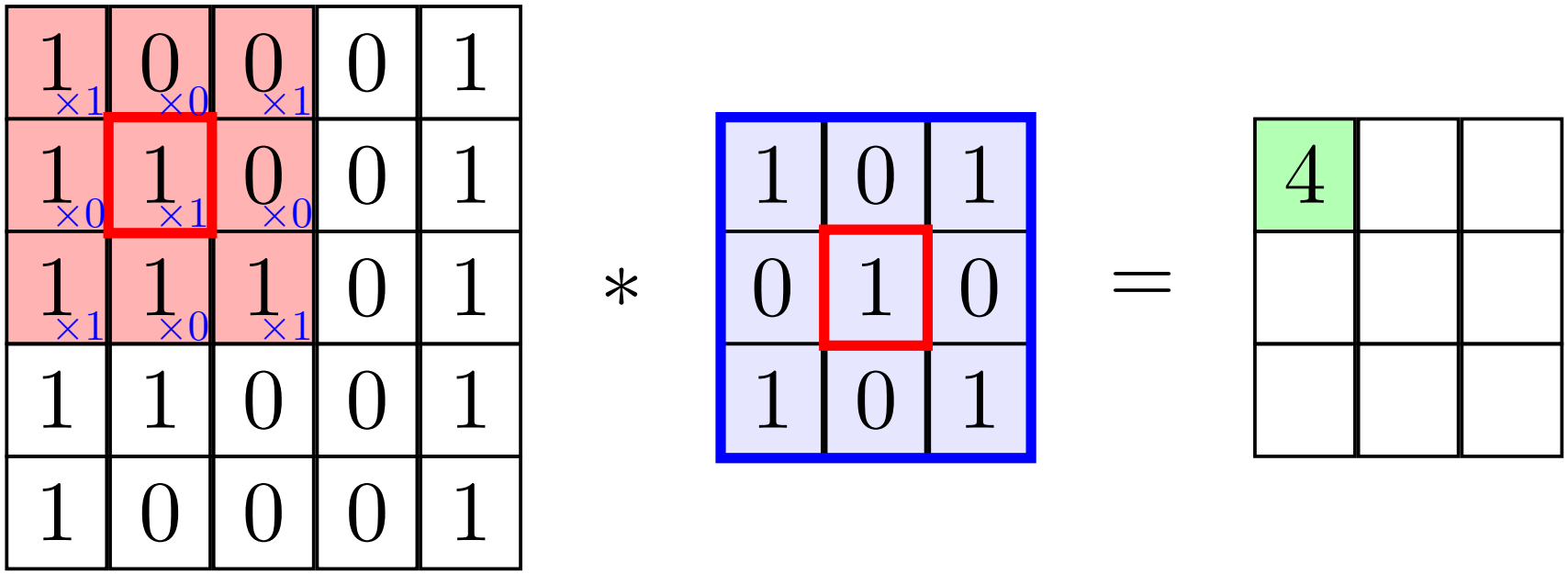
To obtain the $I_2 = I_1 * M$

1. First flip the mask M in both dimensions.
2. For each pixel p in I_1
   - 2.1 Place mask origin on top of the pixel.
   - 2.2 Multiply each mask weight with the pixel value under it.
   - 2.3 Sum the result and put in location of the pixel p in I_2.

