CONVOLUTIONAL NEURAL NETWORKS

CNN / ConvNet

IN LESS THAN 50 MINUTES

PAUSING BEFORE STARTING

START DOWNLOAD OF DOG / CAT DATA

PREVIOUS MODEL

```
from keras import models

from keras import layers


network = models.Sequential()
network.add(layers.Dense(512, activation='relu',
            input_shape=(28 * 28,)))

network.add(layers.Dense(256, activation='relu'))

network.add(layers.Dense(10, activation='softmax'))
```

LAYERS.DENSE
ALL INPUTS CONNECTED TO ALL NODES

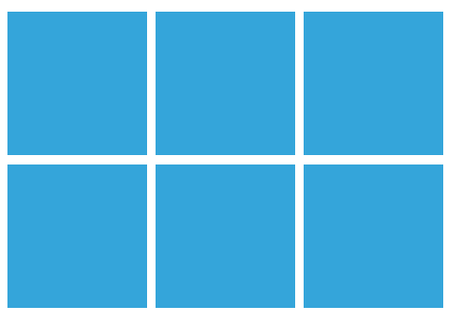
CONVNET MODEL

```
from keras import models

from keras import layers
```

```
model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(28, 28, 1)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
```

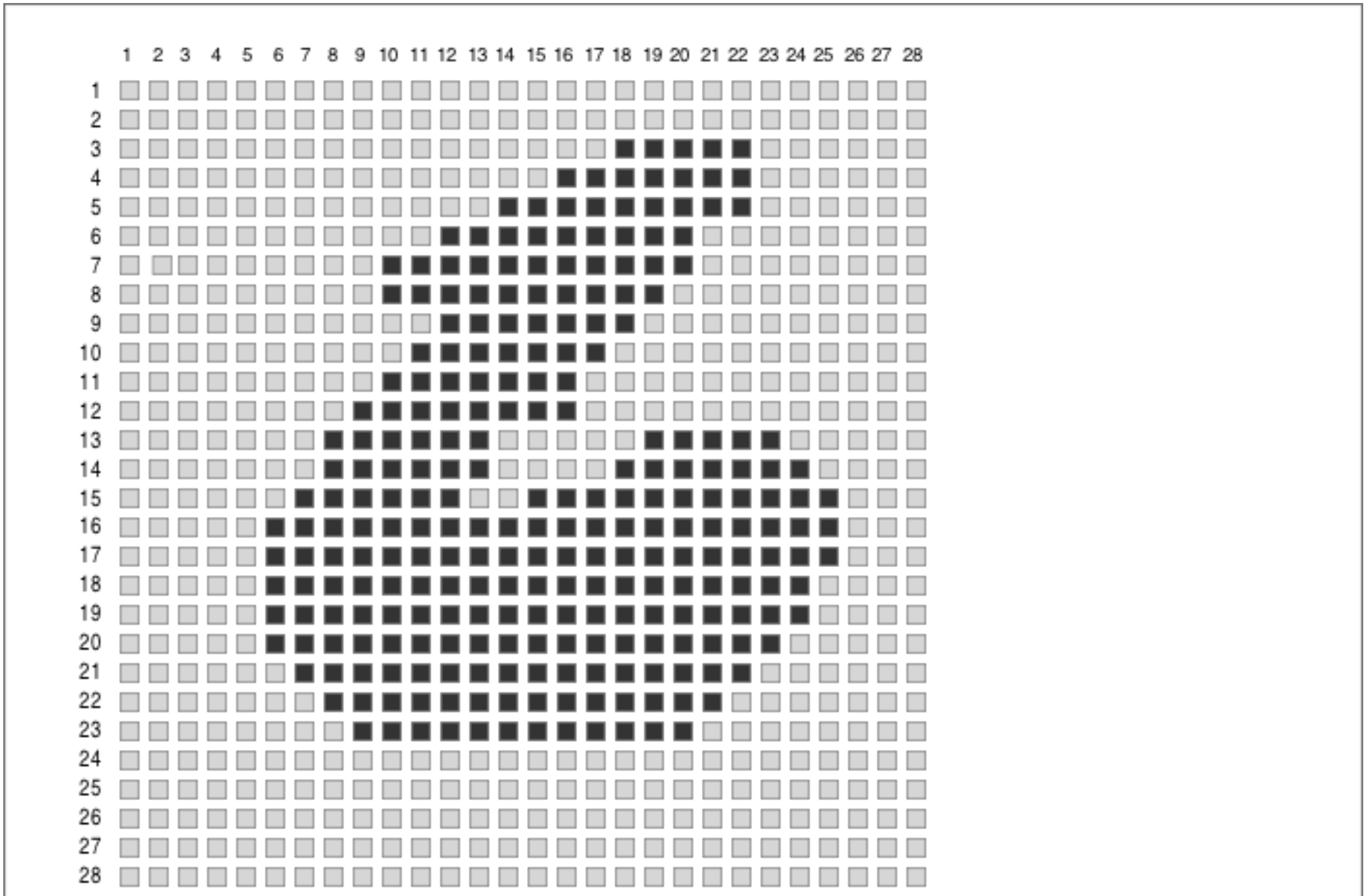
CONVNET MODEL

```
from keras import models

from keras import layers
```

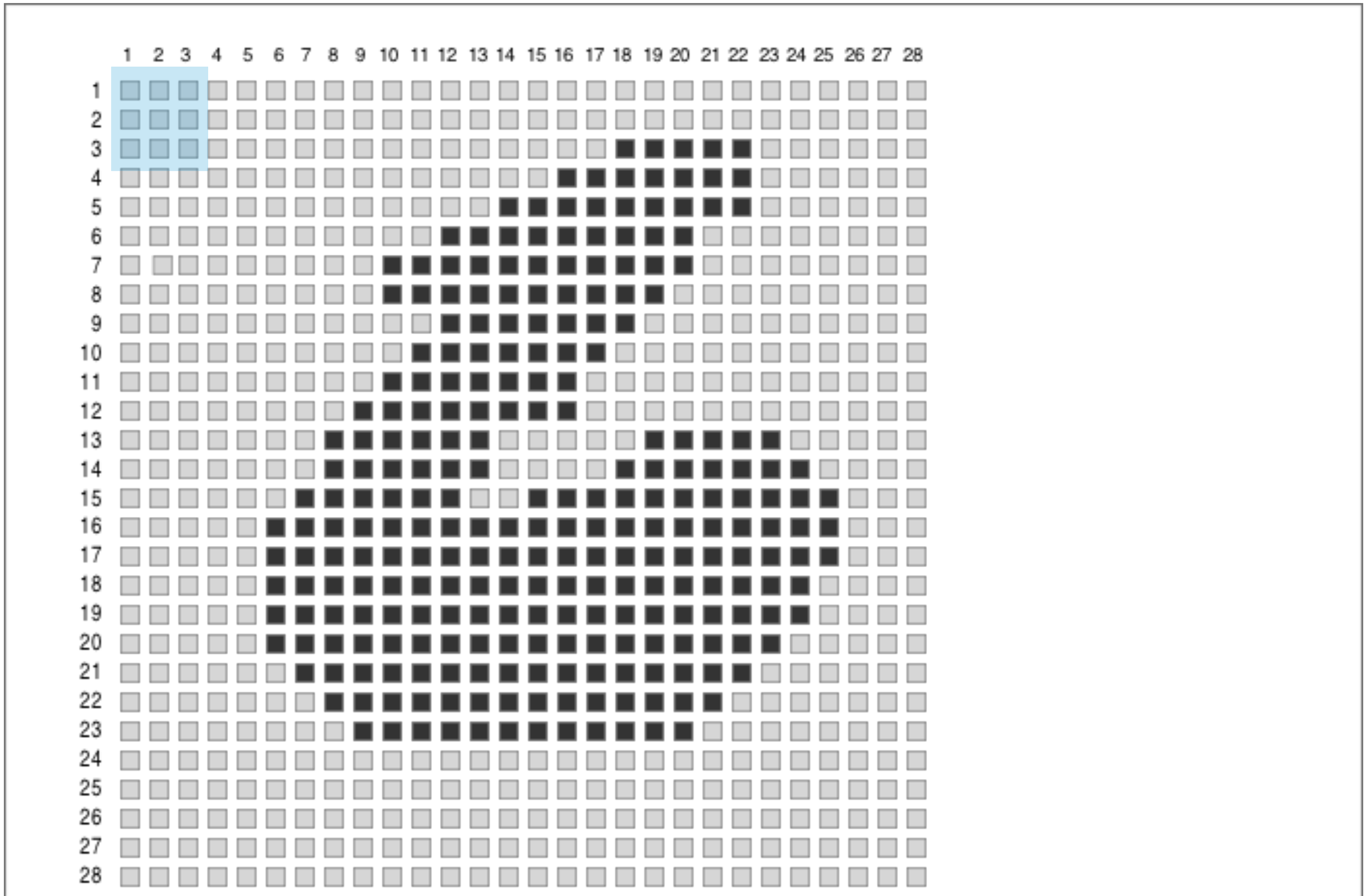


```
model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(28, 28, 1)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
```

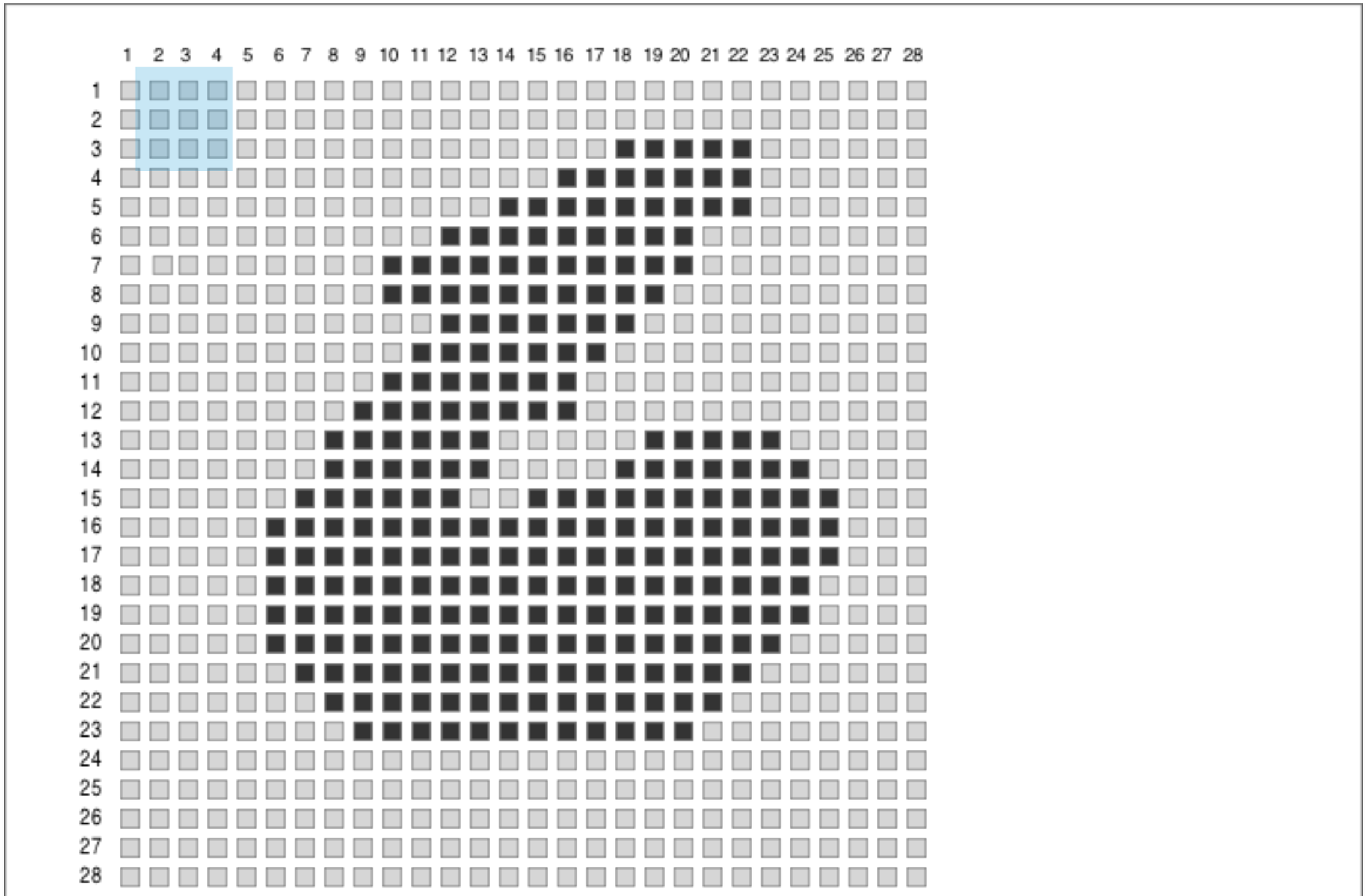
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

