투빅스 11기 정규과정

ToBig's 10기 박성진

Convolutional Neural Networks

Contents

Unit 01 | Intro

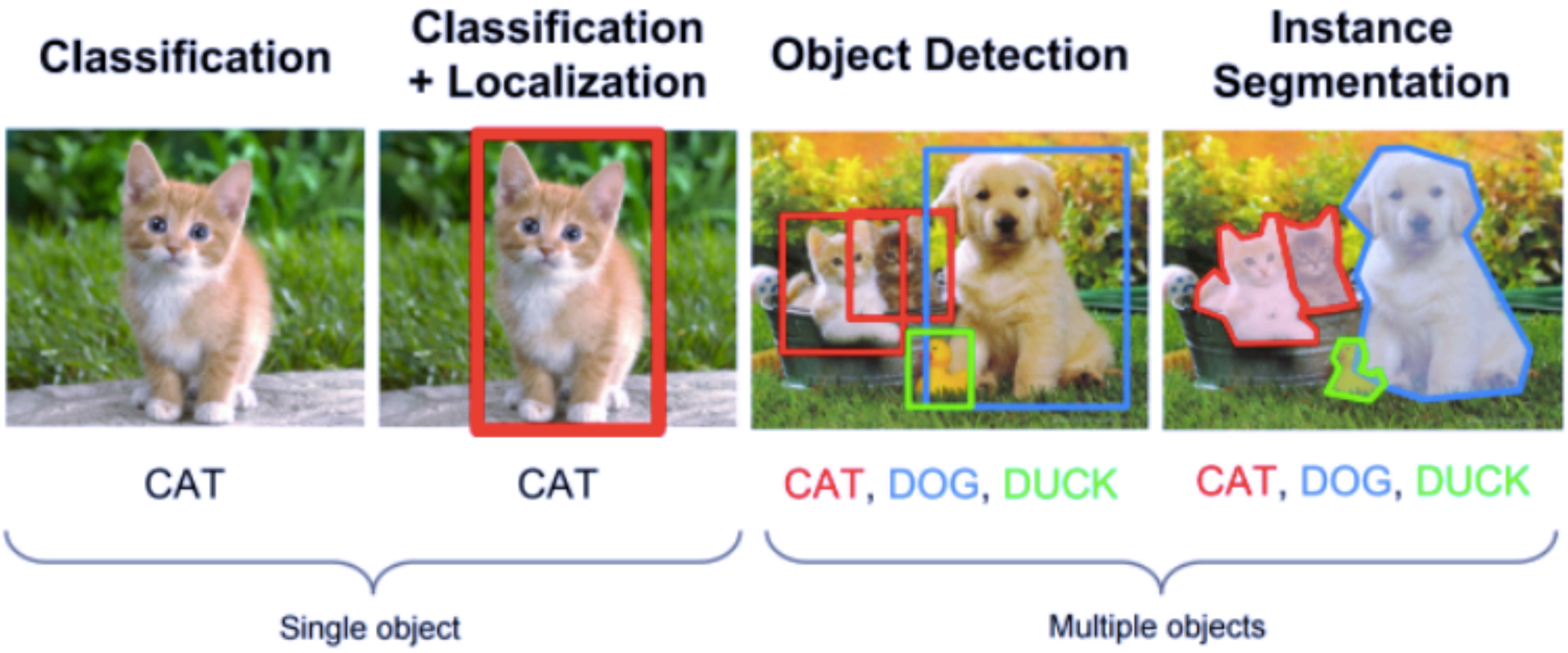
Unit 02 | Layers in CNN

Unit 03 | Convolution

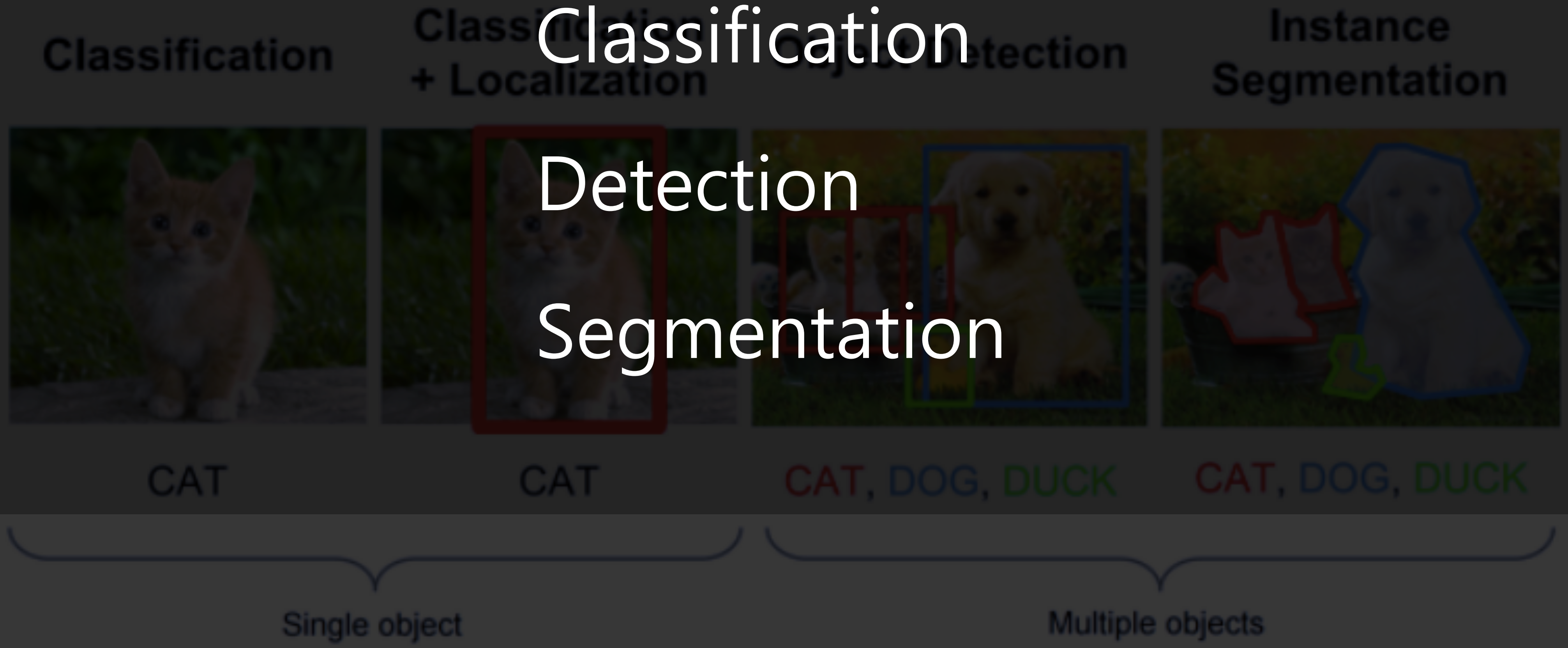
Unit 04 | Sub-Sampling

Unit 05 | Summary

◆ Intro - Applications of CNN



◆ Intro - Applications of CNN

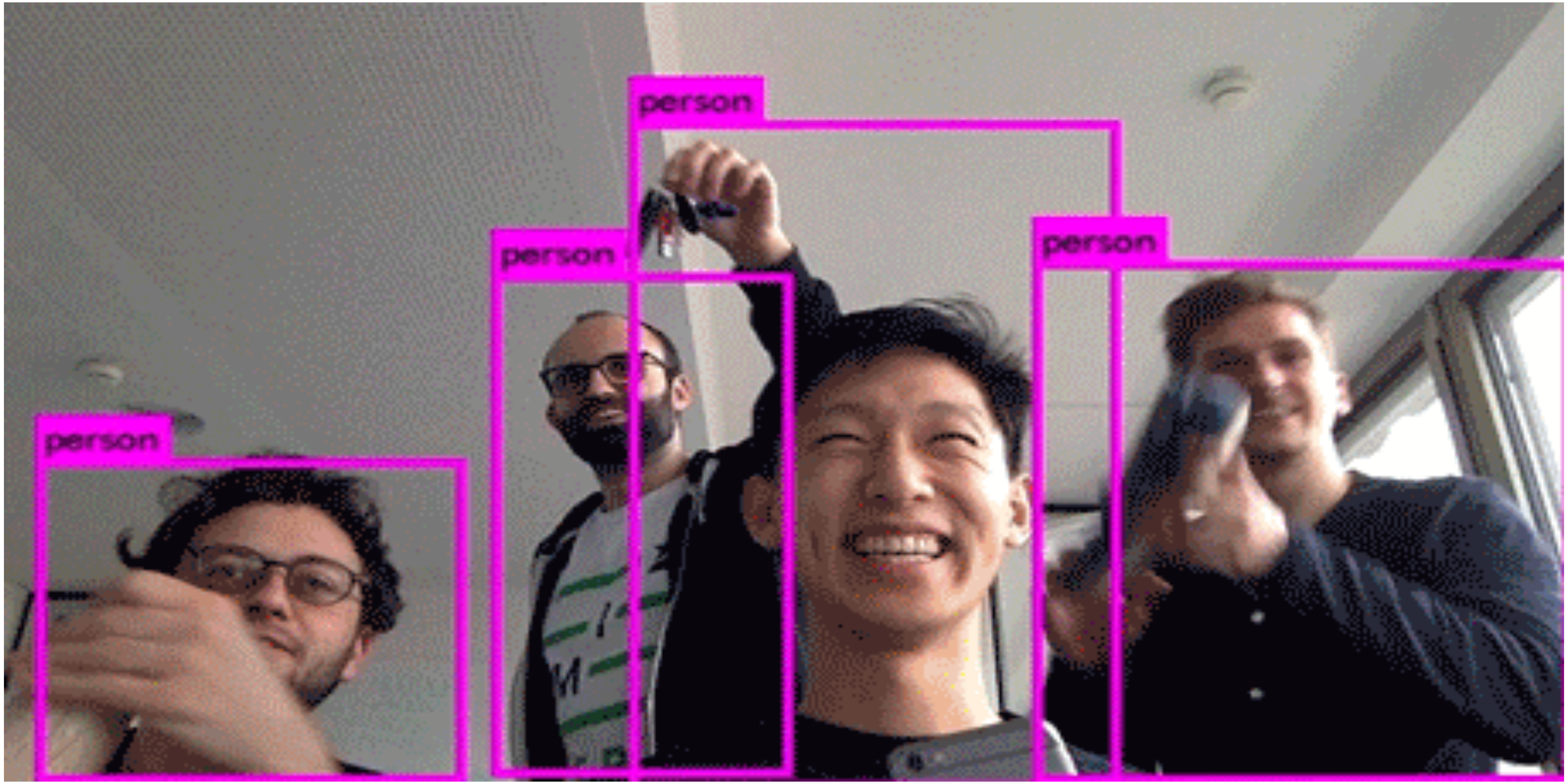


Classification

Detection

Segmentation

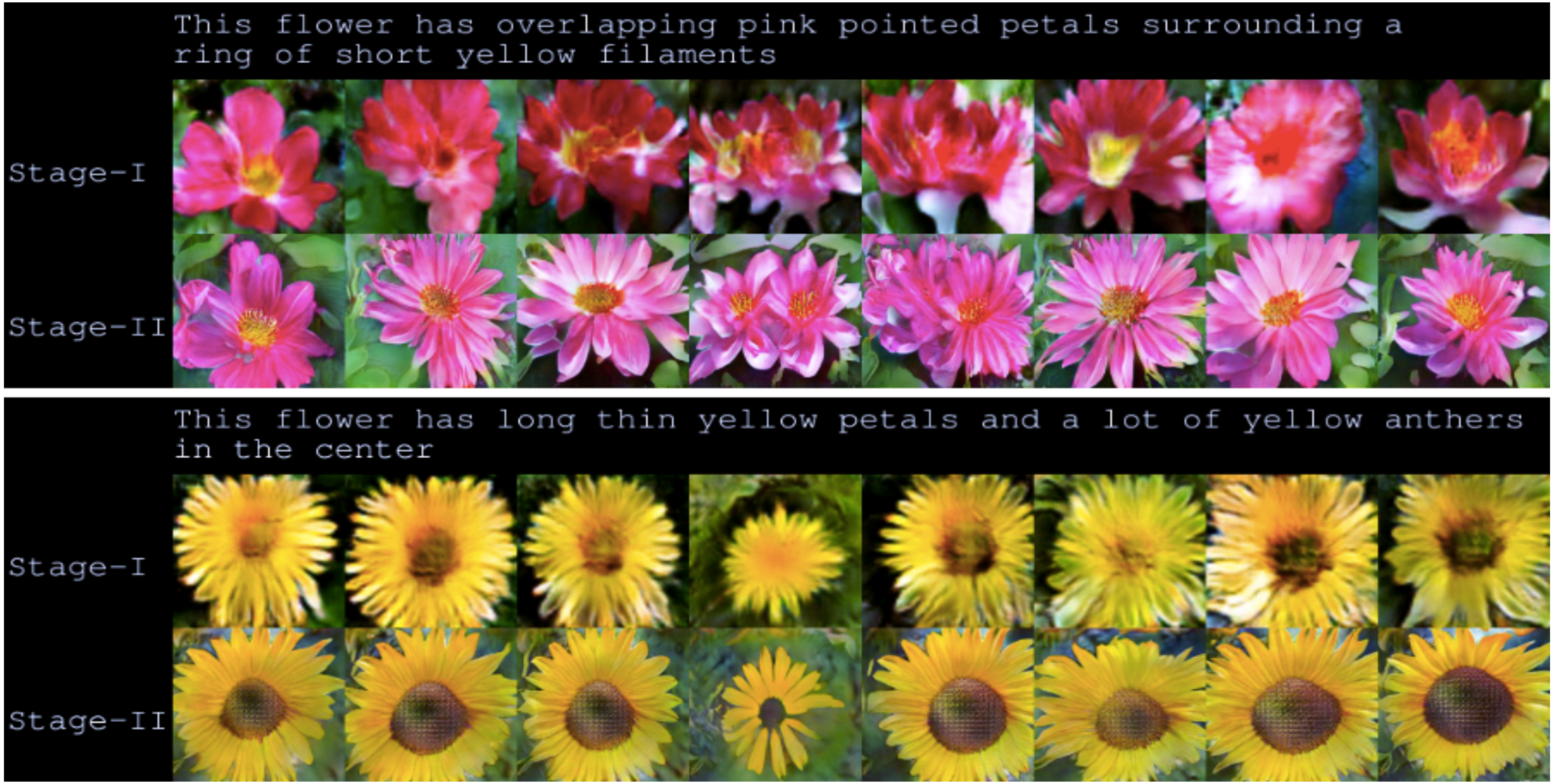
◆ Intro - Applications of CNN



◆ Intro - Applications of CNN

Real Time Object Detection

◆ Intro - Applications of CNN


This flower has overlapping pink pointed petals surrounding a ring of short yellow filaments
Stage-I
Stage-II
This flower has long thin yellow petals and a lot of yellow anthers in the center
Stage-I
Stage-II