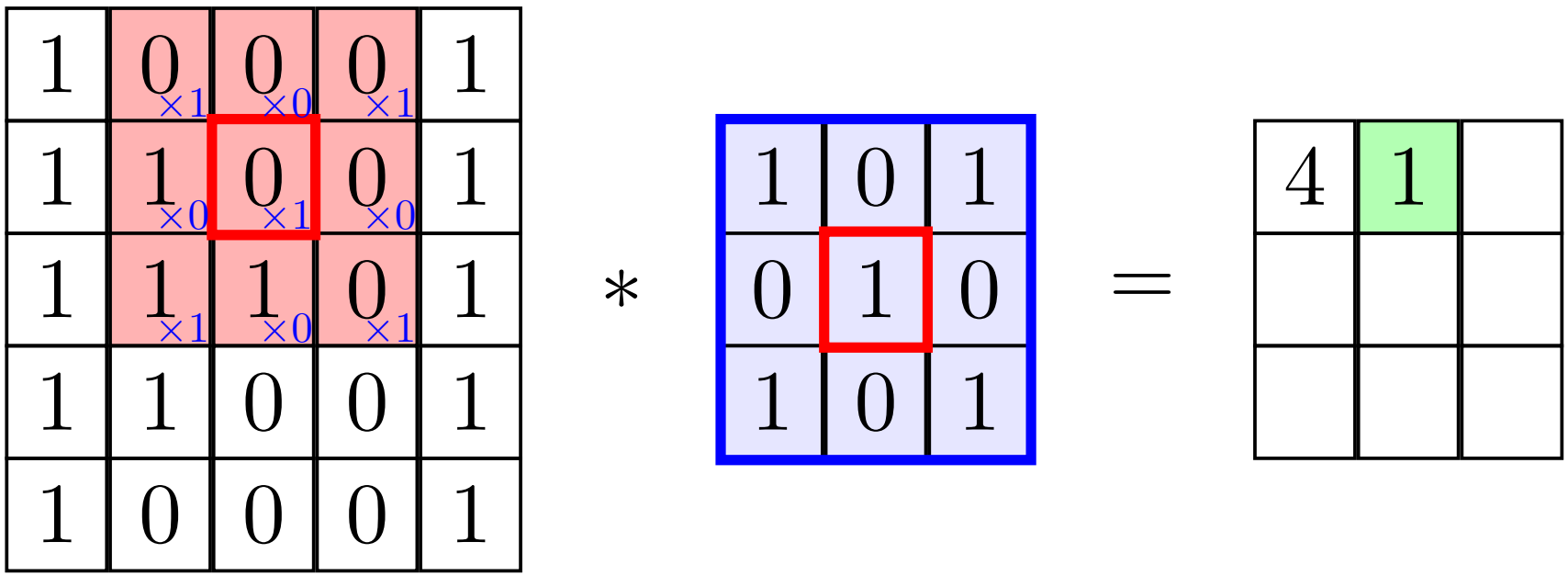
Convolution: Example



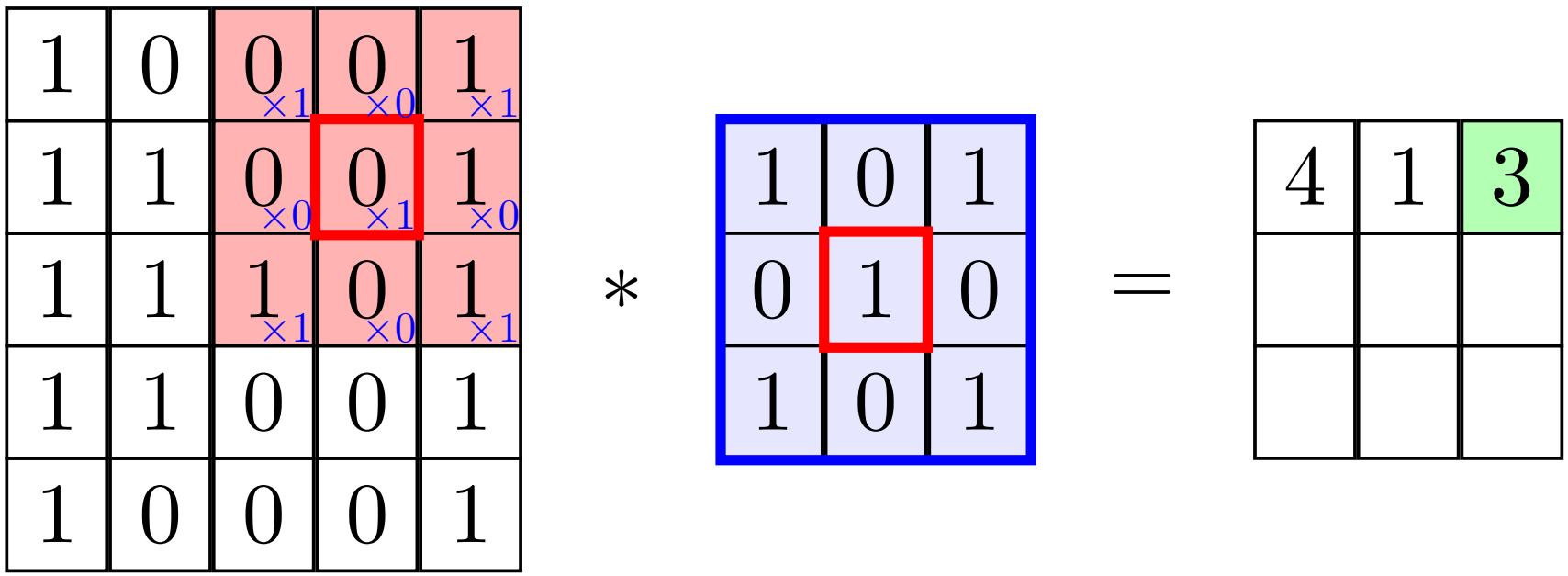
Convolution: Example



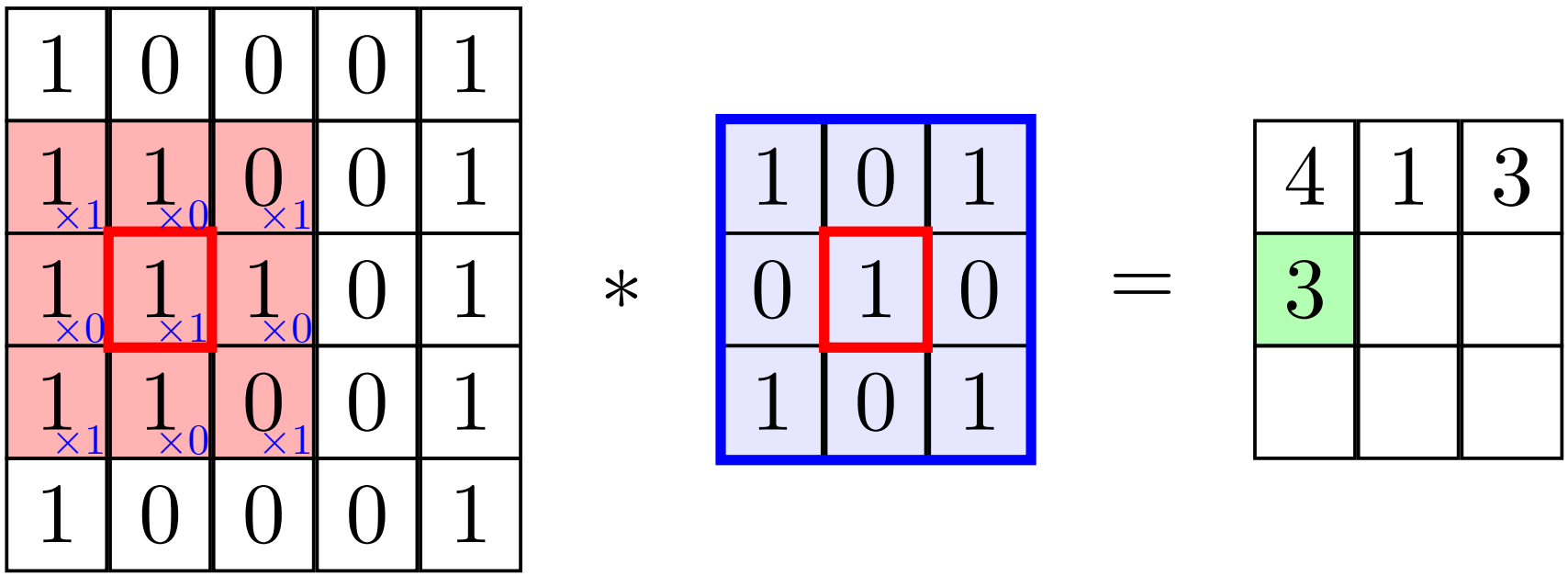
Convolution: Example



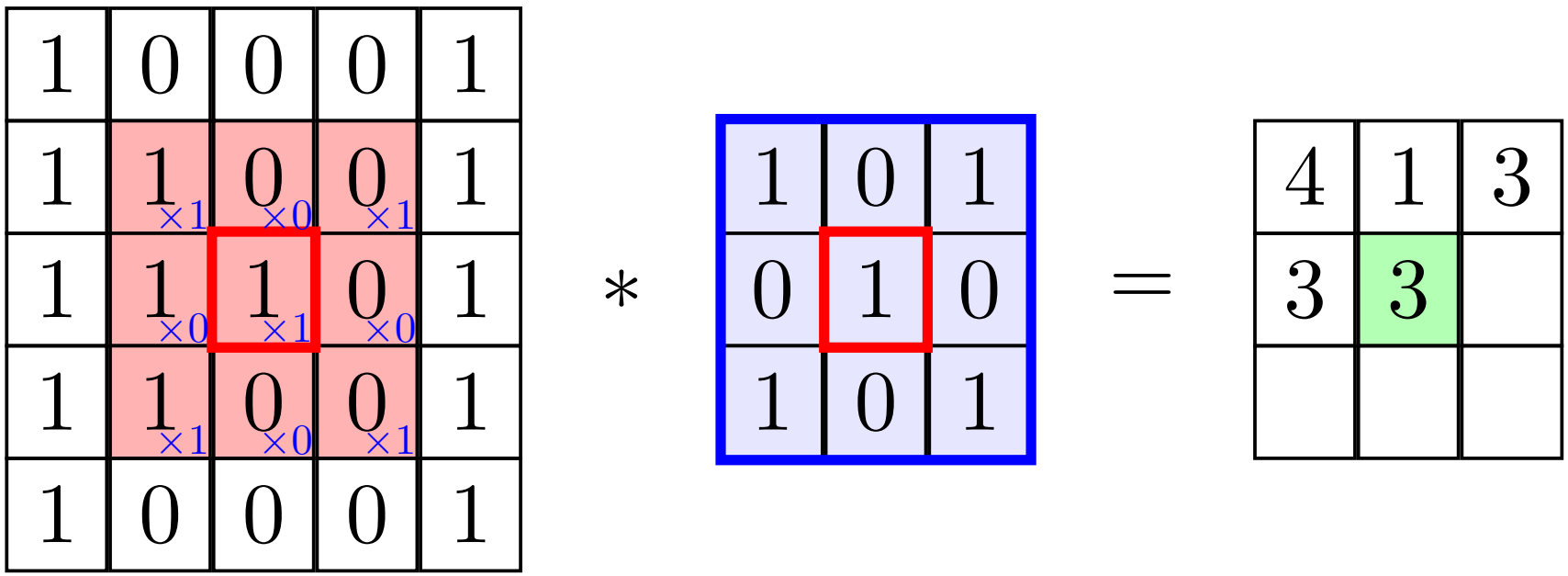
Convolution: Example



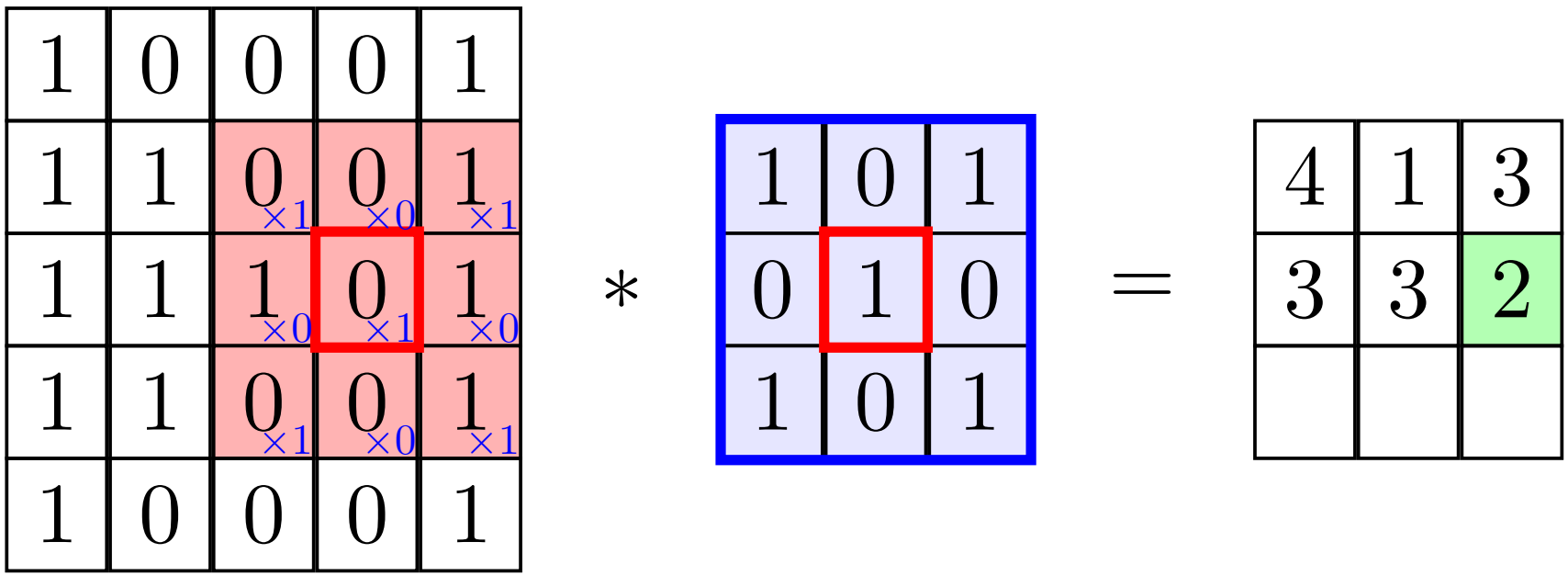
Convolution: Example



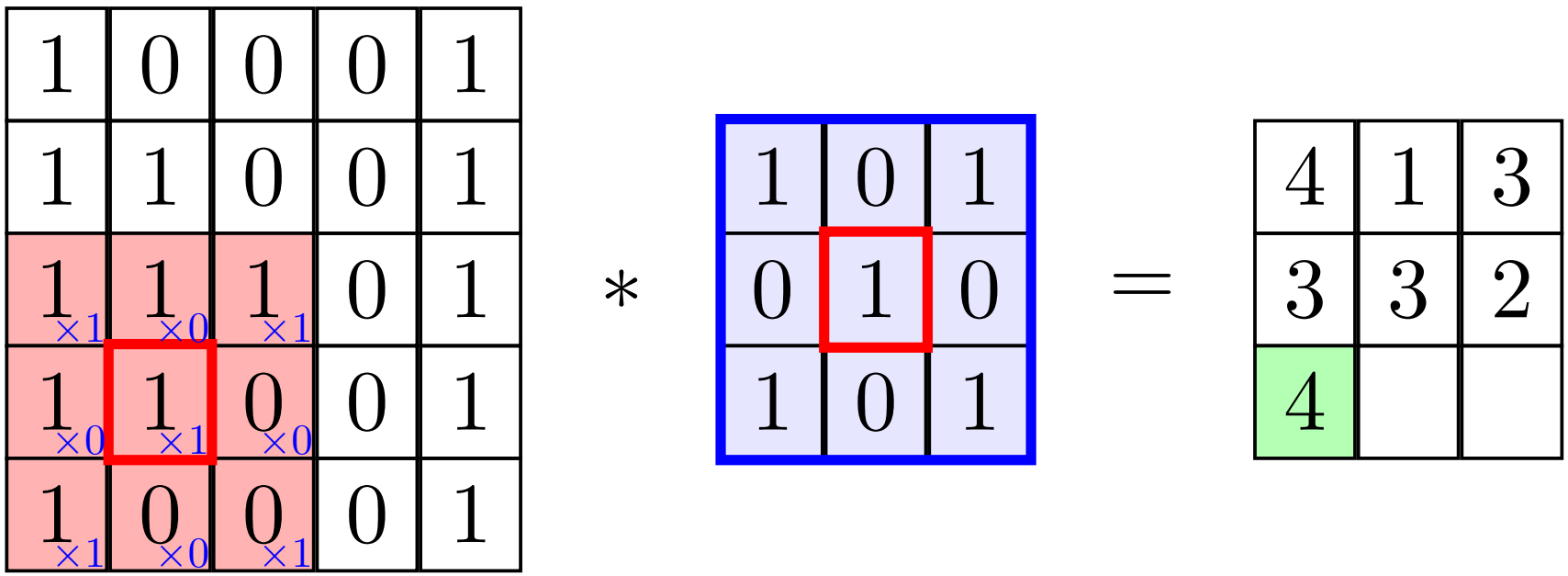