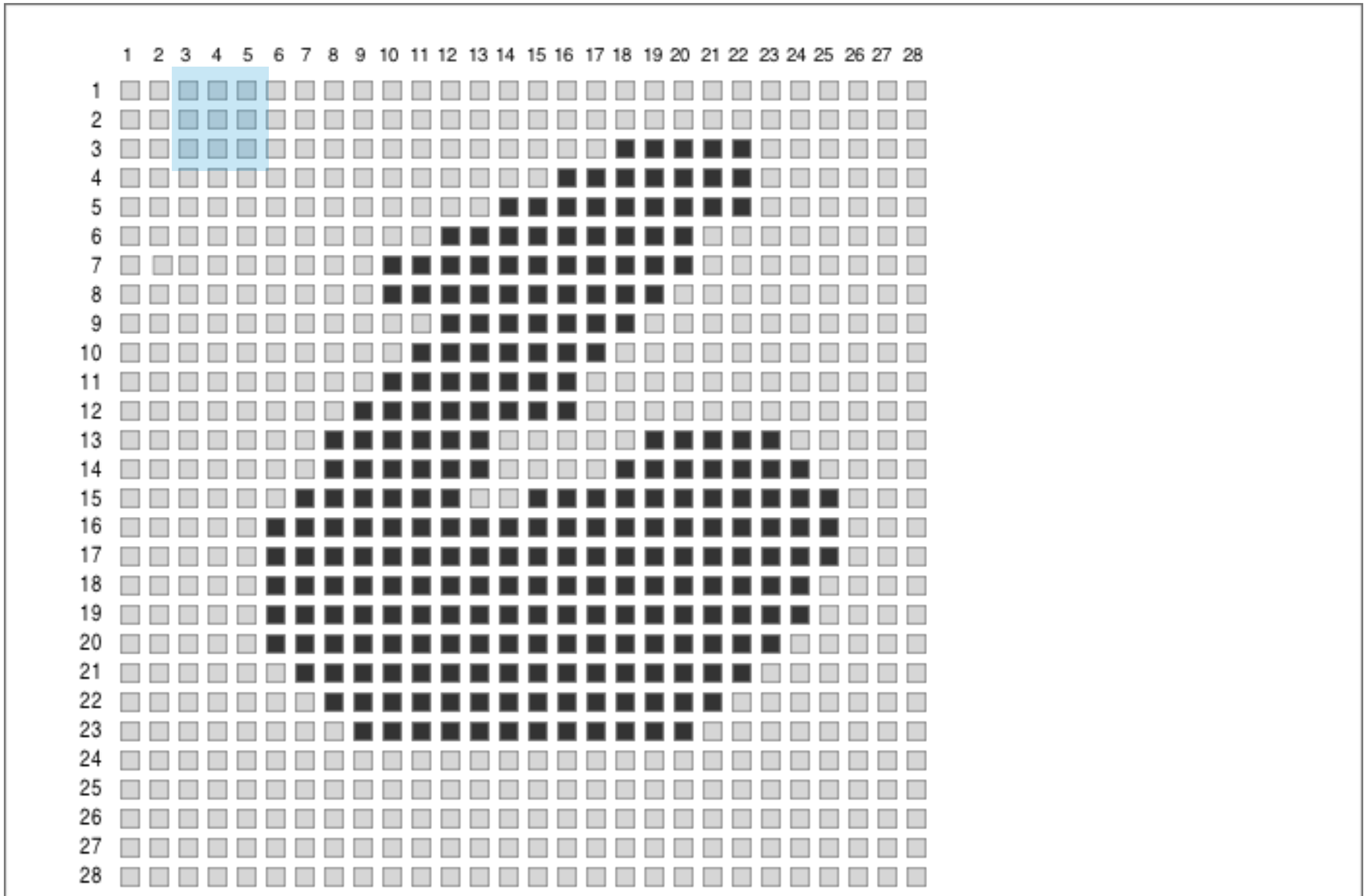
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

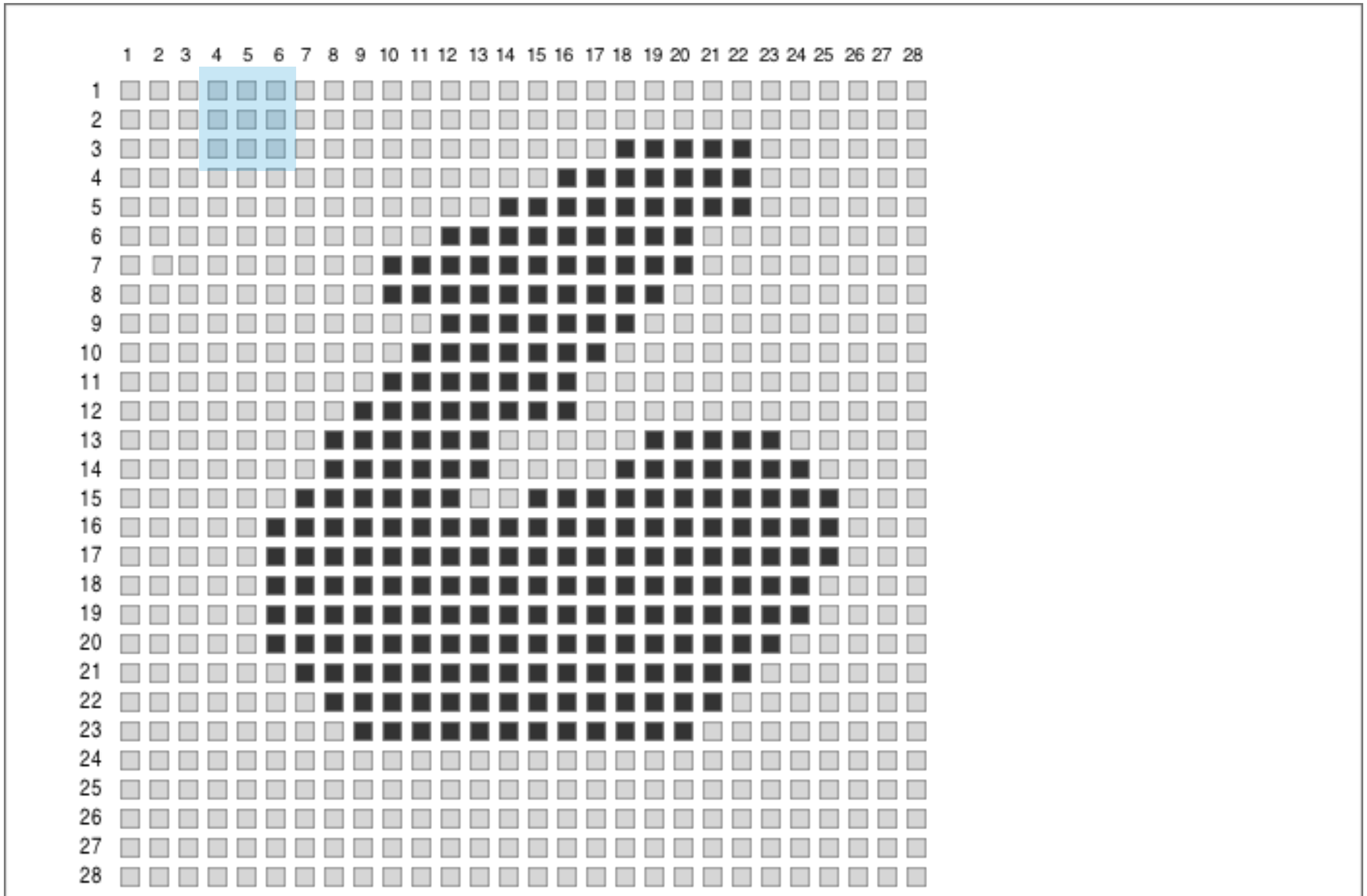
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

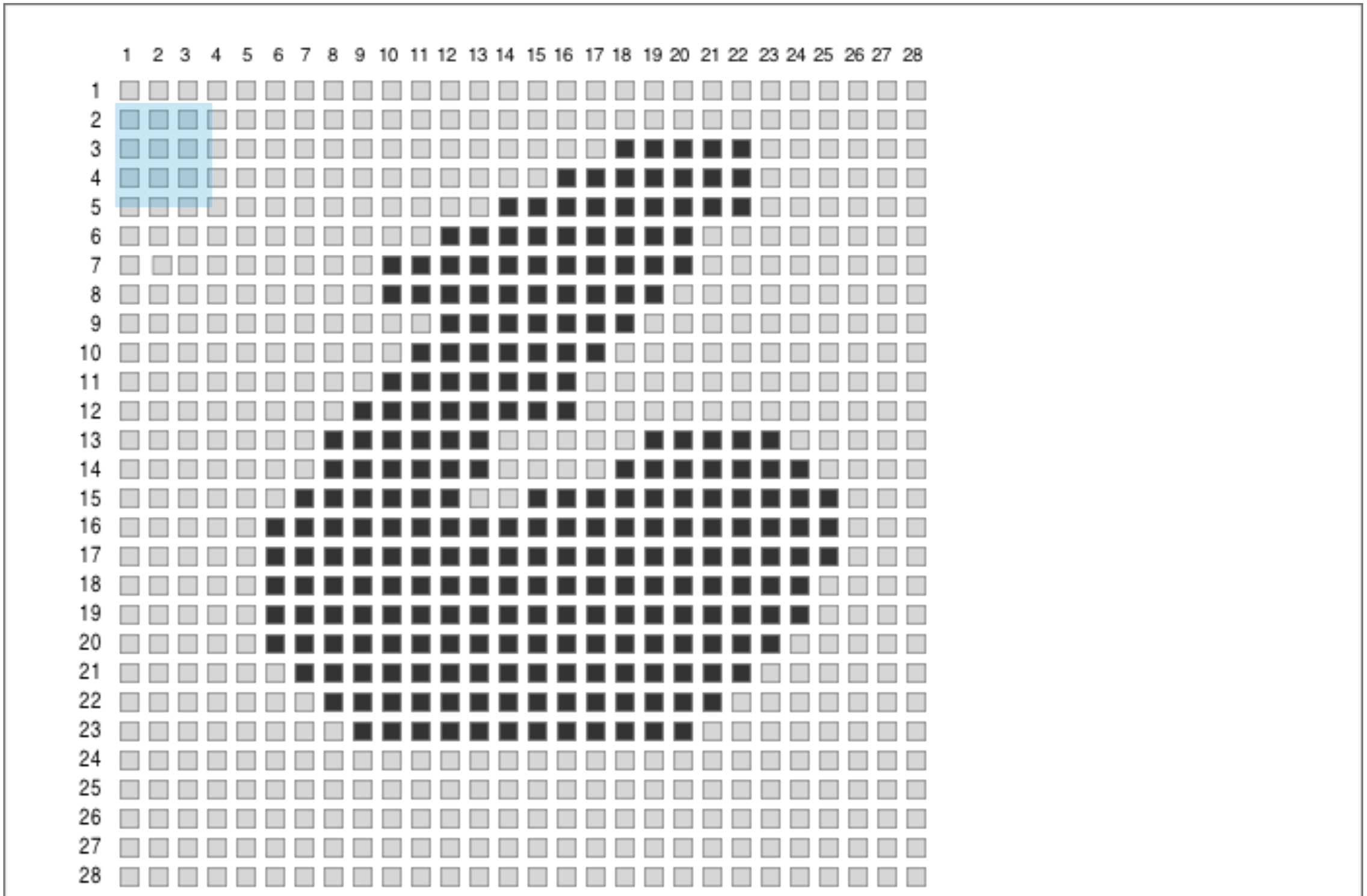
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

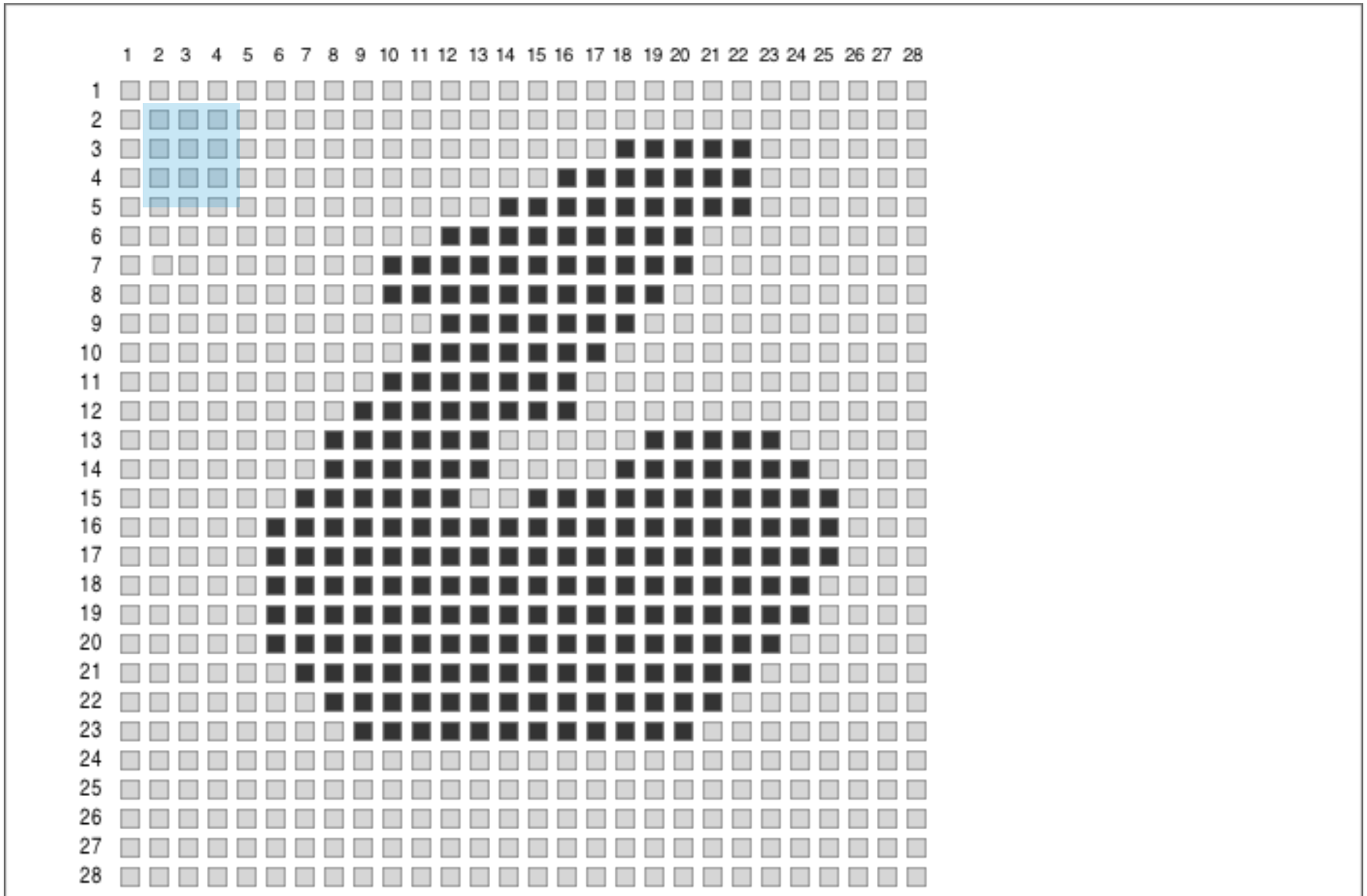
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

