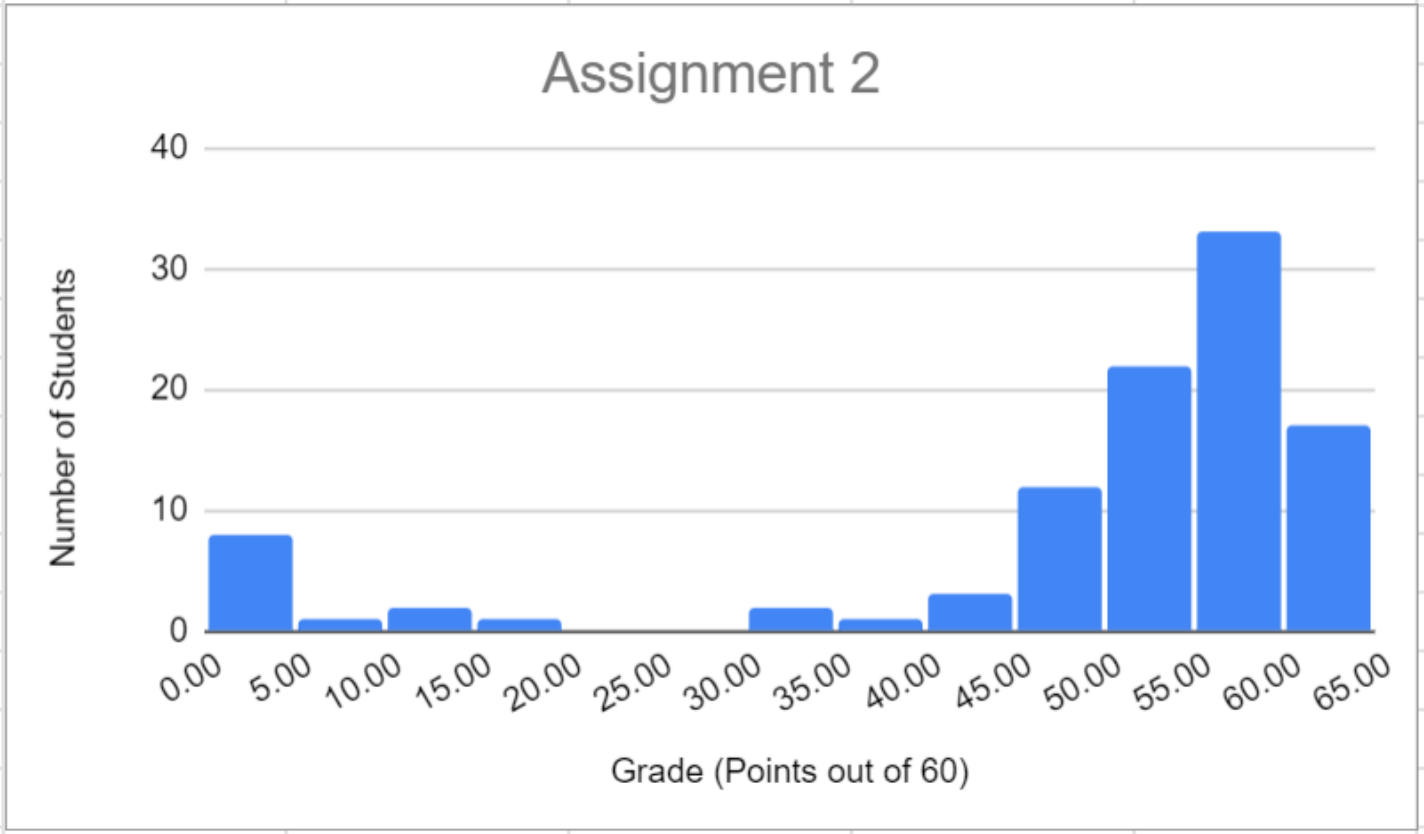
LECTURE 17

WINTER 2021
APPLIED MACHINE LEARNING
CIHANG XIE

SLIDE CREDIT:
NARGES NOROUZI

HW3



Mean:	47.41
Median:	54.25
SD:	16.89676871

EXTRA CREDITS

- 4 HWs in total (50%)
  - You can earn extra credits by completing the bonus questions
- 5 Quizzes in total (30%)
  - **Quiz 5 10 pts**

  Total quiz credit will be capped at 30 pts
- Group Activities (10%)
- Class Participation (10%)
  - **Come to my next office hour, I will provide a small test**

EXTRA CREDITS

Course Feedback

Student Experience of Teaching Surveys - Spring 2021 for CSE-144-01-64331-Applied ML

Medium	Online
Timing	Scheduled

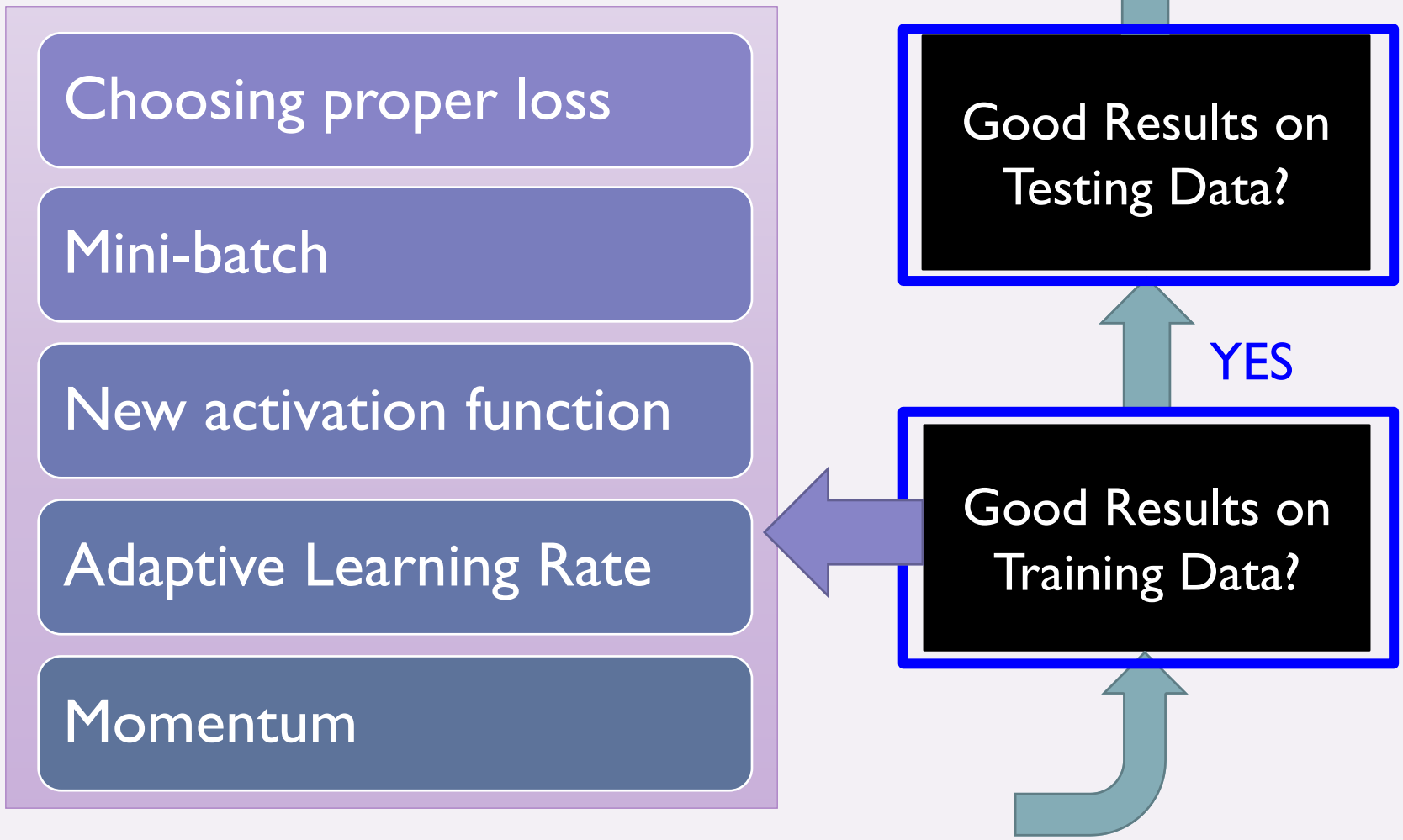
- Start Date 2021-05-24 08:00
- End Date 2021-06-06 23:59

Response Rate

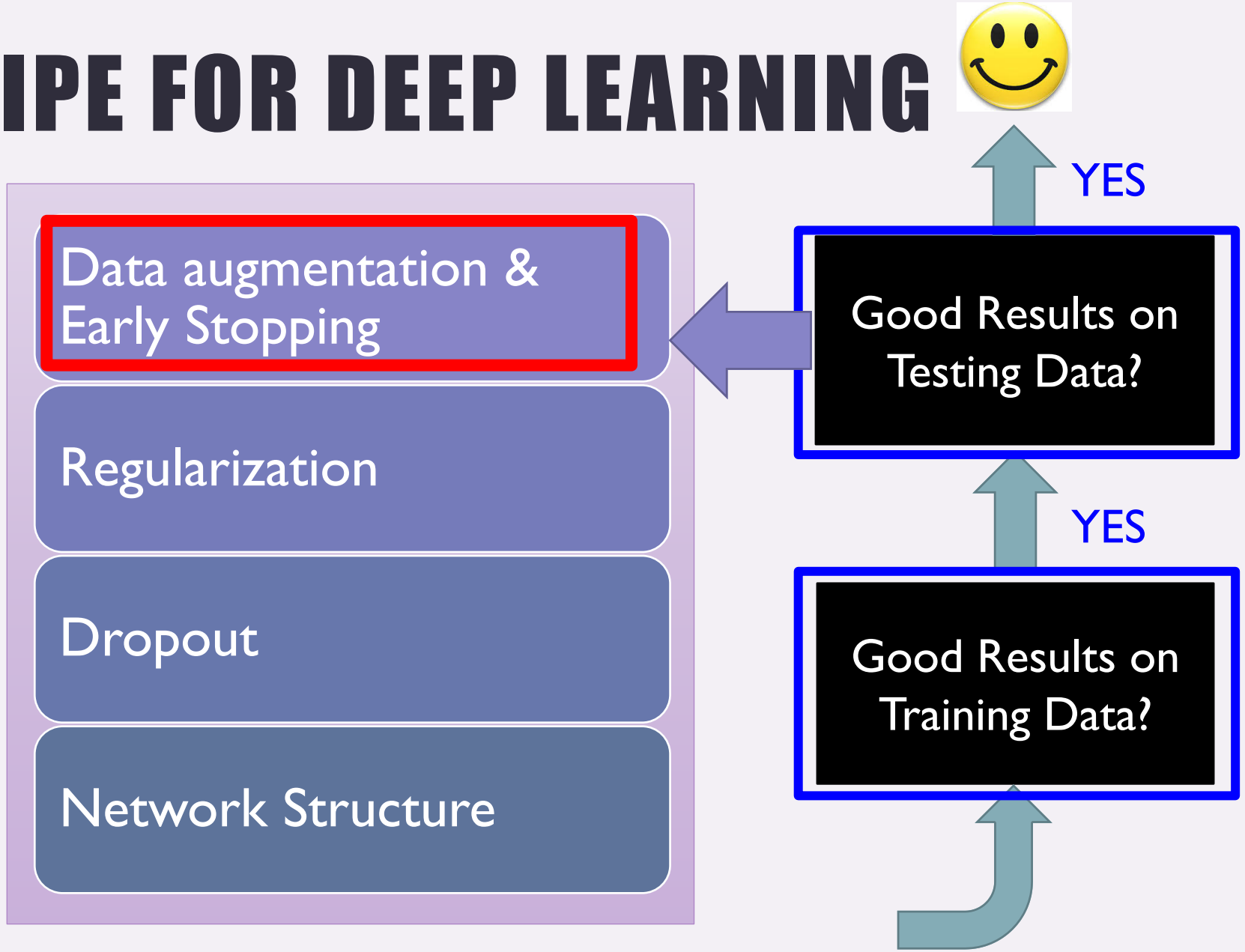
	Responded	Invited	% Rate
Students	2	102	1.96%

