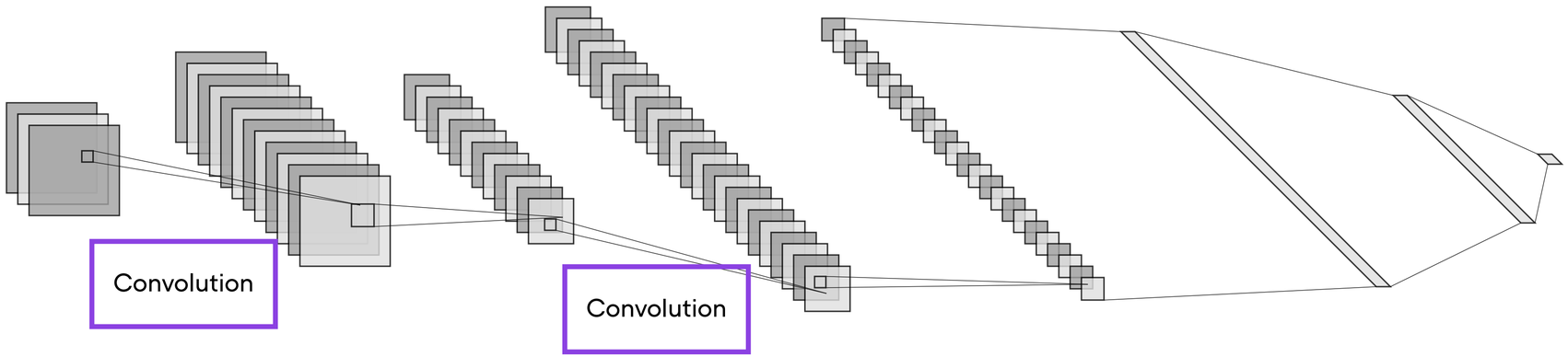
7.2

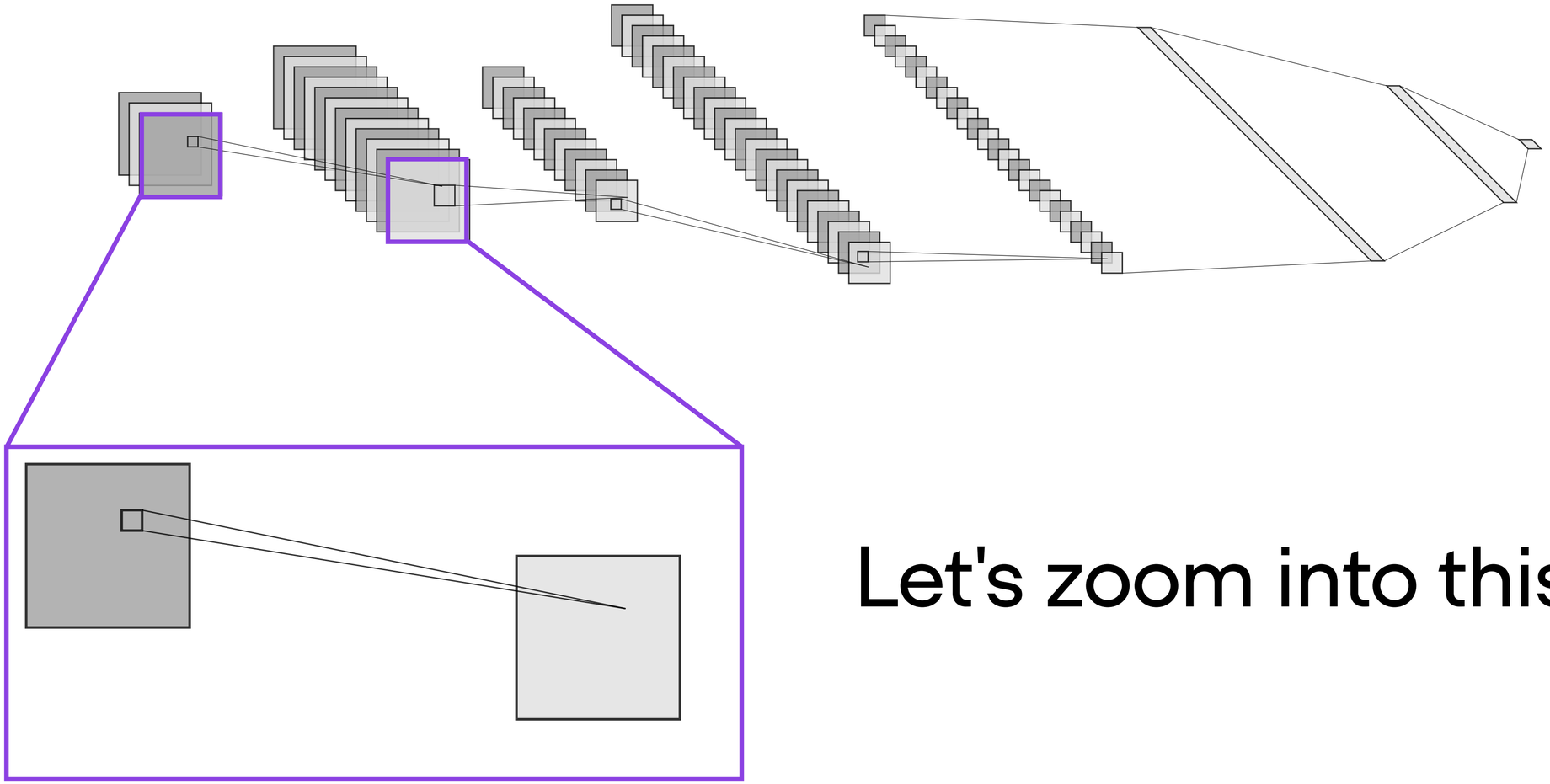
How Convolutional Neural Networks Work

Part 2: Convolutional Layers

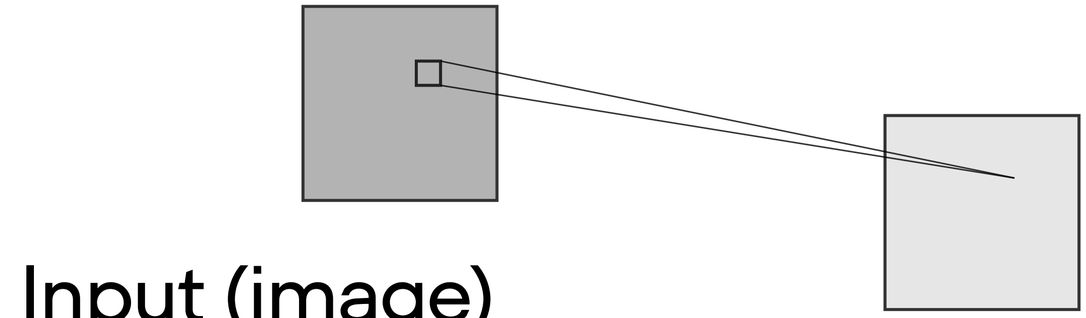
Sebastian Raschka and the Lightning AI Team

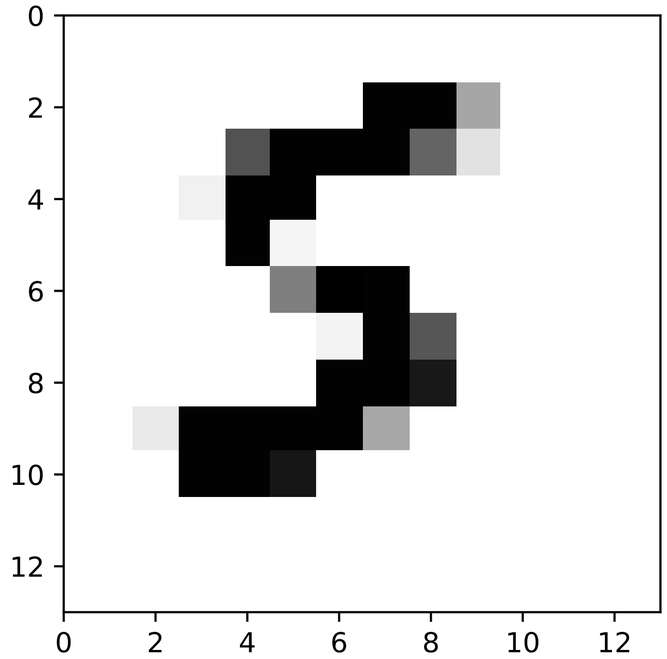
Looking at convolutional layers in more detail



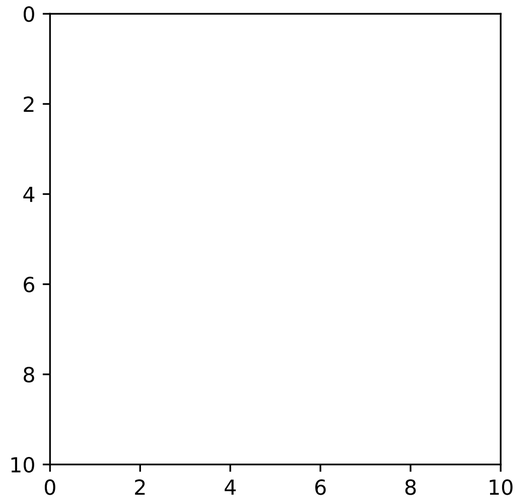


Let's zoom into this part

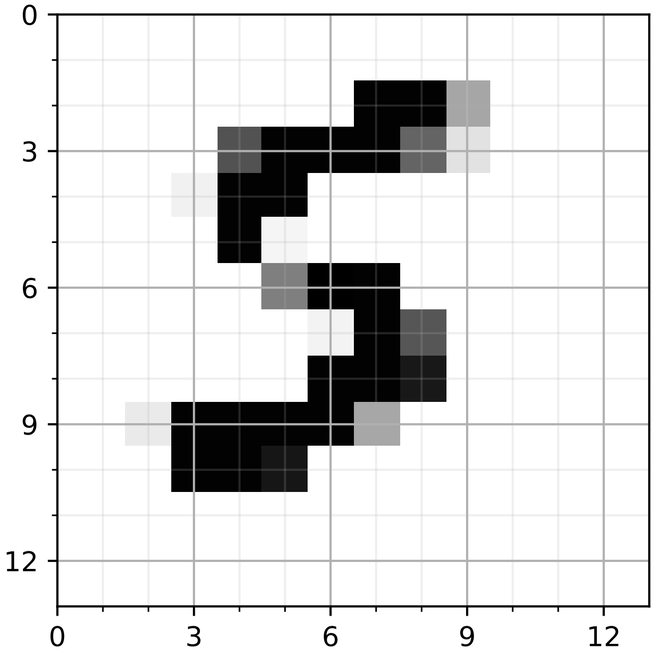

Input (image)
Feature map



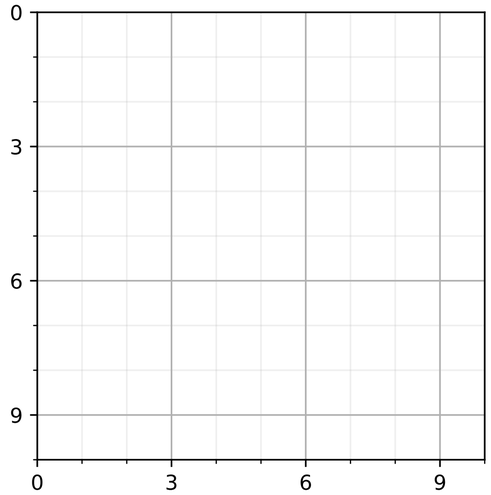
Input (image)