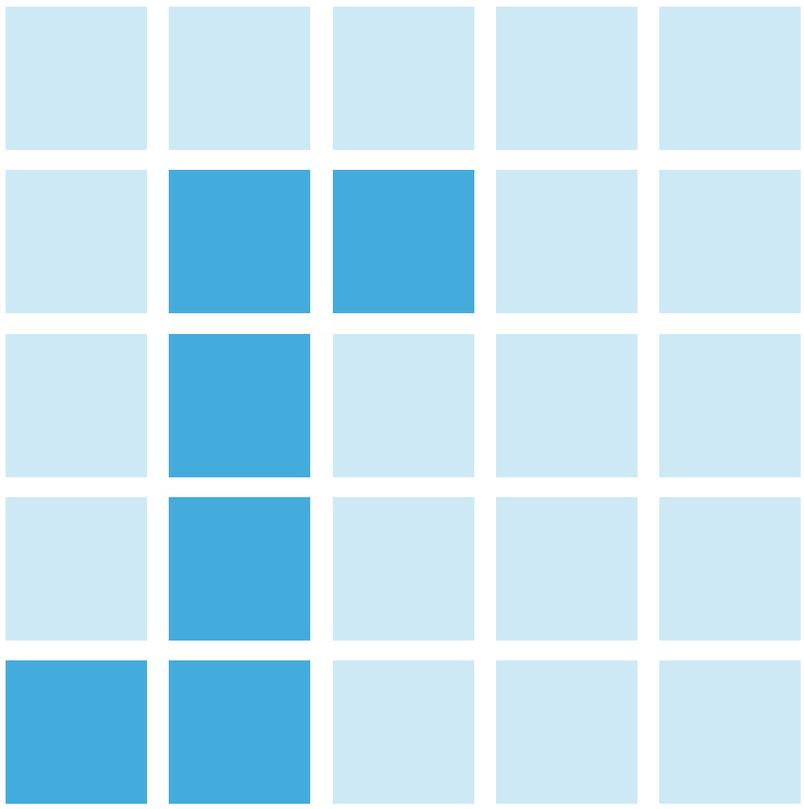
3 X 3

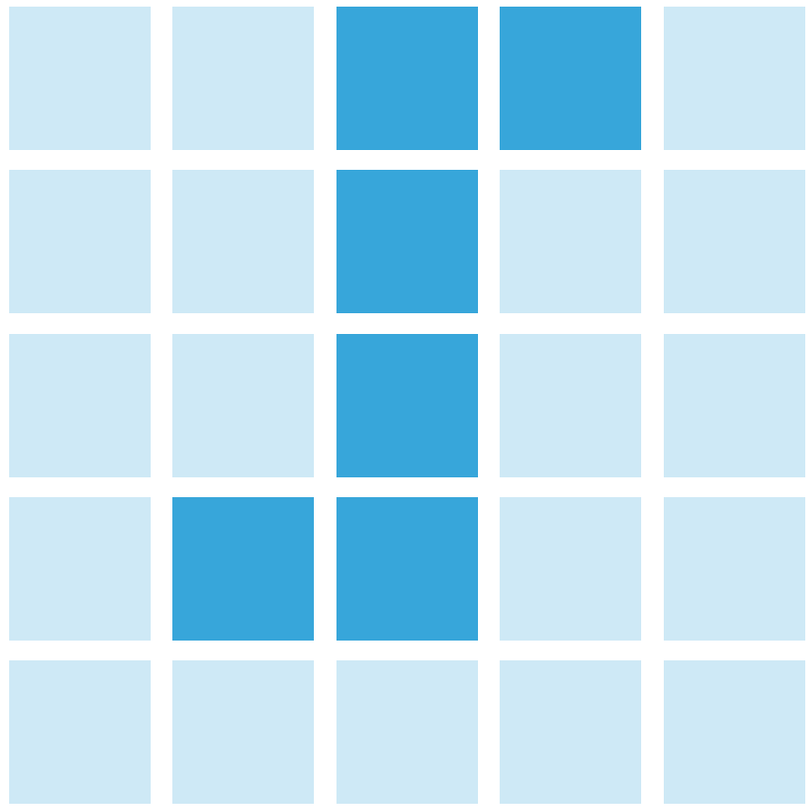

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

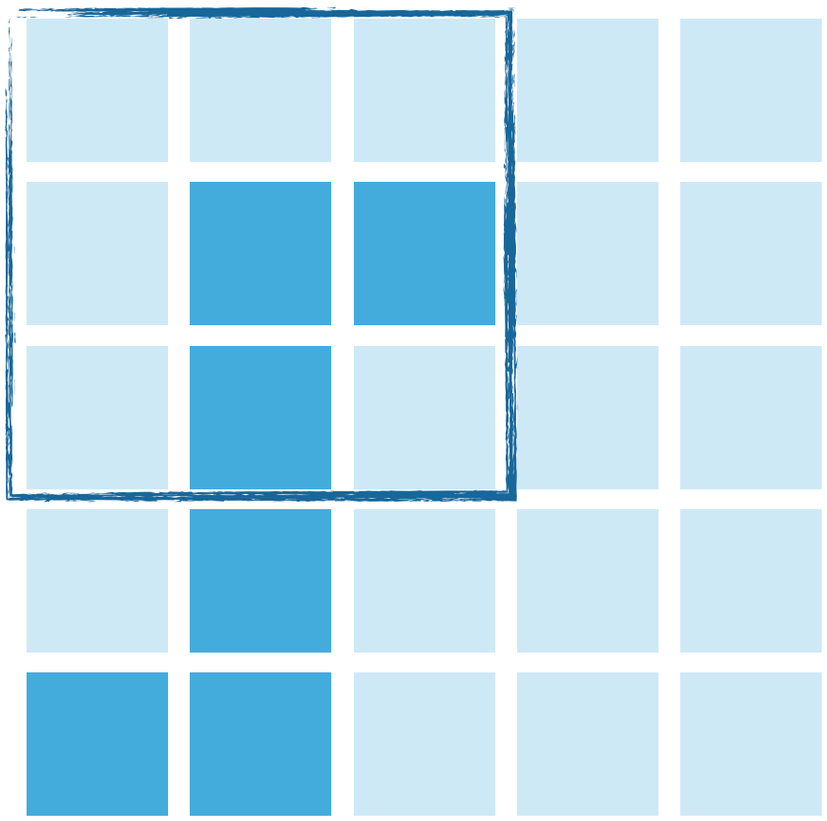
WHY?

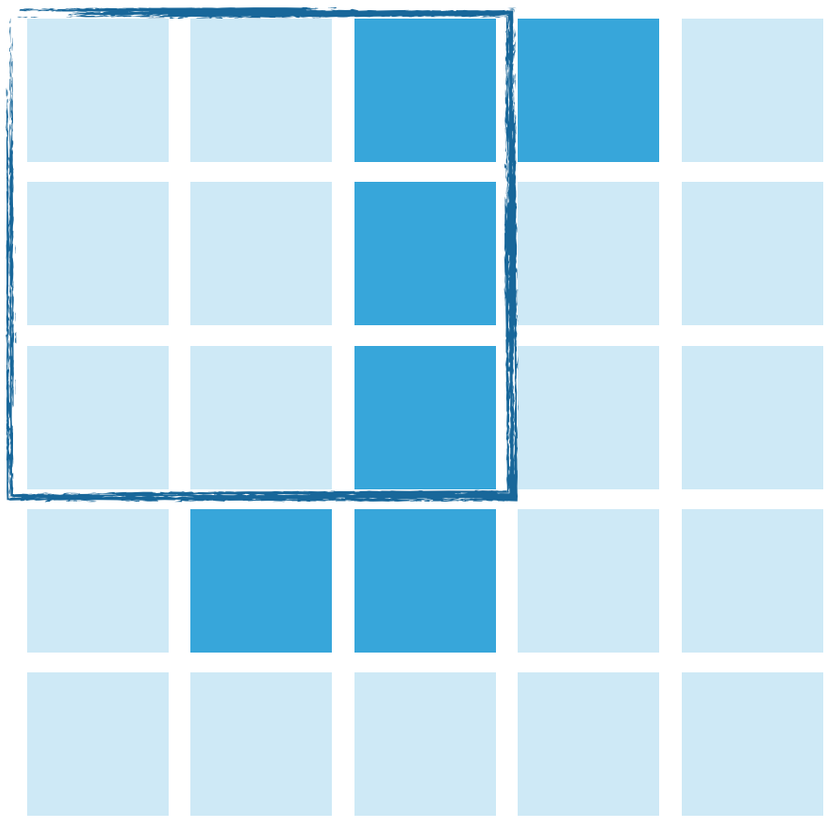




To a densely connected layer - these look different.

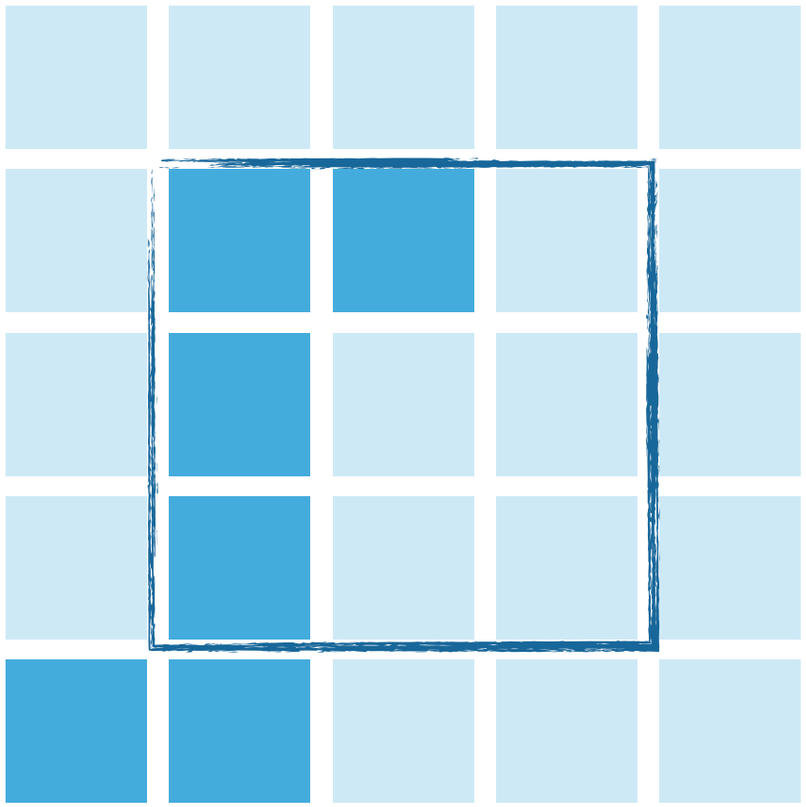
WHY?

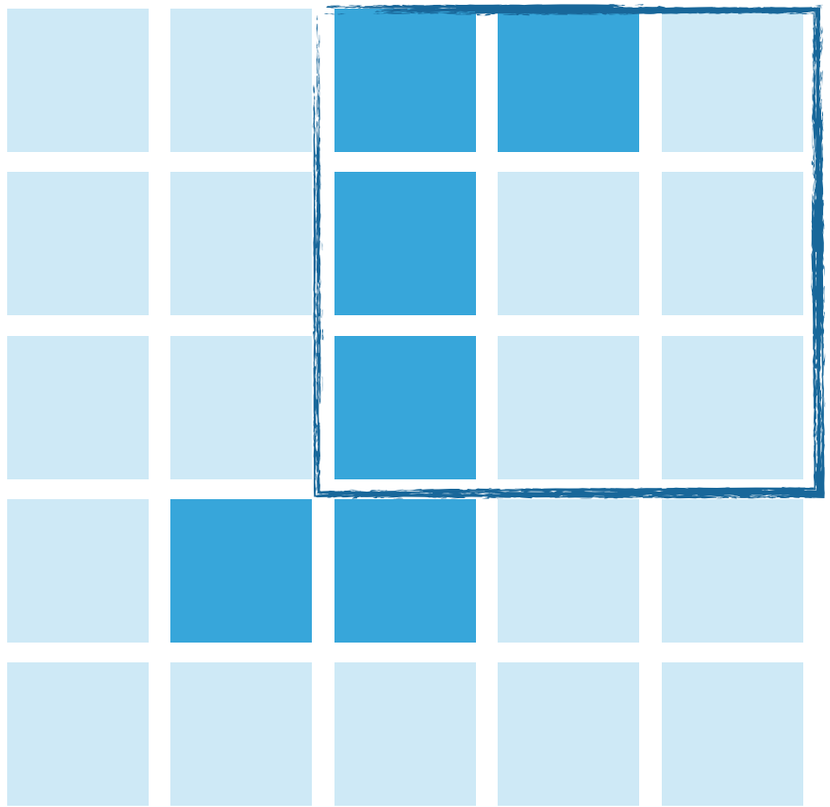




To a ConvNet layer ...