60+ responded --- everyone get 1pt
80+ responded --- everyone get 2pts
All responded --- everyone get 3pts

RECIPE FOR DEEP LEARNING



RECIPE FOR DEEP LEARNING



DATA AUGMENTATION

Image credit: https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/

EARLY STOP

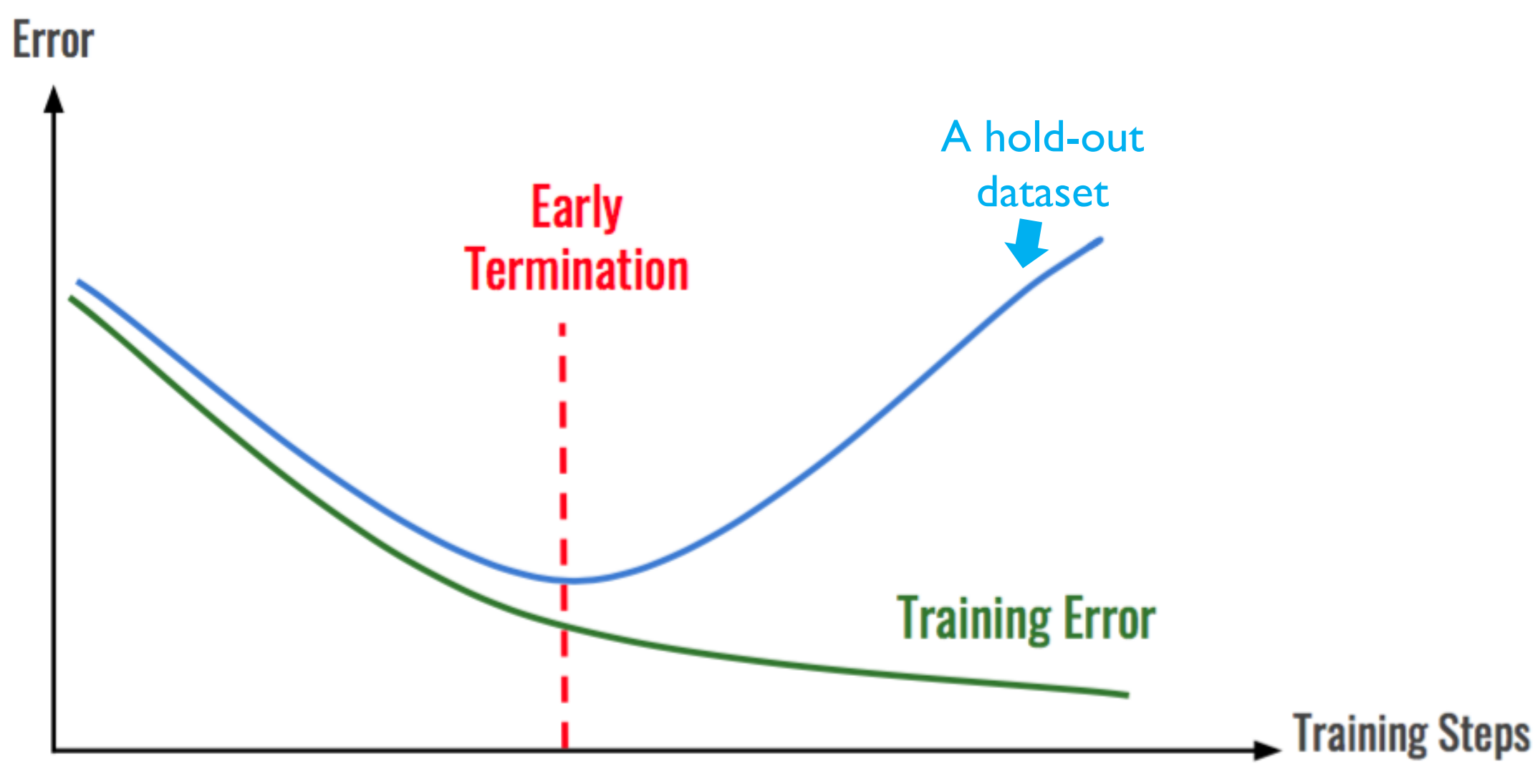
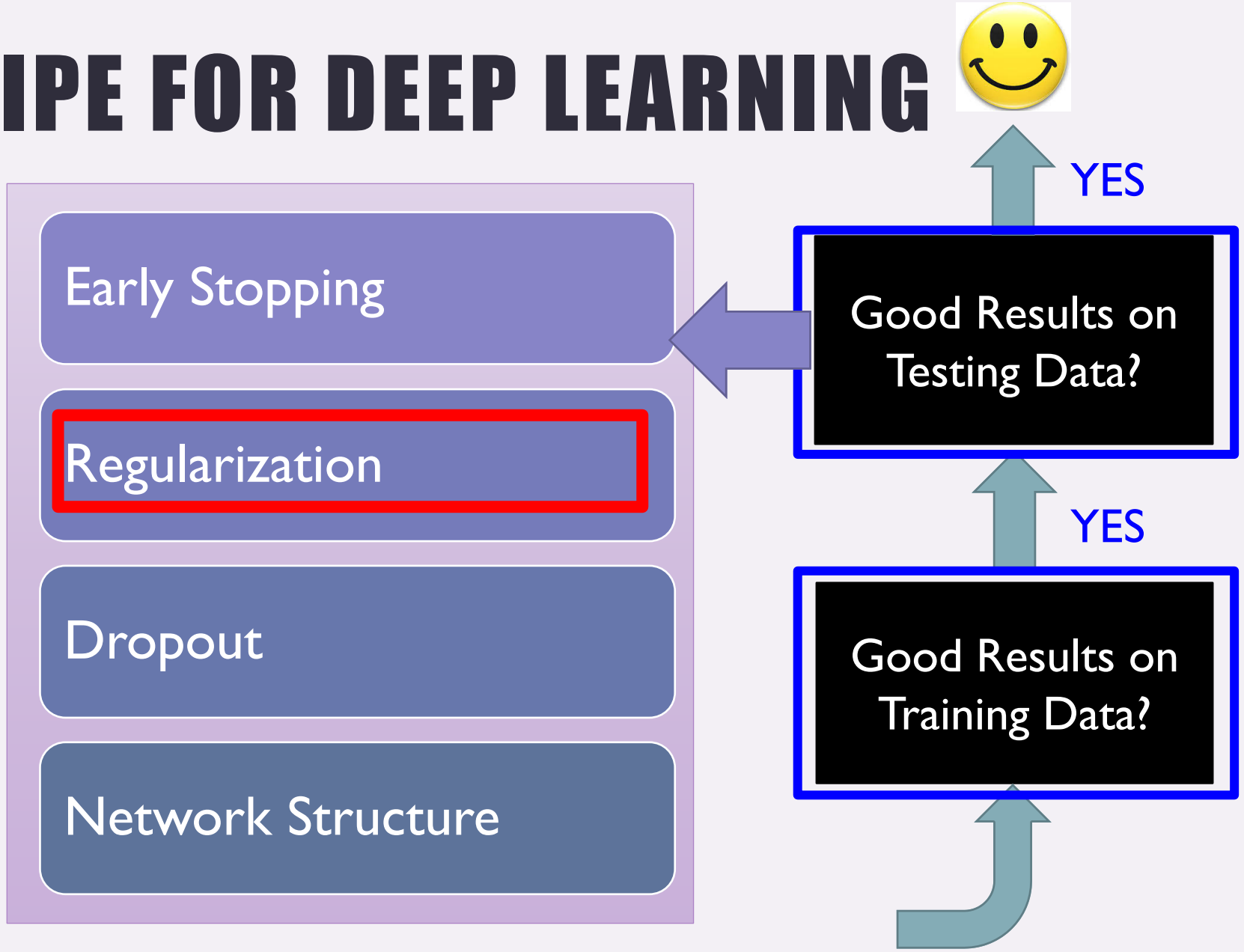


Image credit: https://medium.com/analytics-vidhya/early-stopping-with-pytorch-to-restrain-your-model-from-overfitting-dce6de4081c5

RECIPE FOR DEEP LEARNING



COMMON REGULARIZERS

- Sum of the weights

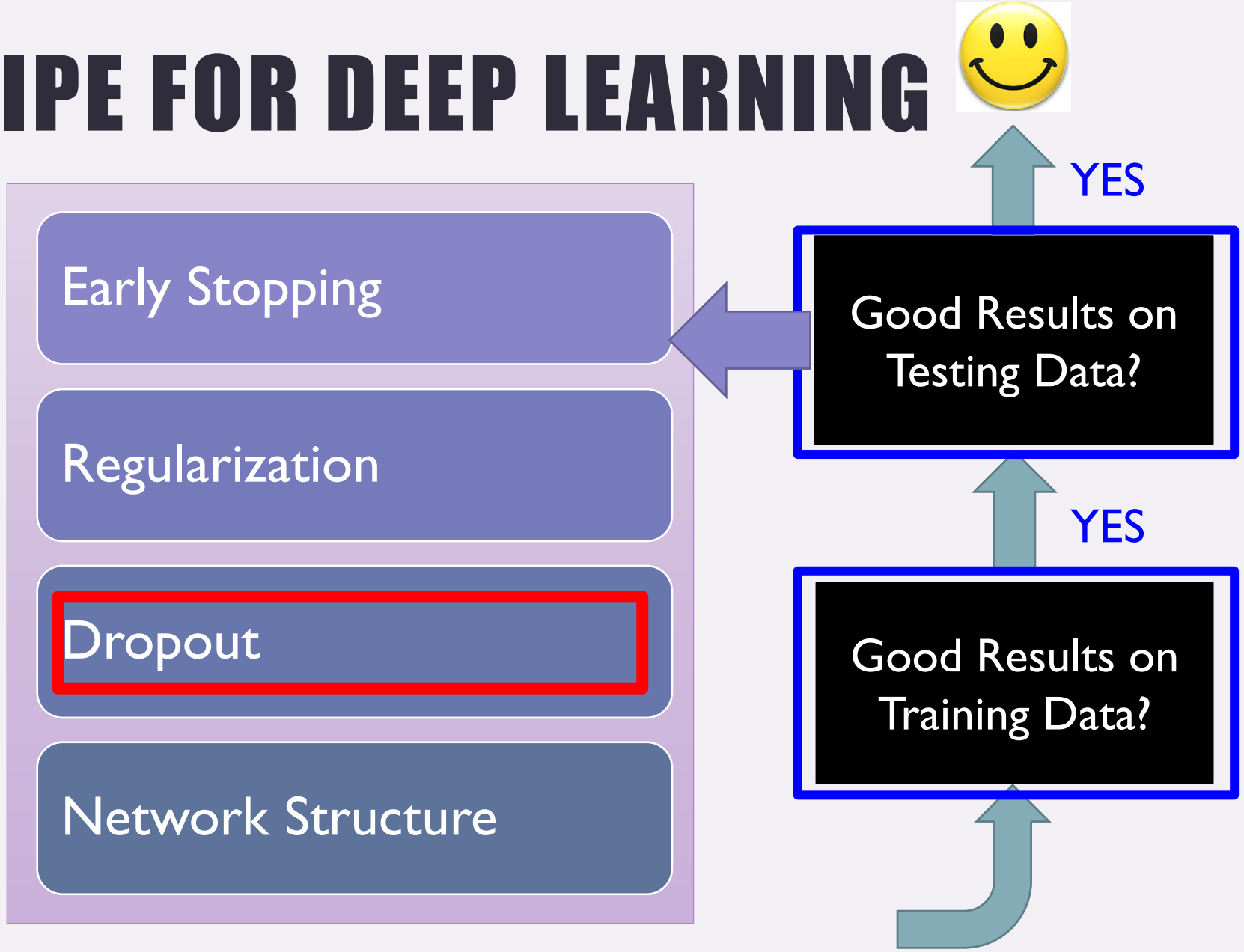
$$r(\theta) = \sum_{\theta_j} |\theta_j|$$