Convolution: Example



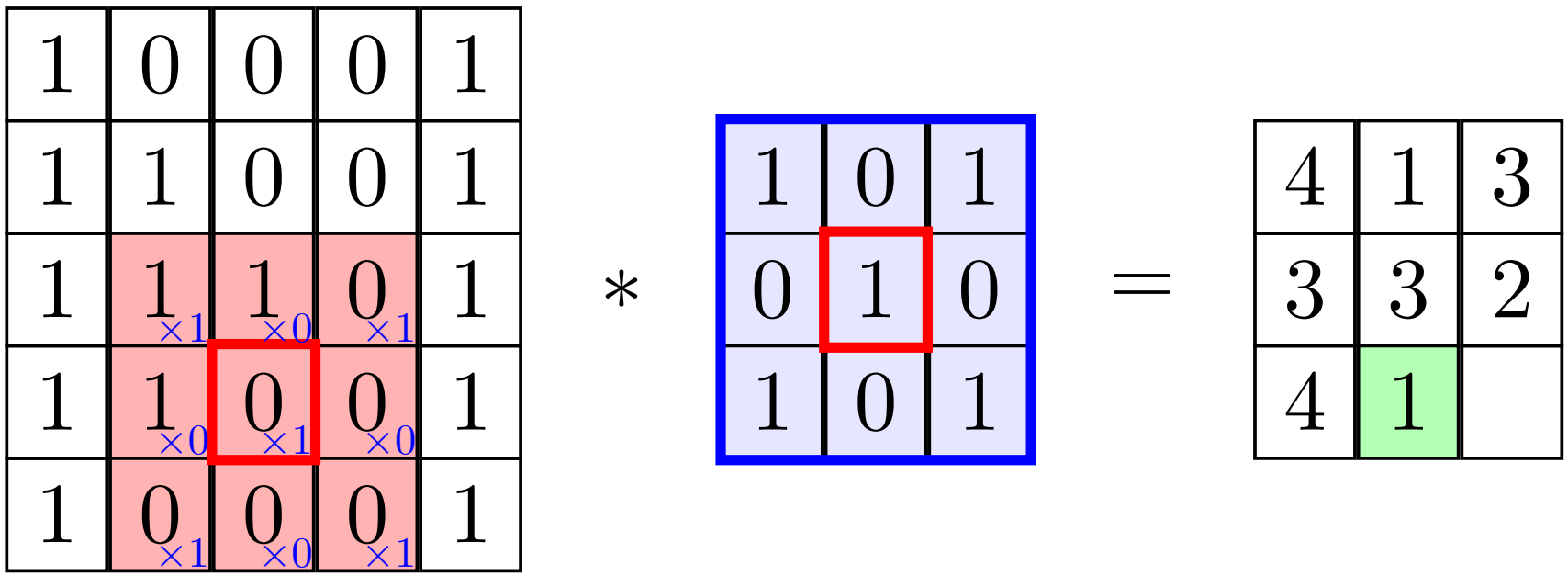
Convolution: Example



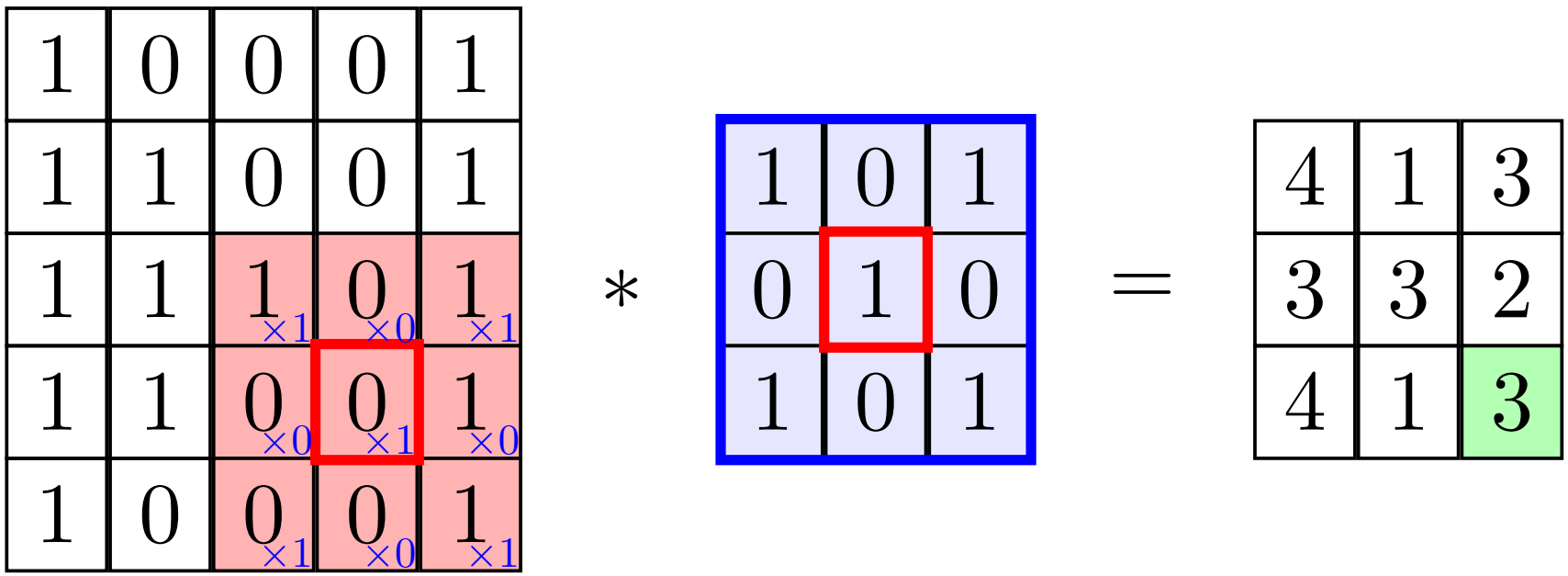
Convolution: Example



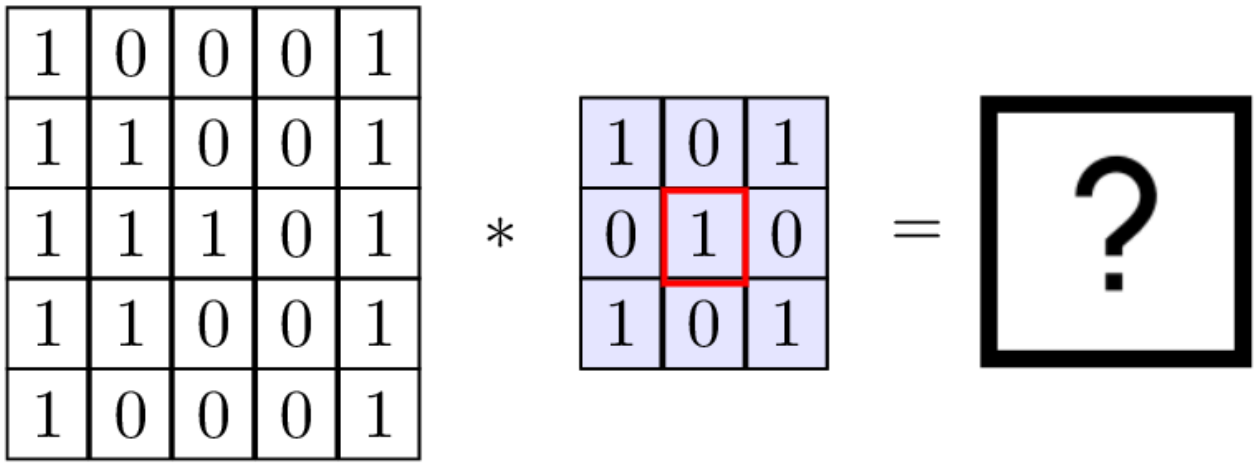
Convolution: Example



Convolution: Example



Convolution: How to determine the output image size?