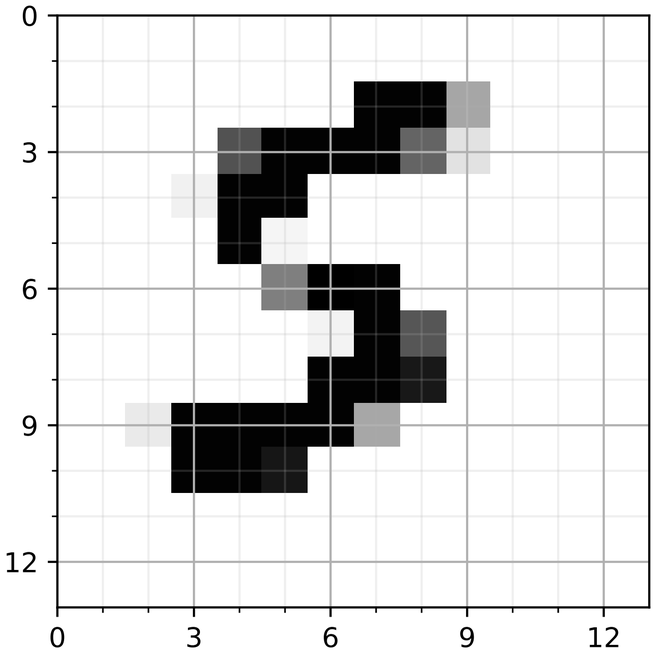
Feature map



Input (image)

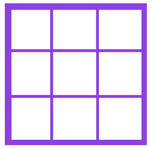


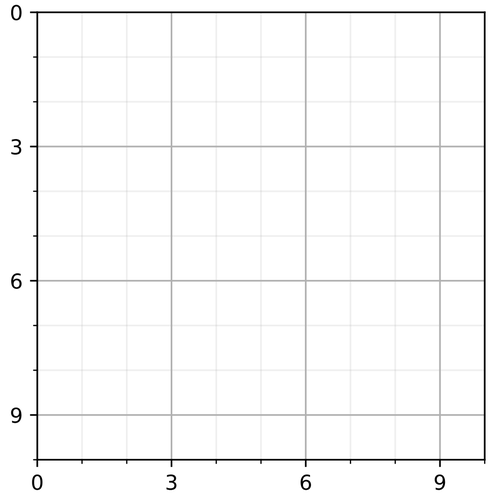
Feature map



Input (image)

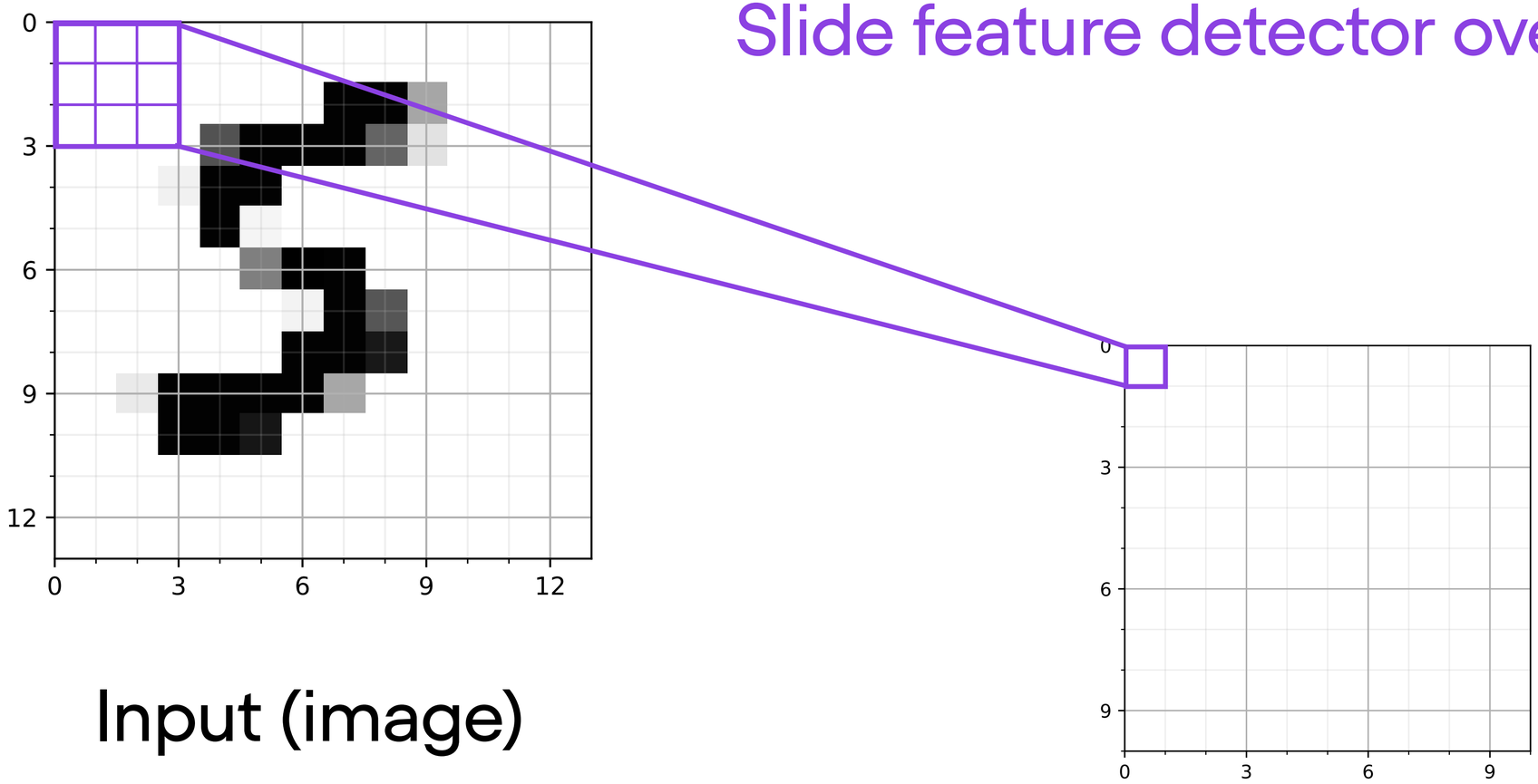
3x3 feature detector (kernel, filter)





Feature map

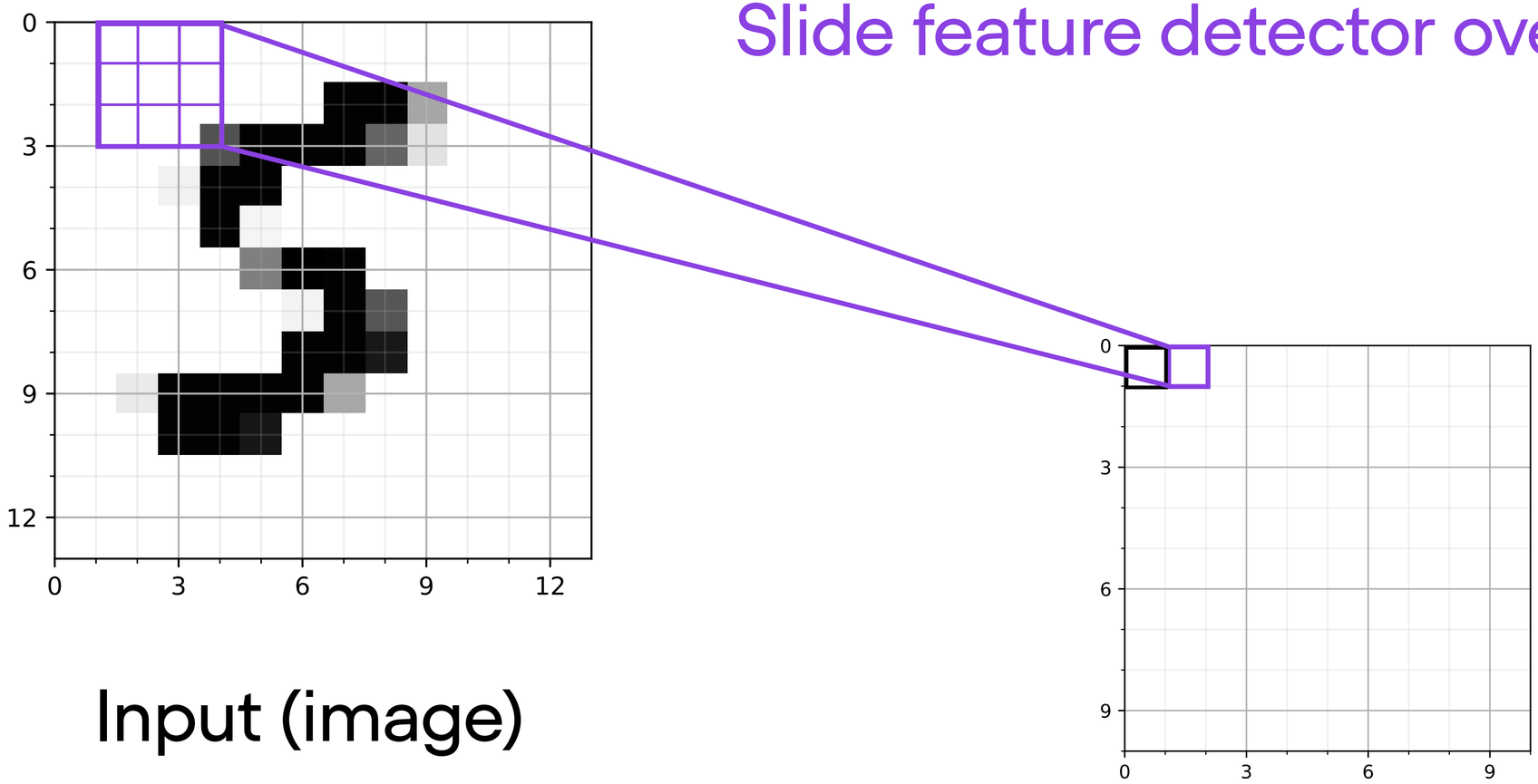
Slide feature detector over inputs



Input (image)