- Sum of the squared weights

$$r(\theta) = \sqrt{\sum_{\theta_j} |\theta_j|^2}$$

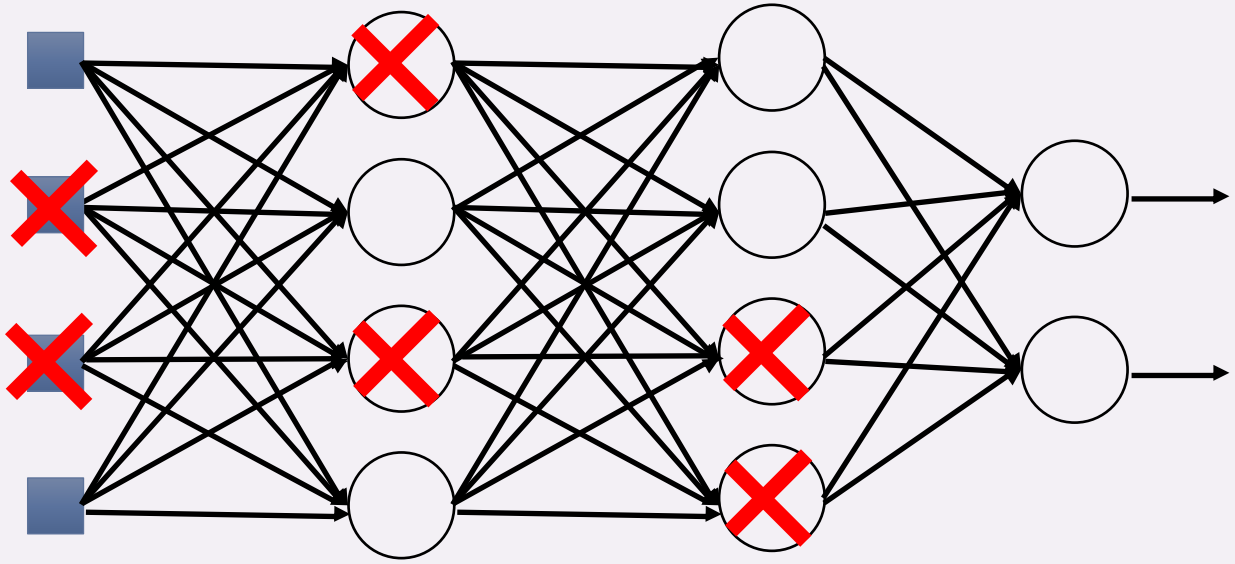
Squared weights penalizes large values more.

RECIPE FOR DEEP LEARNING



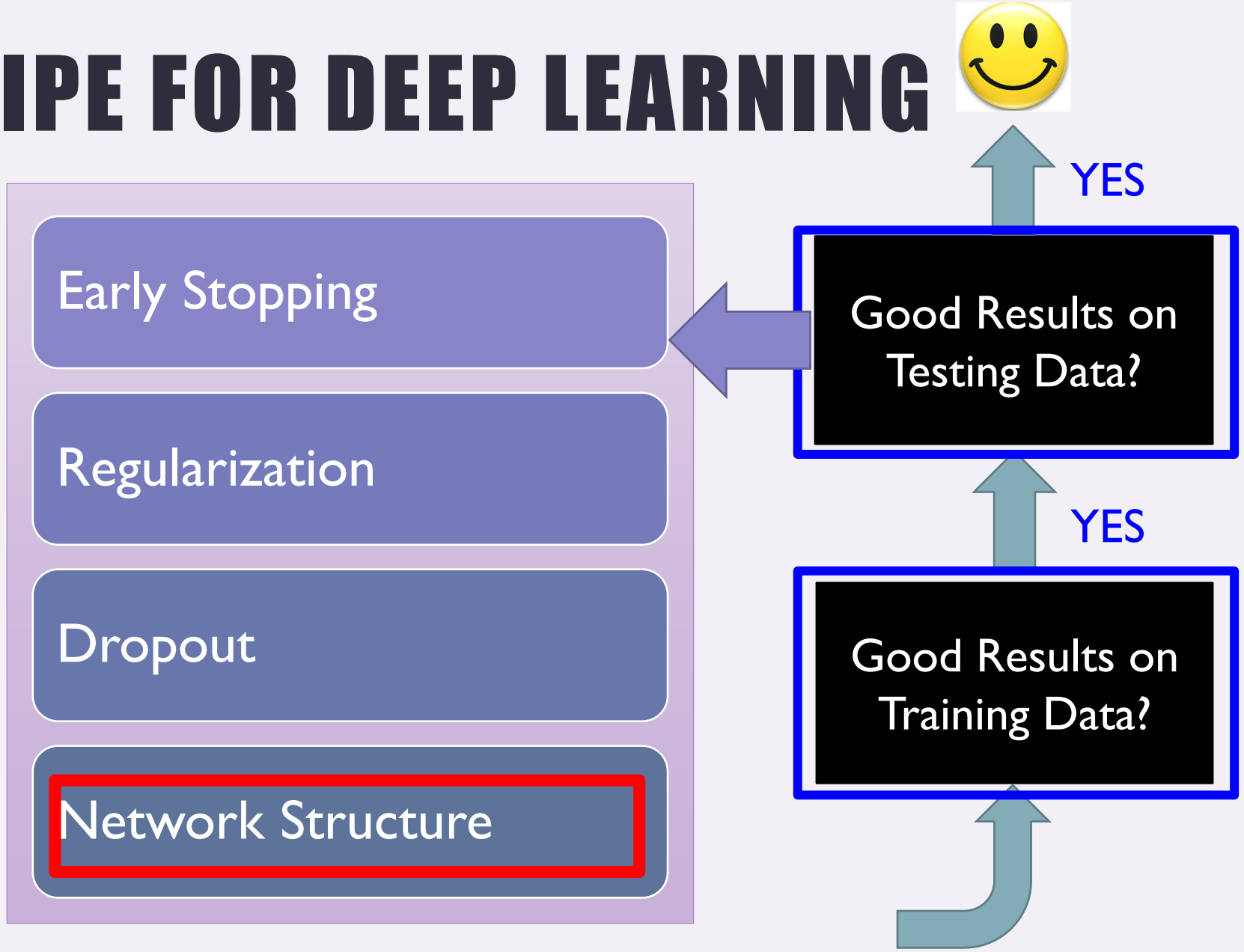
DROPOUT

Training:



- **Each time before updating the parameters**
  - Each neuron has $p\%$ to dropout

RECIPE FOR DEEP LEARNING



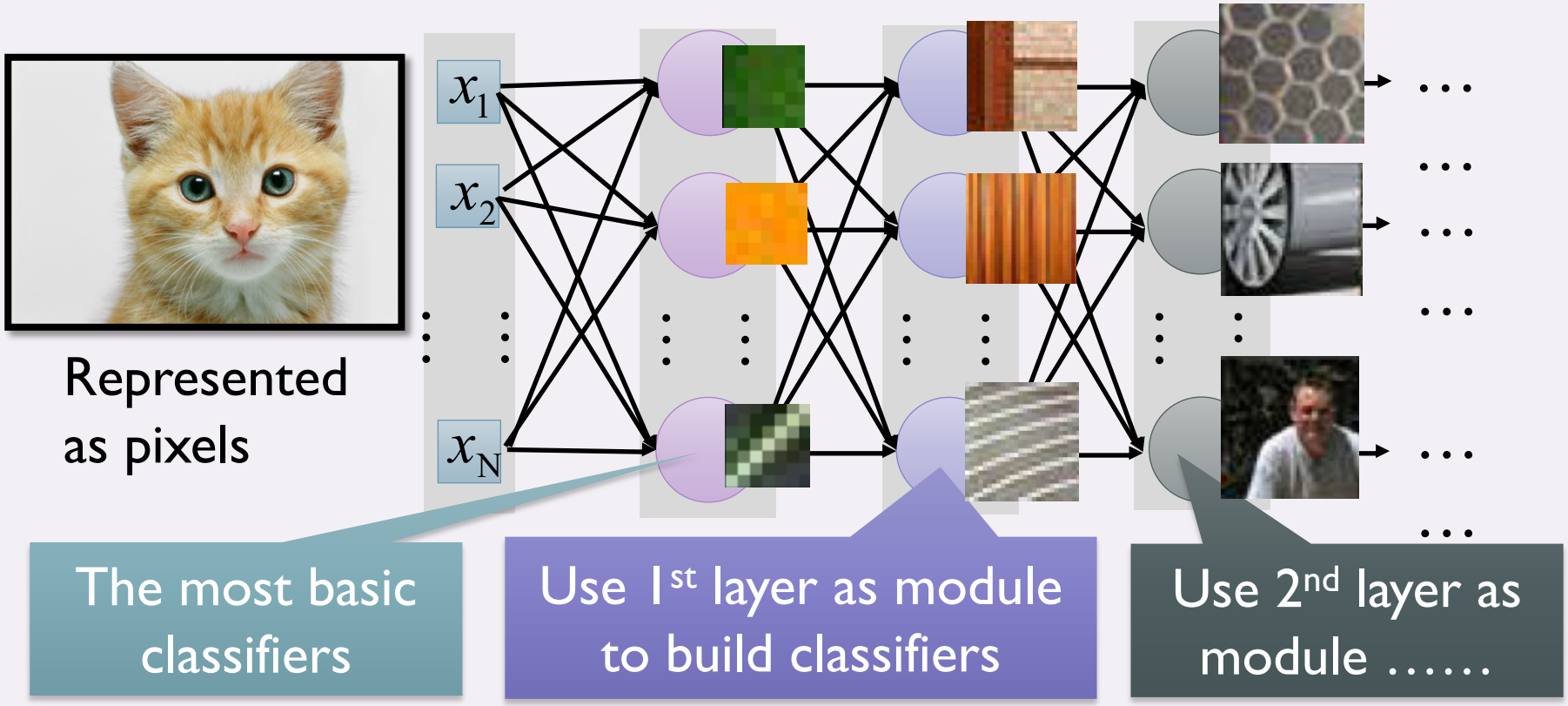
VARIANTS OF NEURAL NETWORKS

- Convolutional Neural Network (CNN)
- Transformer
- Graph Neural Network (GNN)
- Recurrent Neural Network (RNN)

TODAY

- A new deep learning architecture – Convolutional Neural Network
  - Convolution layer
  - Pooling layer
  - TensorFlow implementation

WHY CNN FOR IMAGE?