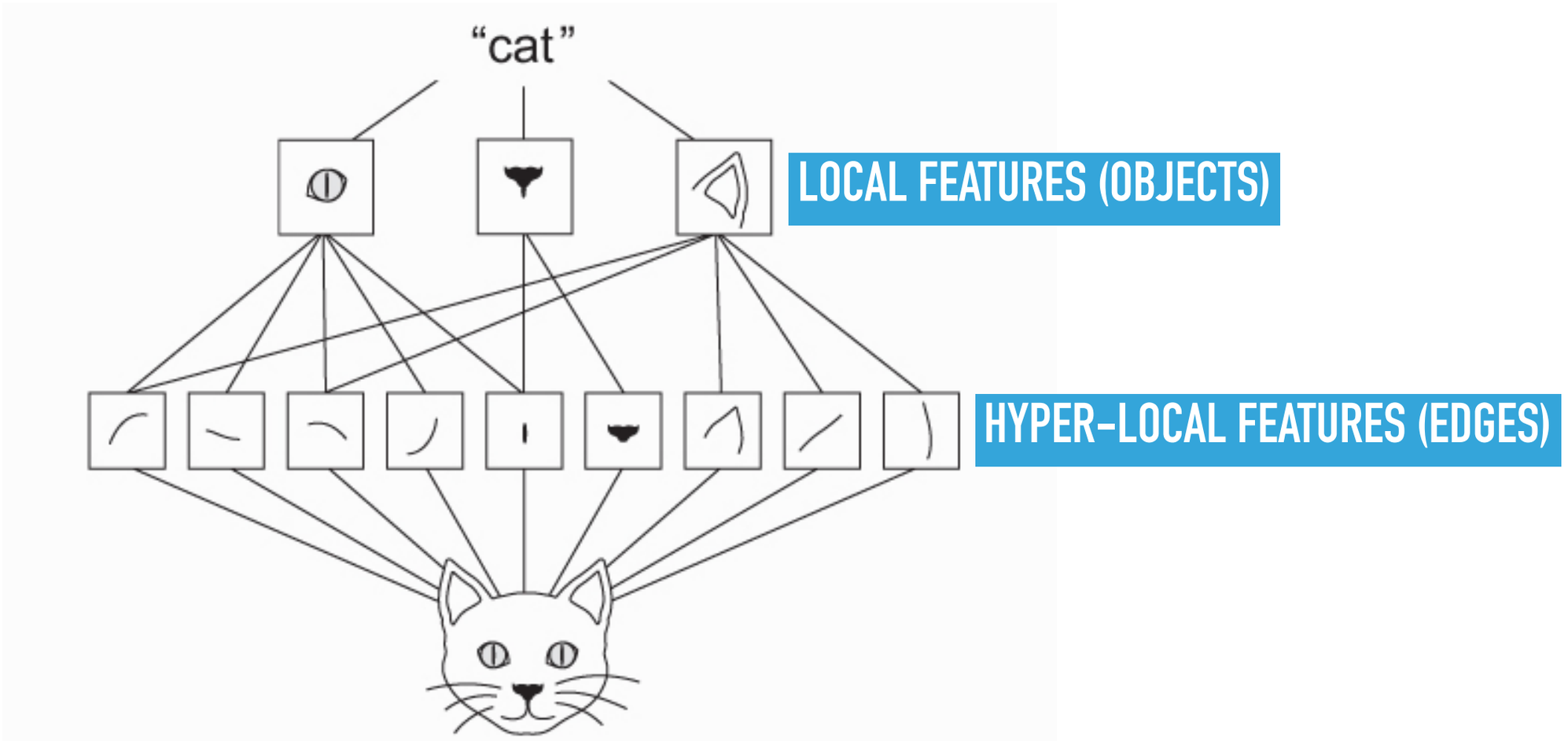
WHY?



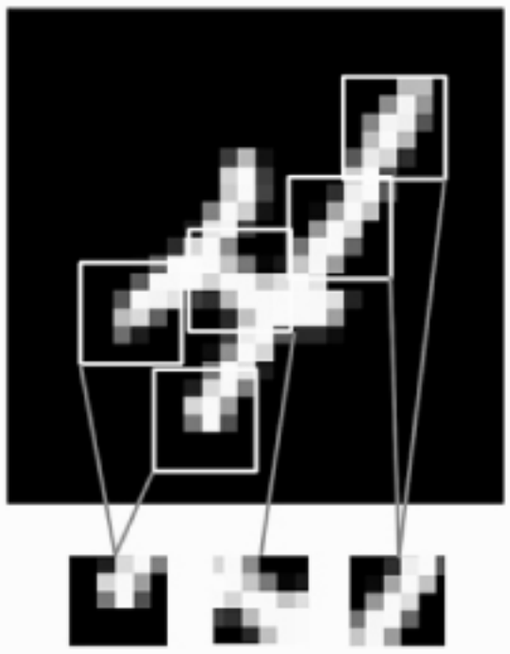


To a ConvNet layer ...

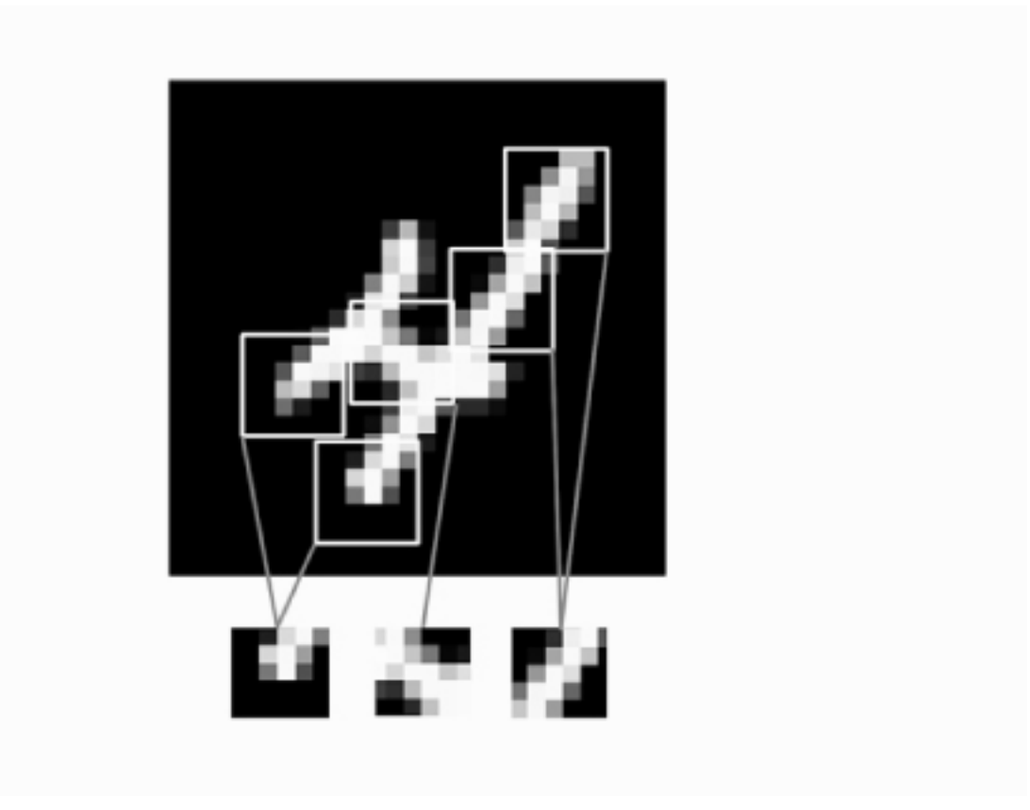
EXTRACTING FEATURES



LOCAL PATTERNS

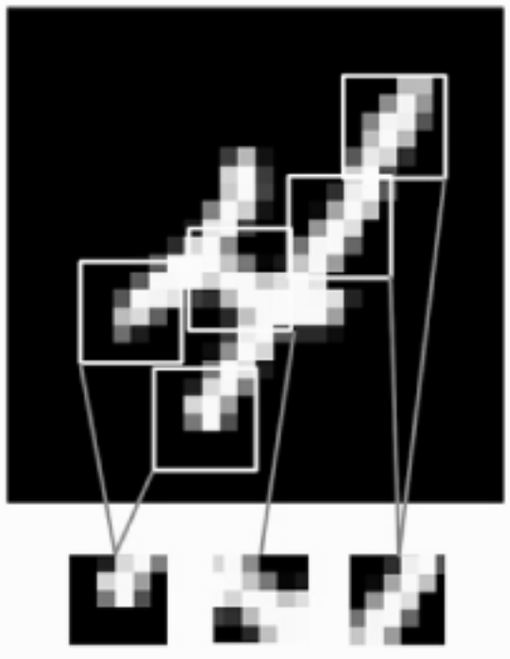


2 KEY CHARACTERISTICS



1. The patterns they learn are **translational invariant.** If they learn a pattern in one location (lower left) they can recognize it anywhere

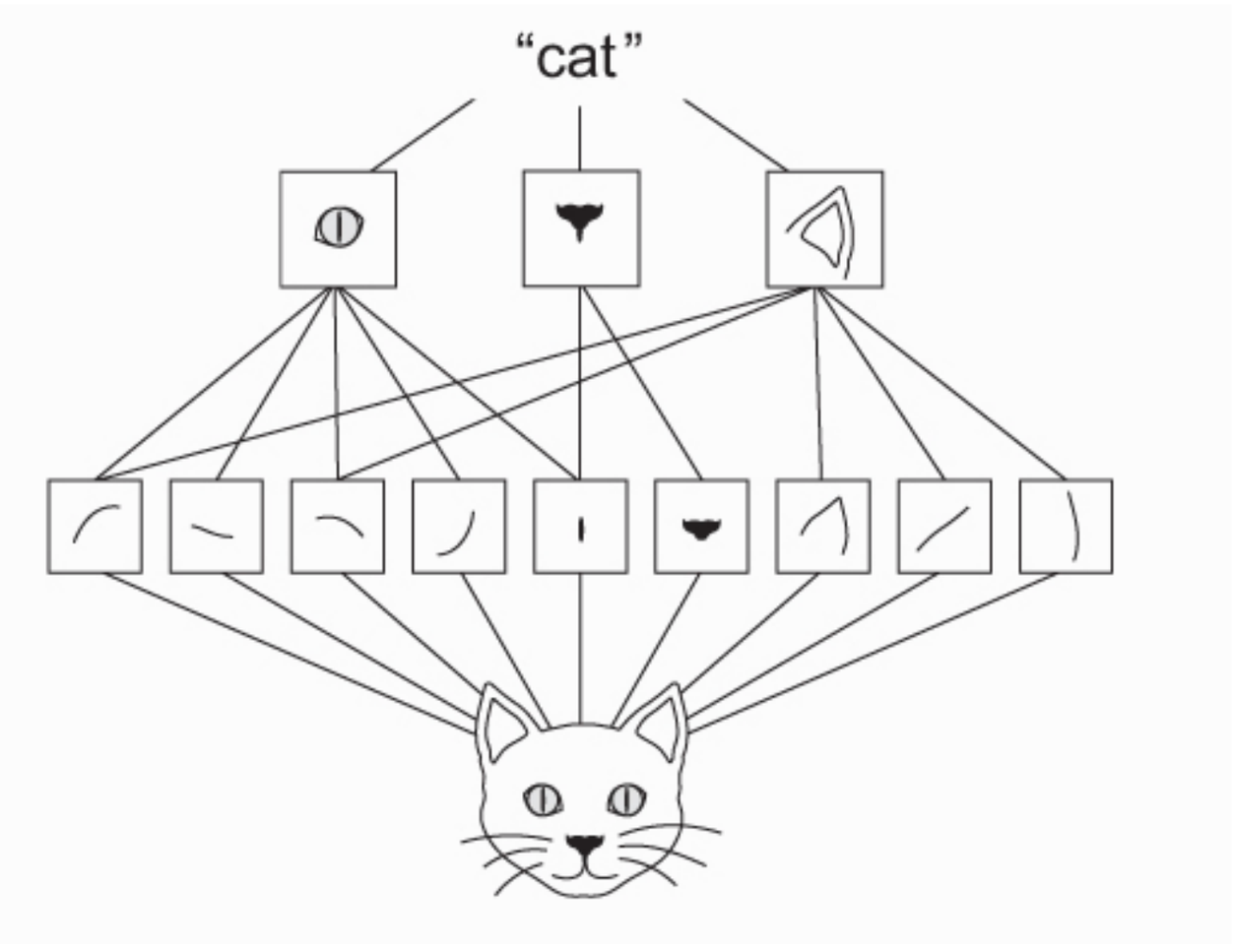
2 KEY CHARACTERISTICS



1. The patterns they learn are **translational invariant.** If they learn a pattern in one location (lower left) they can recognize it anywhere.

The real world is translation invariant.

2 KEY CHARACTERISTICS



1. The patterns they learn are translational invariant. If they learn a pattern in one location (lower left) they can recognize it anywhere.