- If there is an image of size $k \times l$ and we convolve it with an $m \times m$ filter, then the size of resulting output is $(k - m + 1) \times (k - m + 1)$.
- For example, if image $I_1 \in 5 \times 5$ and mask (kernel/filter) is $M \in 3 \times 3$ then the output is $I_2 \in 3 \times 3$.

Padding

- The convolution operation is shrinking our output image. If we perform this operation multiples times, we will lose a lot of data.
- To solve this problem, extend the image I_1 before convolution by adding some row, columns to it with virtual pixels.
- If an image is of size $k \times l$, mask is $m \times m$ then extend the image with p rows (top, bottom) and columns (left, right).
- The p value can be computed as $p = \left\lfloor \frac{m-1}{2} \right\rfloor$
- Mask M is usually odd because in this way, it has a central value.

Padding

Extend with zeros

5	3	3
1	1	2
6	3	1

0	0	0	0	0
0	5	3	3	0
0	1	1	2	0
0	6	3	1	0
0	0	0	0	0

Padding

Replicating Boundary

5	3	3
1	1	2
6	3	1

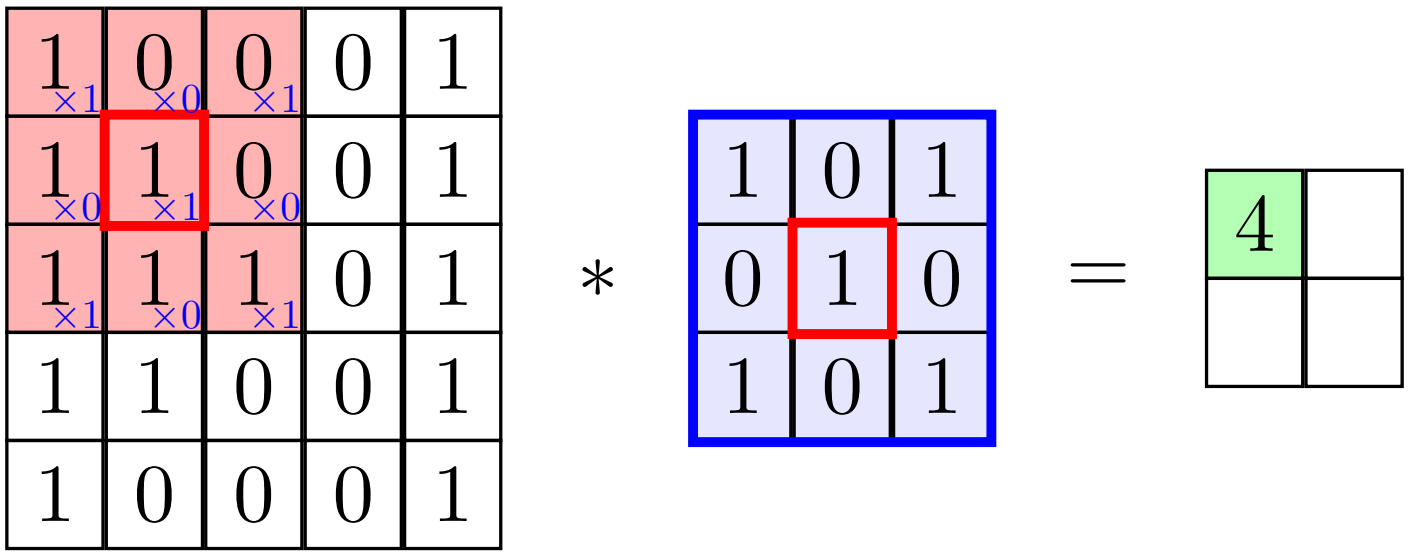
5	5	3	3	3
5	5	3	3	3
1	1	1	2	2
6	6	3	1	1
6	6	3	1	1