Can the network be simplified by considering the properties of images?

WHY CNN FOR IMAGE

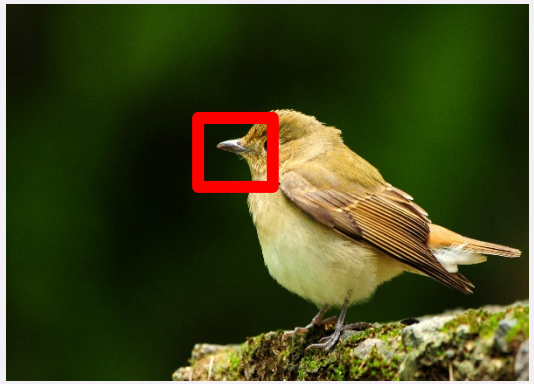
- Some patterns are much smaller than the whole image

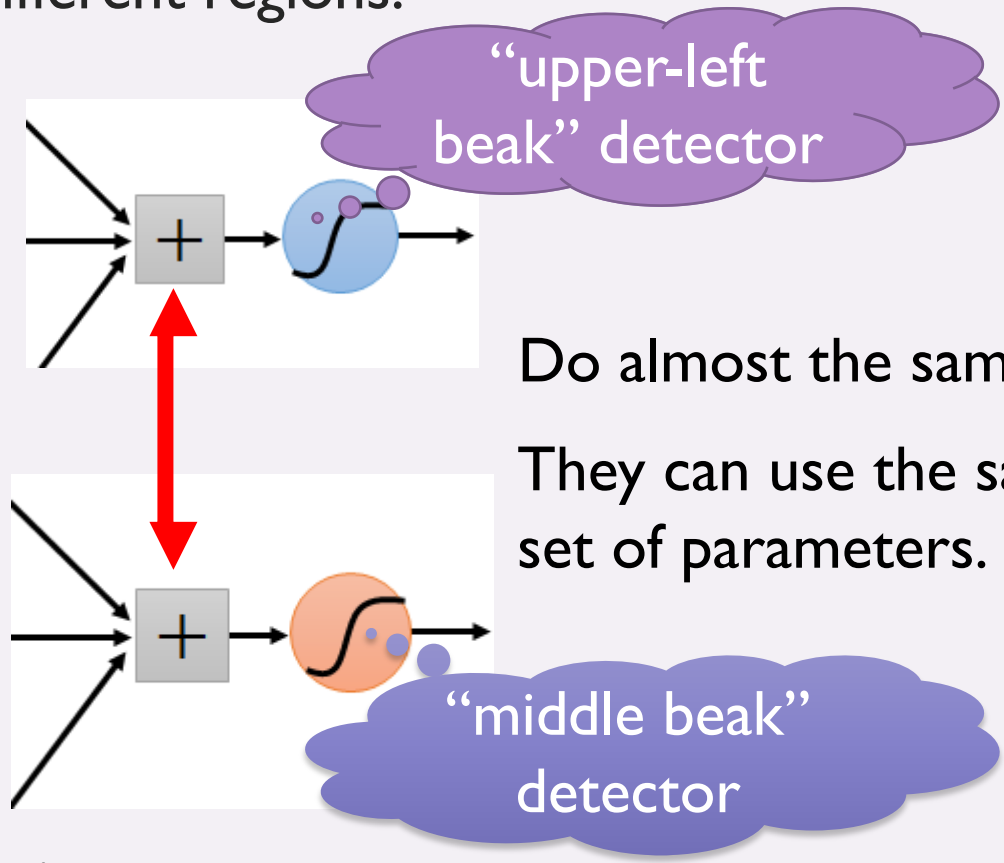
A neuron does not have to see the whole image to discover the pattern.

Connecting to small region with less parameters

WHY CNN FOR IMAGE

- The same patterns appear in different regions.





Do almost the same thing

They can use the same set of parameters.

WHY CNN FOR IMAGE

- Subsampling the pixels will not change the object

bird

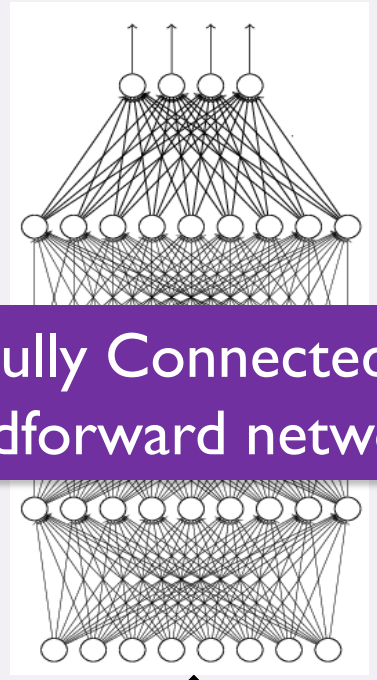
subsampling

bird

We can subsample the pixels to make image smaller

→ Less parameters for the network to process the image

THE WHOLE CNN



cat dog ……

Fully Connected Feedforward network

Flatten

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

THE WHOLE CNN

Property 1

- Some patterns are much smaller than the whole image

Property 2

- The same patterns appear in different regions.

Property 3

- Subsampling the pixels will not change the object

Convolution

Max Pooling

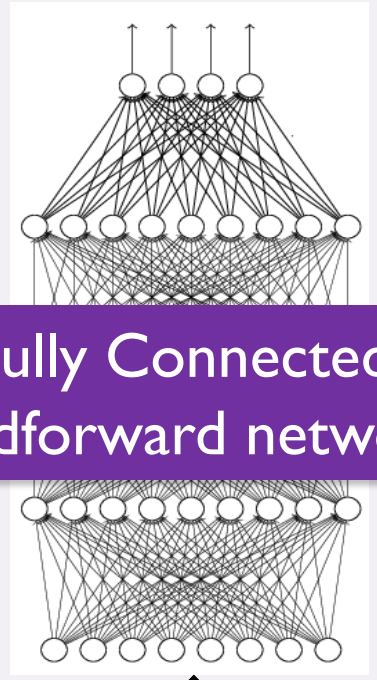
Convolution

Max Pooling

Can repeat many times

Flatten

THE WHOLE CNN



cat dog ……

Fully Connected Feedforward network

Flatten

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

CNN – CONVOLUTION

These are the network parameters to be learned.

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

Matrix

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

Matrix

⋮ ⋮

Property 1

Each filter detects a small pattern (3 x 3).

CNN – CONVOLUTION

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

If stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

3 -1

CNN – CONVOLUTION

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

If stride=2

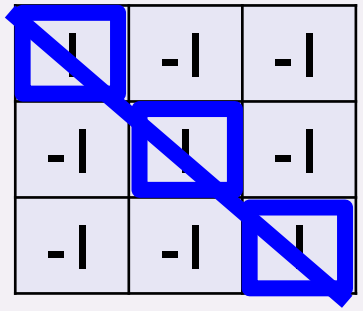
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