◆ Intro - Applications of CNN

This flower has overlapping pink pointed petals surrounding a ring of short yellow filaments
Stage-I
Stage-II
This flower has long thin yellow petals and a lot of yellow anthers in the center
Stage-I
Stage-II

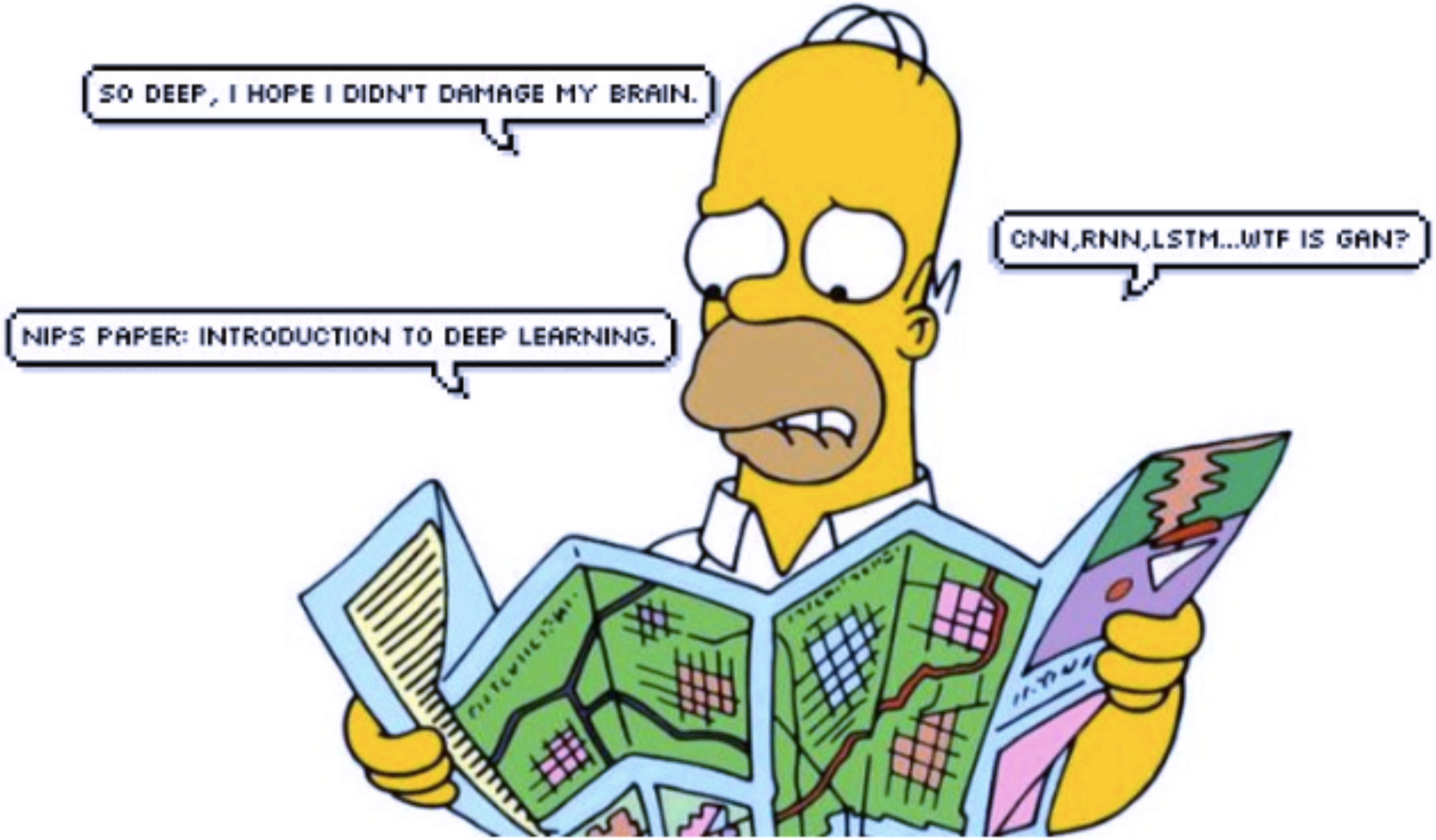
Image Caption Generation

◆ Intro - Applications of CNN

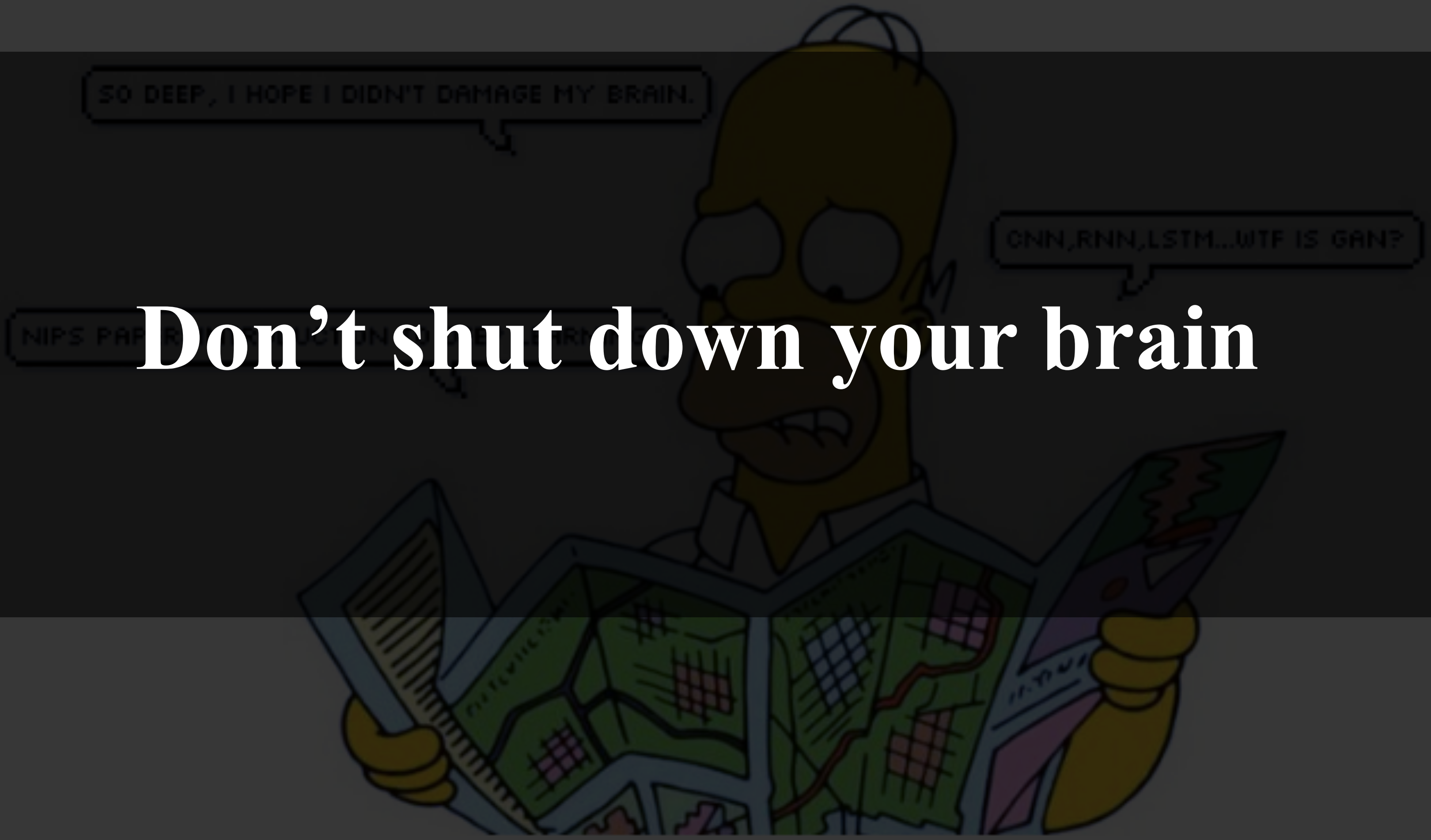
◆ Intro - Applications of CNN

Real Time Pose Estimation

◆ Intro - Applications of CNN


SO DEEP, I HOPE I DIDN'T DAMAGE MY BRAIN.
CNN,RNN,LSTM...WTF IS GAN?
NIPS PAPER: INTRODUCTION TO DEEP LEARNING.

◆ Intro - Applications of CNN


SO DEEP, I HOPE I DIDN'T DAMAGE MY BRAIN.
CNN,RNN,LSTM...WTF IS GAN?
NIPS PAP

Don’t shut down your brain

Contents

Unit 01 | Intro

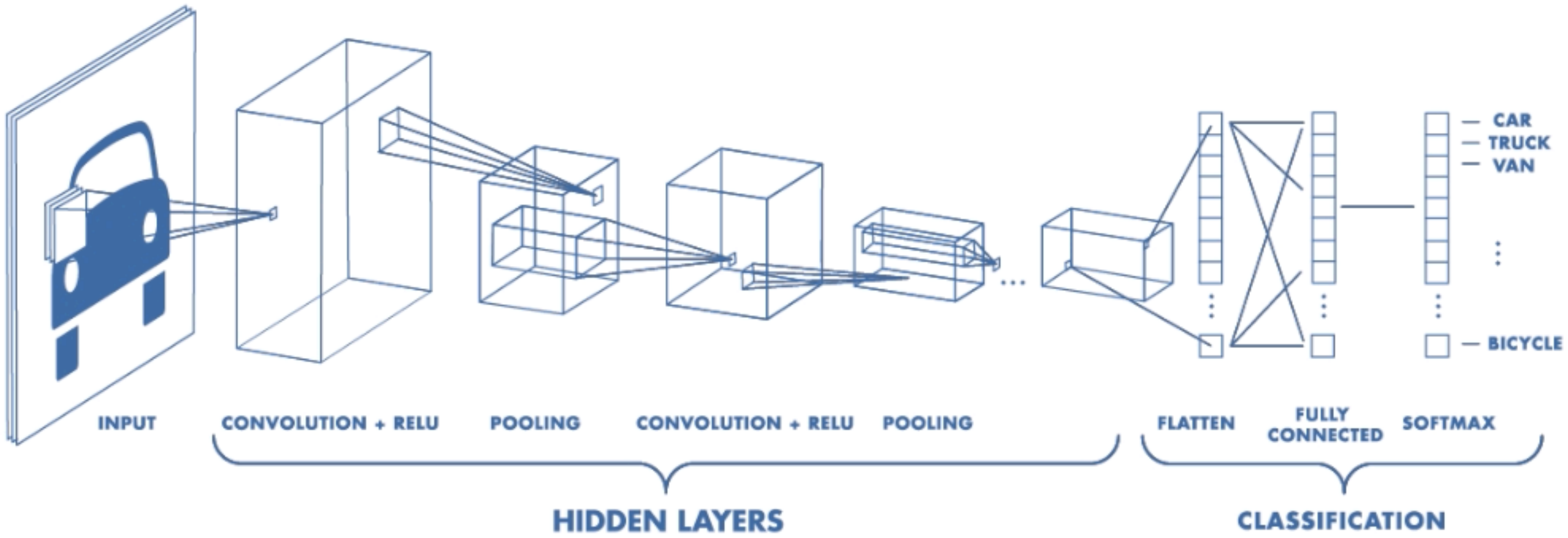
Unit 02 | Layers in CNN

Unit 03 | Convolution

Unit 04 | Sub-Sampling

Unit 05 | Summary

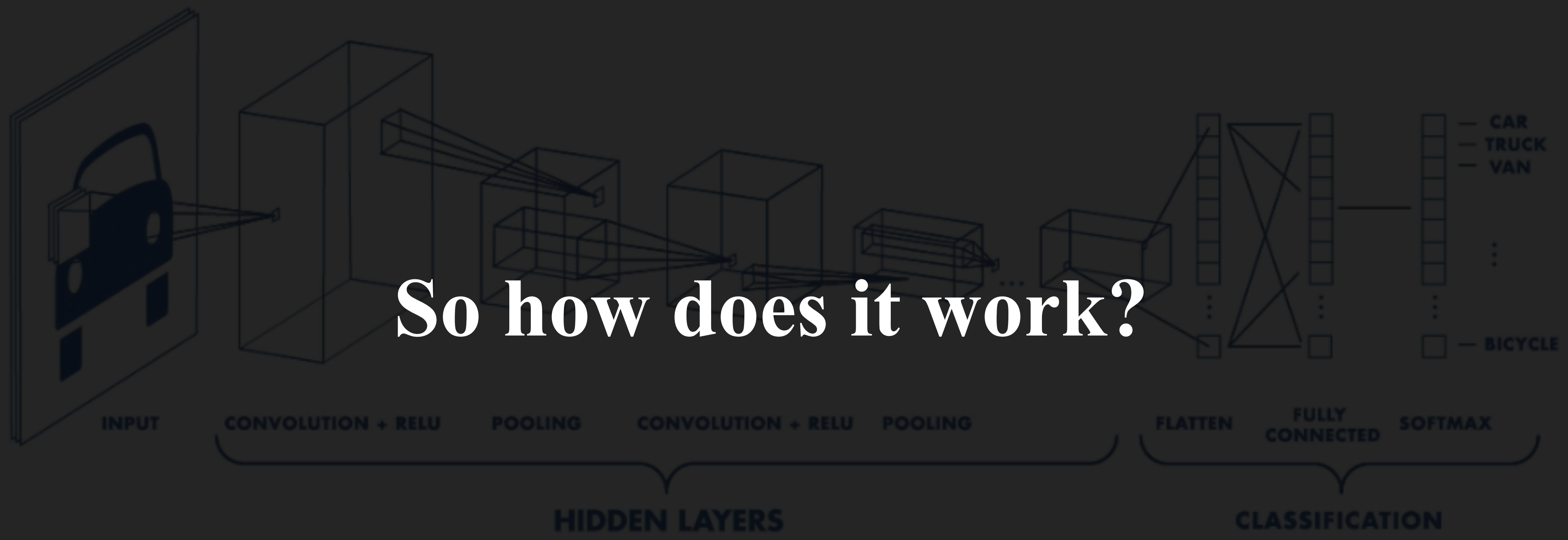
◆ Layers in CNN



- **This is Pretty much everything about the convolutional neural network**
- **Convolution + Subsampling + Full Connection**