Feature map

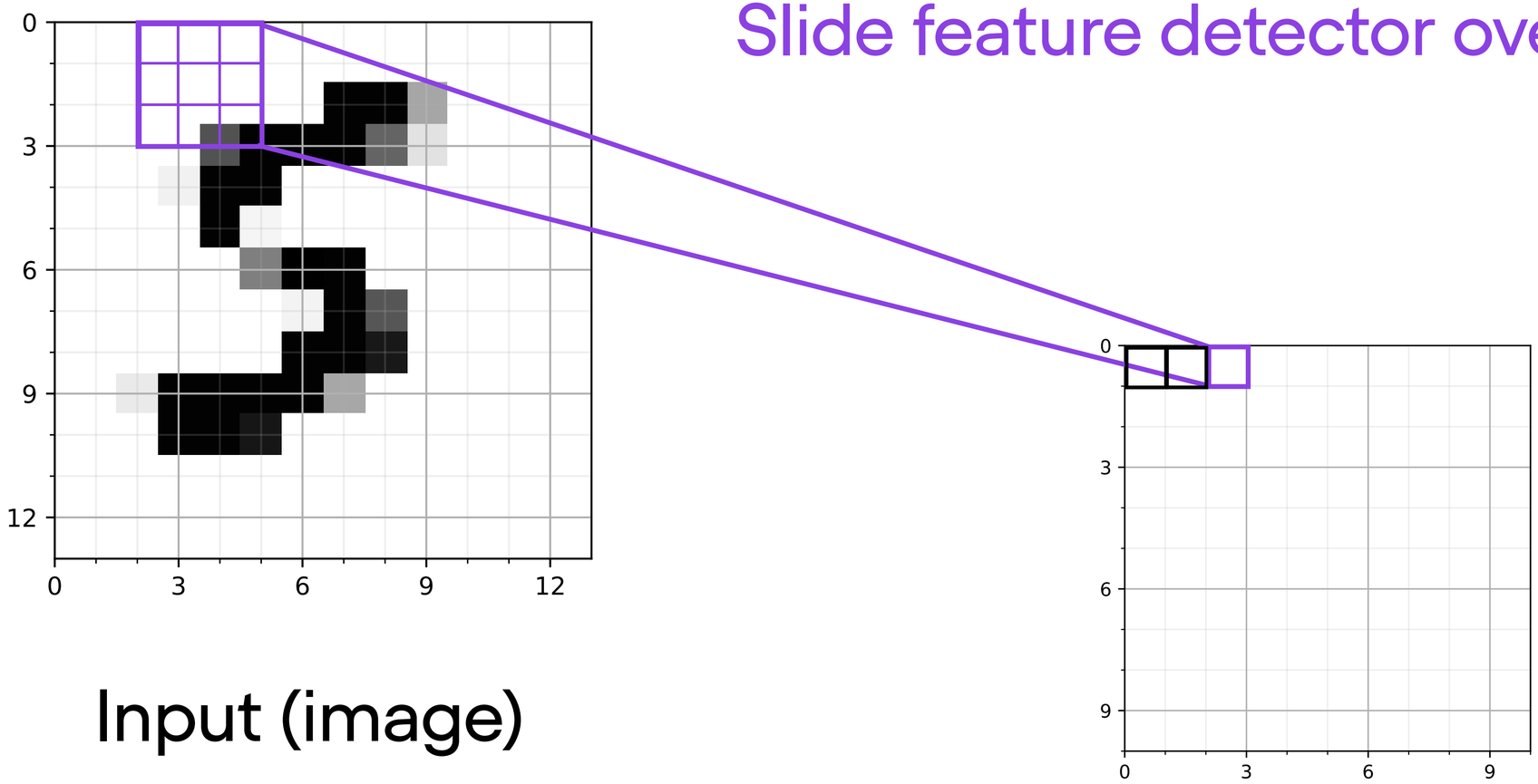
Slide feature detector over inputs



Input (image)

Feature map

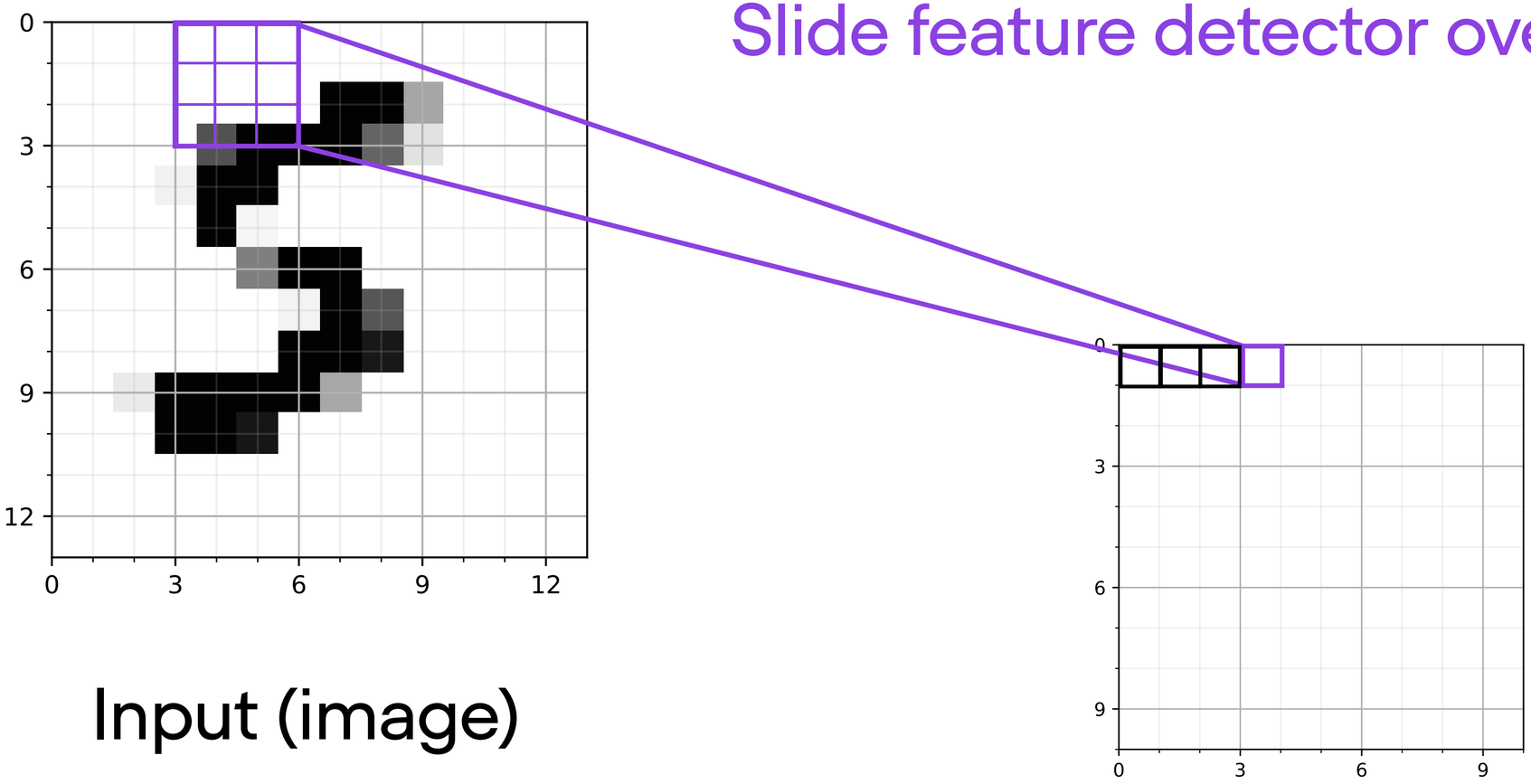
Slide feature detector over inputs



Input (image)