3 -3

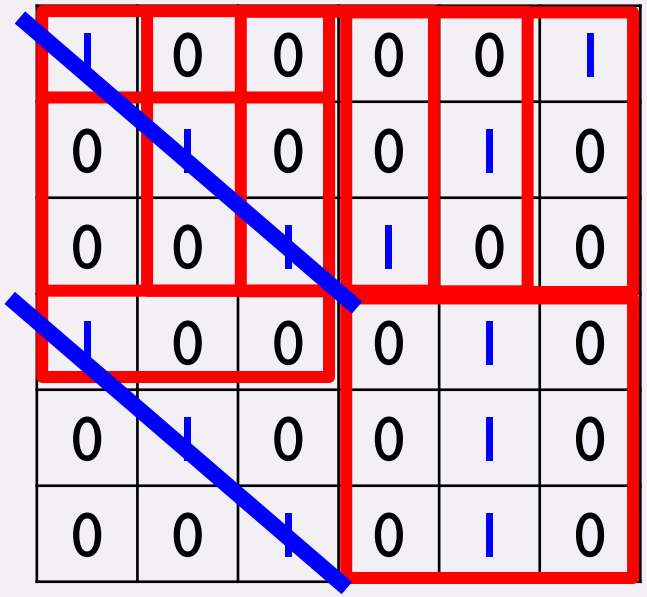
We set stride=1 below

CNN – CONVOLUTION

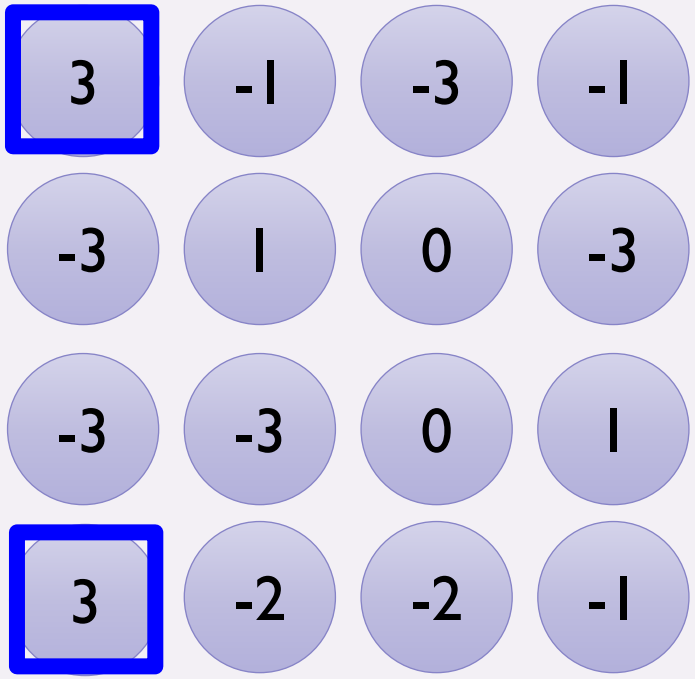


Filter 1

stride=1



6 x 6 image



Property 2

CNN – CONVOLUTION

-1	1	-1
-1	1	-1
-1	1	-1

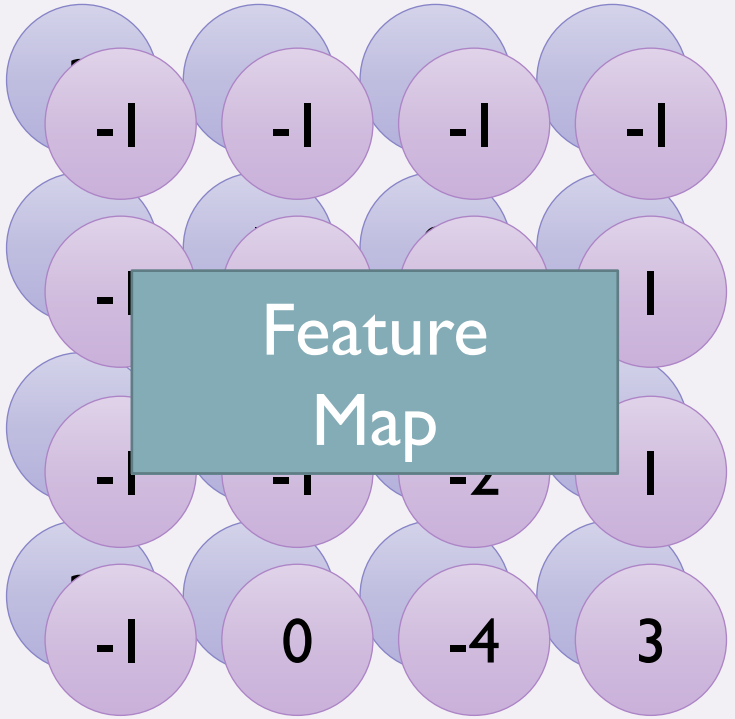
Filter 2

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

Do the same process for every filter

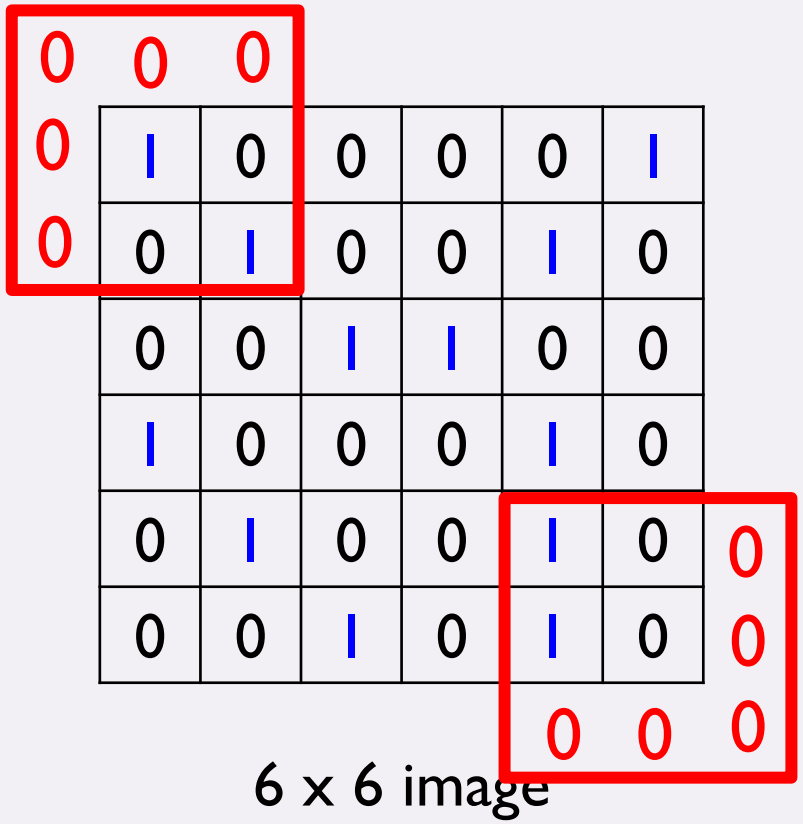


4 x 4 image

CNN – ZERO PADDING

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

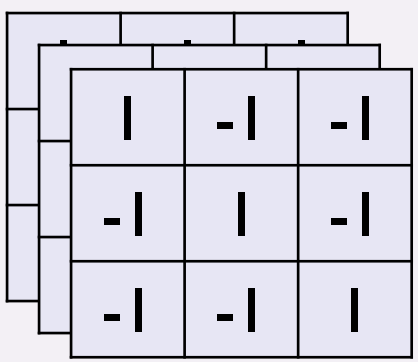


6 x 6 image

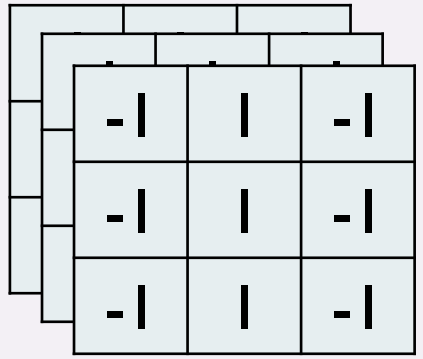
You will get another 6 x 6 images in this way