UNIT 02. Layers in CNN

◆ Layers in CNN

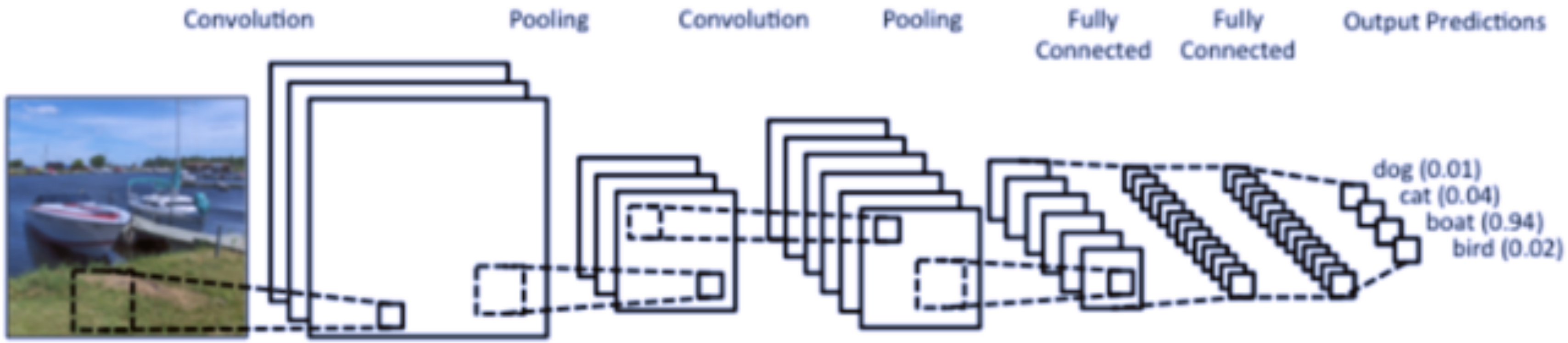


So how does it work?

- This is Pretty much everything about the convolutional neural network
- Convolution + Subsampling + Full Connection

UNIT 02. Layers in CNN

◆ Layers in CNN



- CNN = Convolutions followed by subsampling and fully connected layer
- Feature Extraction <- Convolution + subsampling layers
- Classifier <- Fully connected layer

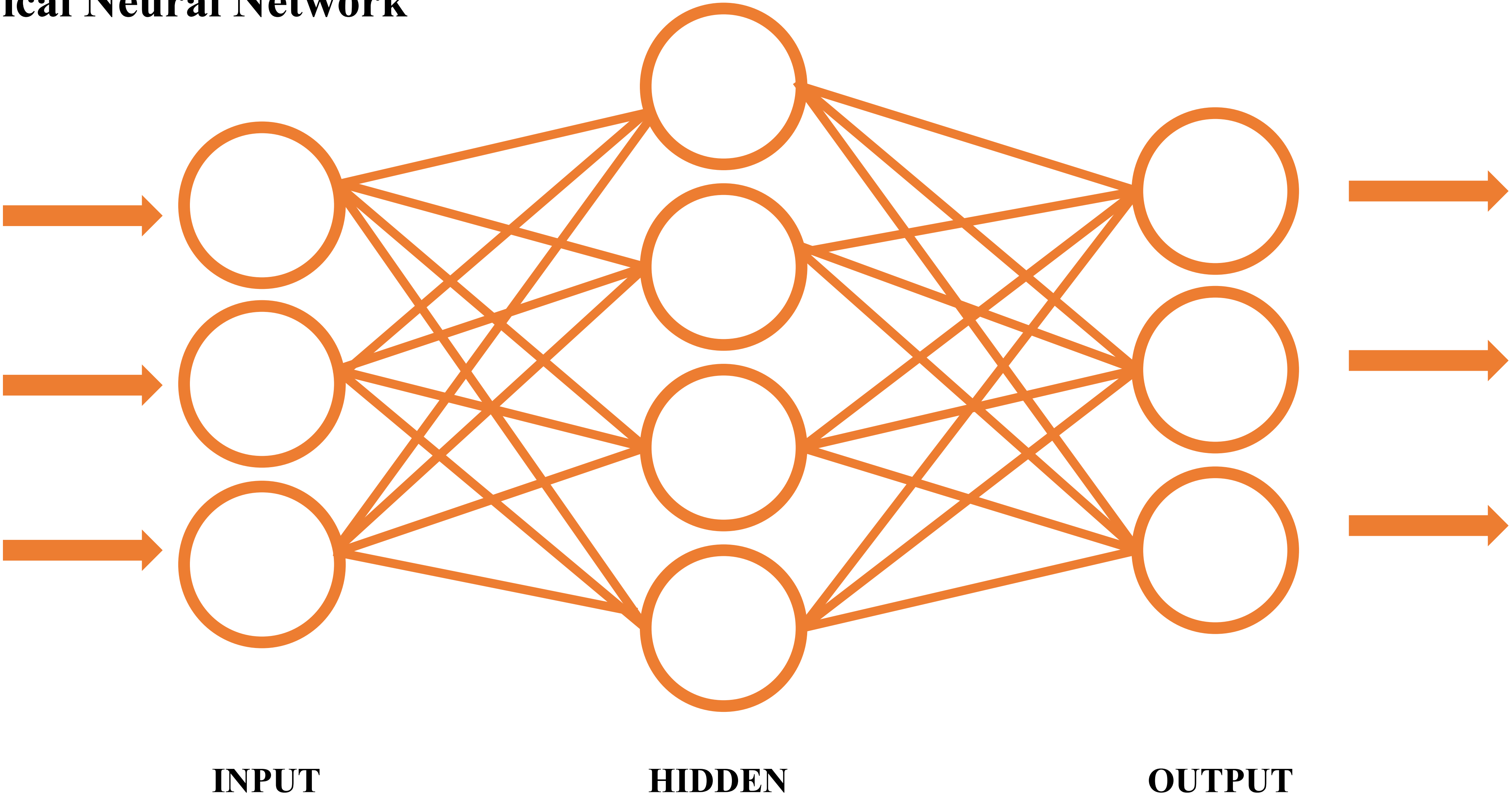
◆ Layers in CNN

Why is so powerful?

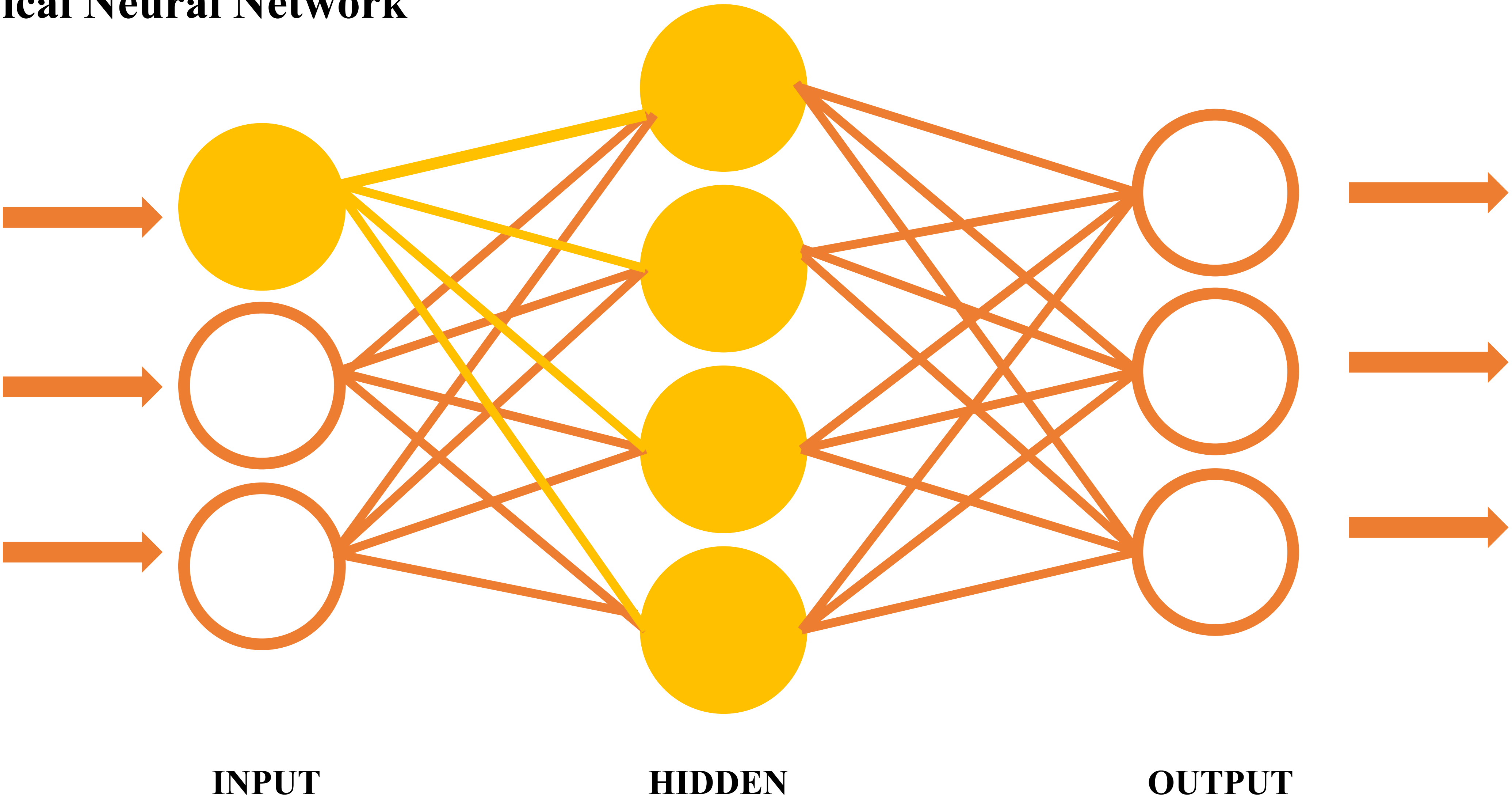
- **Local connectivity(**receptive field**)**
- **Shared Weights and Biases**
- **Compositionality**