Feature map

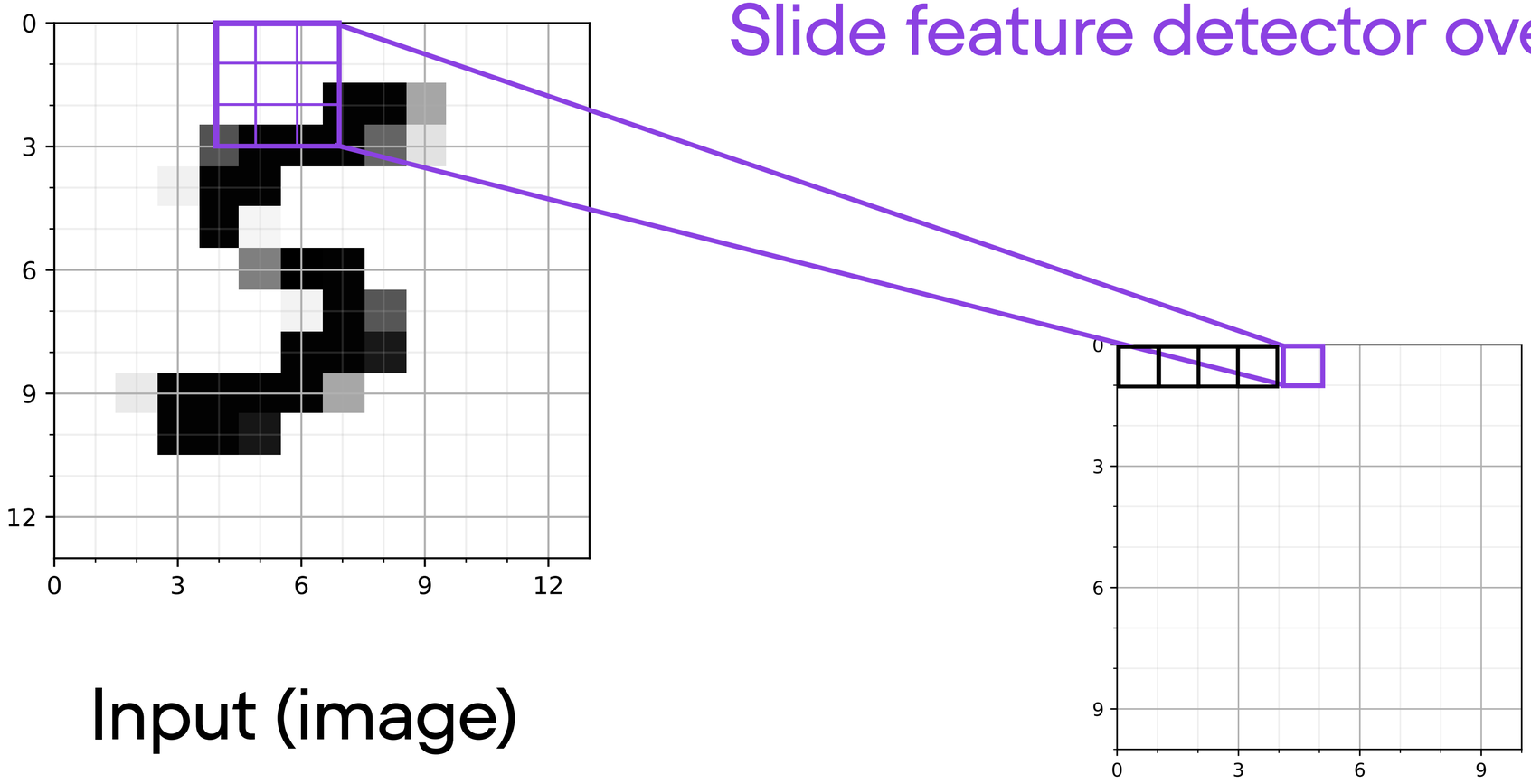
Slide feature detector over inputs



Input (image)

Feature map

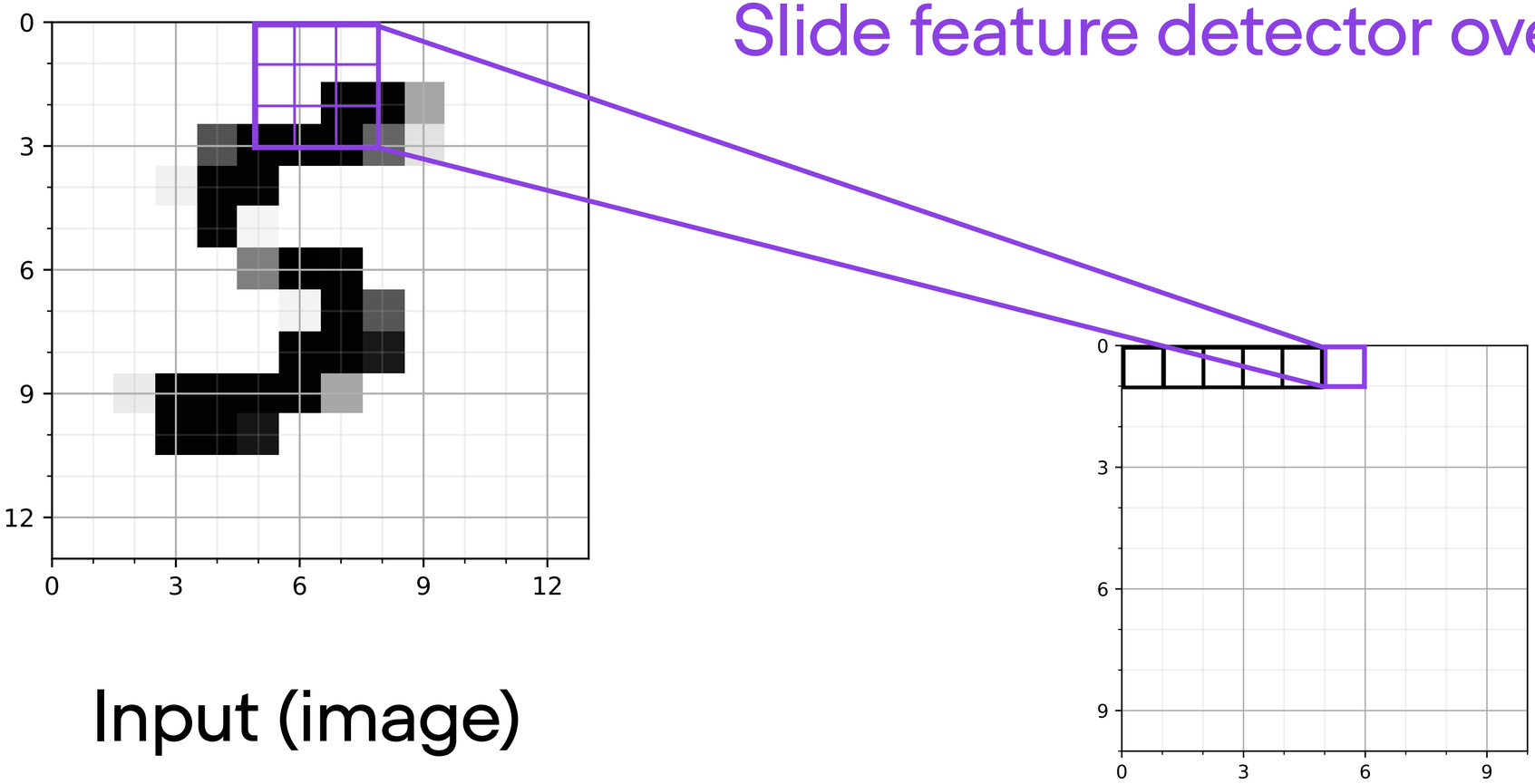
Slide feature detector over inputs



Input (image)

Feature map

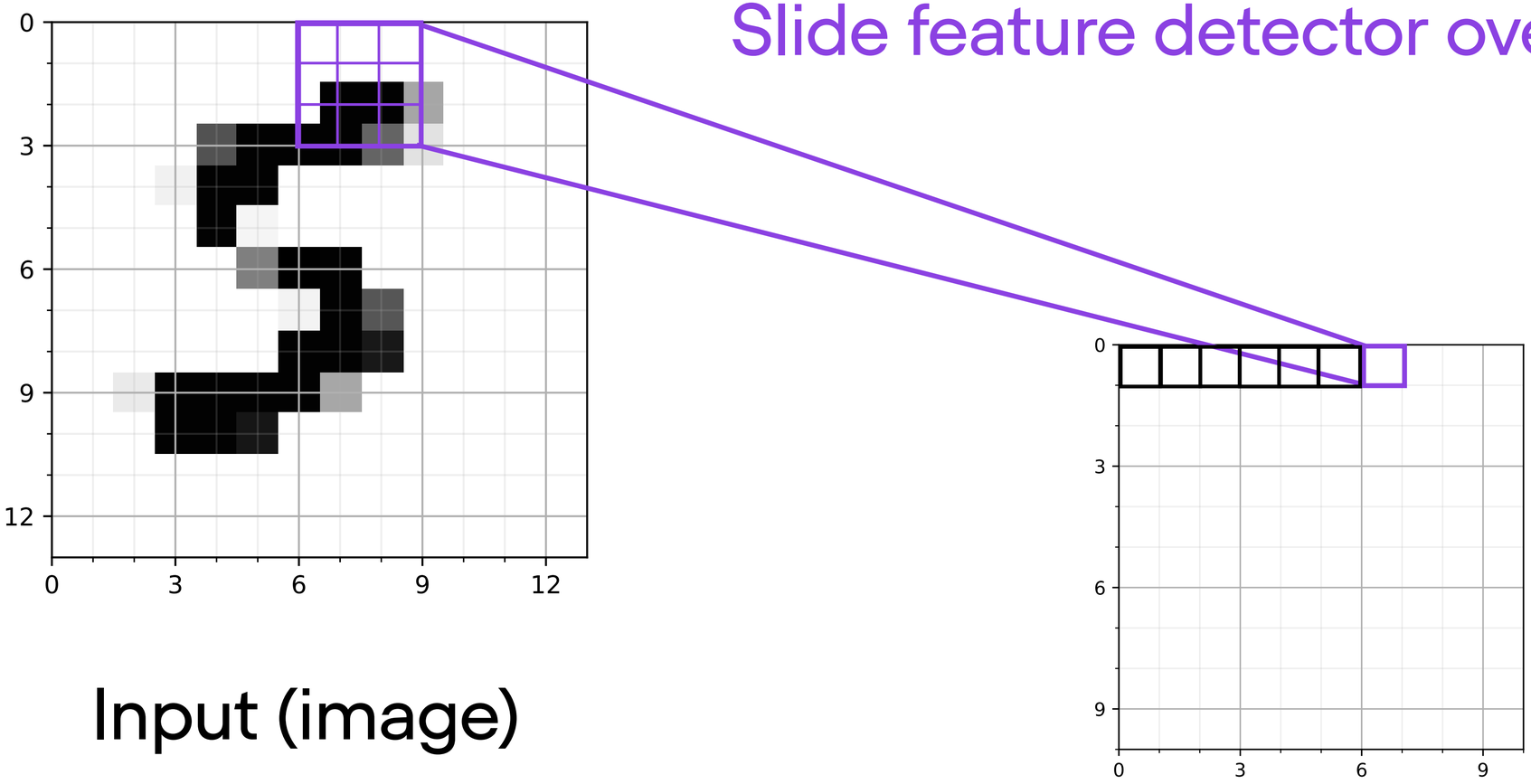
Slide feature detector over inputs



Input (image)