→ Zero padding

CNN – COLORFUL IMAGE

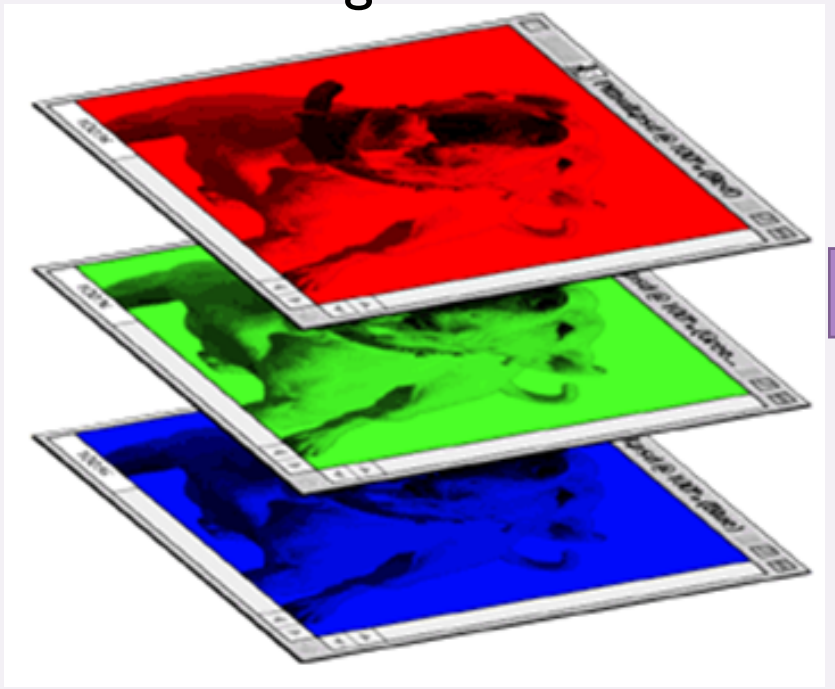


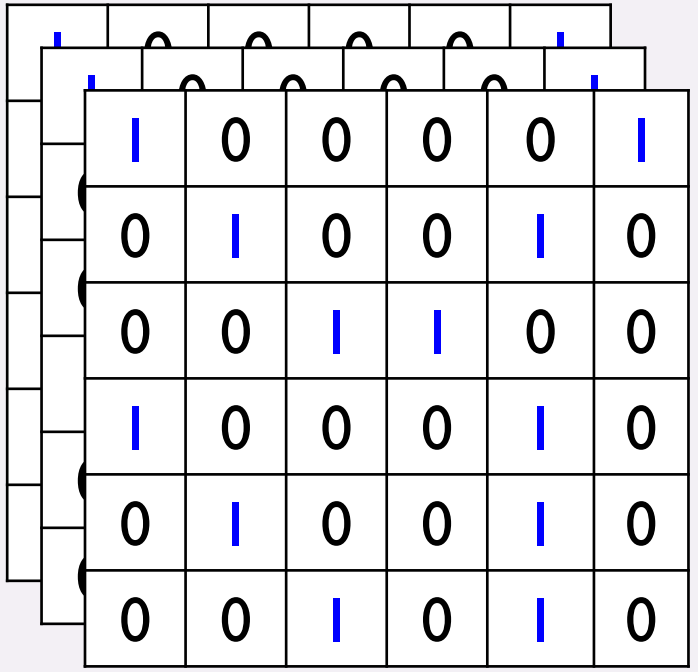
Filter 1



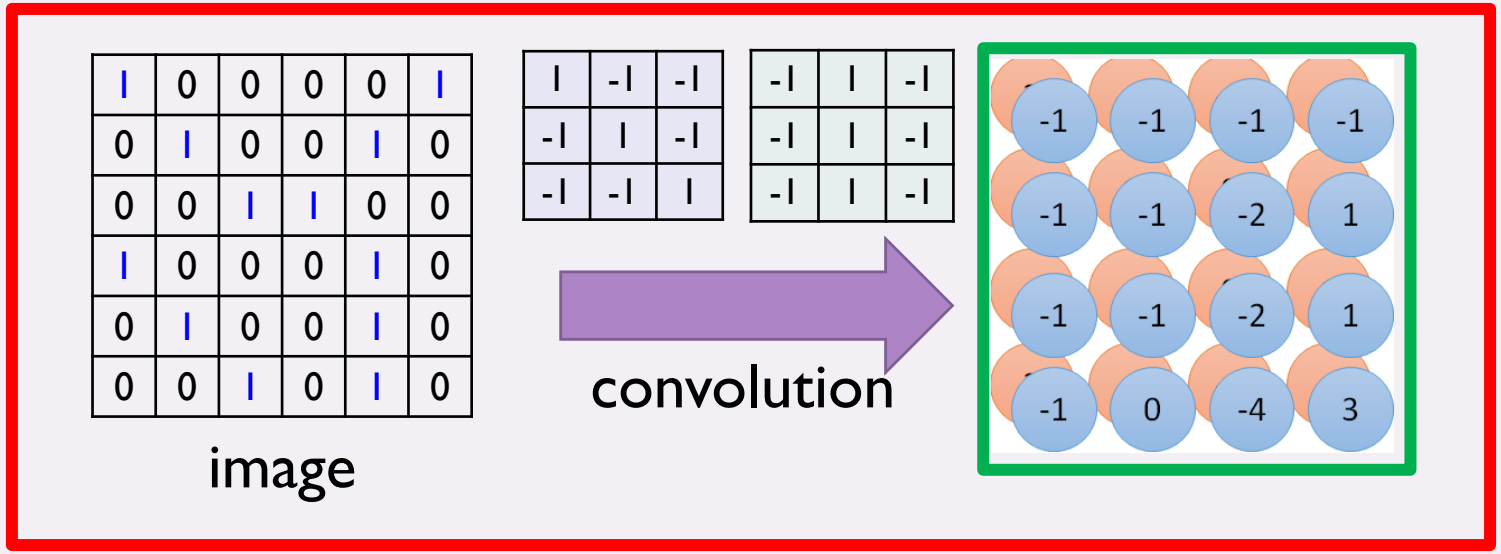
Filter 2

Colorful image

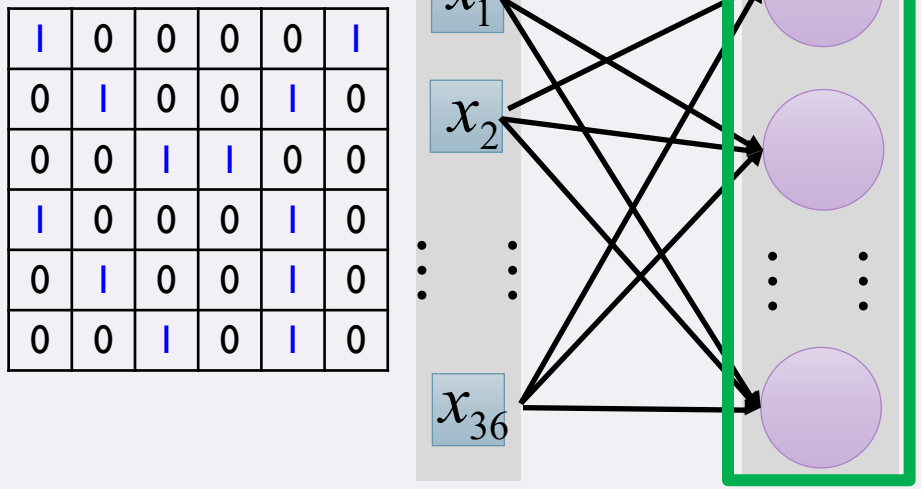


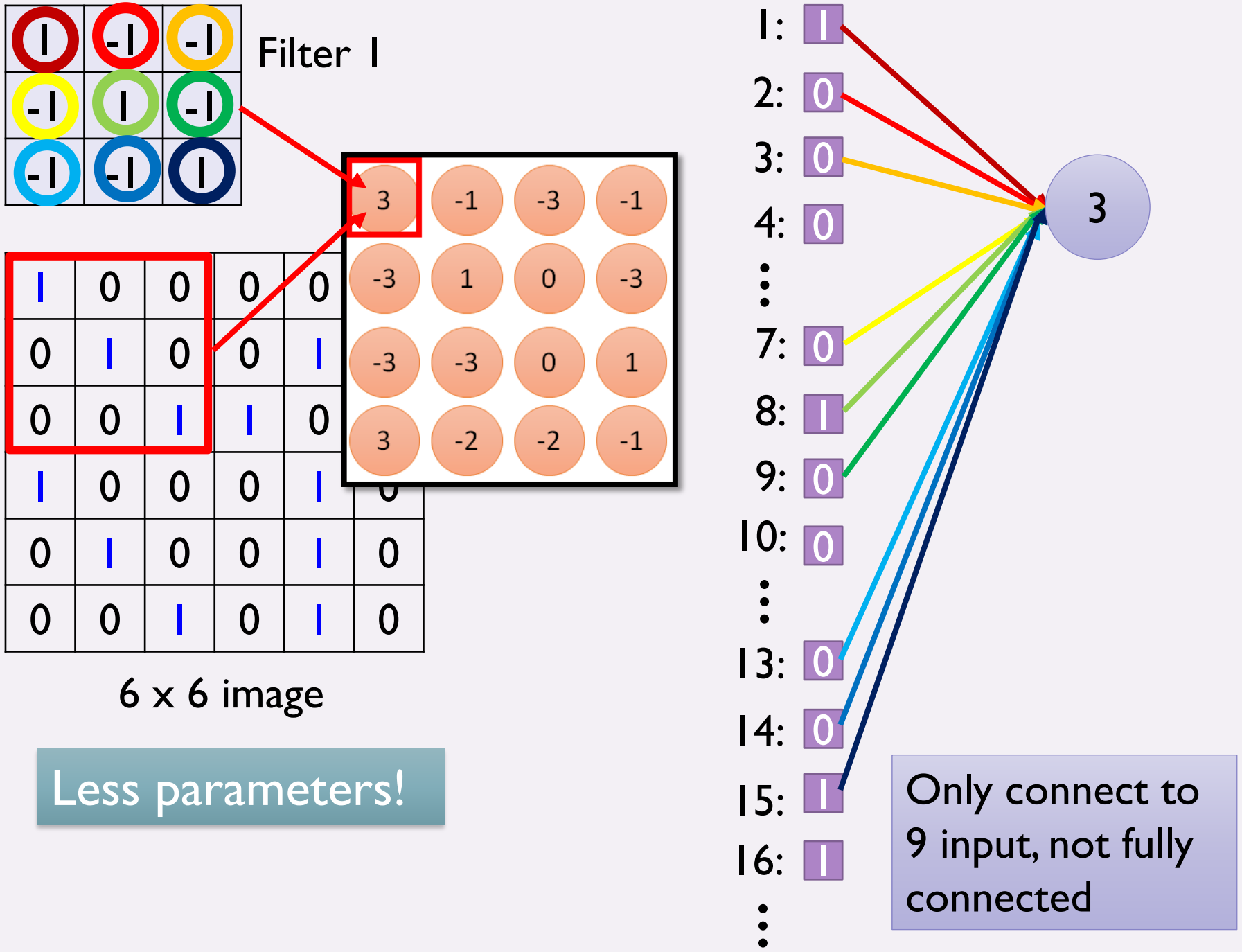


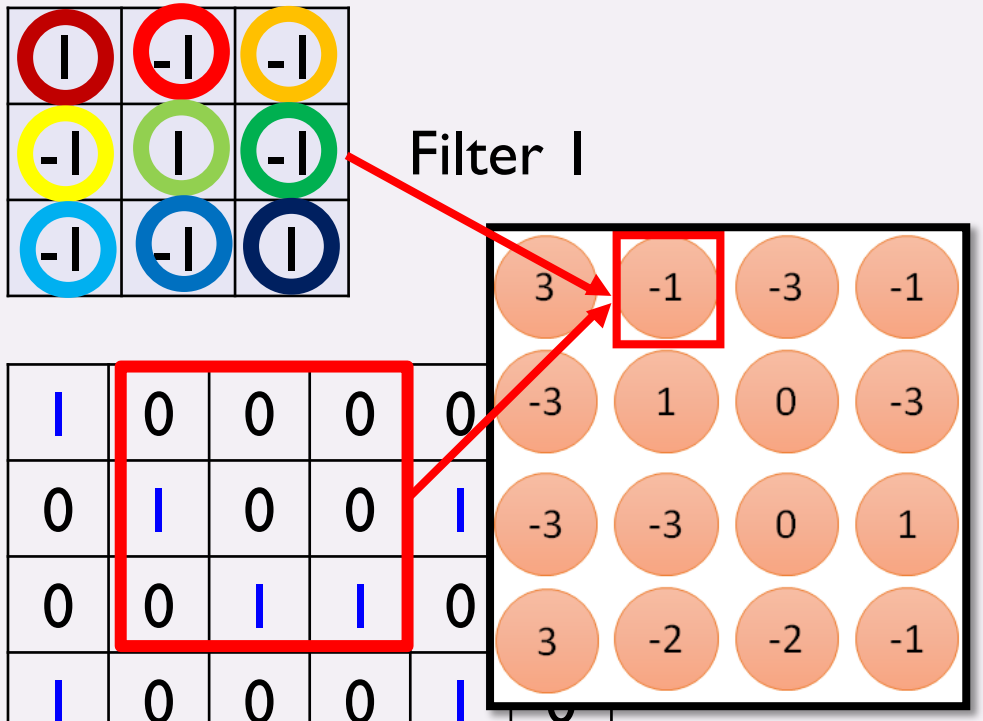
CONVOLUTION VS FULLY-CONNECTED



Fully-connected

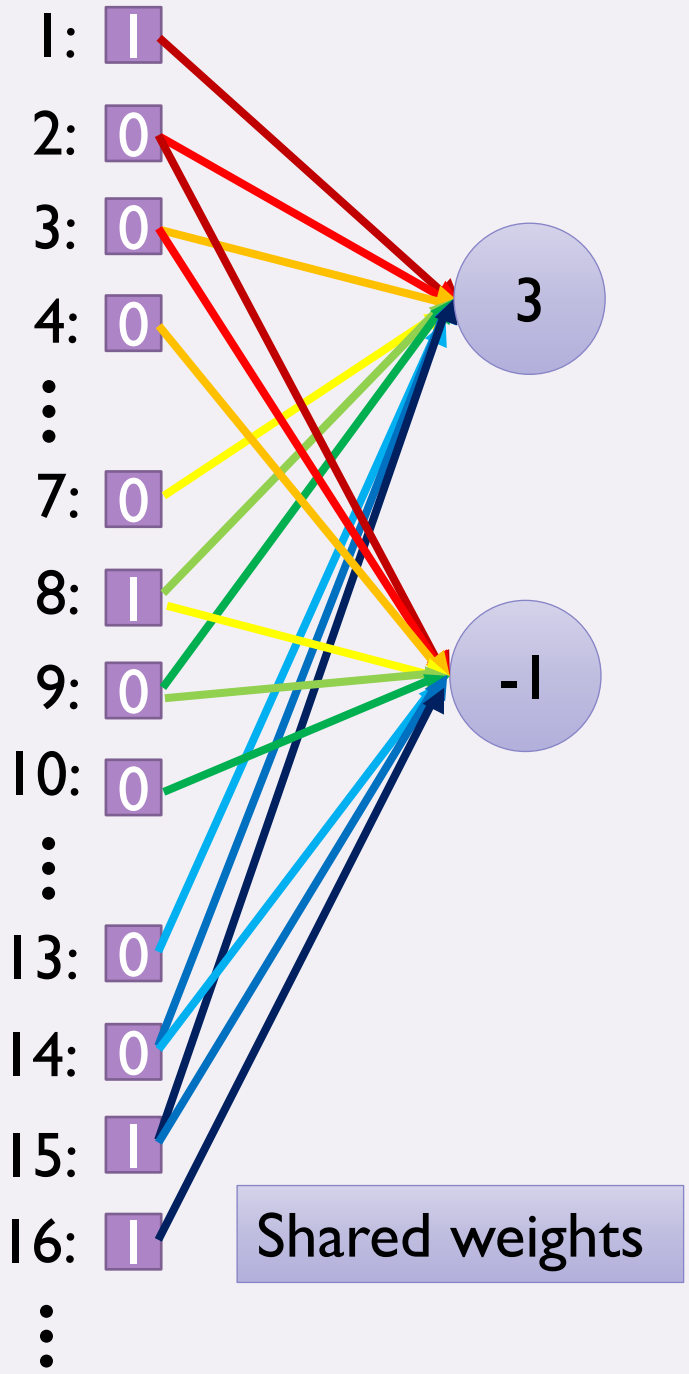



1 -1 -1
-1 1 -1
-1 -1 1
Filter 1
3 -1 -3 -1
-3 1 0 -3
-3 -3 0 1
3 -2 -2 -1
1 0 0 0 0
0 1 0 0 1
0 0 1 1 0
1 0 0 0 1 0
0 1 0 0 1 0
0 0 1 0 1 0
6 x 6 image
Less parameters!
1:
2:
3:
4:
7:
8:
9:
10:
13:
14:
15:
16:
3
Only connect to 9 input, not fully connected

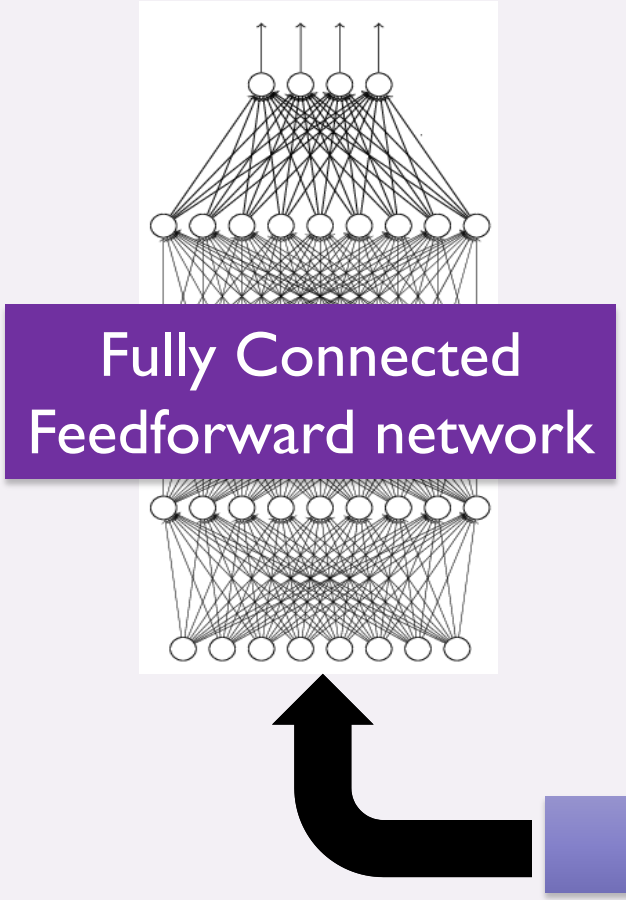


Less parameters!