Strided convolution

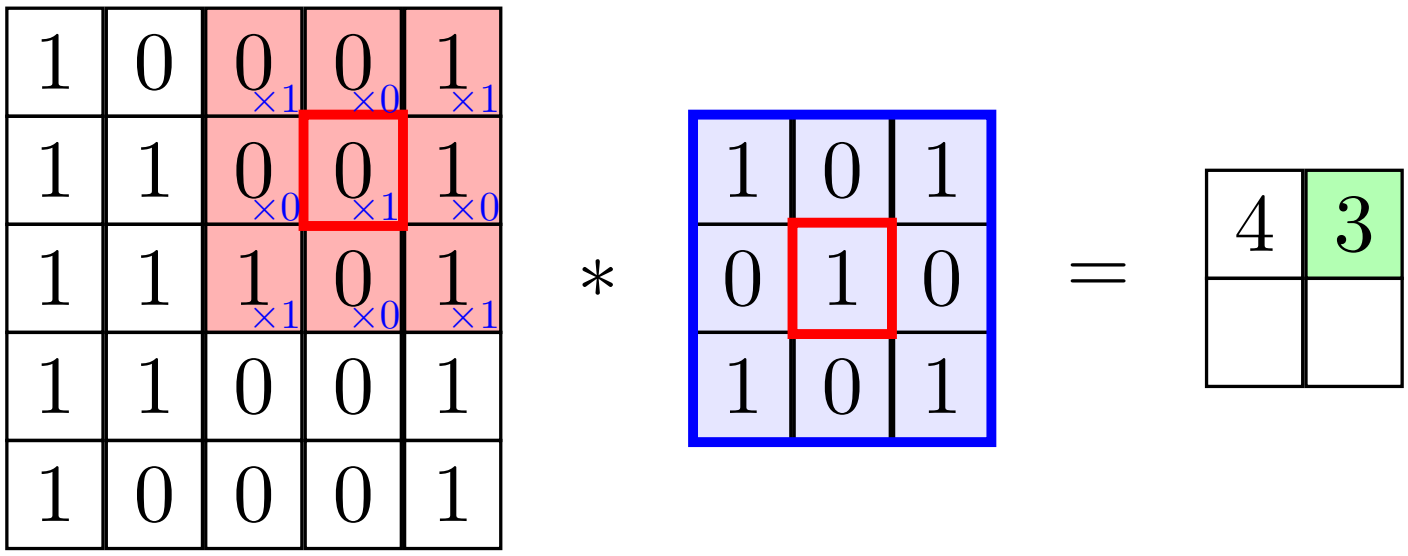
- In the above examples, the mask convolves the input image by moving one step at a time.
- Stride is the step size of the mask.
- We can change this step size value to obtain strided convolution.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 1 \\ 3 & 3 \end{bmatrix}$$