Feature map

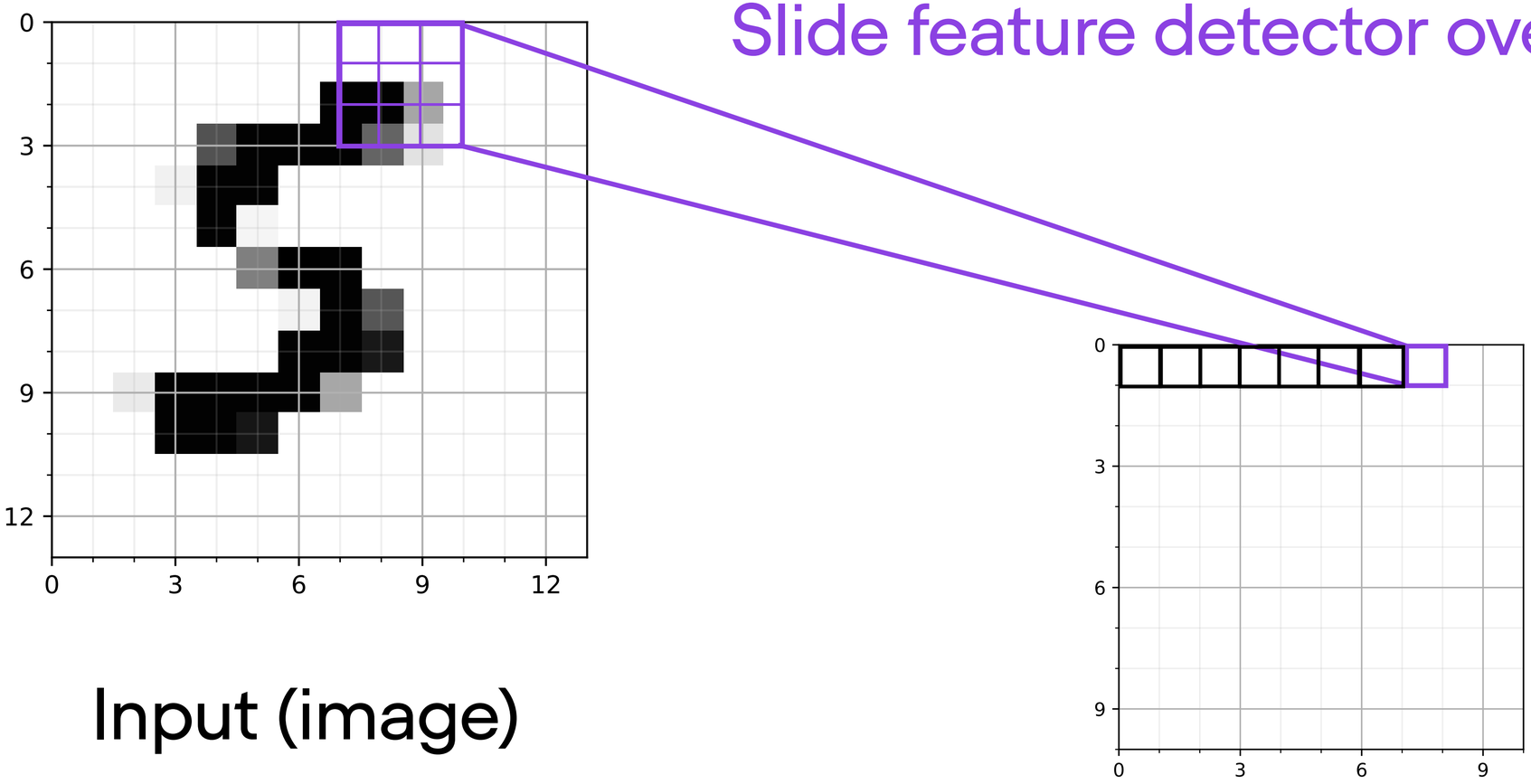
Slide feature detector over inputs



Input (image)

Feature map

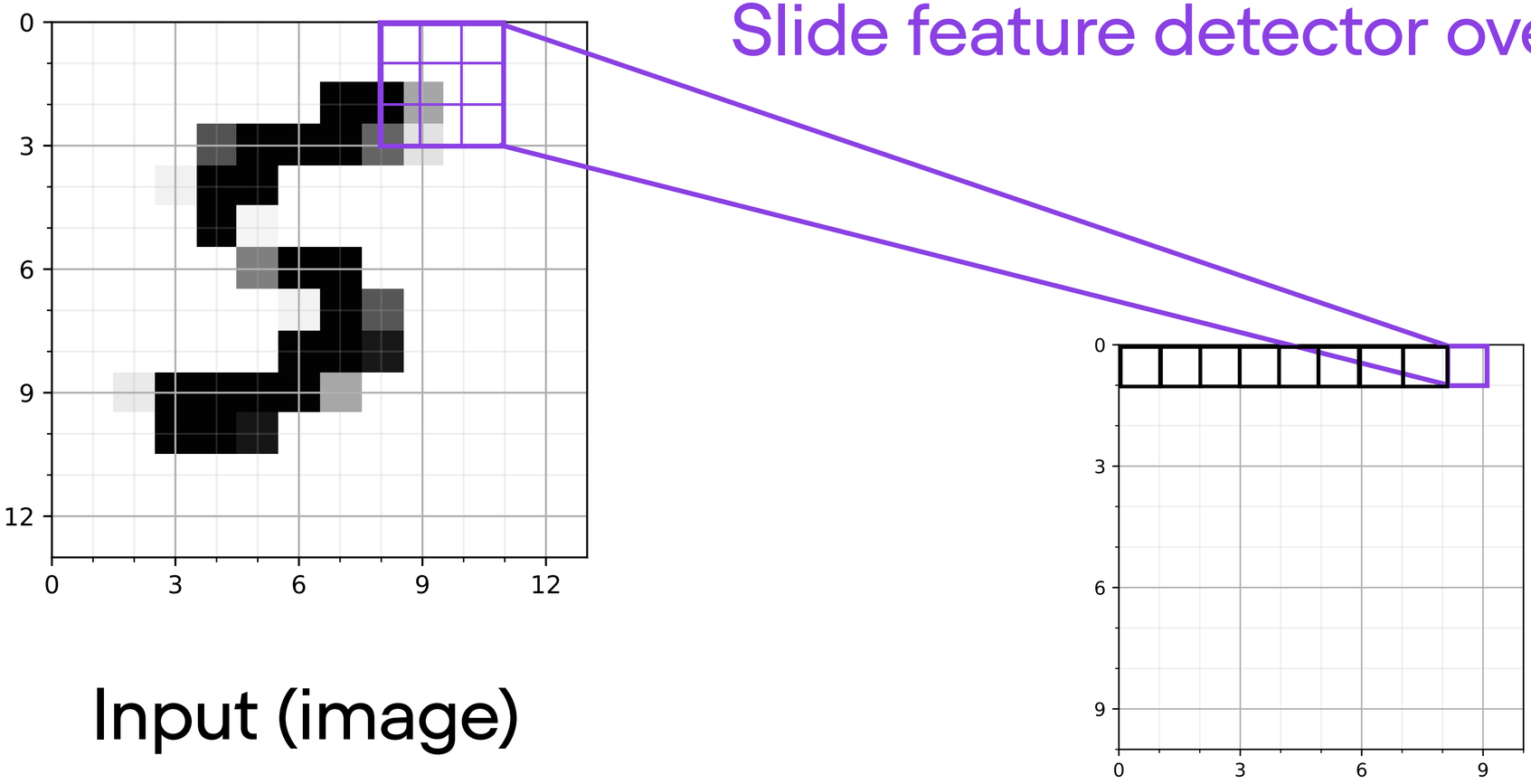
Slide feature detector over inputs



Input (image)

Feature map

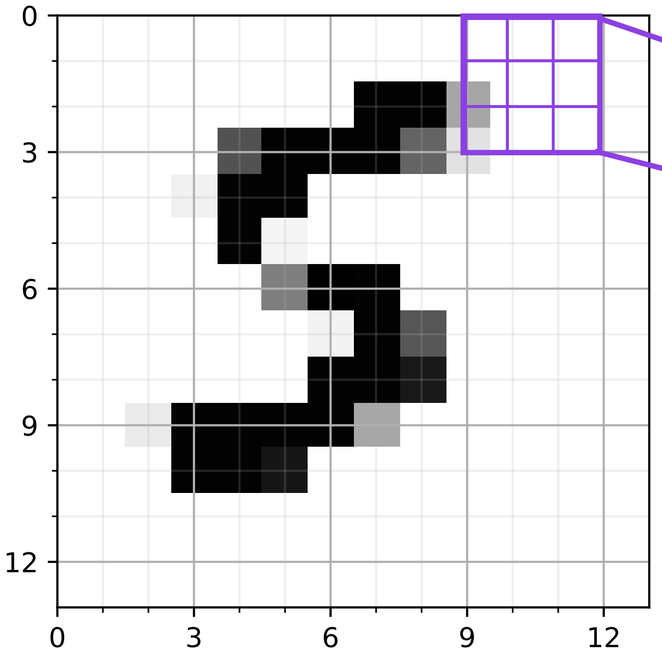
Slide feature detector over inputs



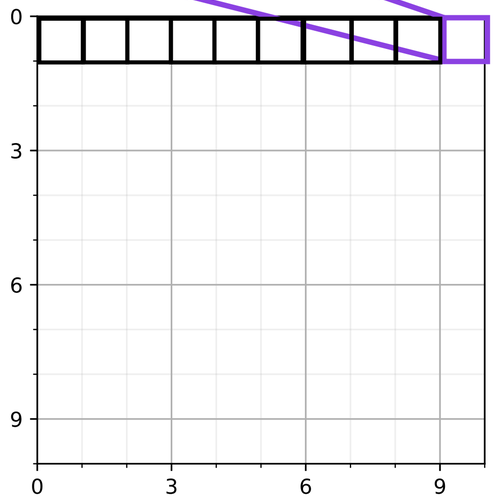
Input (image)