UNIT 02. Layers in CNN

◆ Typical Neural Network

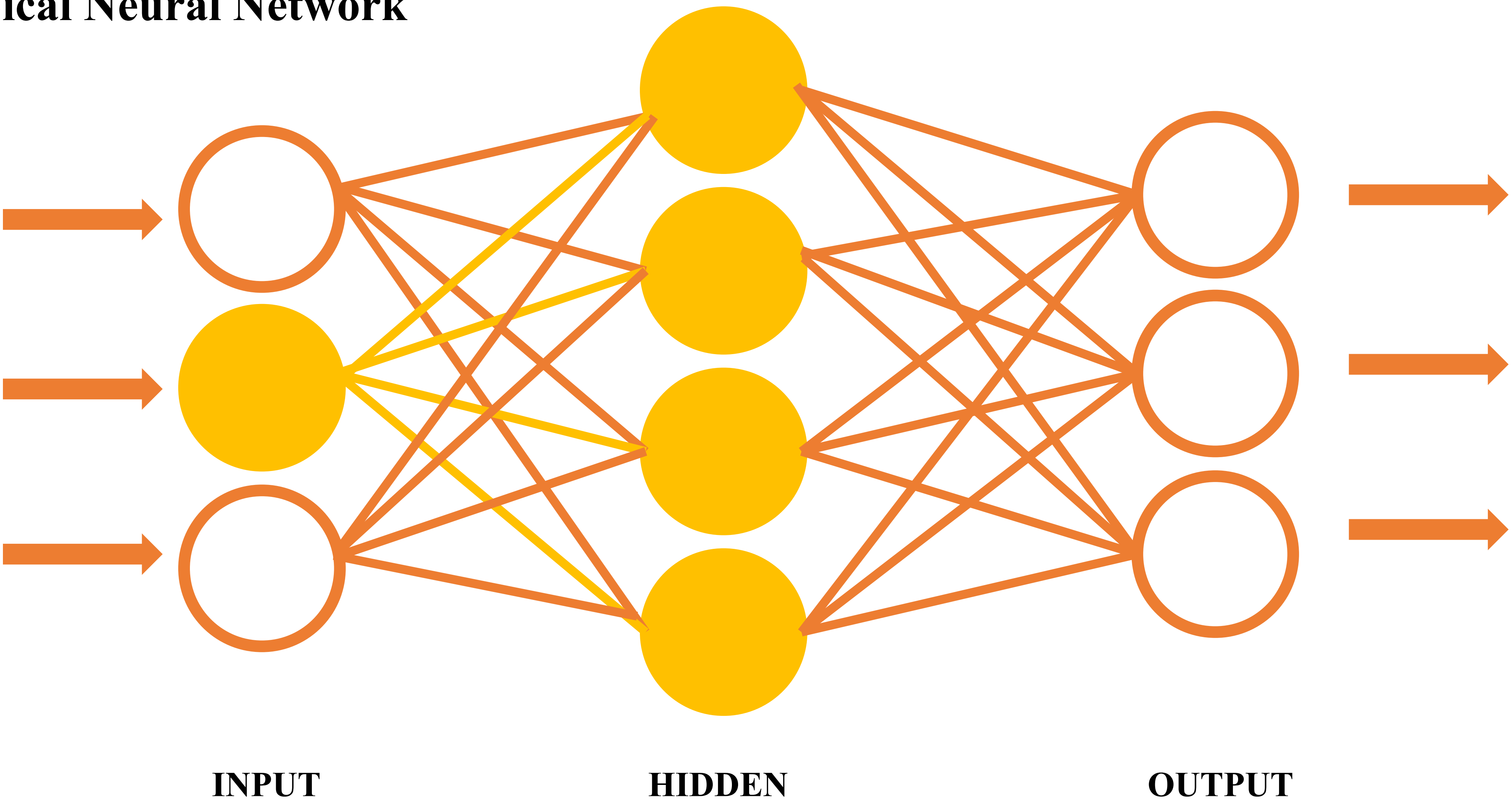


◆ Typical Neural Network



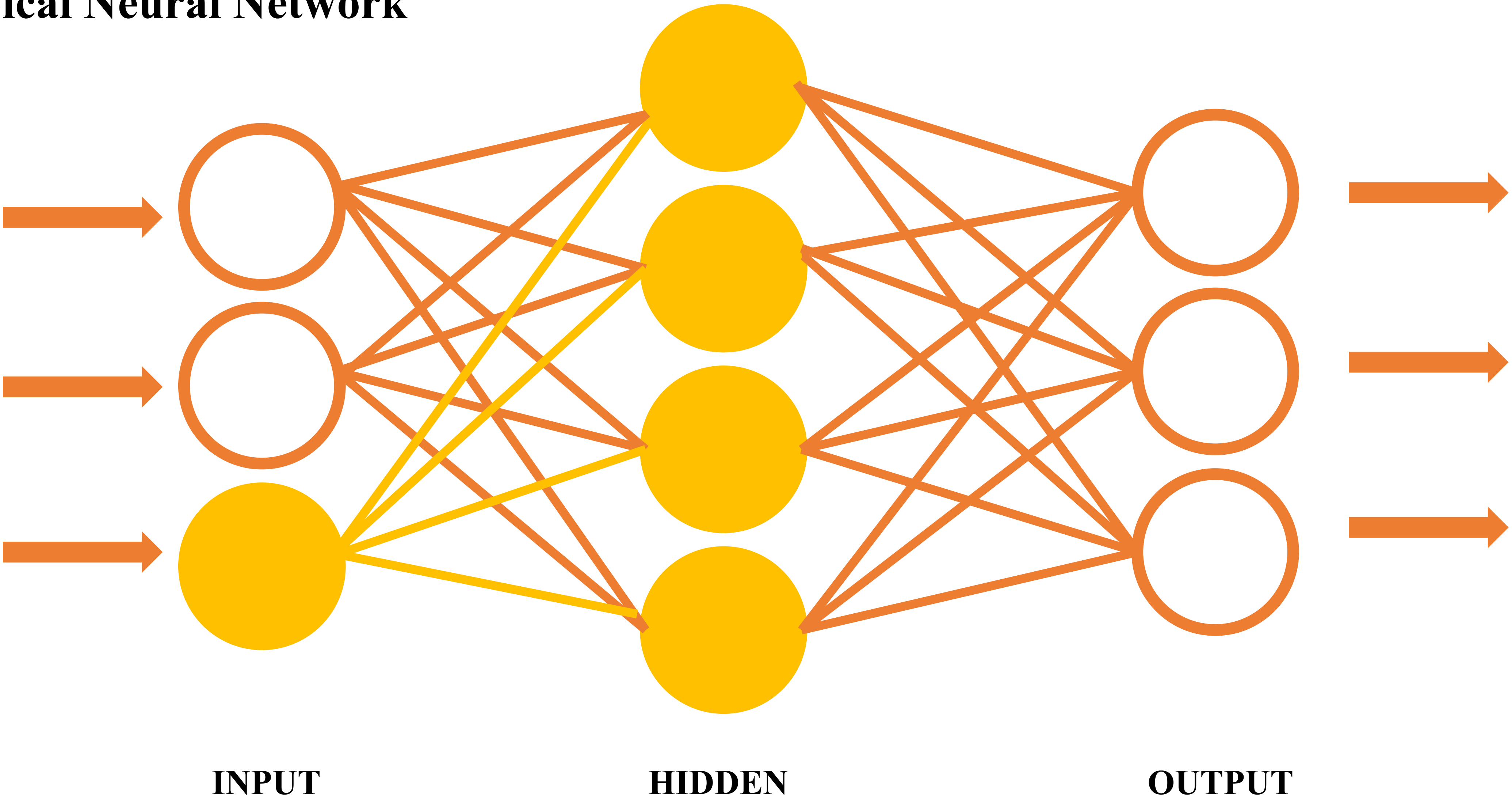
UNIT 02. Layers in CNN

◆ Typical Neural Network

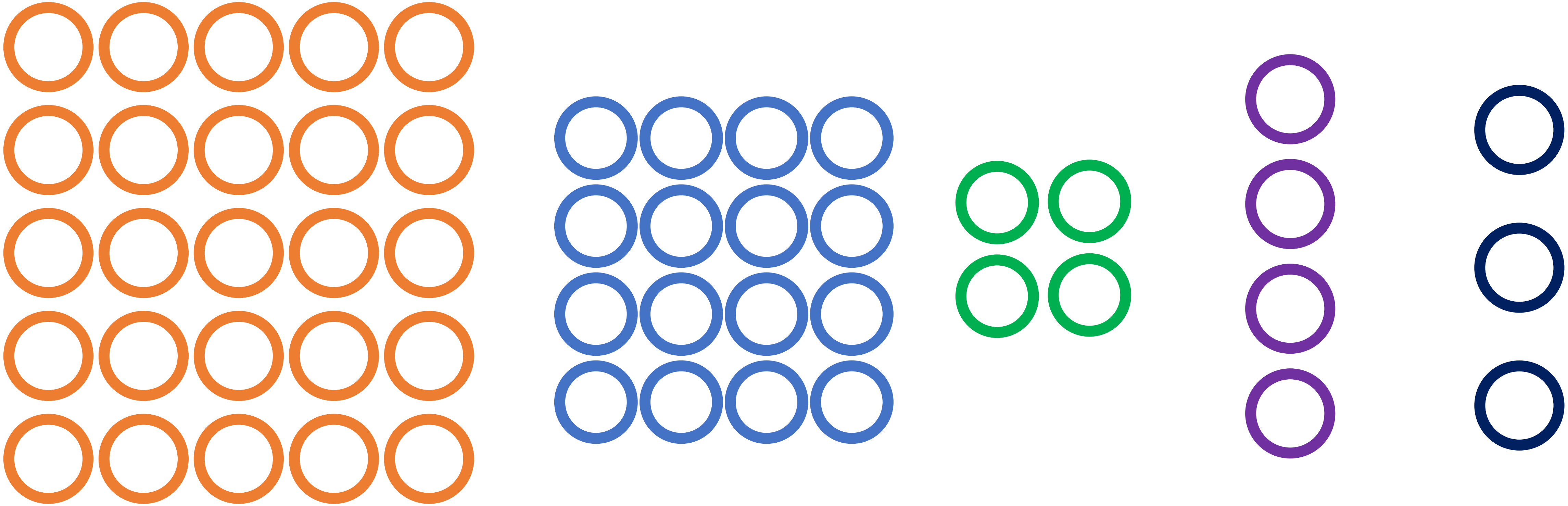


UNIT 02. Layers in CNN

◆ Typical Neural Network



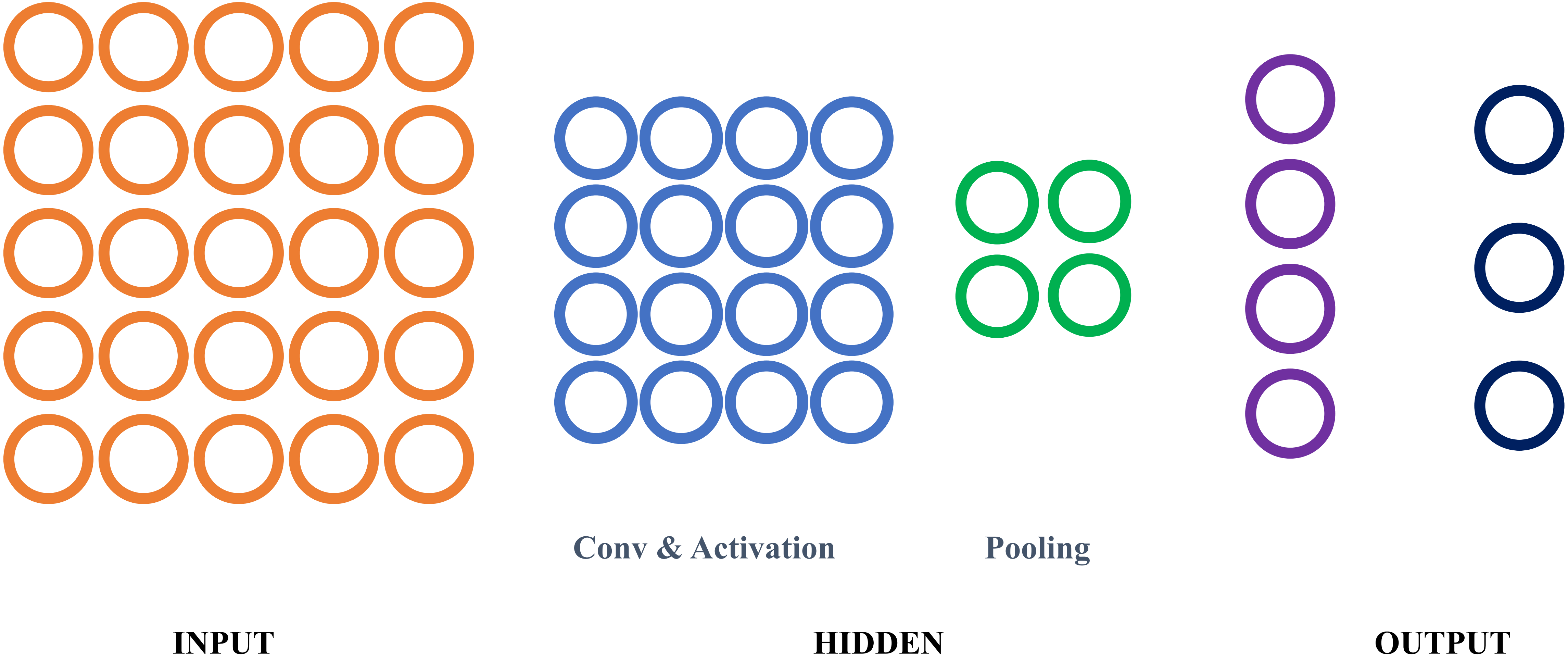
◆ Convolutional Neural Network



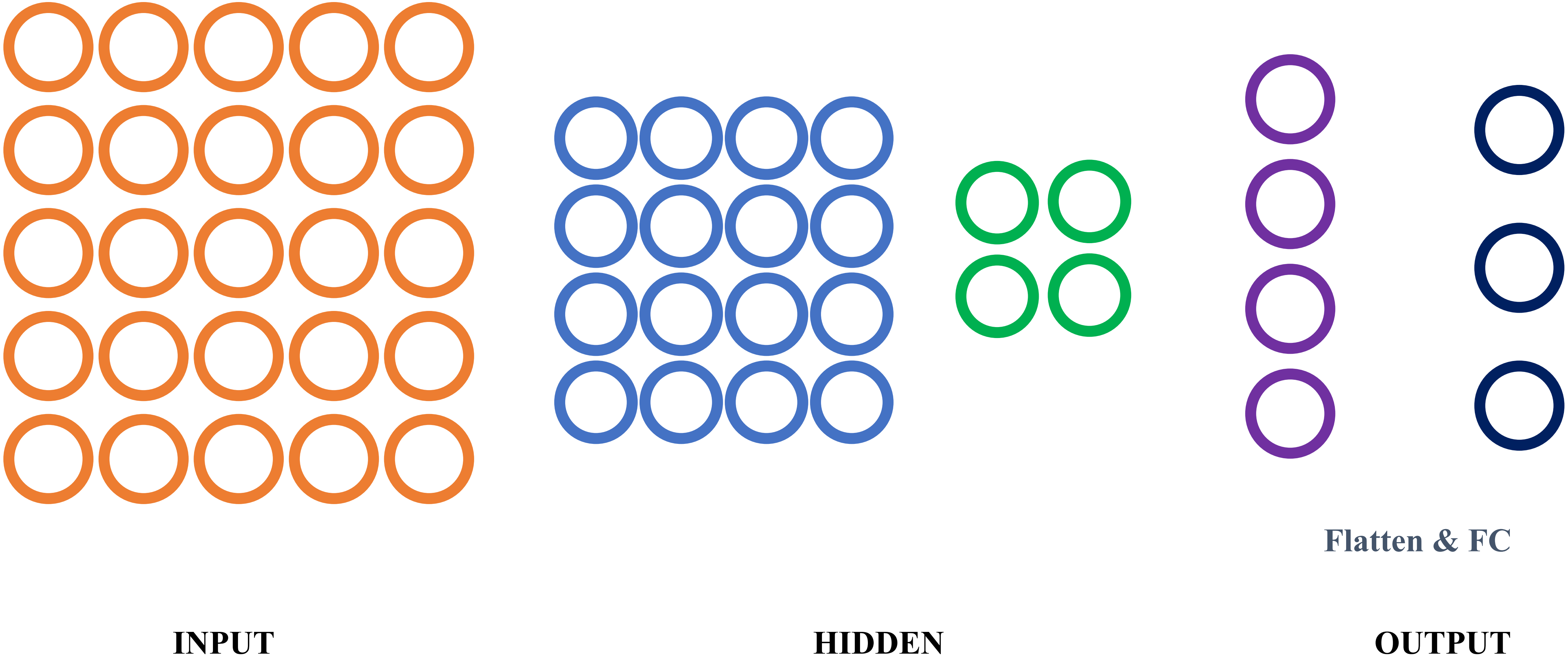