2. They can learn spatial hierarchies of patterns. Hyper local edges combine to form local objects which combine to form ‘cat’

THAT WAS THE

BIG PICTURE

NOW THE NEXT LAYER DOWN

MORE INFO ON THE

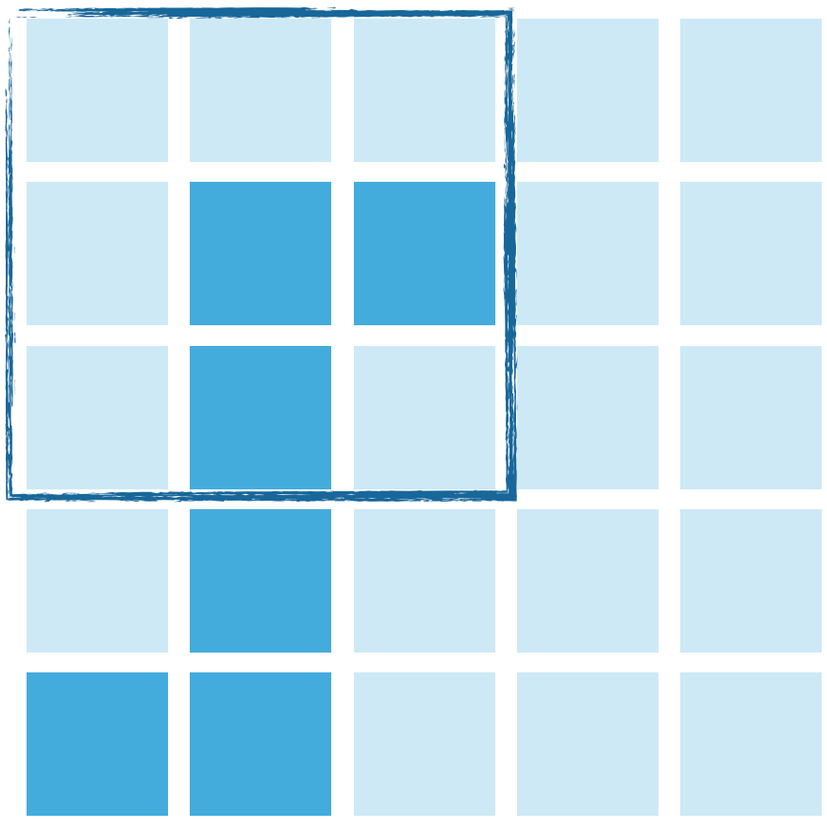
3 BY 3

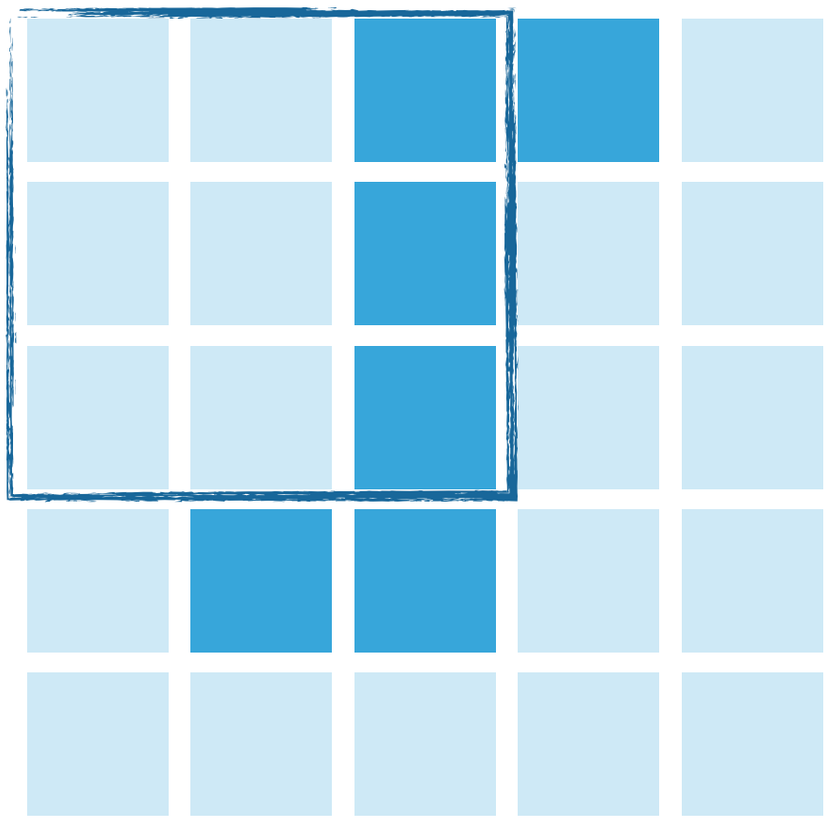
TEXT

THIS 3 X 3 AREA IS CALLED A WINDOW OR PATCH

3X3 AND 5X5 ARE THE COMMON SIZES.

EACH PATCH FEEDS INTO ONE NEURON IN THE LAYER.



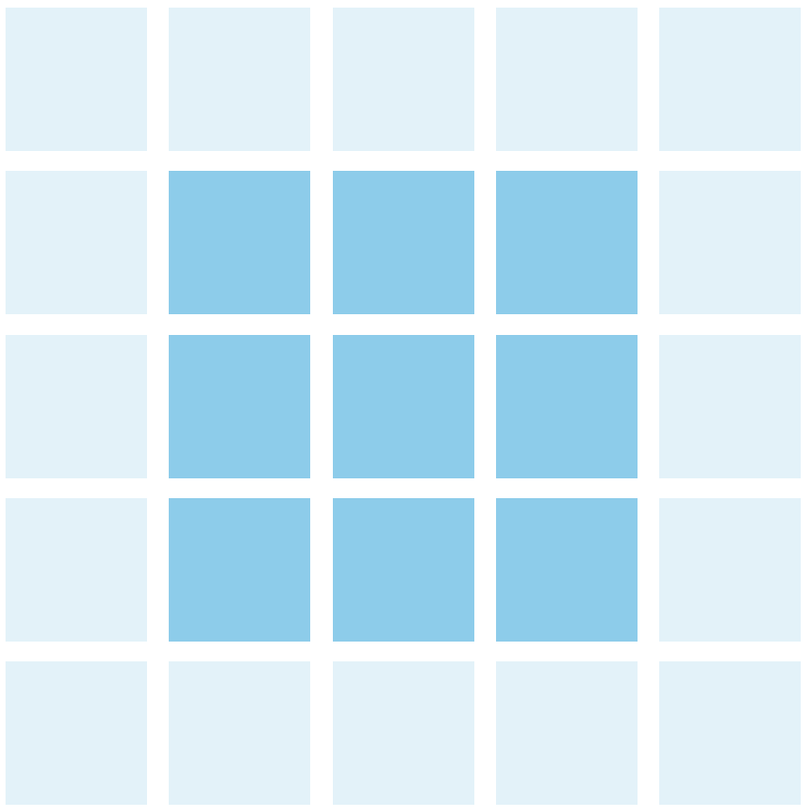


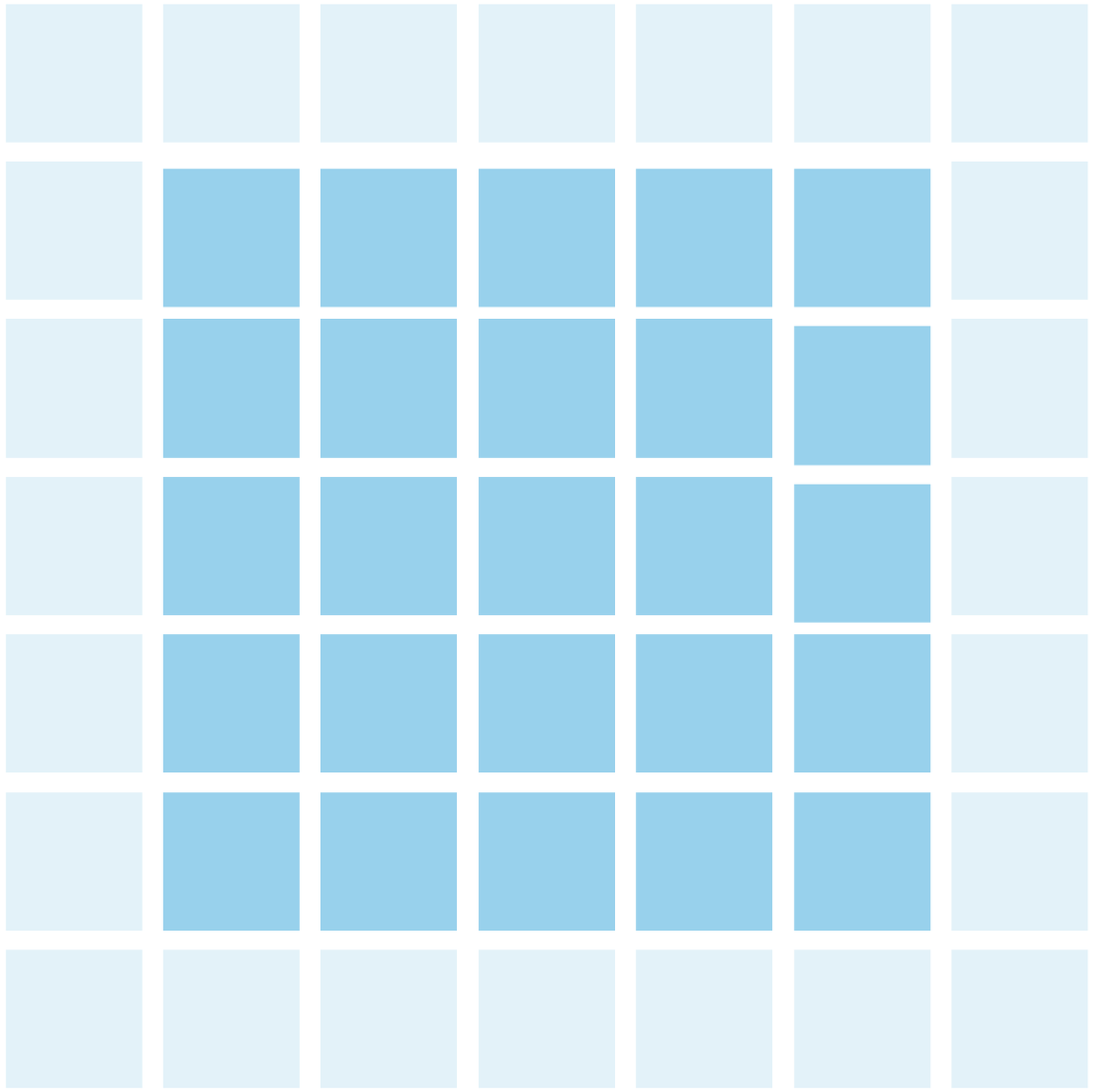
2 team questions: (assume the patch is 3x3)

If the input is 5 x 5 what are the output dimensions?

If the input is a 28 x 28 digit image what are the output dimensions?

PATCHES





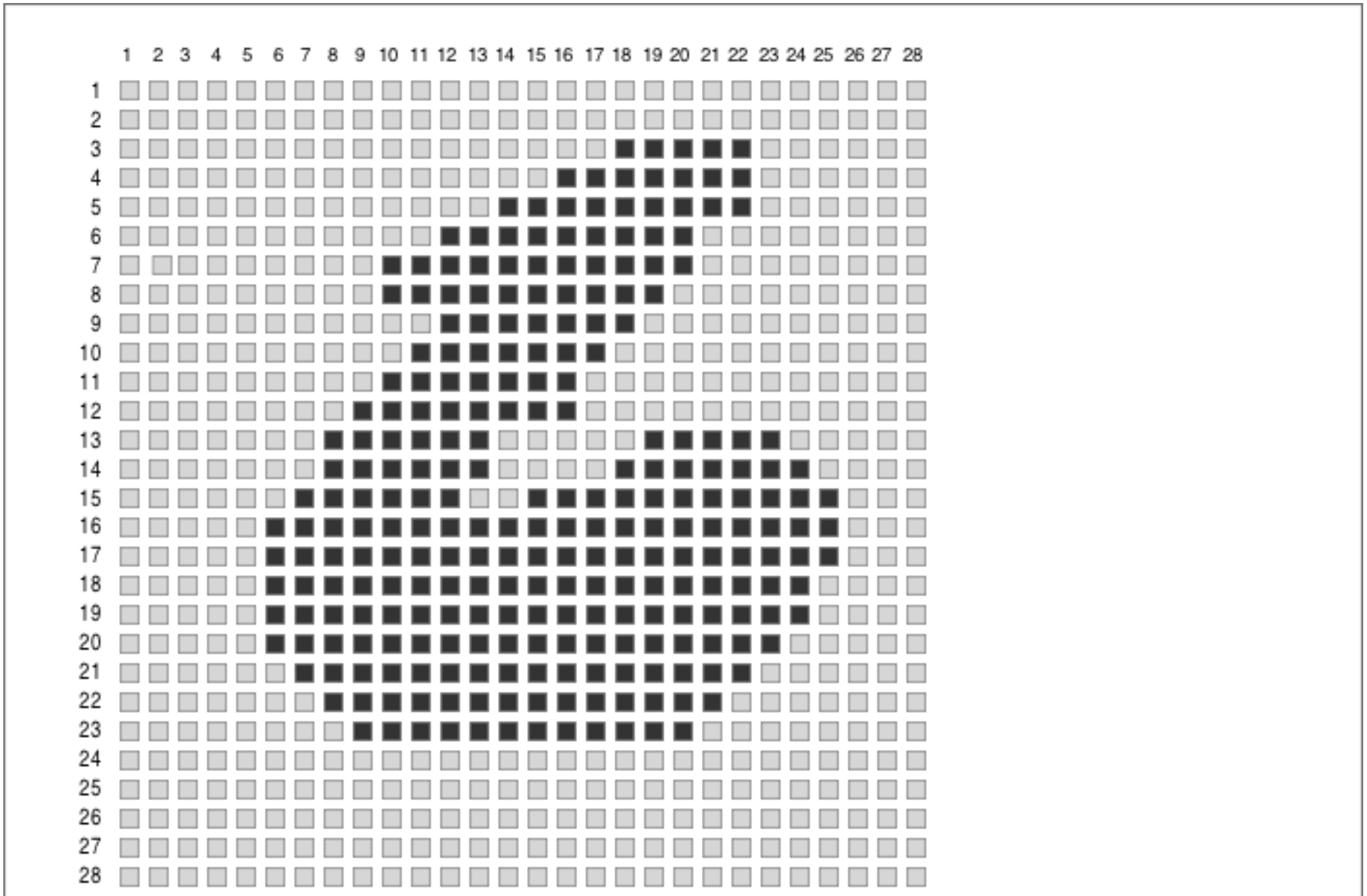
Padding

```
model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(28, 28, 1), padding='same')). (valid)
```