Even less parameters!



THE WHOLE CNN



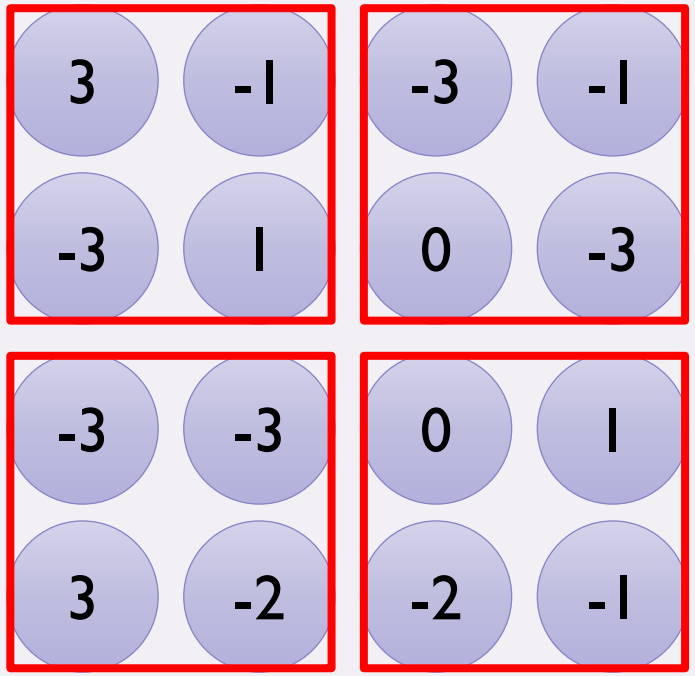
CNN – MAX POOLING

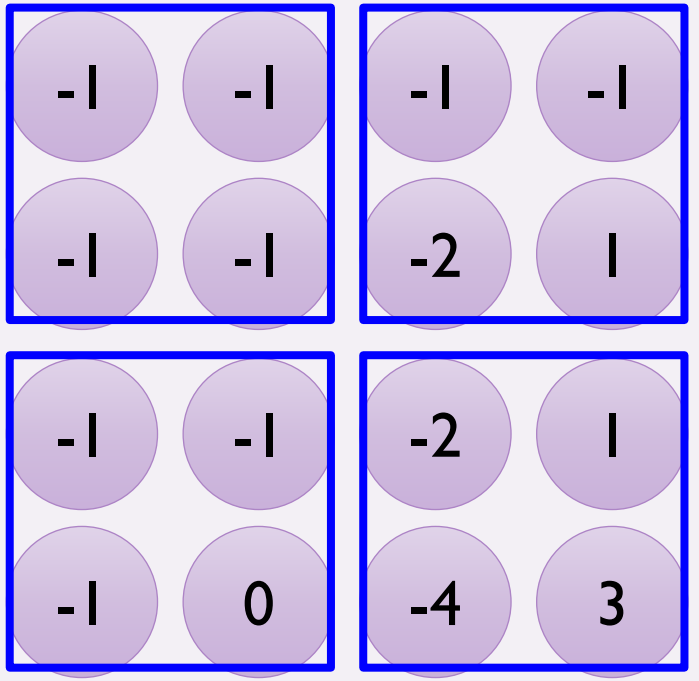
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