INPUT HIDDEN OUTPUT

UNIT 02. Layers in CNN

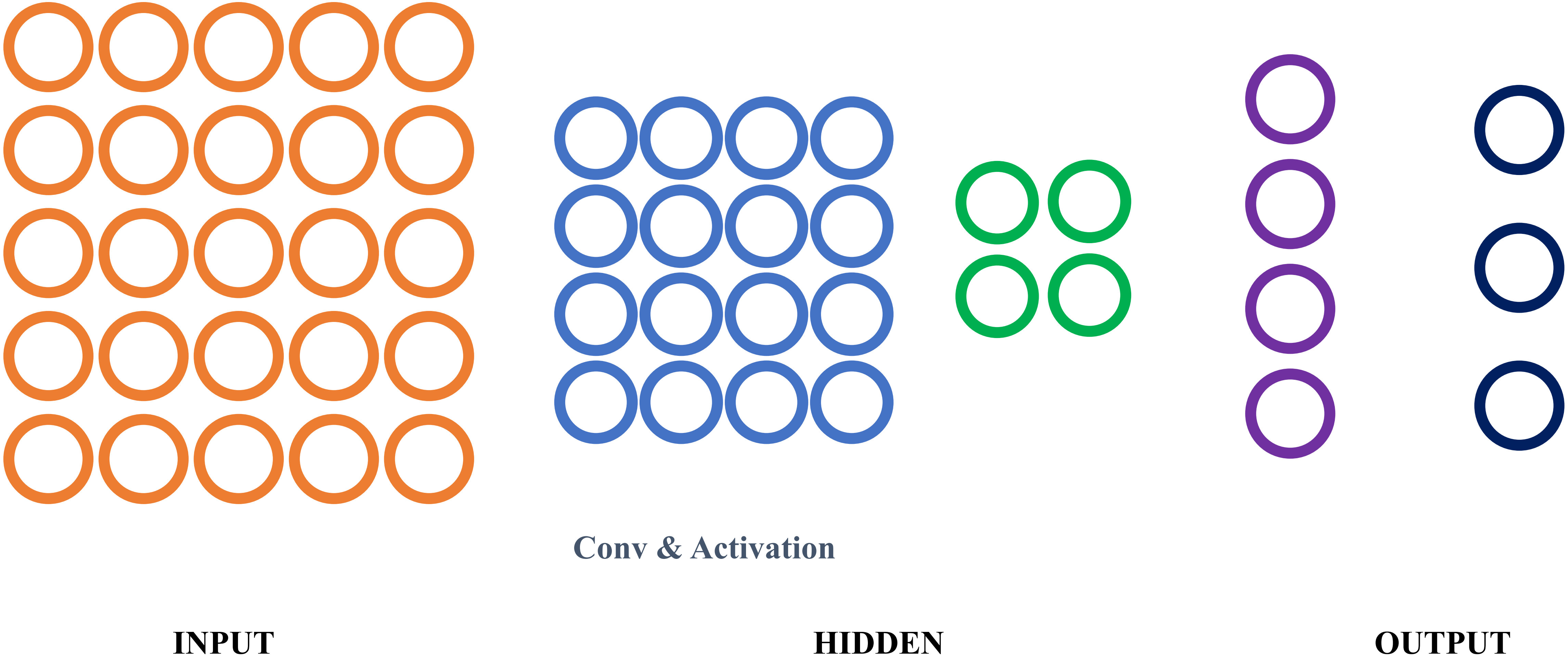
◆ Convolutional Neural Network



◆ Convolutional Neural Network

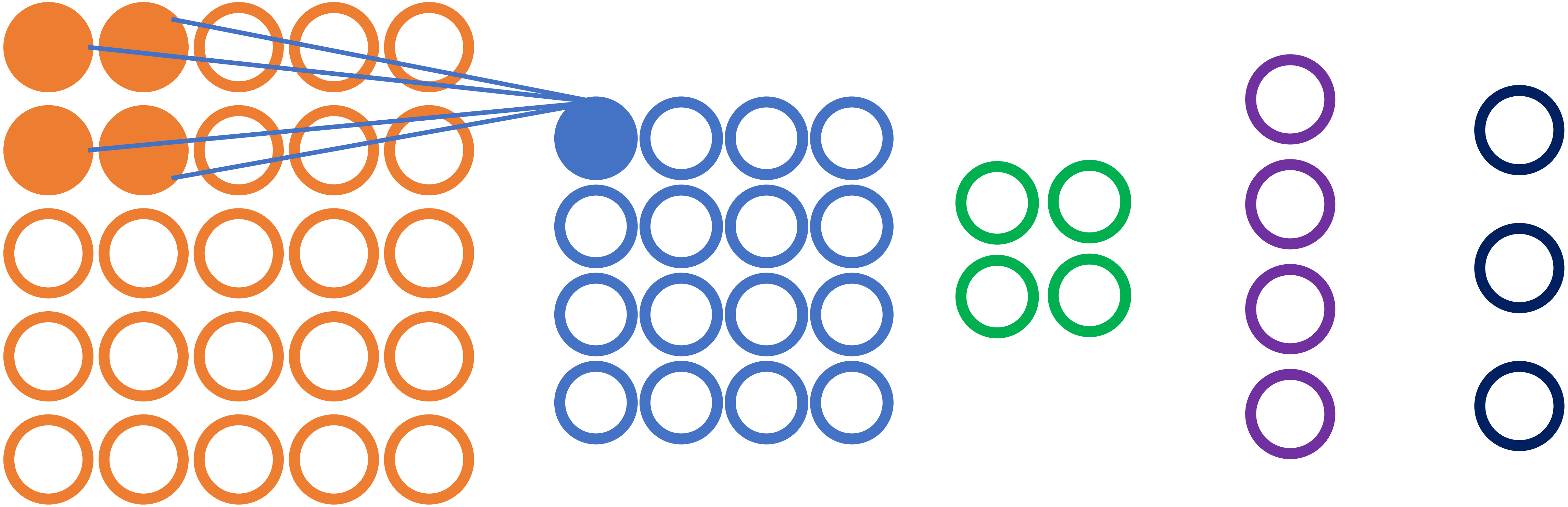


◆ Convolutional Neural Network



UNIT 02. Layers in CNN

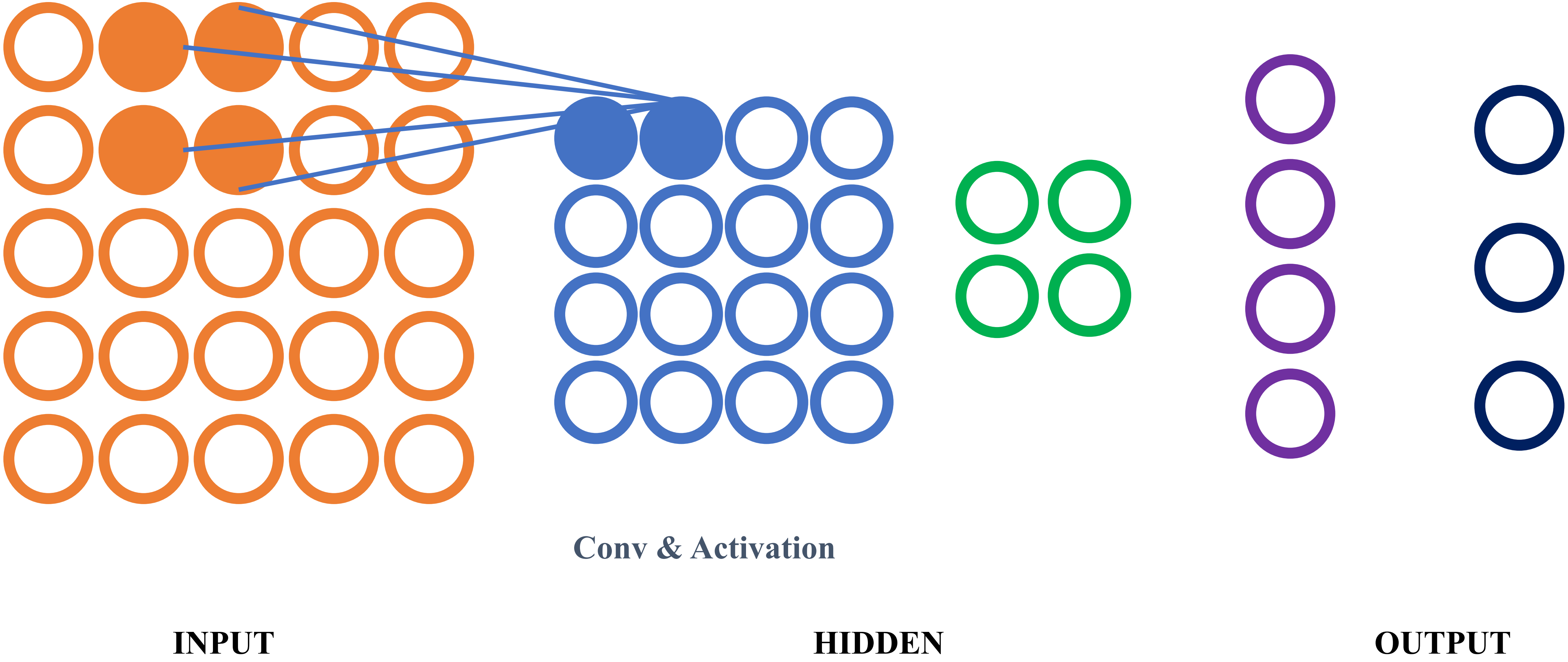
◆ Convolutional Neural Network



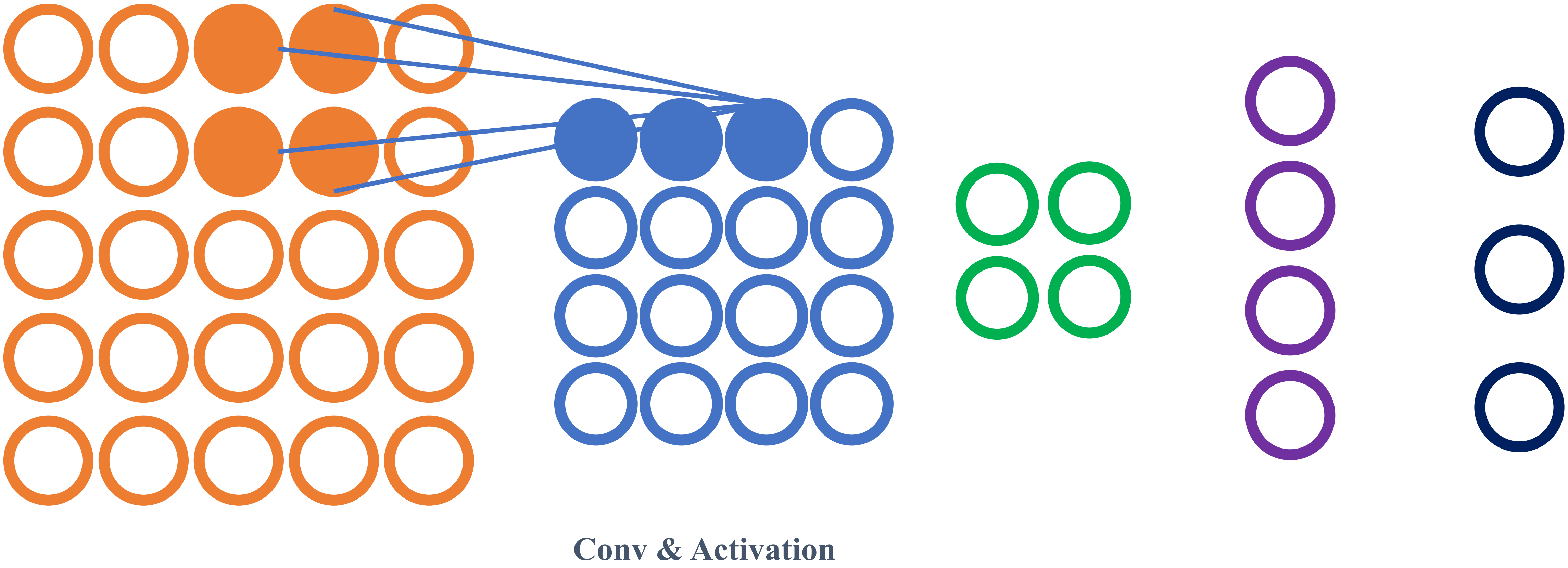
Conv & Activation

INPUT HIDDEN OUTPUT

◆ Convolutional Neural Network