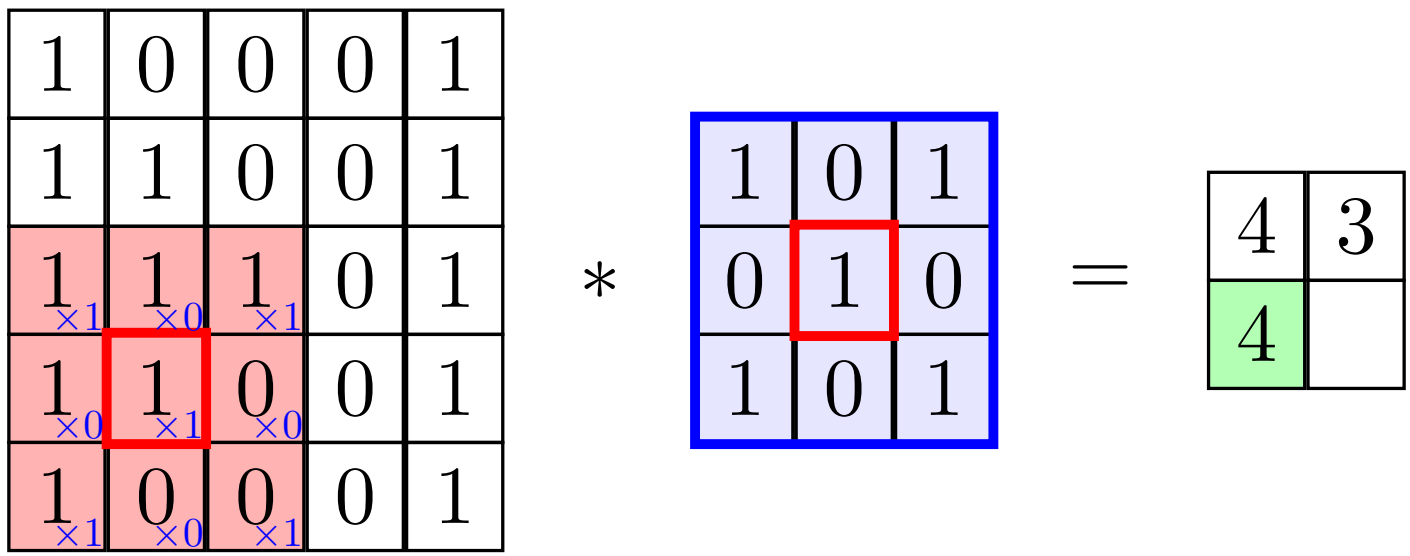
Strided convolution



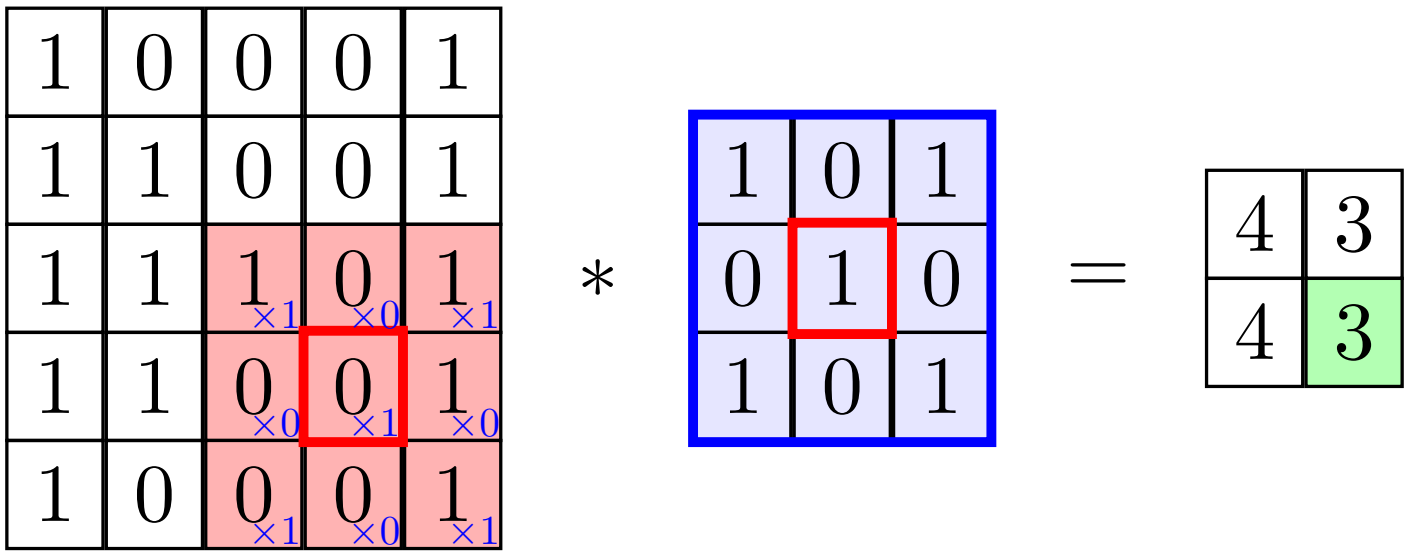
Strided convolution



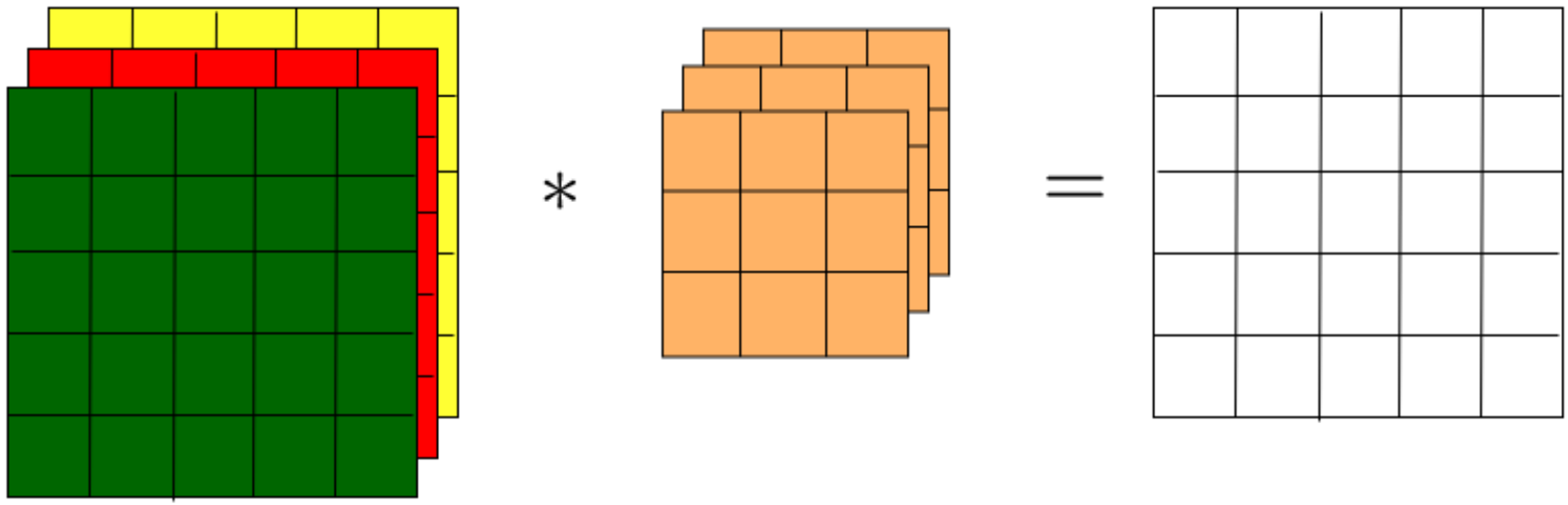
Strided convolution



Strided convolution



Convolution on RGB images



Summary

Image $I_{k\times l}$, mask $M_{m\times m}$, stride s and padding p

- Output image size: $\frac{k-m}{s}+1$
- Padding p when $s=1$: $\lfloor\frac{m-1}{2}\rfloor$
- Padding p when $s>1$: $\lfloor\frac{k*s-k+m-s}{2}\rfloor$
- Output image size (RGB):
  $k\times l\times n_c * m\times m\times n_c = (\frac{k-m}{s}+1)\times(\frac{k-m}{s}+1)\times n_f$
  where n_c is the number of input channels and n_f is the number of output filters.