Feature map

Slide feature detector over inputs

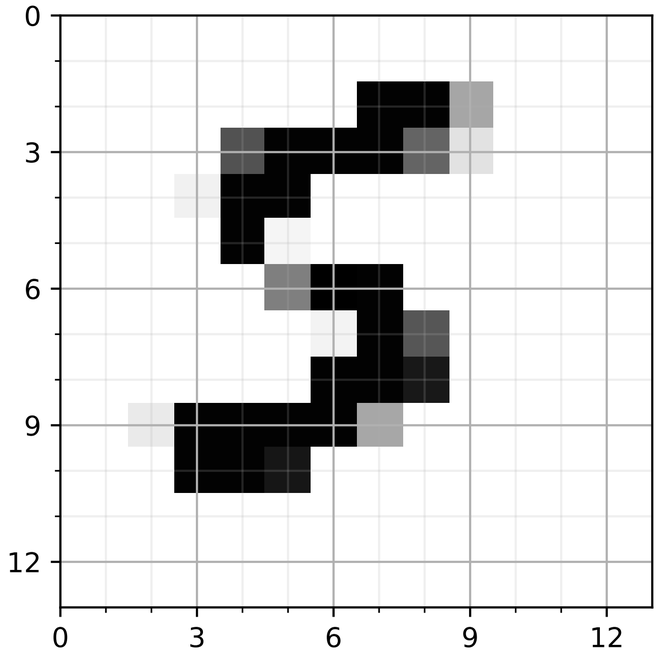


Input (image)

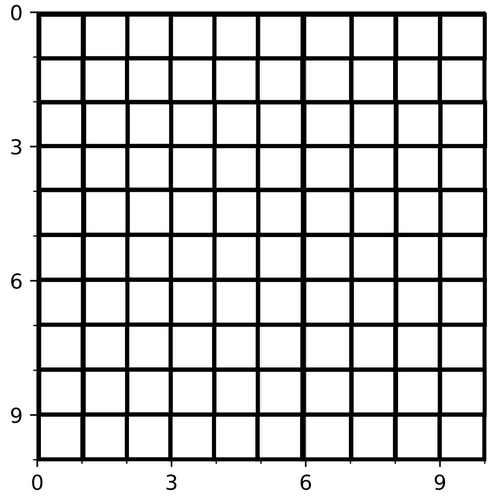


Feature map

Slide feature detector over inputs

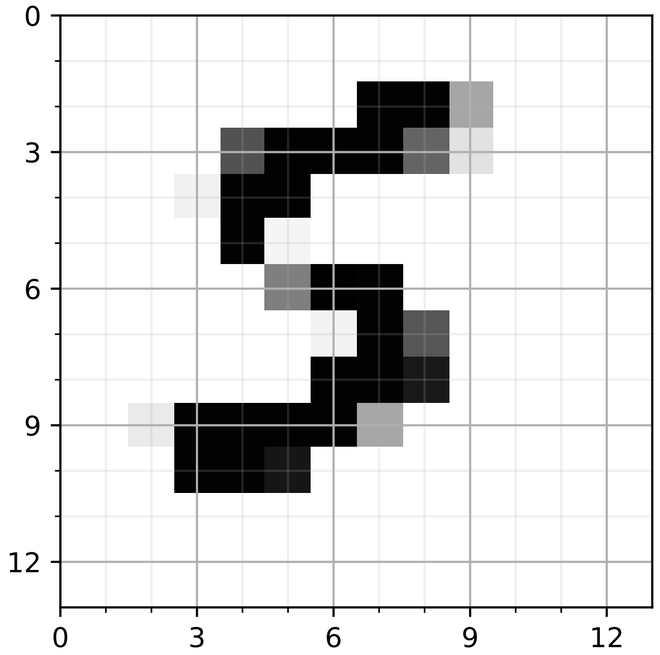


Input (image)

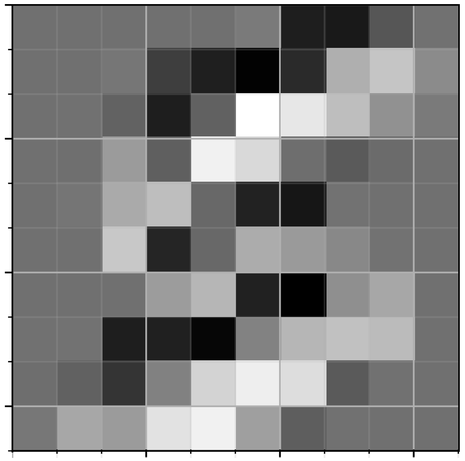


Feature map

Slide feature detector over inputs

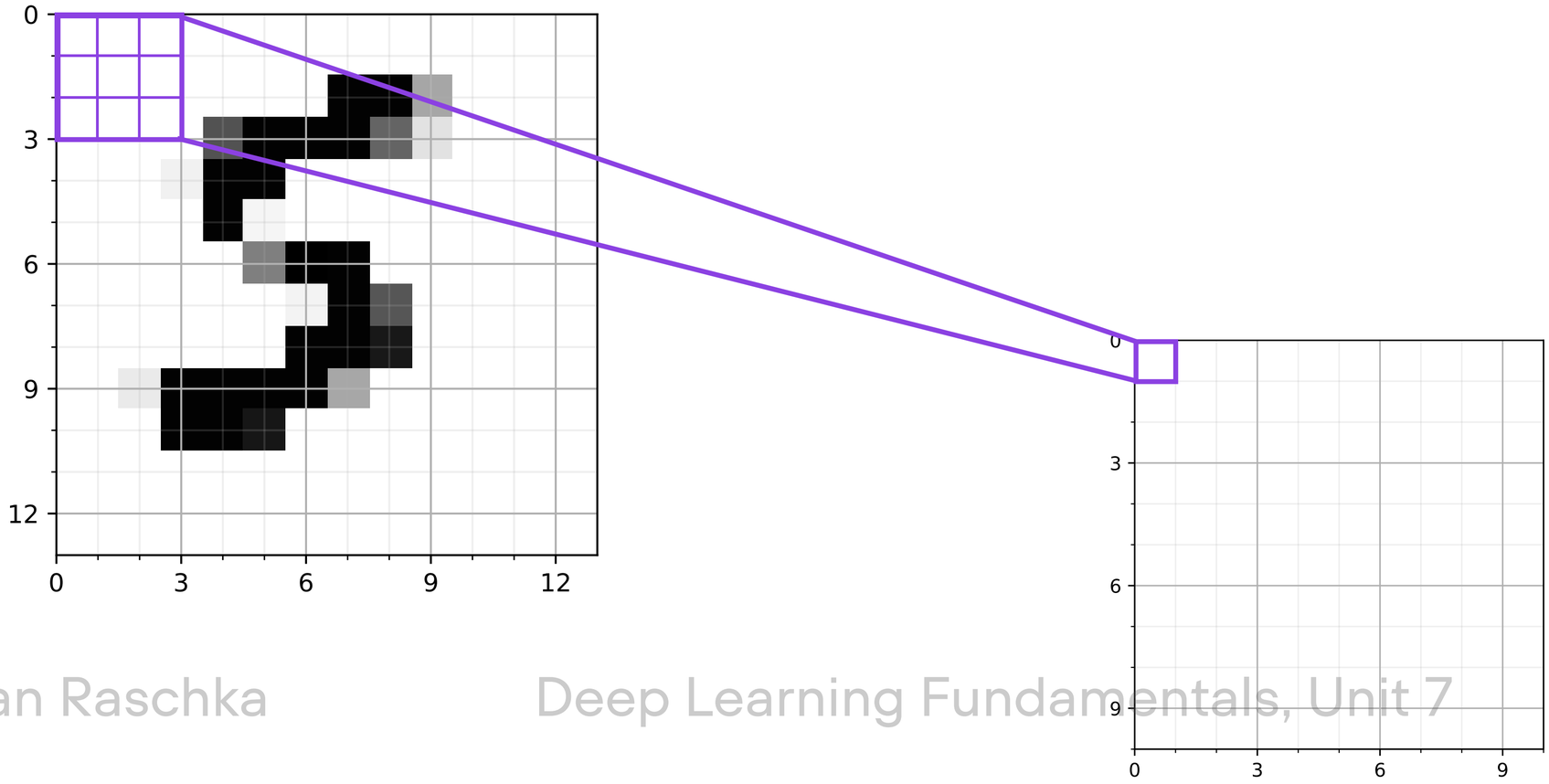


Input (image)