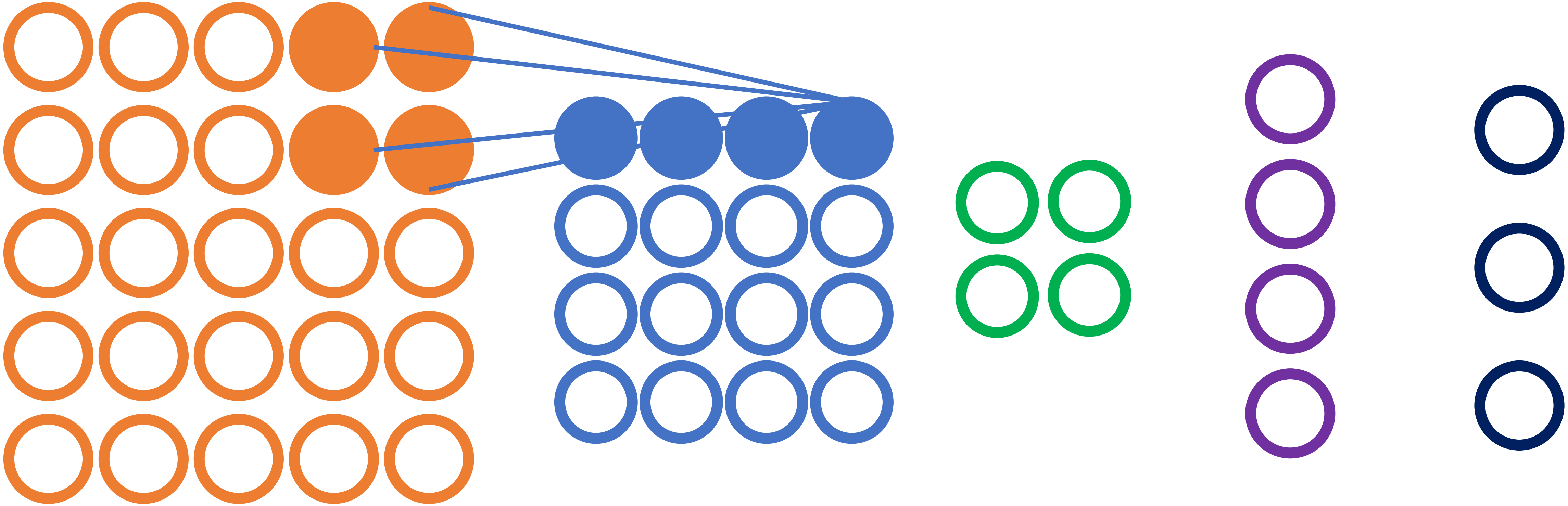
◆ Convolutional Neural Network



INPUT HIDDEN OUTPUT

UNIT 02. Layers in CNN

◆ Convolutional Neural Network

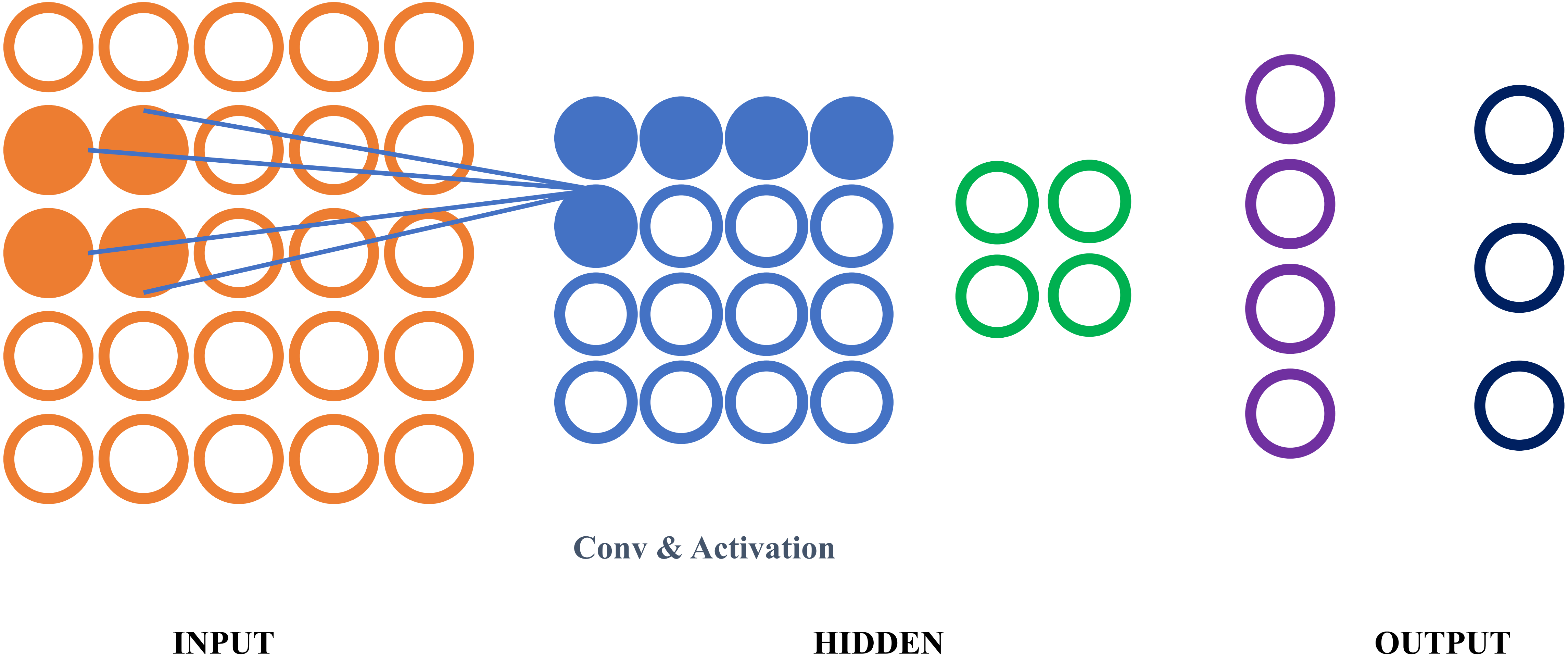


Conv & Activation

INPUT HIDDEN OUTPUT

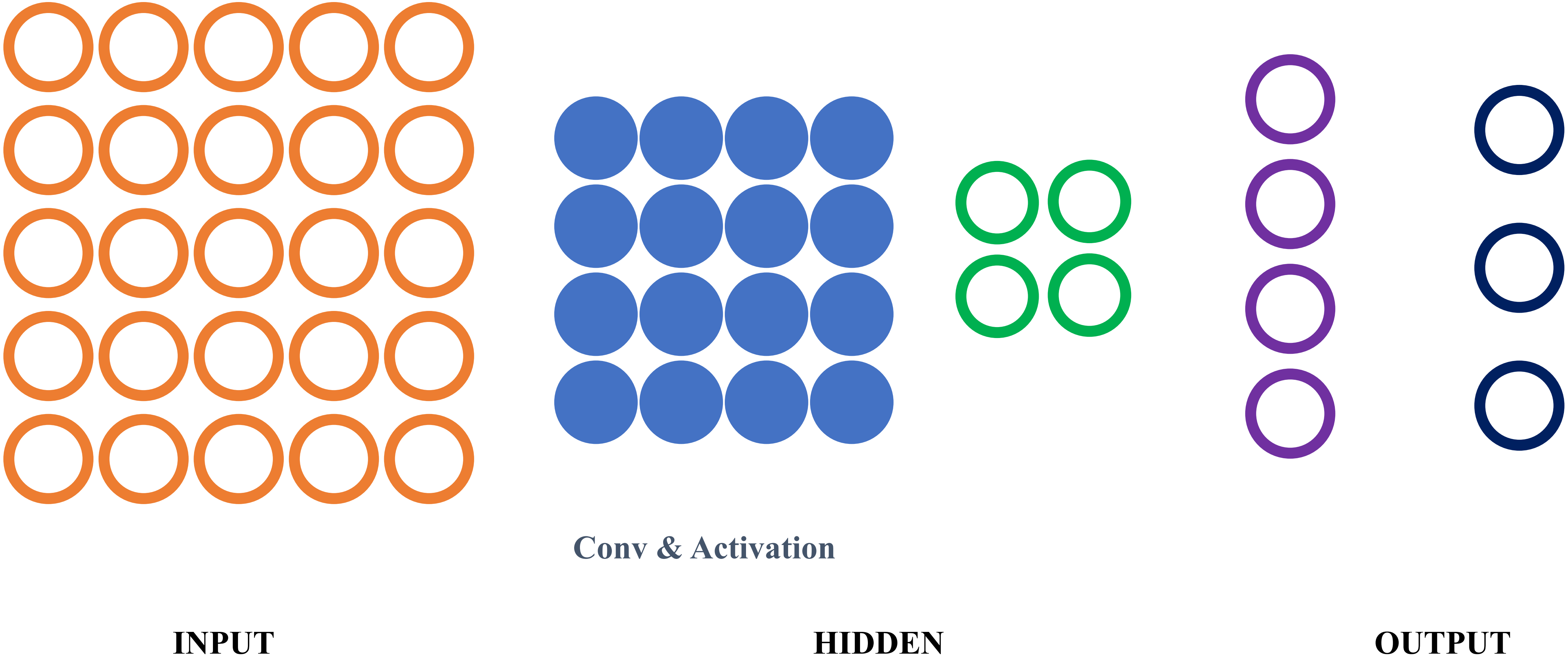
UNIT 02. Layers in CNN

◆ Convolutional Neural Network



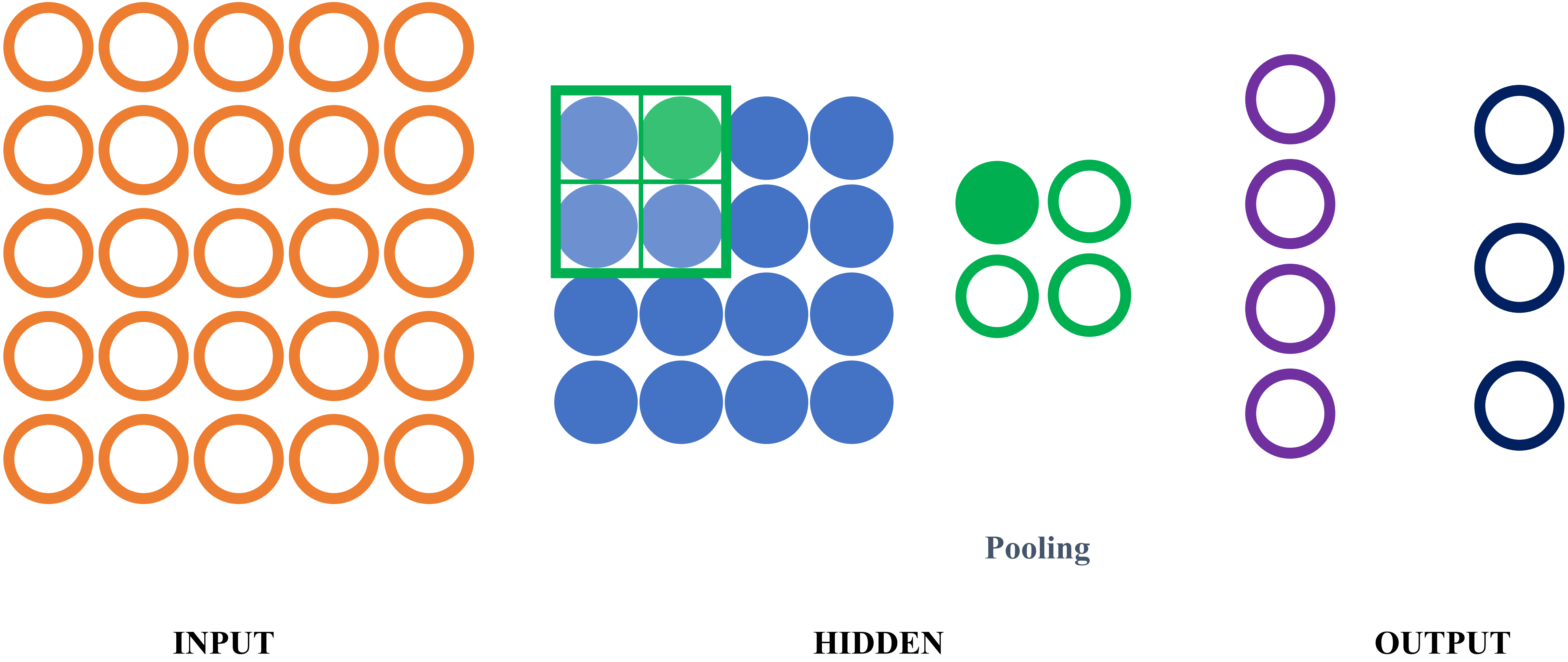
UNIT 02. Layers in CNN

◆ Convolutional Neural Network