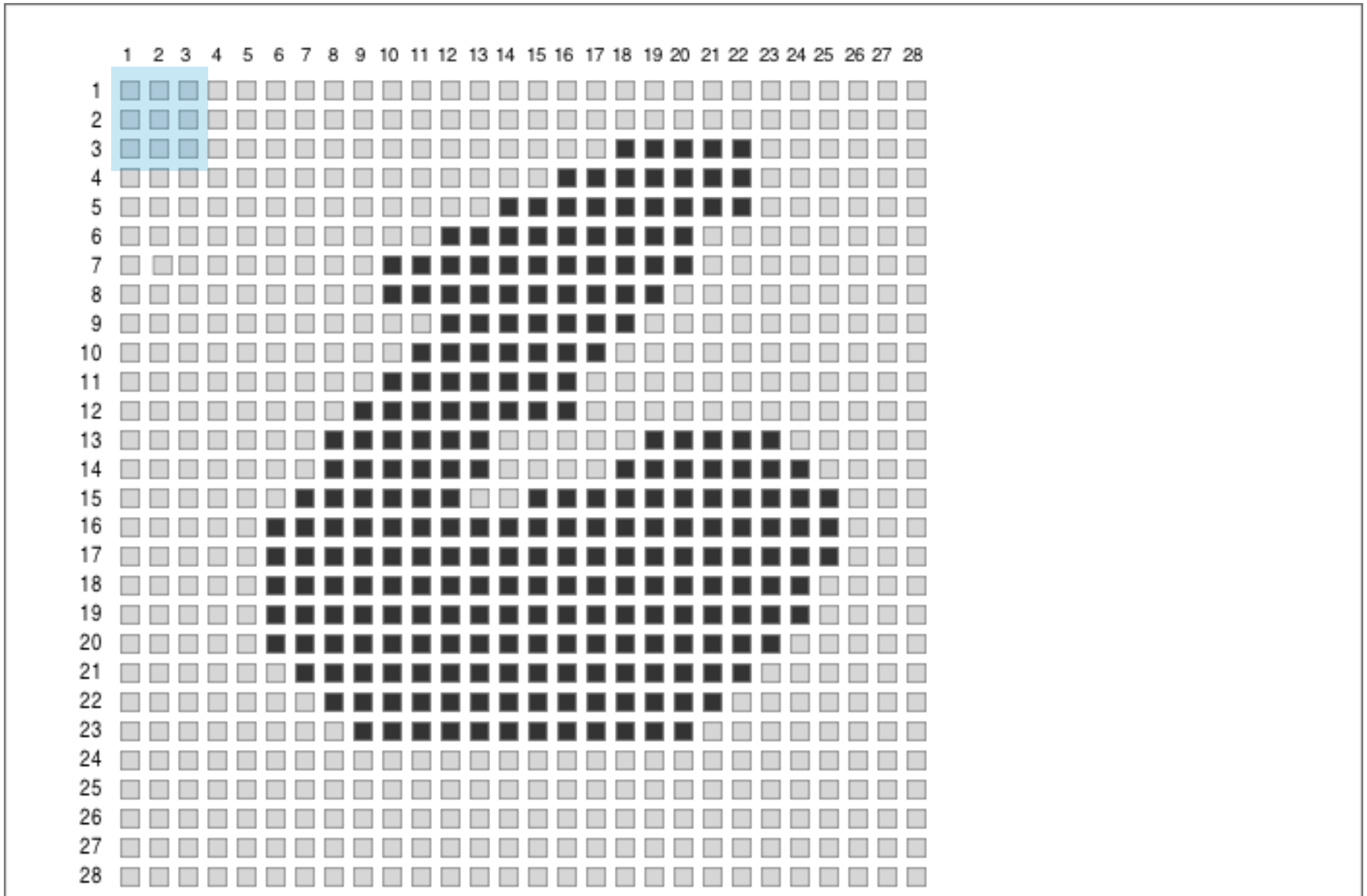
THE DISTANCE BETWEEN SUCCESSIVE WINDOWS IS CALLED ITS STRIDE.

THE DEFAULT STRIDE IS ONE.

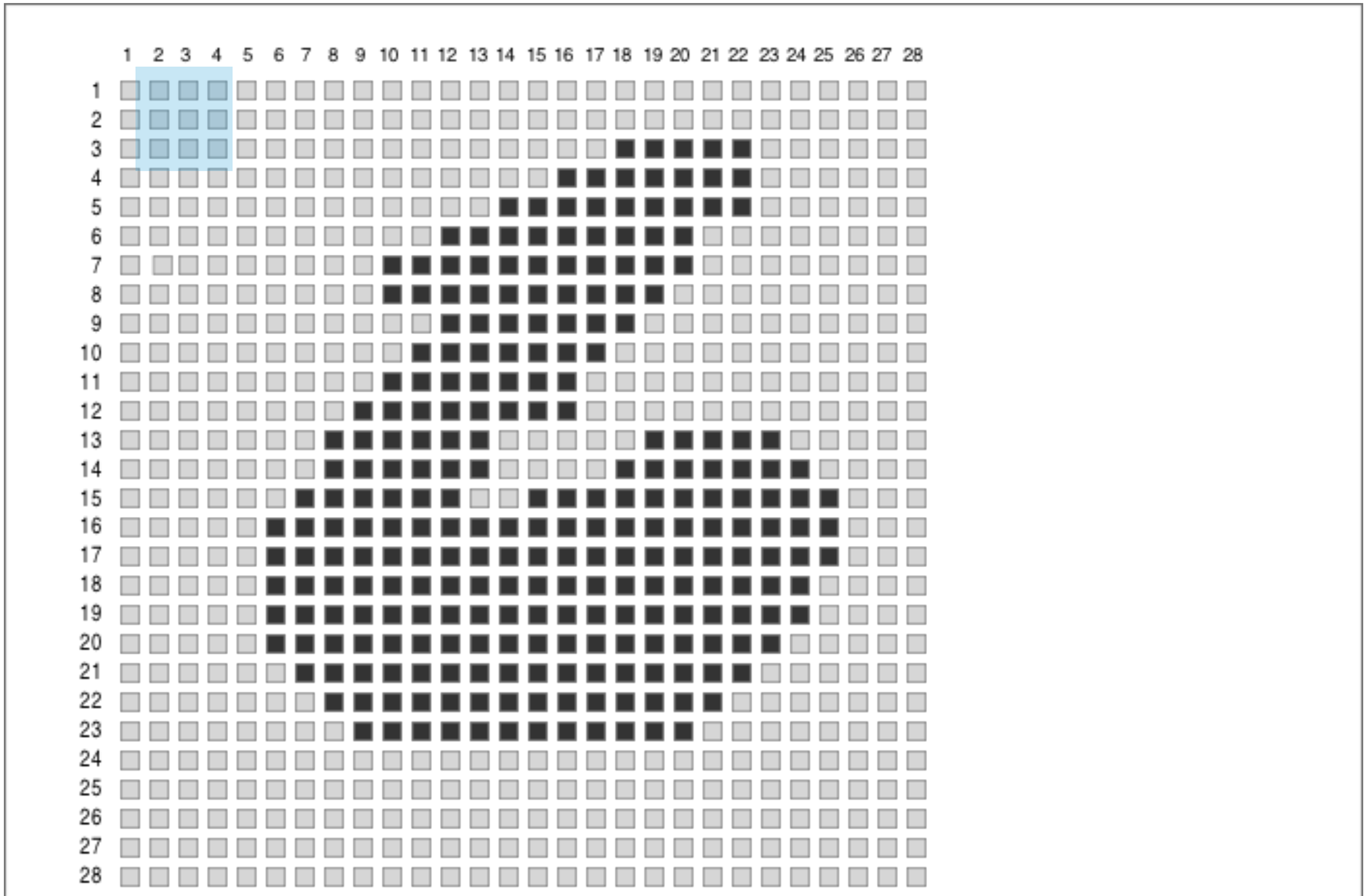
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

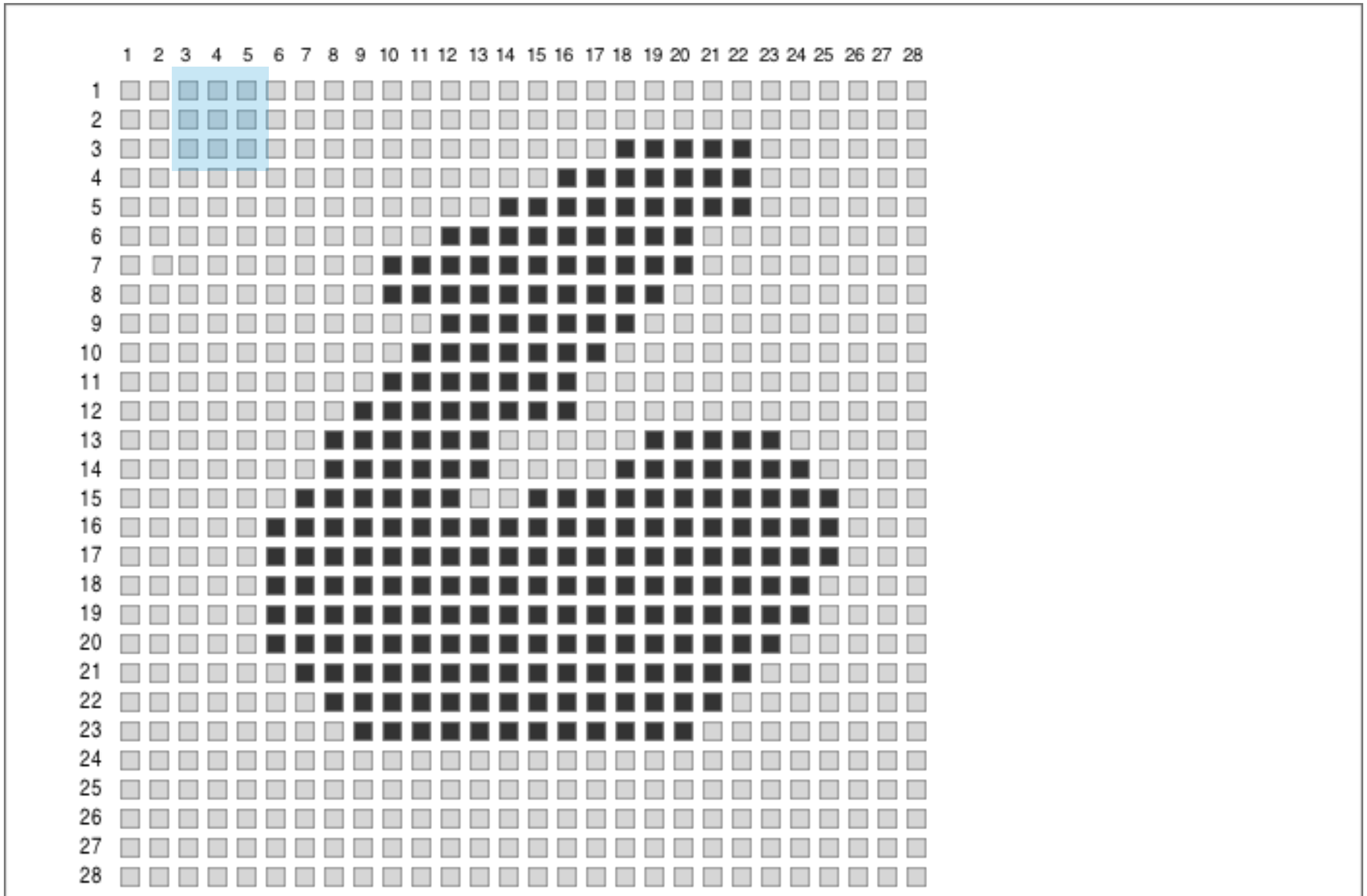
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

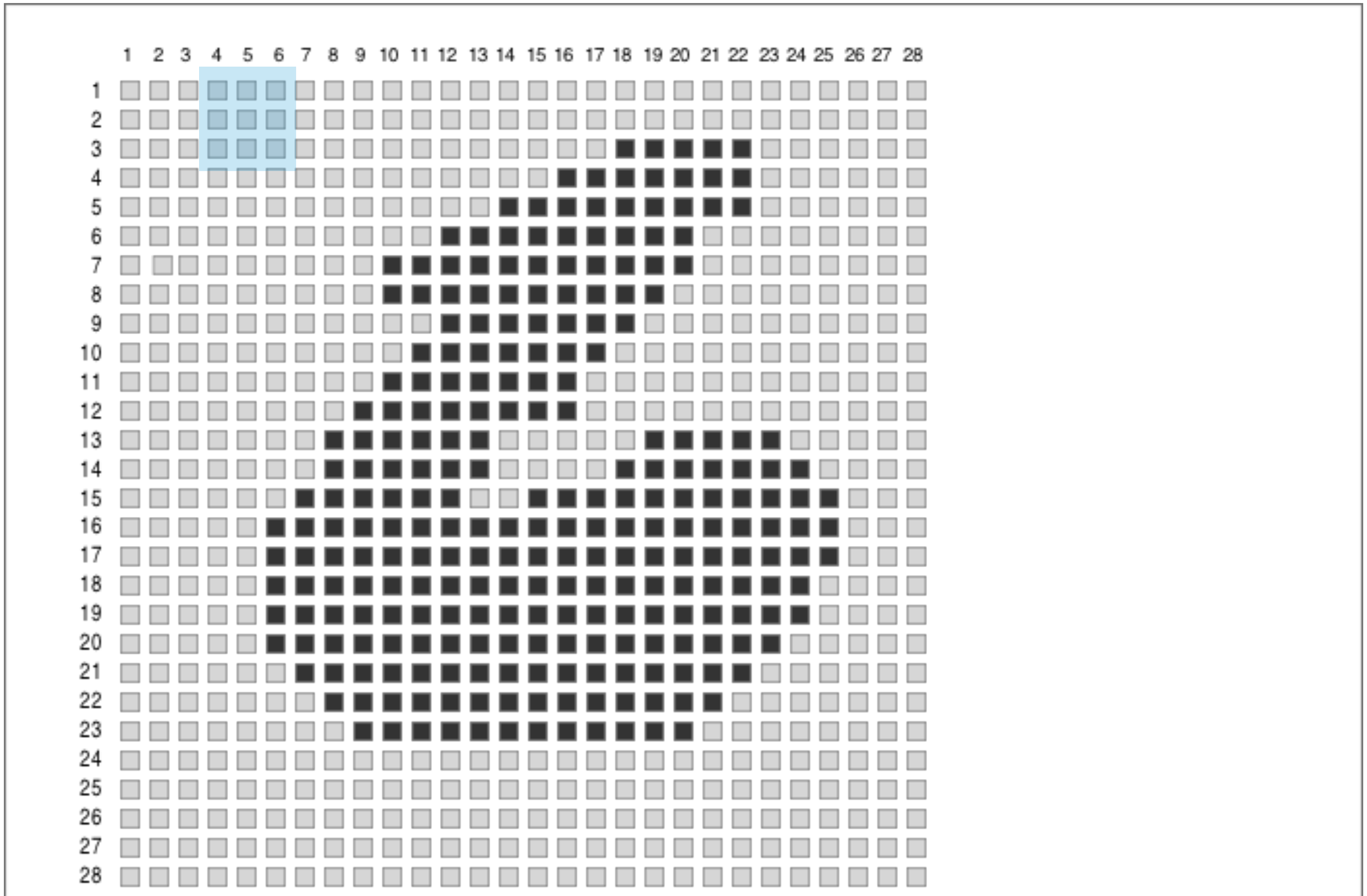
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