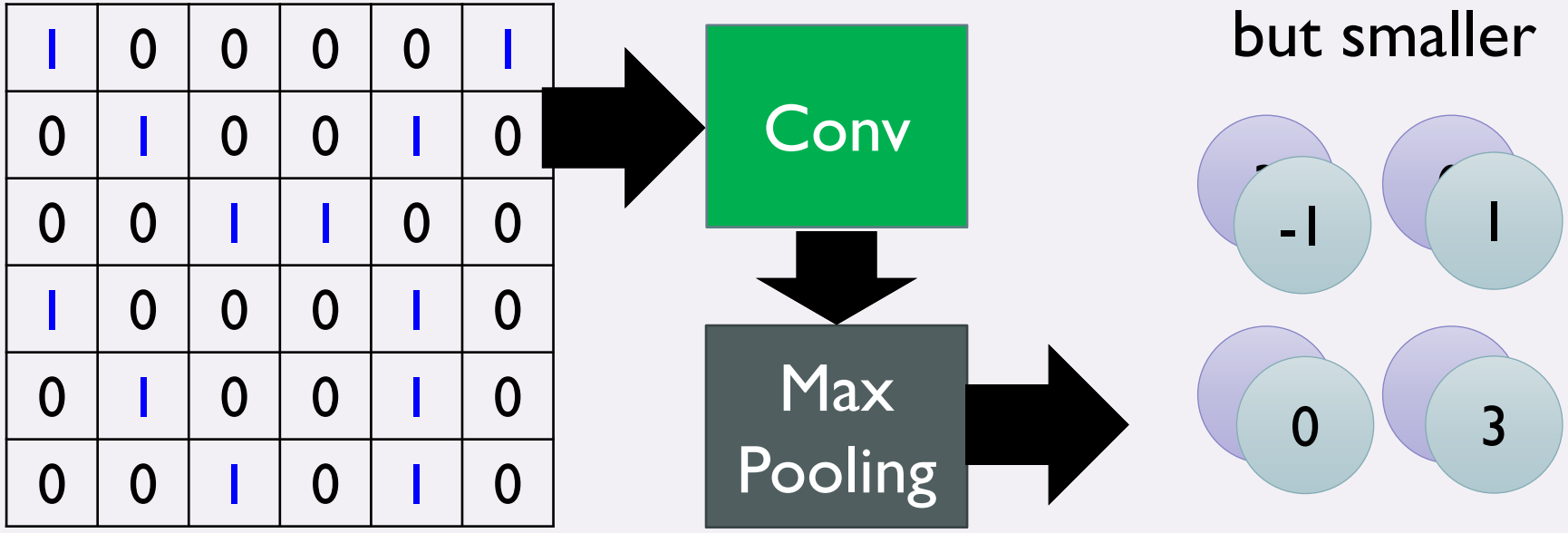
-1	1	-1
-1	1	-1
-1	1	-1

Filter 2





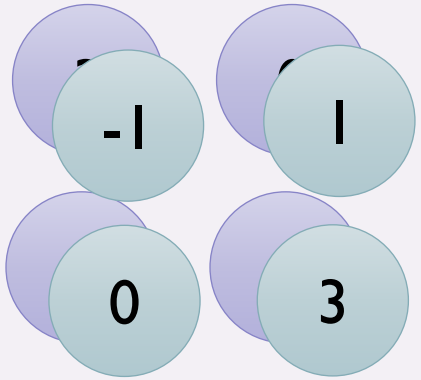
CNN – MAX POOLING



6 x 6 image

Each filter is a channel

THE WHOLE CNN



A new image

Convolution

Max Pooling

Convolution

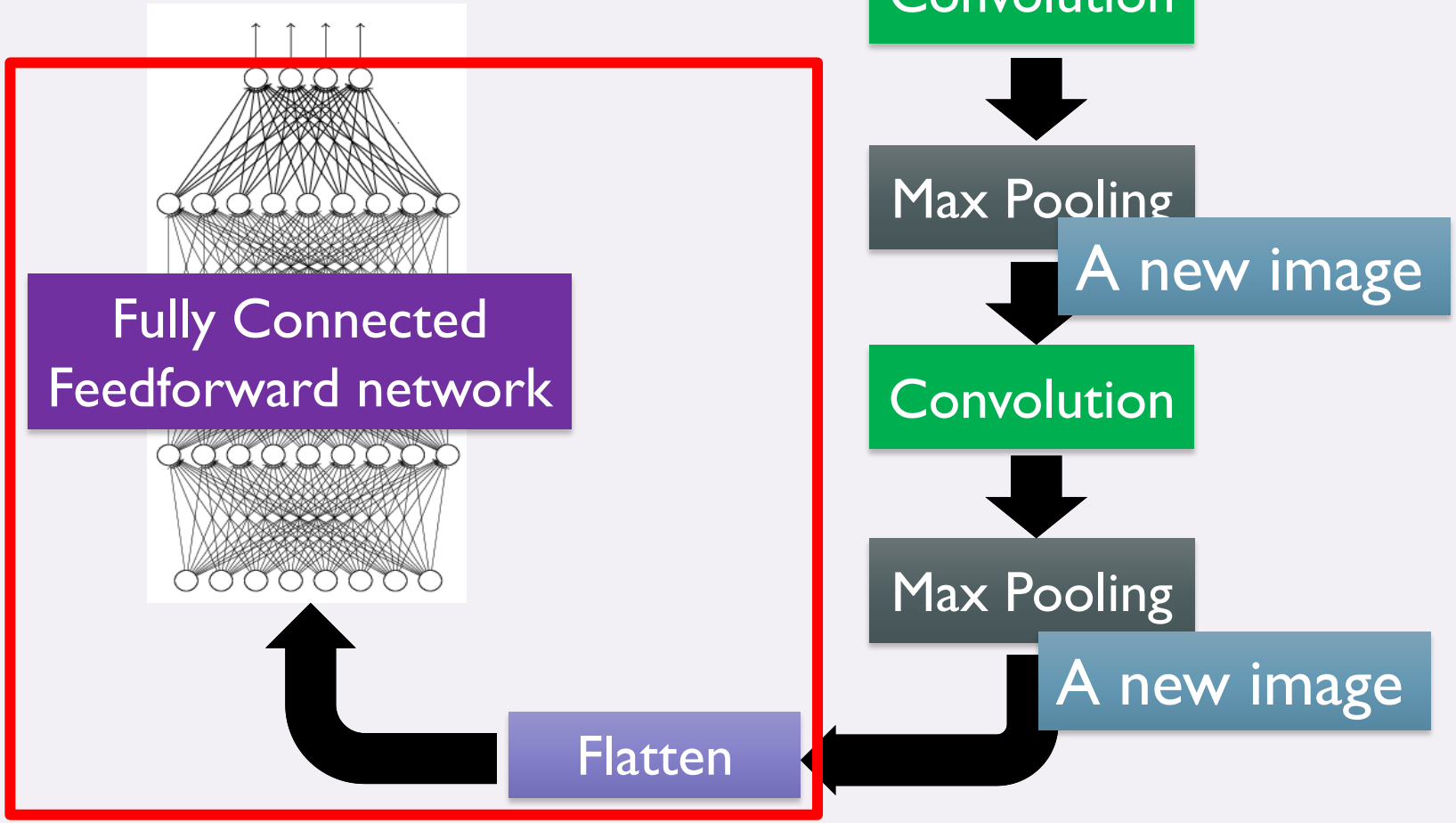
Max Pooling

Can repeat many times

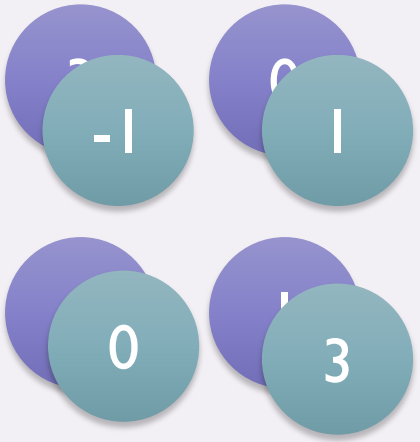
Smaller than the original image

The number of the channel is the number of filters

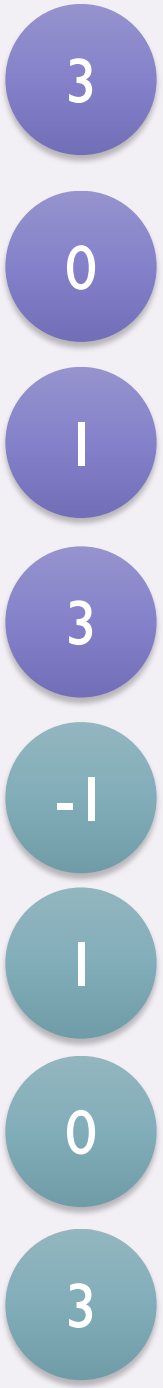
THE WHOLE CNN

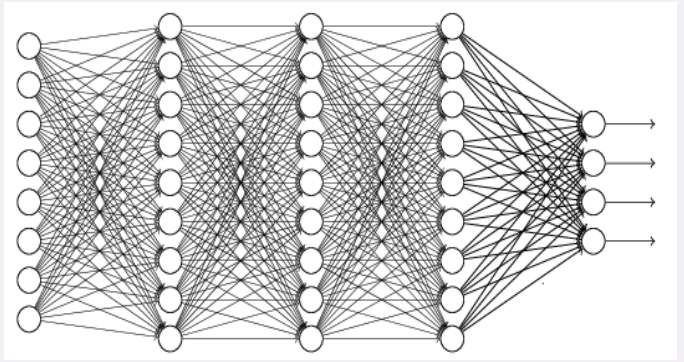


FLATTEN



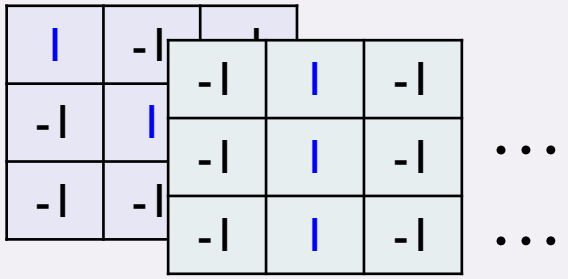
Flatten





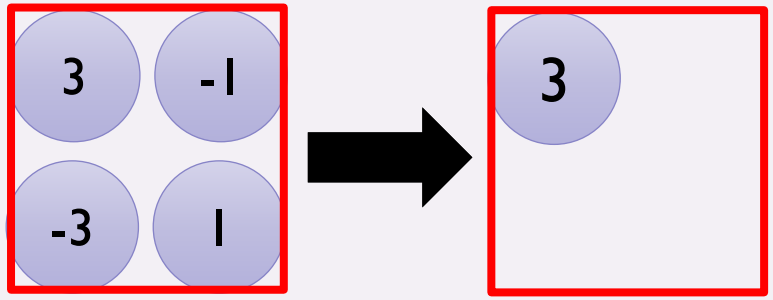
Fully Connected Feedforward network

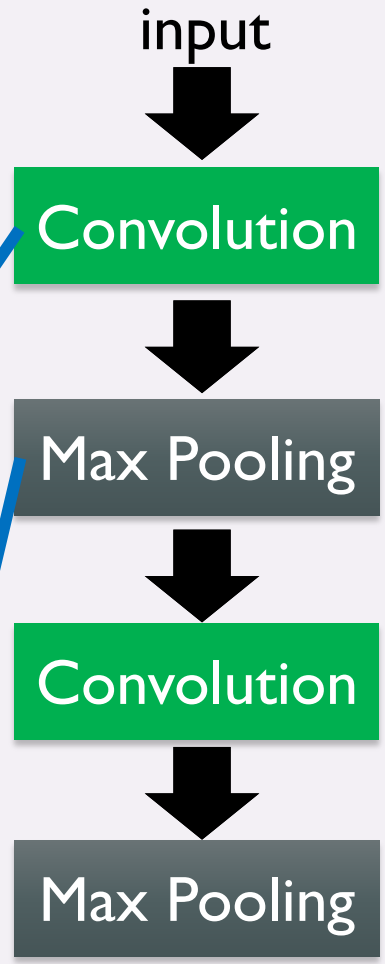
```
model.add(Conv2D(filters = 25,
kernel_size = (3, 3),
padding = 'same',
activation = 'relu',
input_shape = (28, 28, 1)))
```

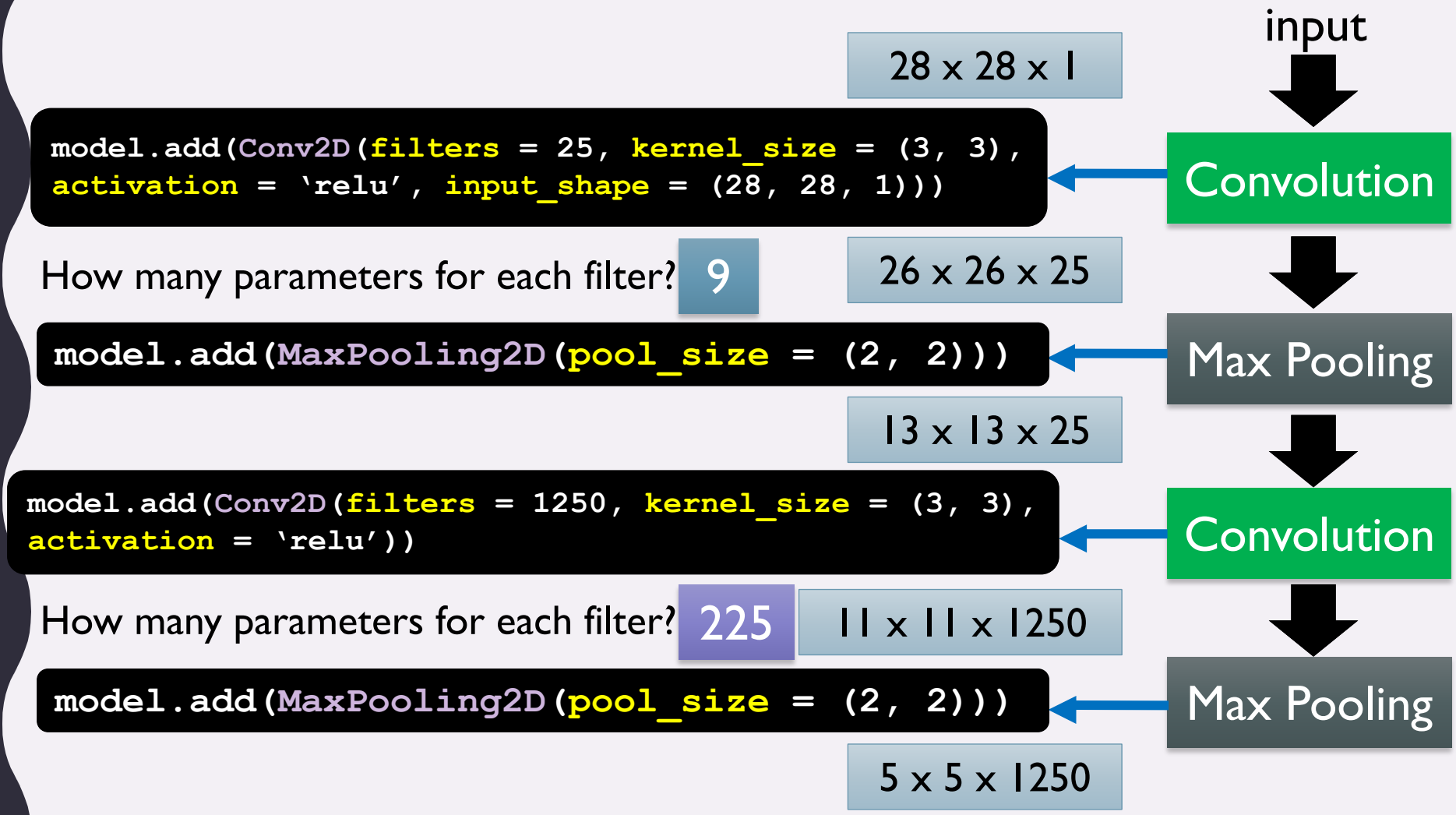


There are 25 3x3 filters.

```
model.add(MaxPooling2D(pool_size = (2, 2)))
```






input
28 x 28 x 1
model.add(Conv2D(filters = 25, kernel_size = (3, 3), activation = 'relu', input_shape = (28, 28, 1)))
Convolution
How many parameters for each filter?
9
26 x 26 x 25
model.add(MaxPooling2D(pool_size = (2, 2)))
Max Pooling
13 x 13 x 25
model.add(Conv2D(filters = 1250, kernel_size = (3, 3), activation = 'relu'))
Convolution
How many parameters for each filter?
225
11 x 11 x 1250
model.add(MaxPooling2D(pool_size = (2, 2)))
Max Pooling
5 x 5 x 1250

input

28 x 28 x 1

Convolution

26 x 26 x 25

Max Pooling

13 x 13 x 25

Convolution

11 x 11 x 1250

Max Pooling

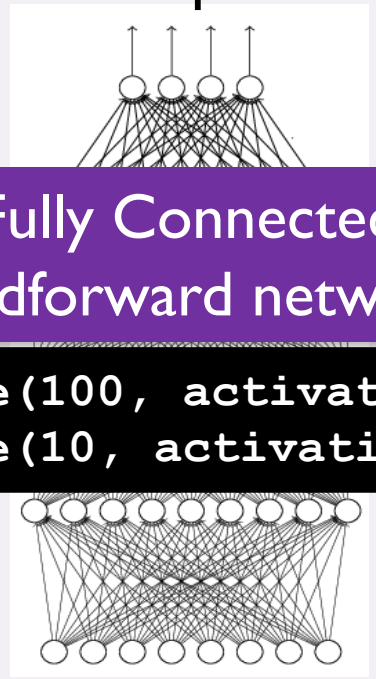
5 x 5 x 1250

```
model.add(Flatten())
```

Flatten

31250

output



Fully Connected Feedforward network

```
model.add(Dense(100, activation='relu'))
model.add(Dense(10, activation='softmax'))
```

EXERCISE

HTTPS://BIT.LY/2RKUDRQ

QUESTIONS?