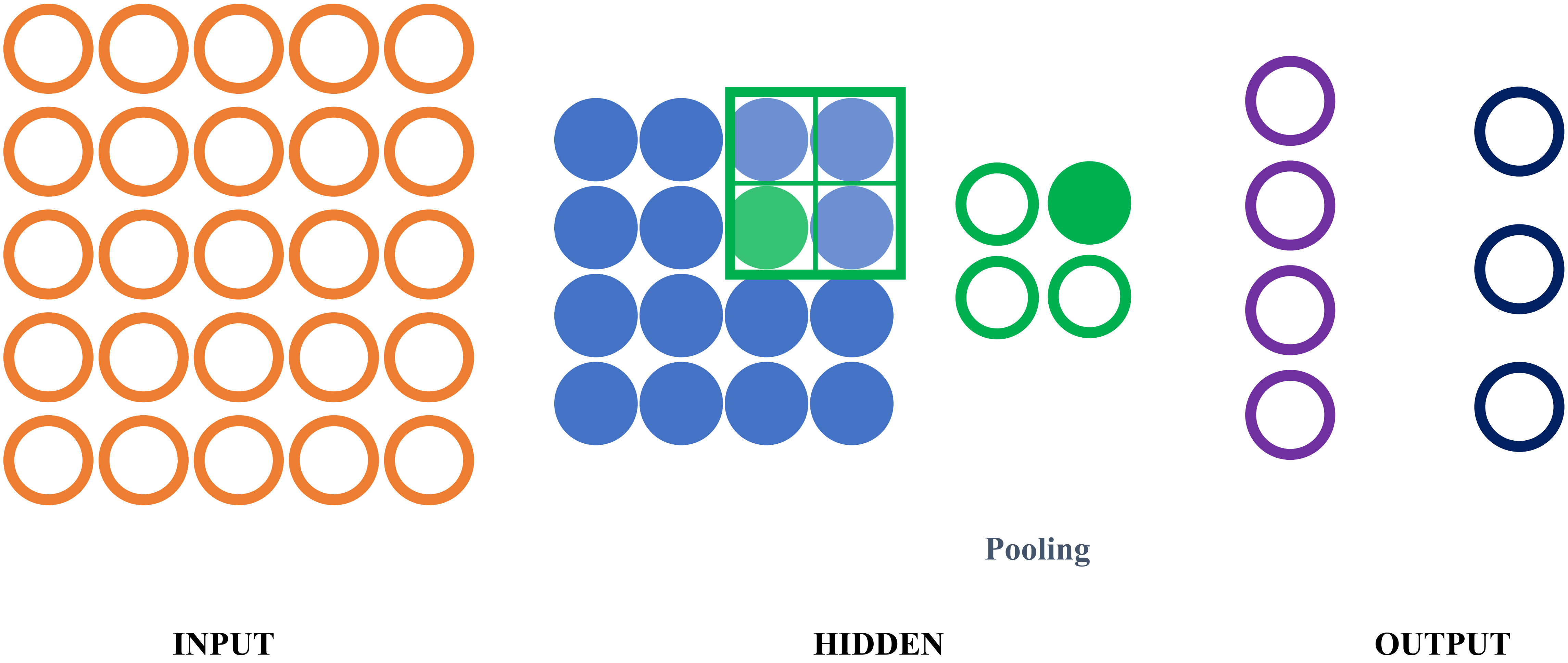


UNIT 02. Layers in CNN

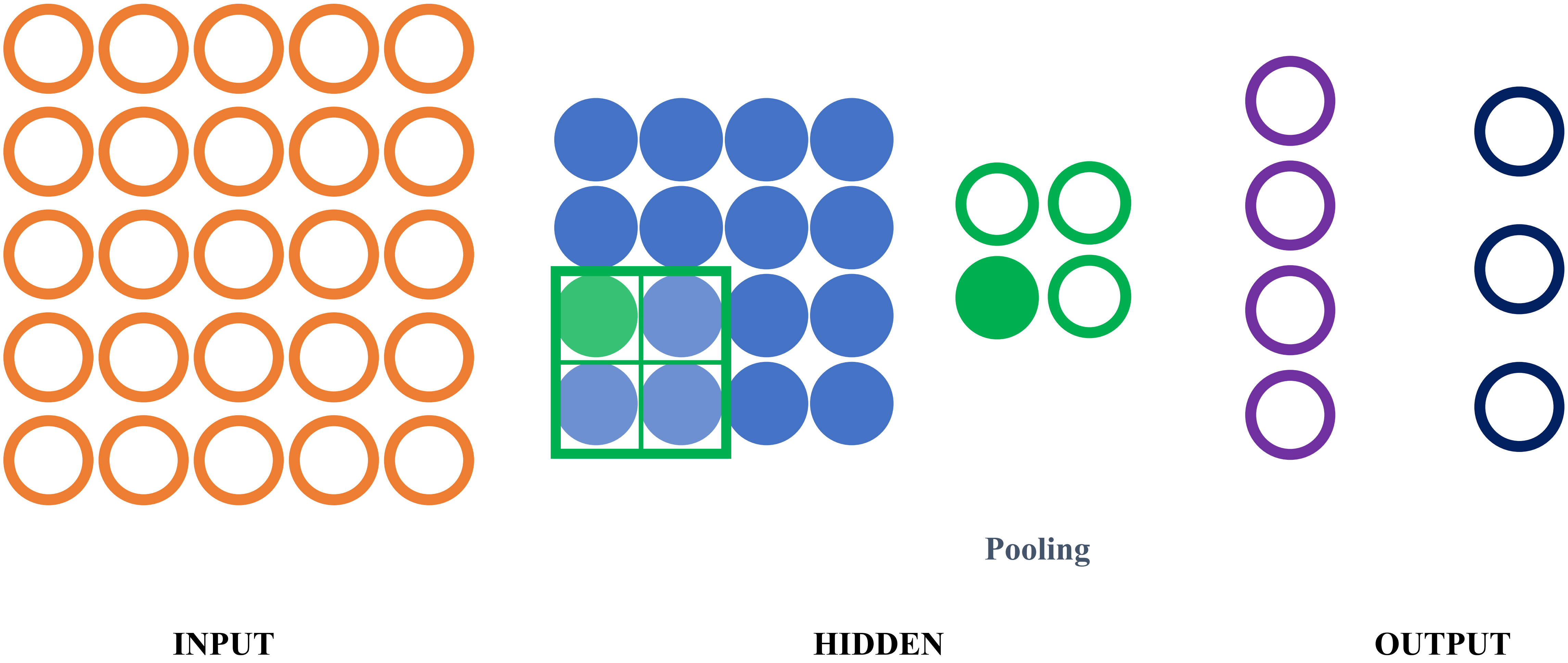
◆ Convolutional Neural Network



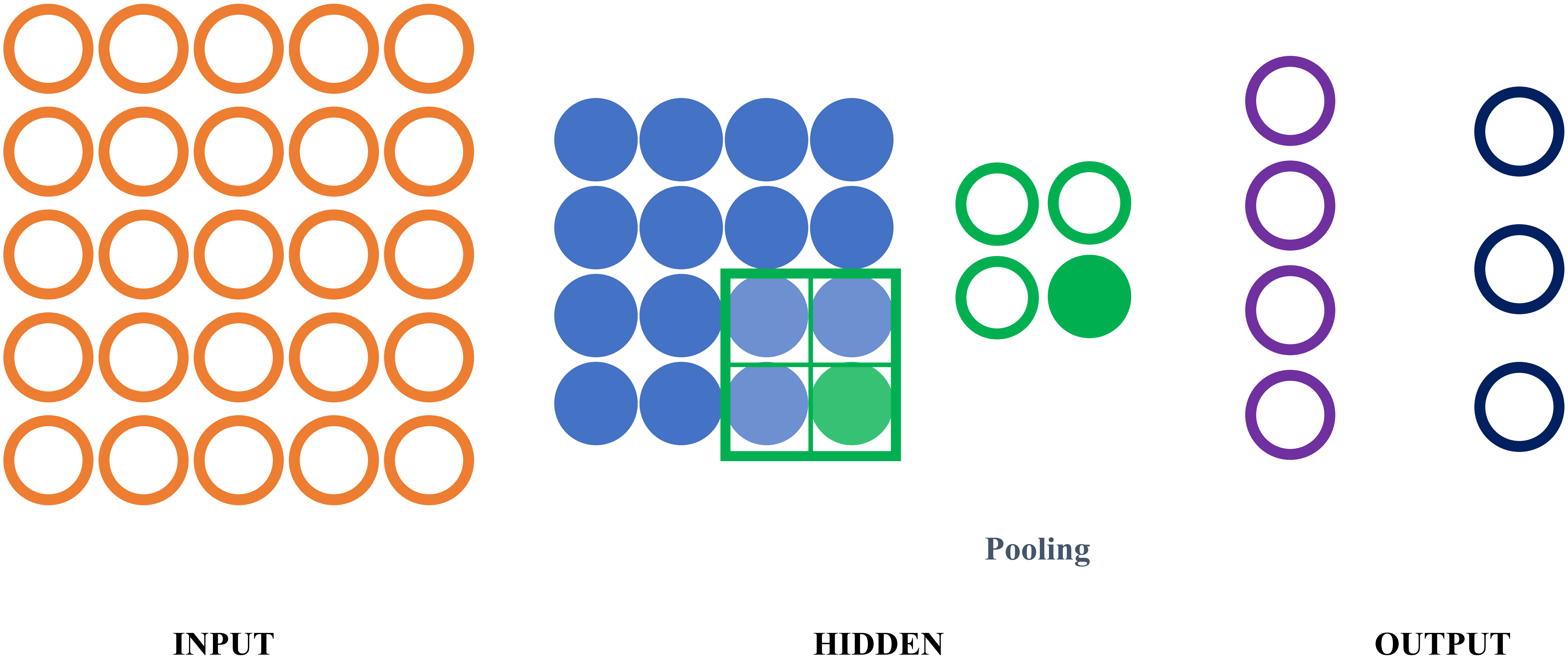
◆ Convolutional Neural Network



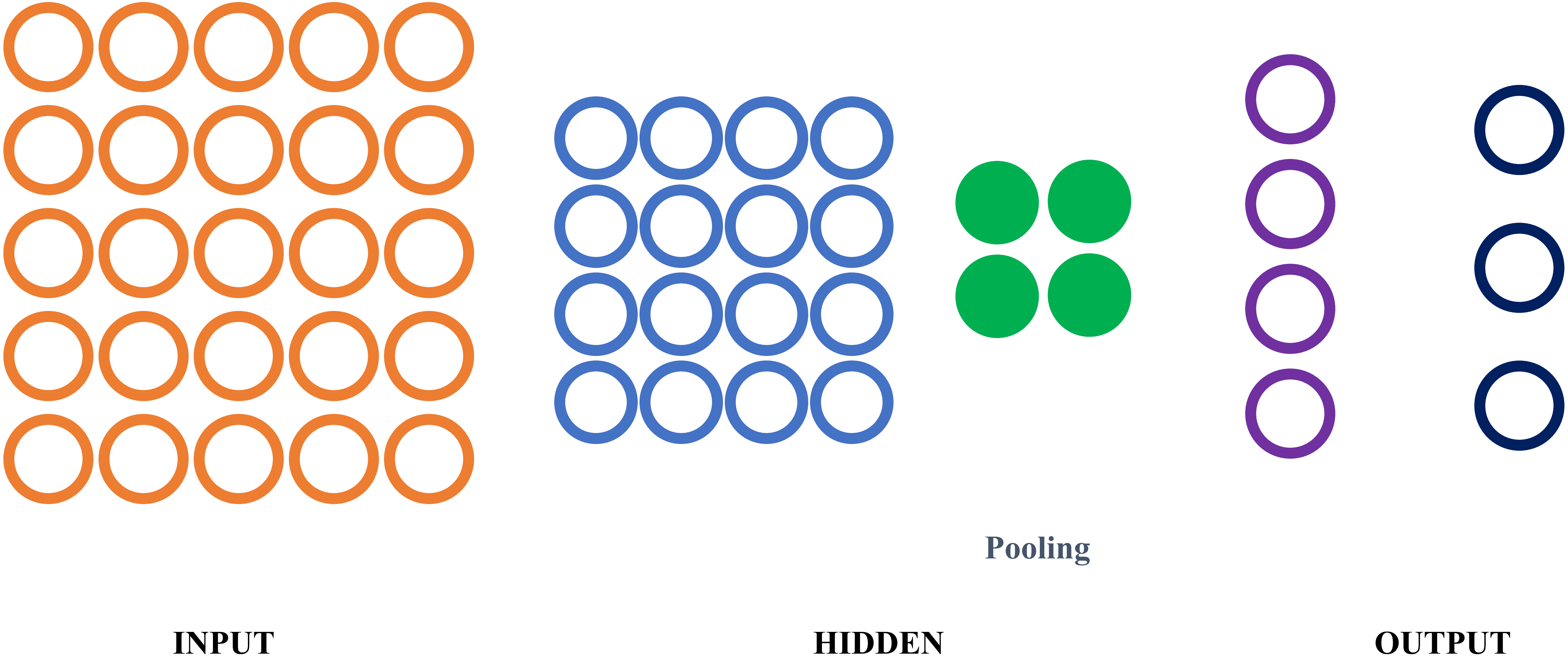
◆ Convolutional Neural Network



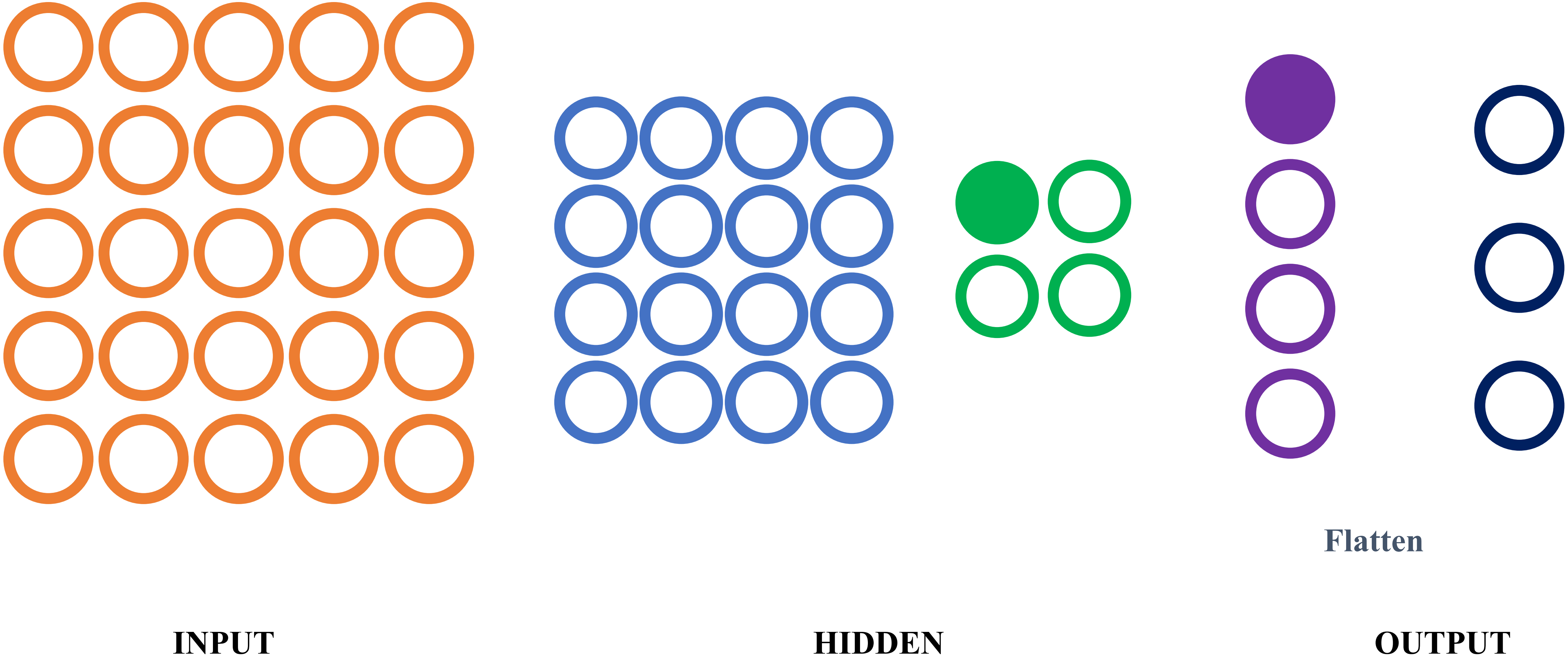
◆ Convolutional Neural Network



◆ Convolutional Neural Network