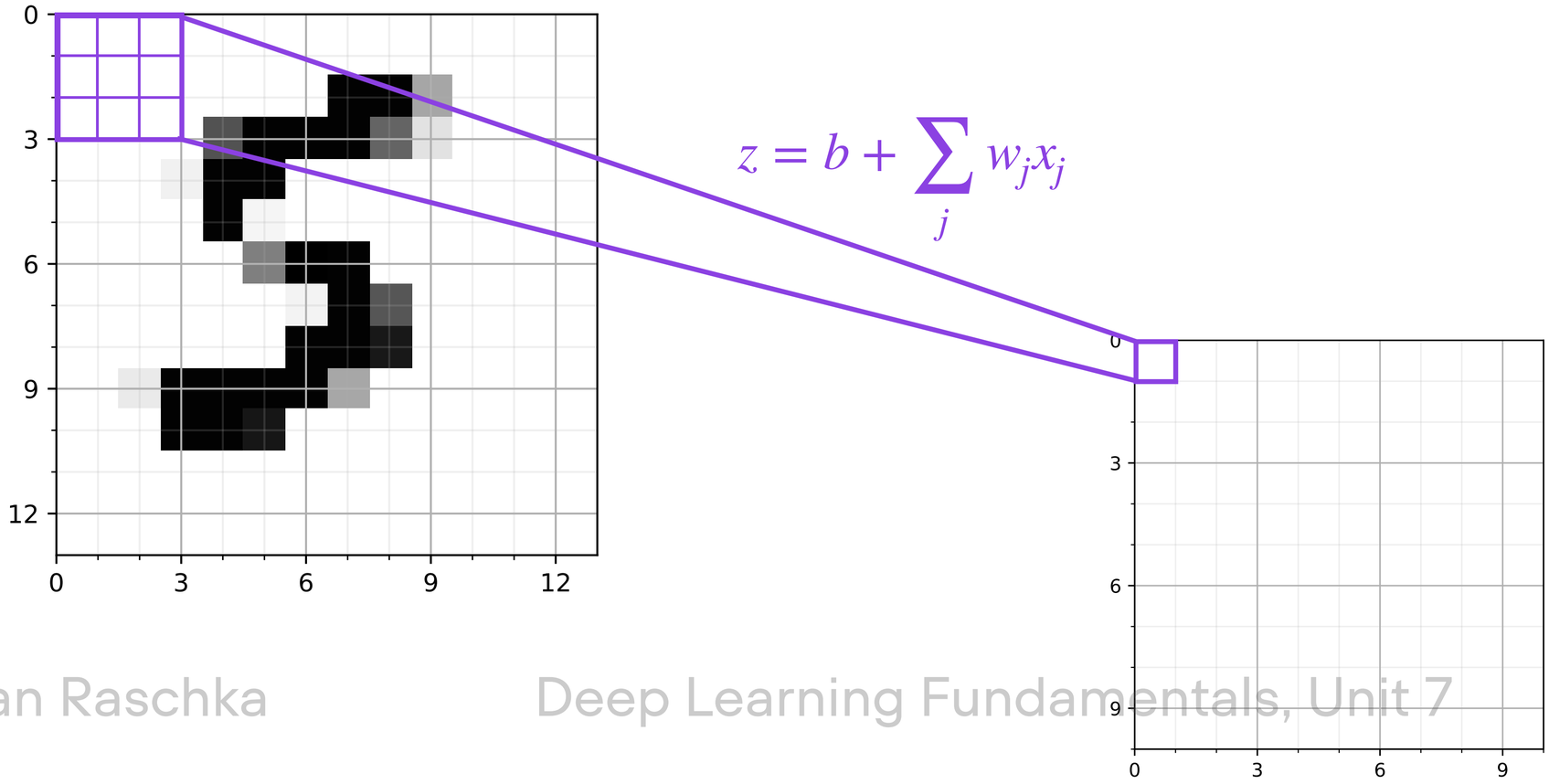


Feature map

What is happening during this operation?

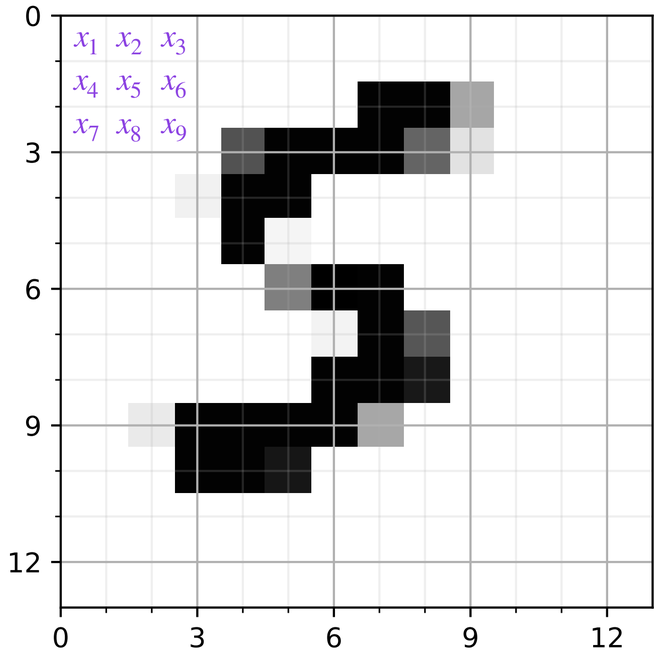


What is happening during this operation?

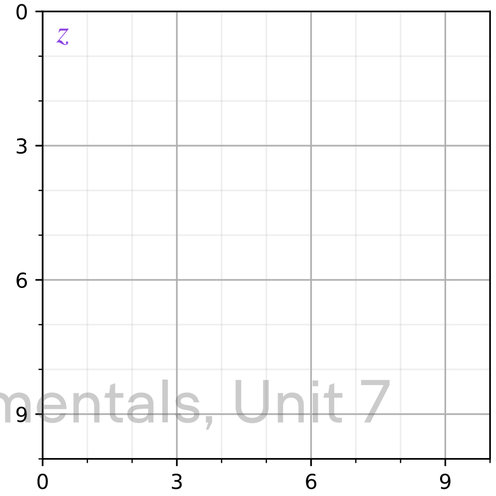


What is happening during this operation?

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9



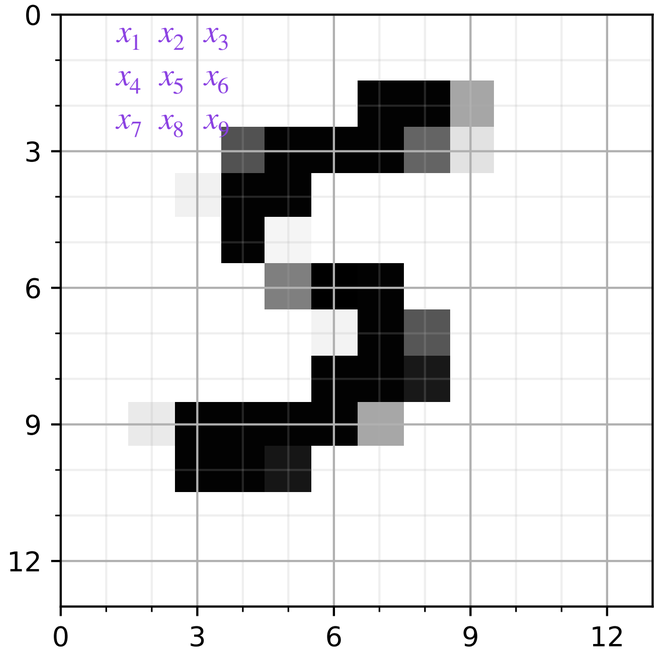
$$z = b + \sum_j w_j x_j$$



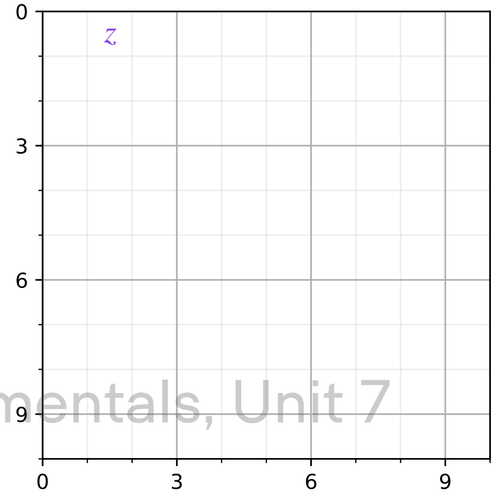
The inputs (x's) differ as we slide over the image.

The weights (w's) do not differ.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9



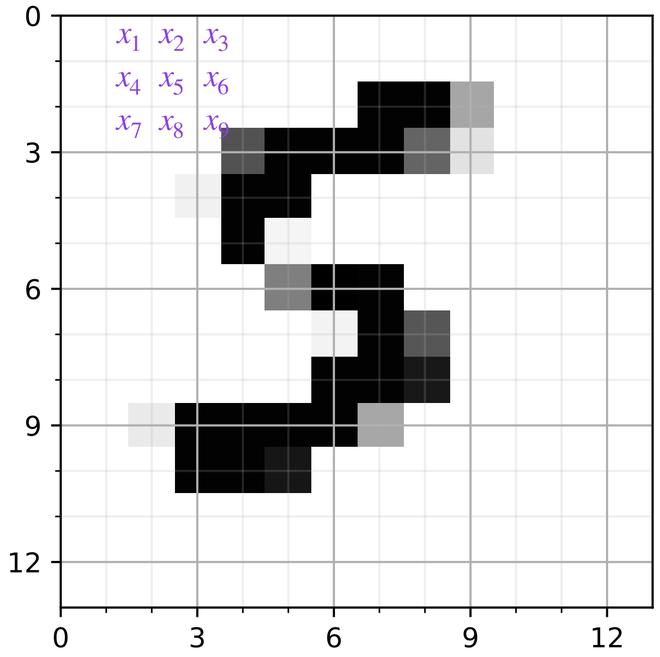
$$z = b + \sum_j w_j x_j$$



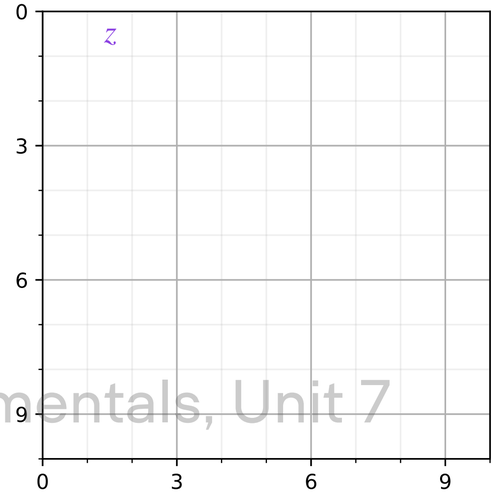
The inputs (x's) differ as we slide over the image.