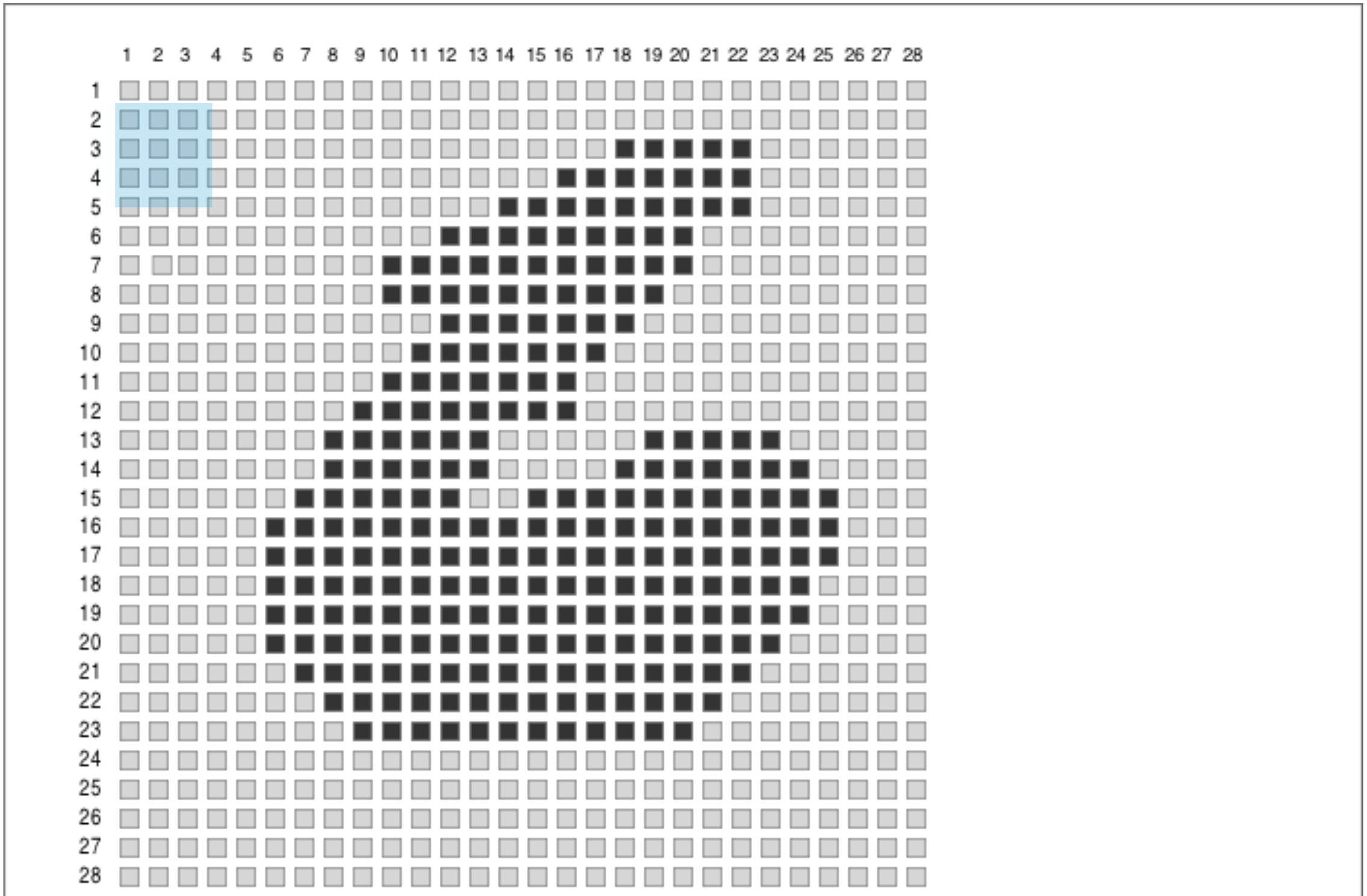
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

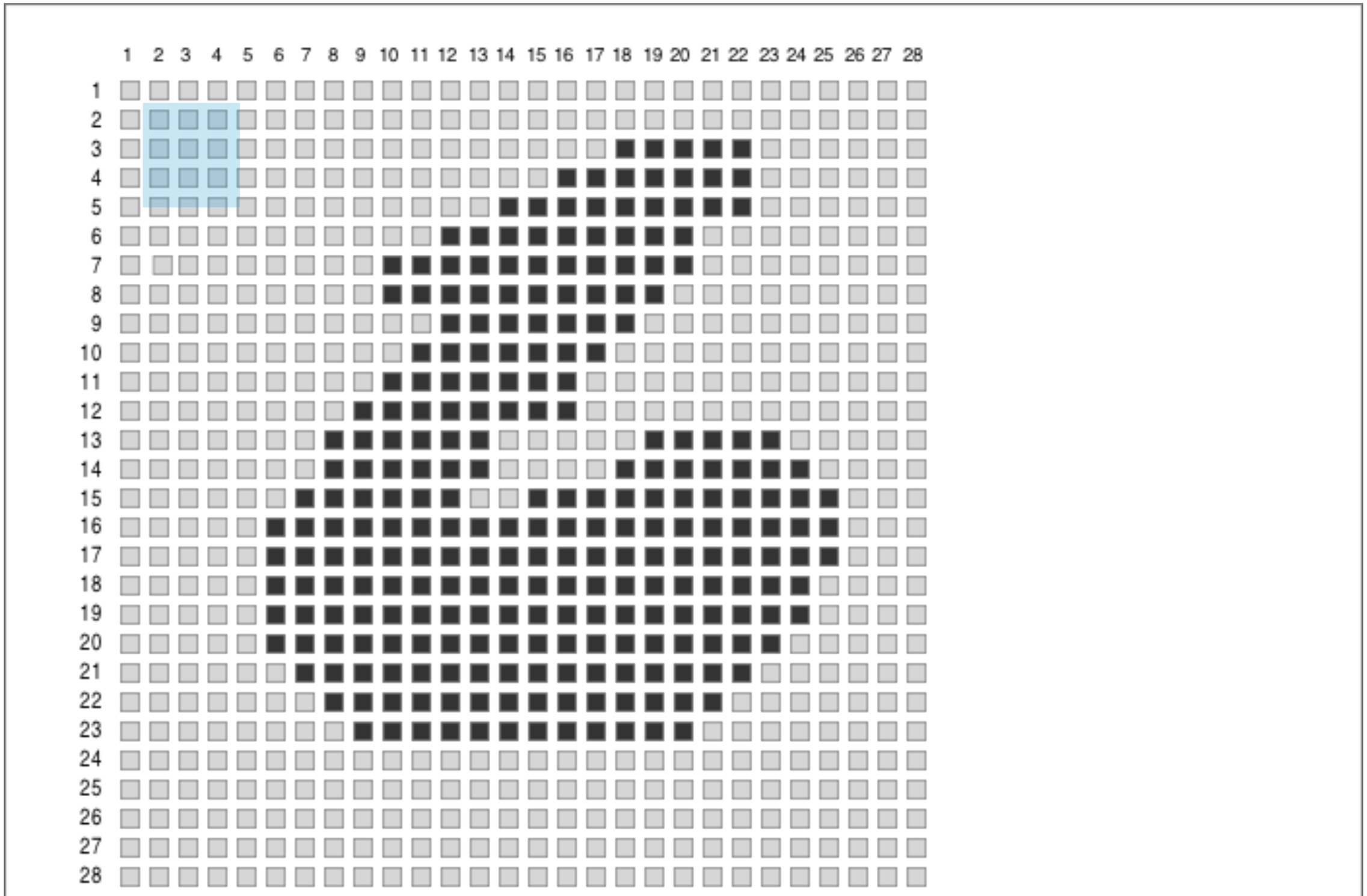
3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

3 X 3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

STRIDES

THE DISTANCE BETWEEN SUCCESSIVE WINDOWS IS CALLED ITS STRIDE.

THE DEFAULT STRIDE IS ONE.

WHEN THE STRIDE IS NOT ONE IT IS CALLED A STRIDED CONVOLUTION

DIVING A BIT DEEPER

THE DIMENSIONS

WE STARTED WITH 28 X 28 X 1 What's the one?

DOG/CAT 150 X 150 X 3

What's the three?

DOG/CAT 150 X 150 X 3

What's the three?

This dimension is called the depth axis or the channel axis.

WE STARTED WITH 28 X 28 X 1

```
from keras import models

from keras import layers
```

```
model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(28, 28, 1)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
```

WE STARTED WITH 28 X 28 X 1

```
from keras import models

from keras import layers
```

```
model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(28, 28, 1)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
```

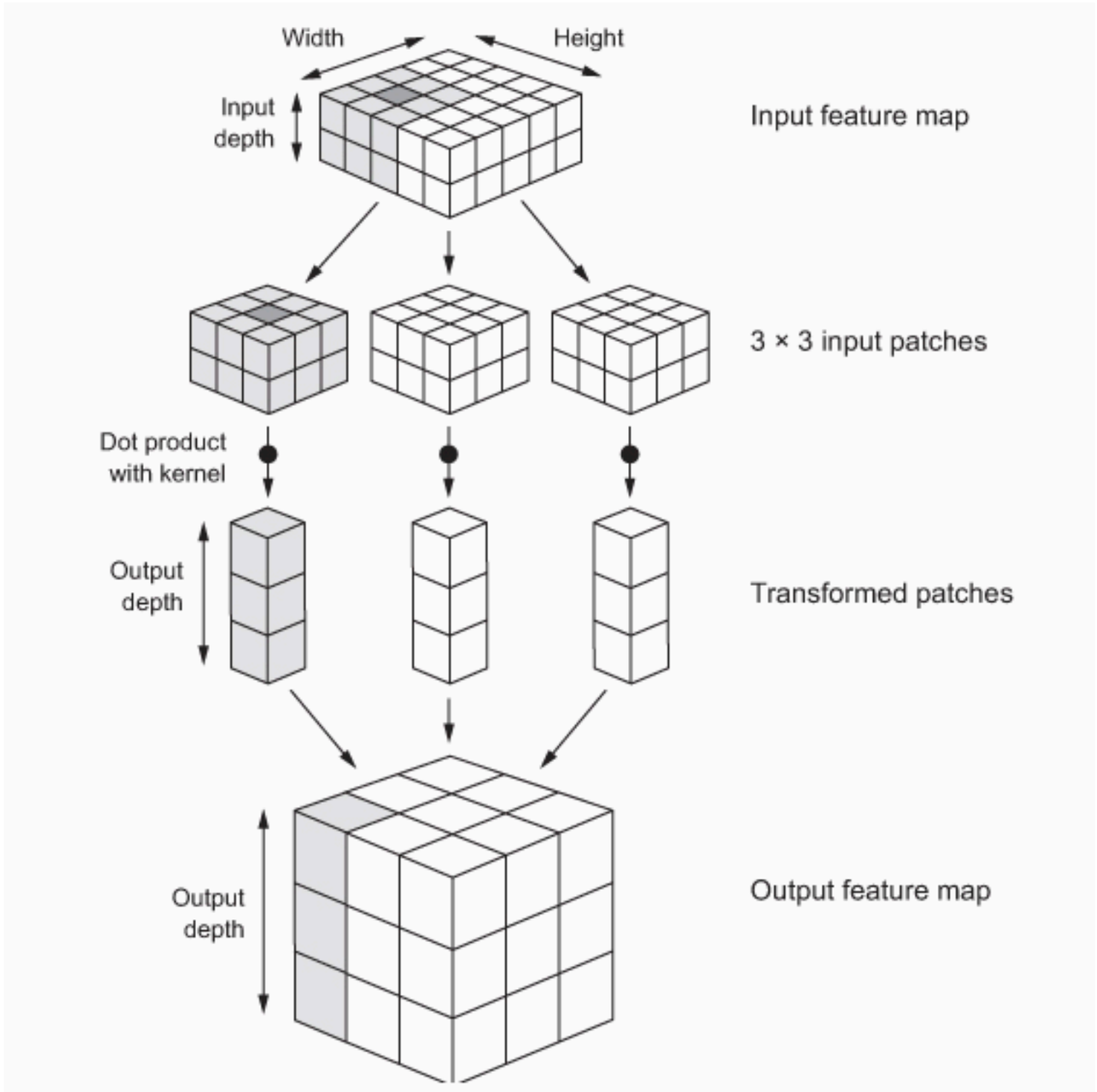
WE STARTED WITH 28 X 28 X 1

```
from keras import models

from keras import layers
```

Now the different channels don't stand for colors but for features or filters.

```
model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(28, 28, 1)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
```

THE FULL PROCESS



THAT WAS A BIT MORE

DETAILED PICTURE

NOW THE NEXT LAYER DOWN

MORE INFO ON THE

3 BY 3

THE MODEL

```
model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(28, 28, 1)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
```

THE MODEL

```
model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(28, 28, 1)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
```

THE MODEL

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 3, 3, 64)          36928
=================================================================
Total params: 55,744
Trainable params: 55,744
Non-trainable params: 0
_________________________________
```

THE MODEL

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 3, 3, 64)          36928
=================================================================
Total params: 55,744
Trainable params: 55,744
Non-trainable params: 0
_________________________________________________________________
```

MaxPooling downsamples by a factor of 2.

WHY DOWNSAMPLE? WHY DON'T WE JUST USE CONVNET LAYERS

```
model_no_max_pool = models.Sequential()
model_no_max_pool.add(layers.Conv2D(32, (3, 3), activation='relu',
                      input_shape=(28, 28, 1)))
model_no_max_pool.add(layers.Conv2D(64, (3, 3), activation='relu'))
model_no_max_pool.add(layers.Conv2D(64, (3, 3), activation='relu'))

model_no_max_pool.summary()
```

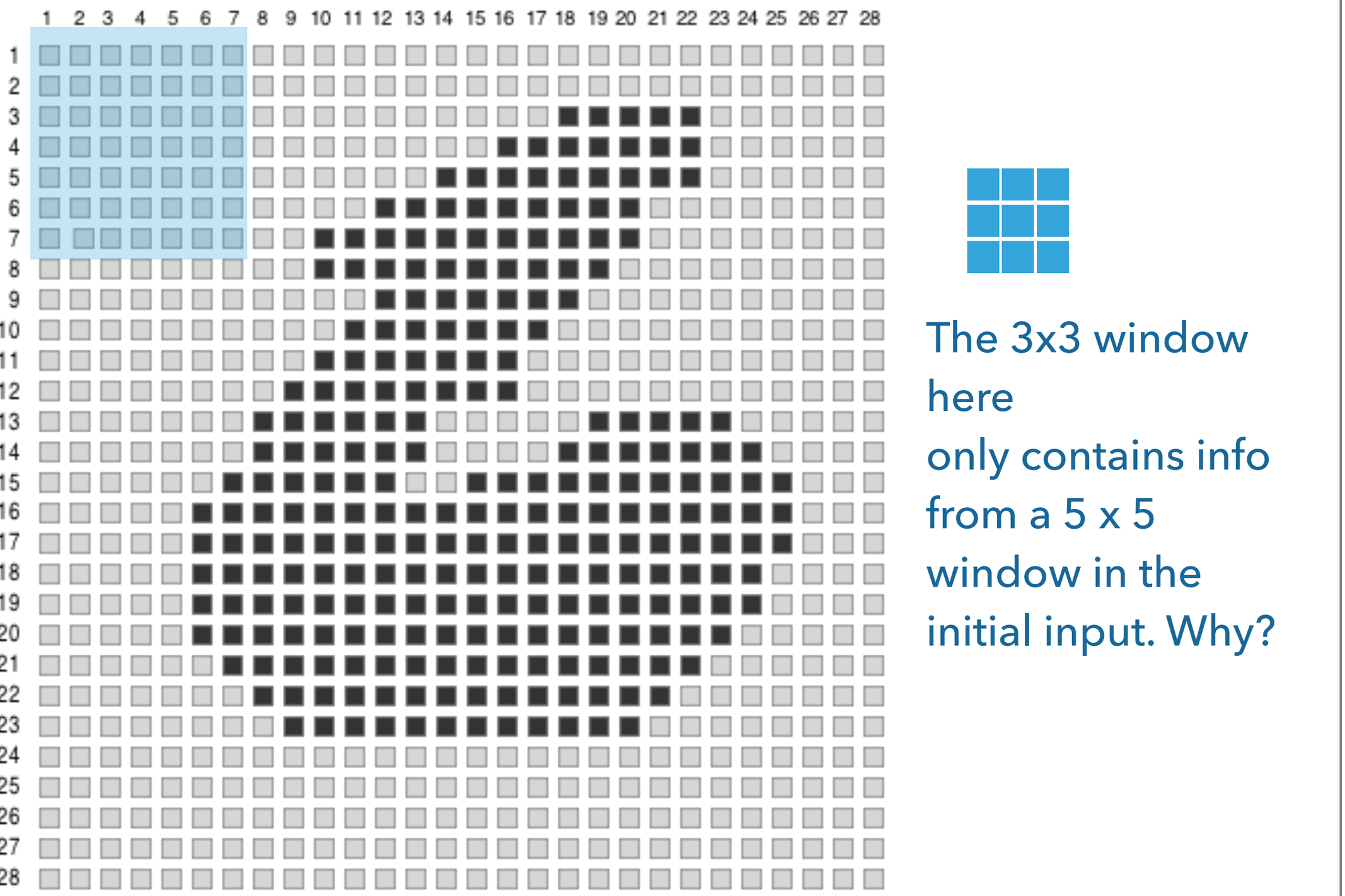
WHY DOWNSAMPLE? WHY DON'T WE JUST USE CONVNET LAYERS

```
model_no_max_pool = models.Sequential()
model_no_max_pool.add(layers.Conv2D(32, (3, 3), activation='relu',
                      input_shape=(28, 28, 1)))
model_no_max_pool.add(layers.Conv2D(64, (3, 3), activation='relu'))
model_no_max_pool.add(layers.Conv2D(64, (3, 3), activation='relu'))

model_no_max_pool.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_7 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 24, 24, 64)        18496
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 22, 22, 64)        36928
=================================================================
Total params: 55,744
Trainable params: 55,744
Non-trainable params: 0
```

First layer 3x3

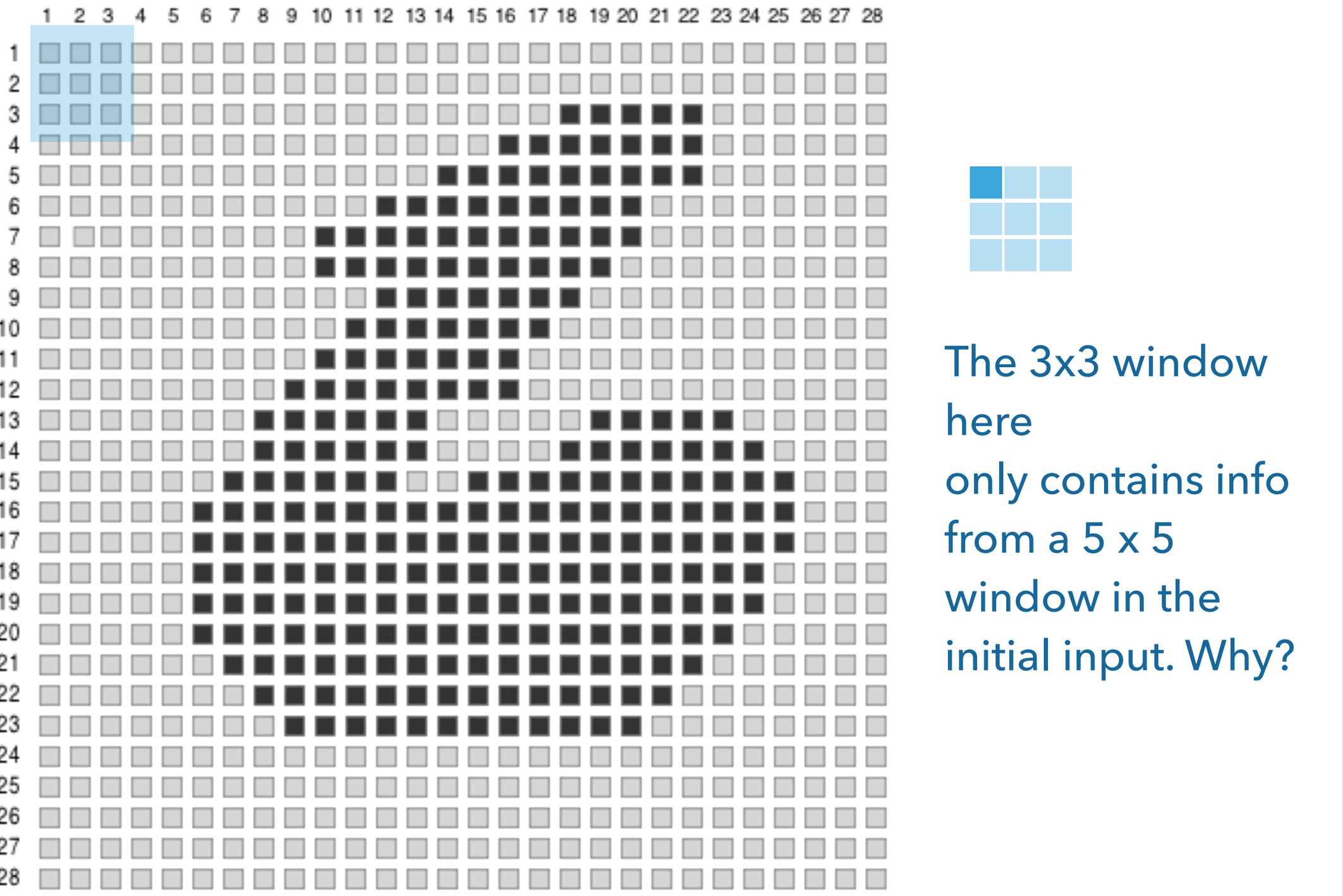
Second layer 3x3



The 3x3 window here
only contains info from a 5 x 5 window in the initial input. Why?

First layer 3x3

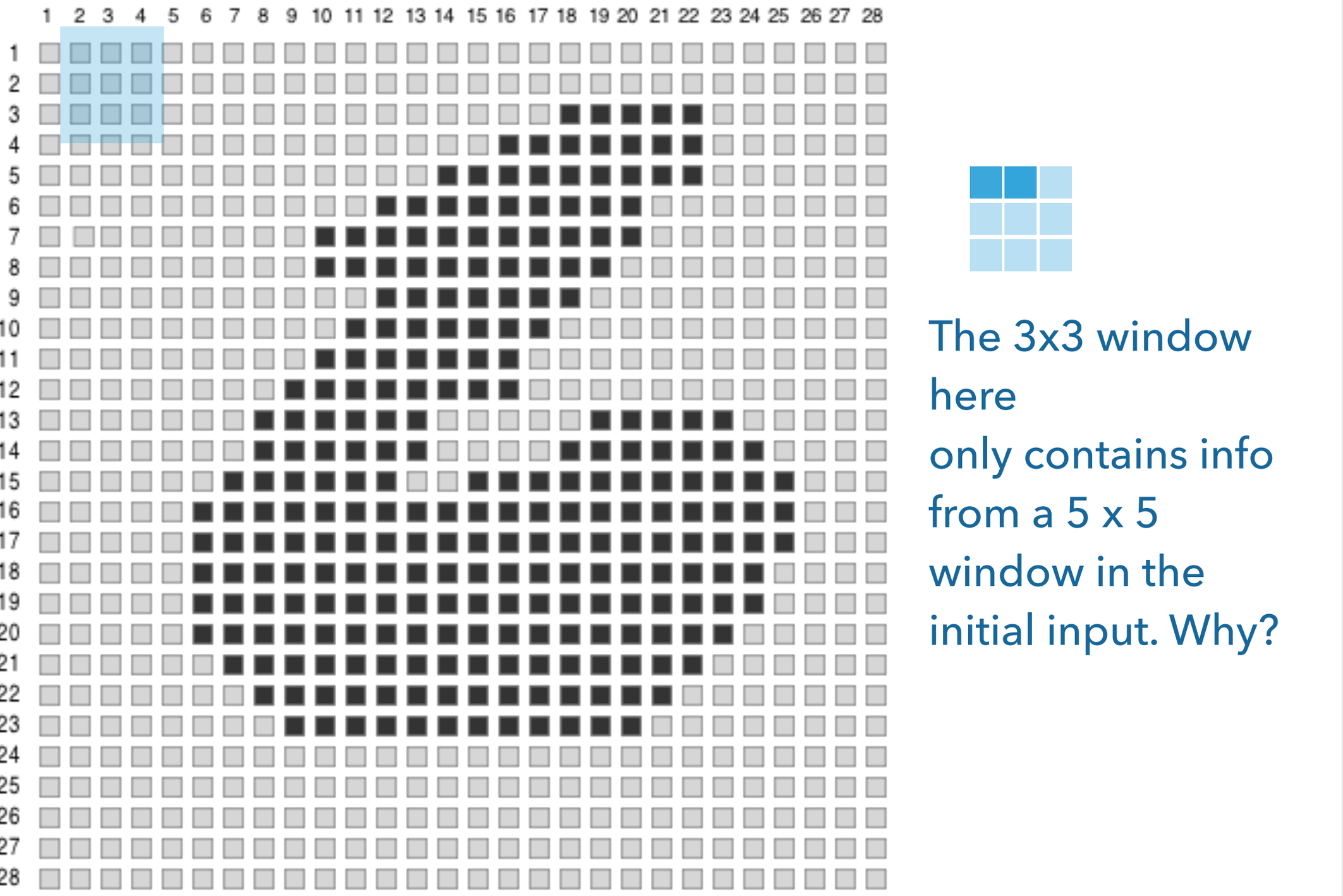
Second layer 3x3



The 3x3 window here
only contains info from a 5 x 5 window in the initial input. Why?

First layer 3x3

Second layer 3x3



The 3x3 window here
only contains info from a 5 x 5 window in the initial input. Why?

PROBLEMS

- Might be difficult to learn to recognize digits with only info from 5 x 5 windows.
- Final feature map has 22 x 22 x 64 = 30,976 coefficients

```
conv2d_3 (Conv2D)               (None, 3, 3, 64)
```

This is huge. If you then flatten with a dense layer you will have 15.8 million parameters!

WHAT IS MAX POOLING? – SIMPLE

- 2 x 2 window with a stride of 2
- Outputs max value of each channel.

NEXT TIME

RECOGNIZING DOGS AND CATS