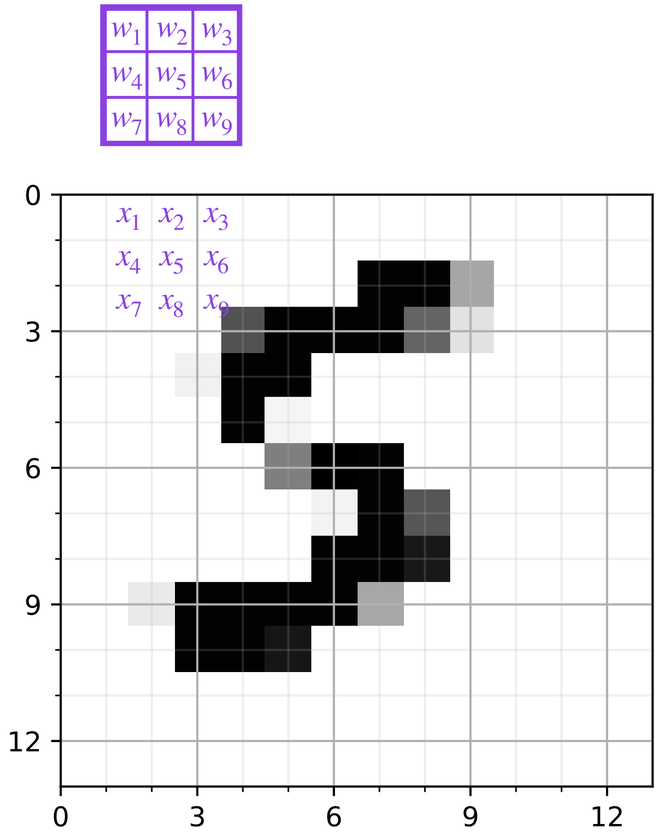
The weights (w's) do not differ. → weight sharing

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9



$$z = b + \sum_j w_j x_j$$

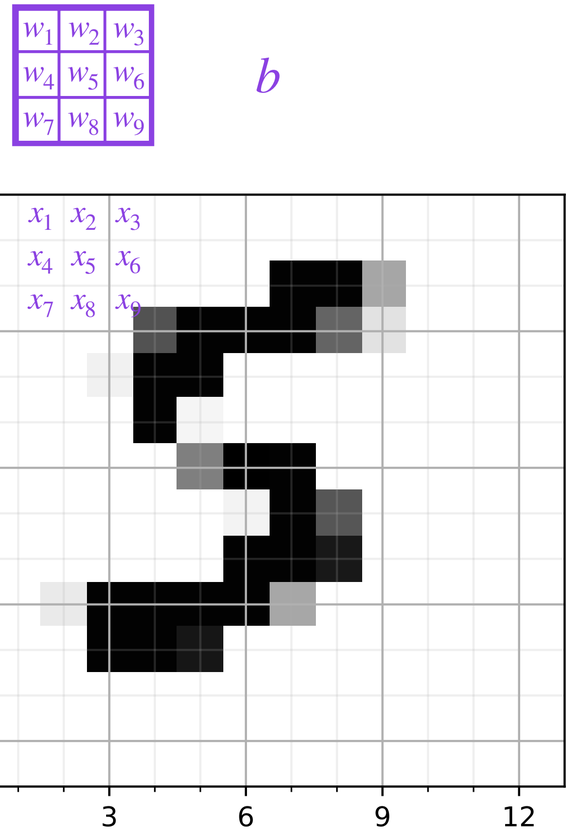




weight sharing

- Rationale: A feature detector that works well in one region may also work well in another region
- A reduction in parameters to fit (compared to MLP)

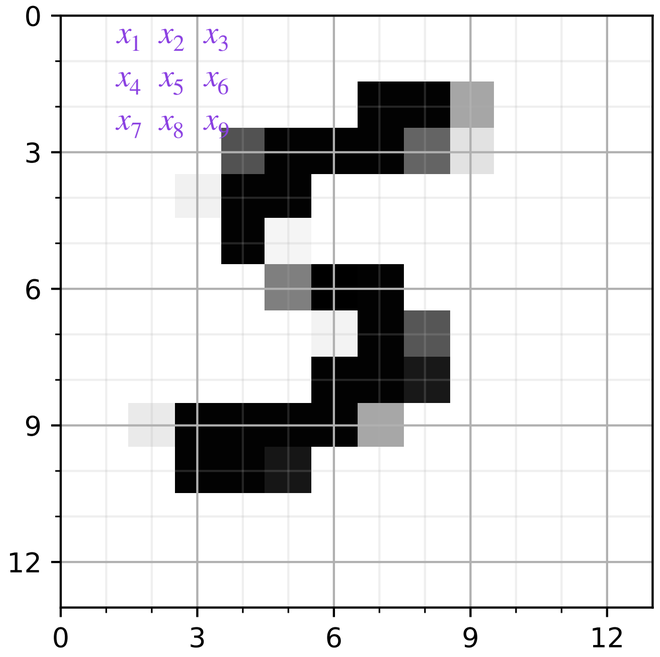
Convolutional layer



Consists of weights

Convolutional layer

$$\begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix} \quad b$$



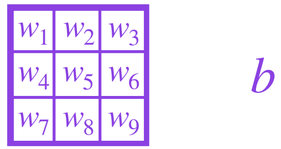
Consists of weights

```
layer = torch.nn.Conv2d(1, 1, kernel_size=3)
```

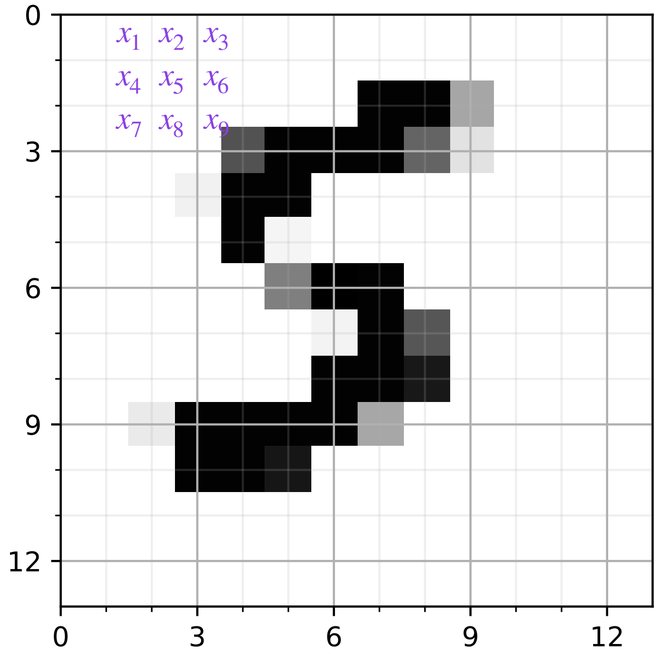
```
layer.weight
```

```
Parameter containing:
tensor([[[[ 0.0930,  0.1602,  0.1177],
          [-0.0802, -0.0701, -0.2747],
          [ 0.1806,  0.2647,  0.2281]]]], requires_grad=True)
```

Convolutional layer



b



Consists of weights & bias unit(s)

```
layer = torch.nn.Conv2d(1, 1, kernel_size=3)
```

```
layer.weight
```

```
Parameter containing:
tensor([[[[ 0.0930,  0.1602,  0.1177],
          [-0.0802, -0.0701, -0.2747],
          [ 0.1806,  0.2647,  0.2281]]]], requires_grad=True)
```

```
layer.bias
```

```
Parameter containing:
tensor([-0.2351], requires_grad=True)
```

Of course, we are free to choose different kernel sizes

```
layer = torch.nn.Conv2d(1, 1, kernel_size=3)
```

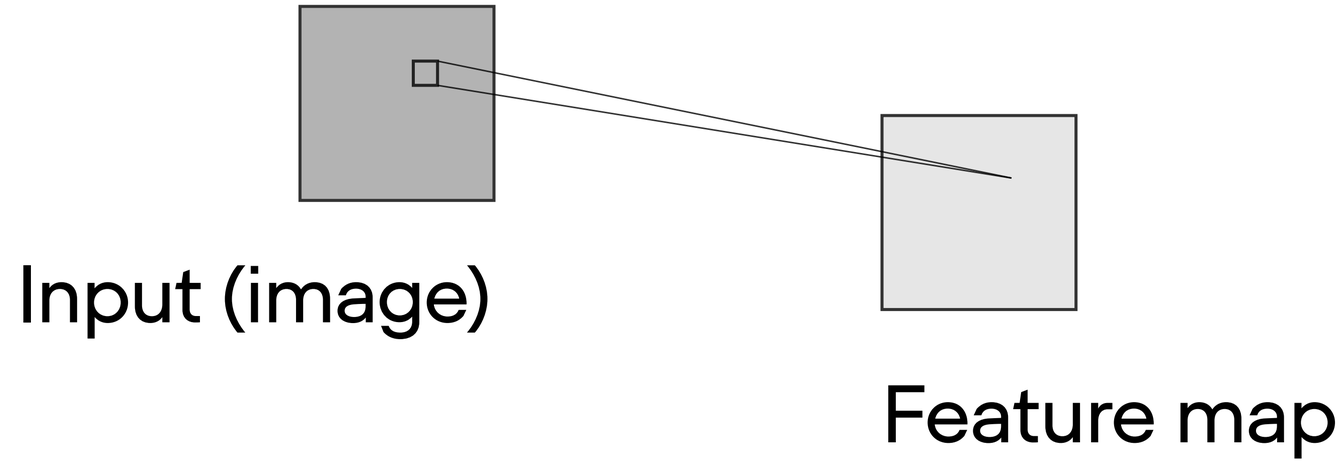
```
layer.weight
```

```
Parameter containing:
tensor([[[[ 0.0930,  0.1602,  0.1177],
          [-0.0802, -0.0701, -0.2747],
          [ 0.1806,  0.2647,  0.2281]]]], requires_grad=True)
```

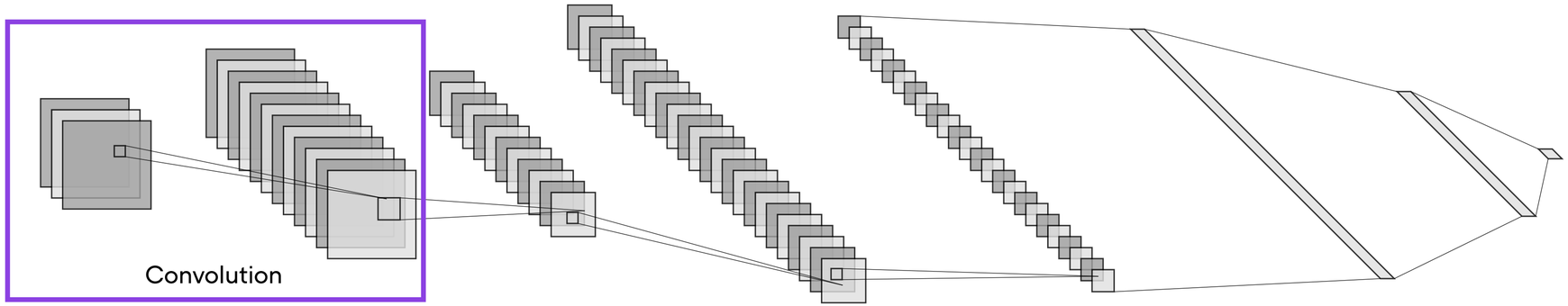
```
layer.bias
```

```
Parameter containing:
tensor([-0.2351], requires_grad=True)
```

So far, we looked at an excerpt



So far, we looked at an excerpt



What about the other channels?

NEXT: let's learn about convolutions with multiple channels