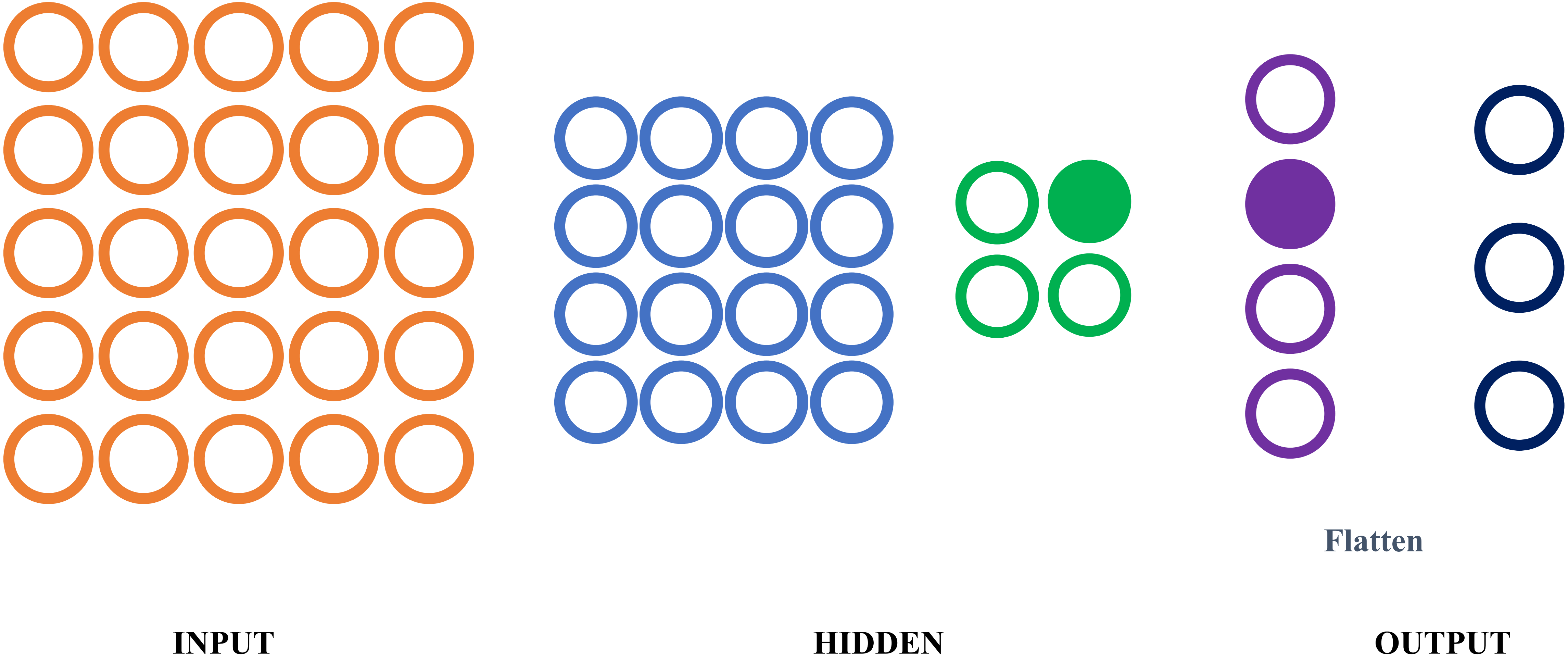


◆ Convolutional Neural Network



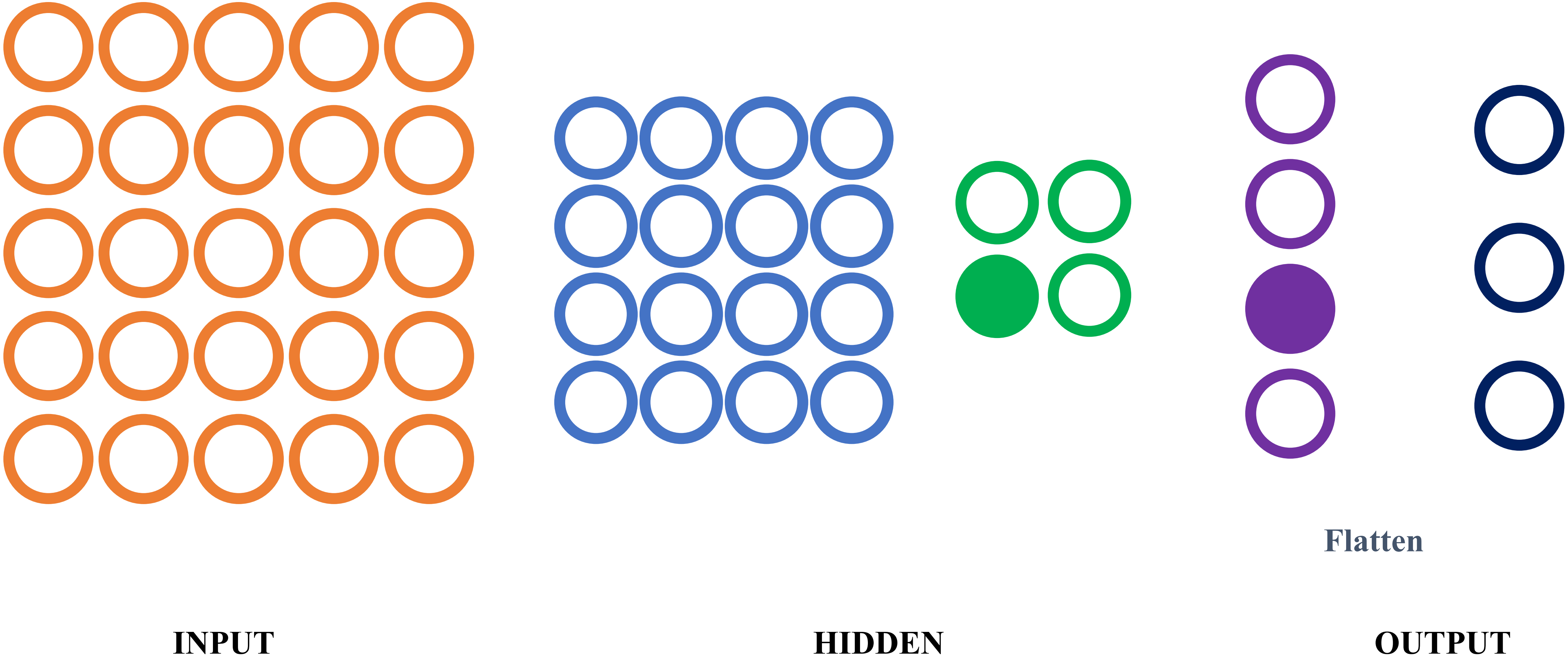
UNIT 02. Layers in CNN

◆ Convolutional Neural Network

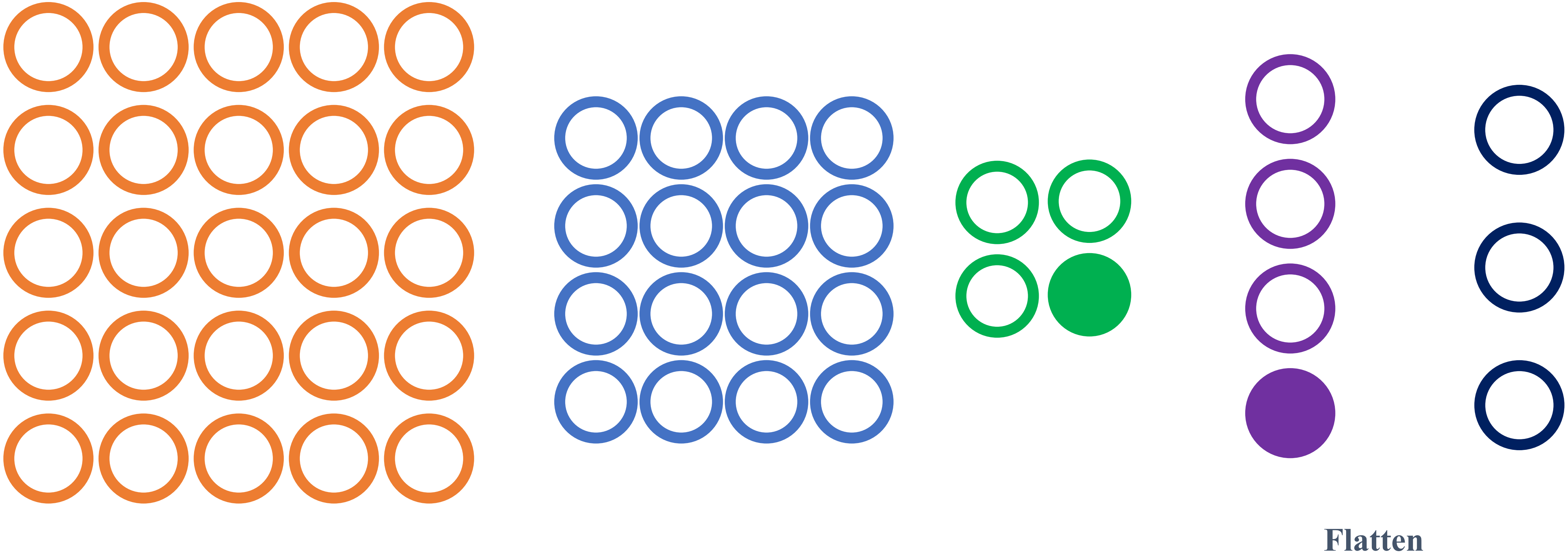


UNIT 02. Layers in CNN

◆ Convolutional Neural Network

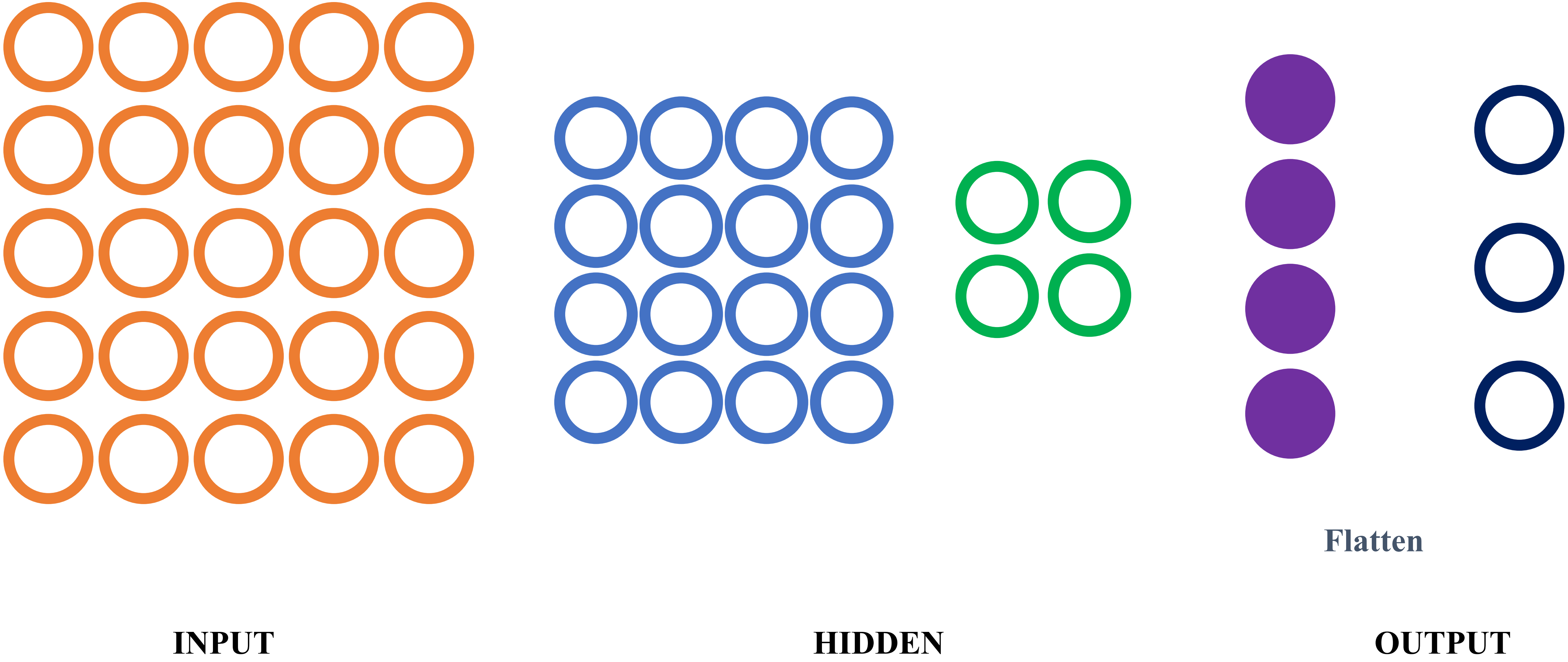


◆ Convolutional Neural Network



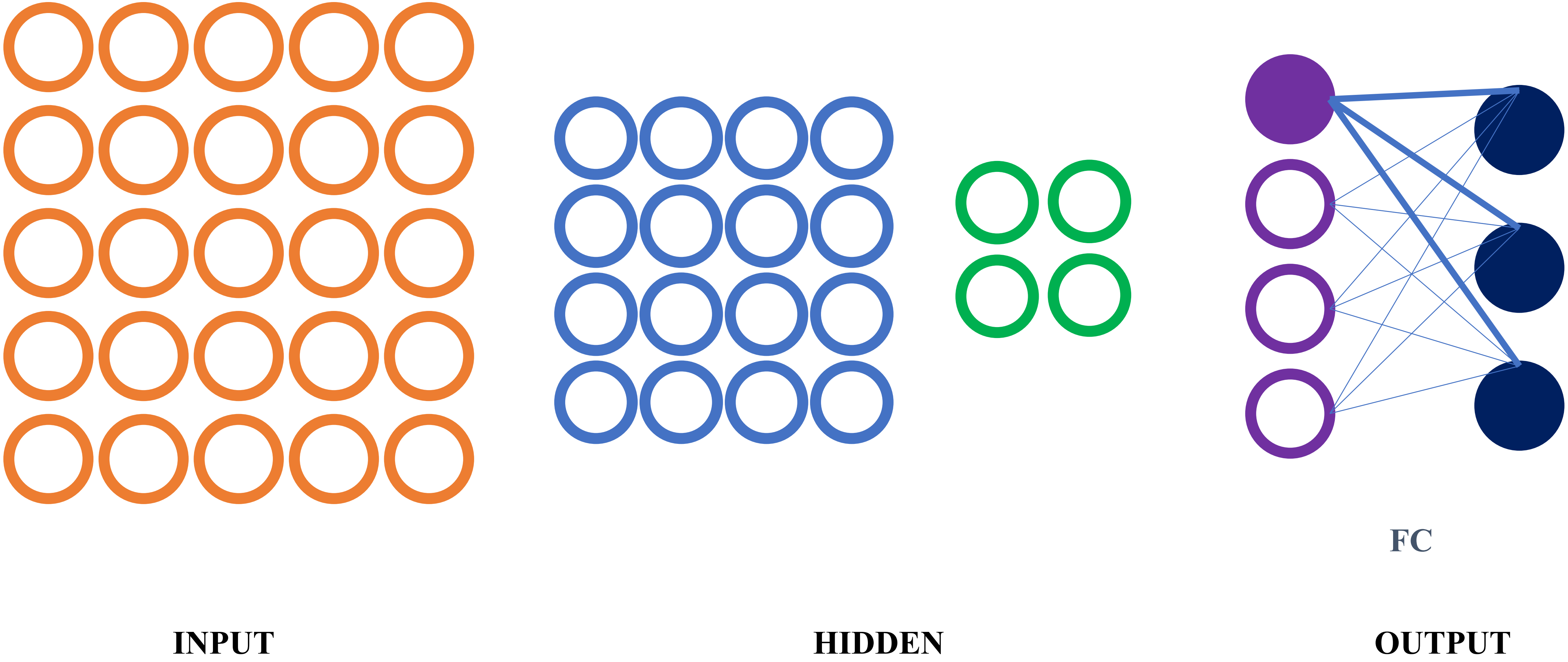
INPUT HIDDEN OUTPUT

◆ Convolutional Neural Network

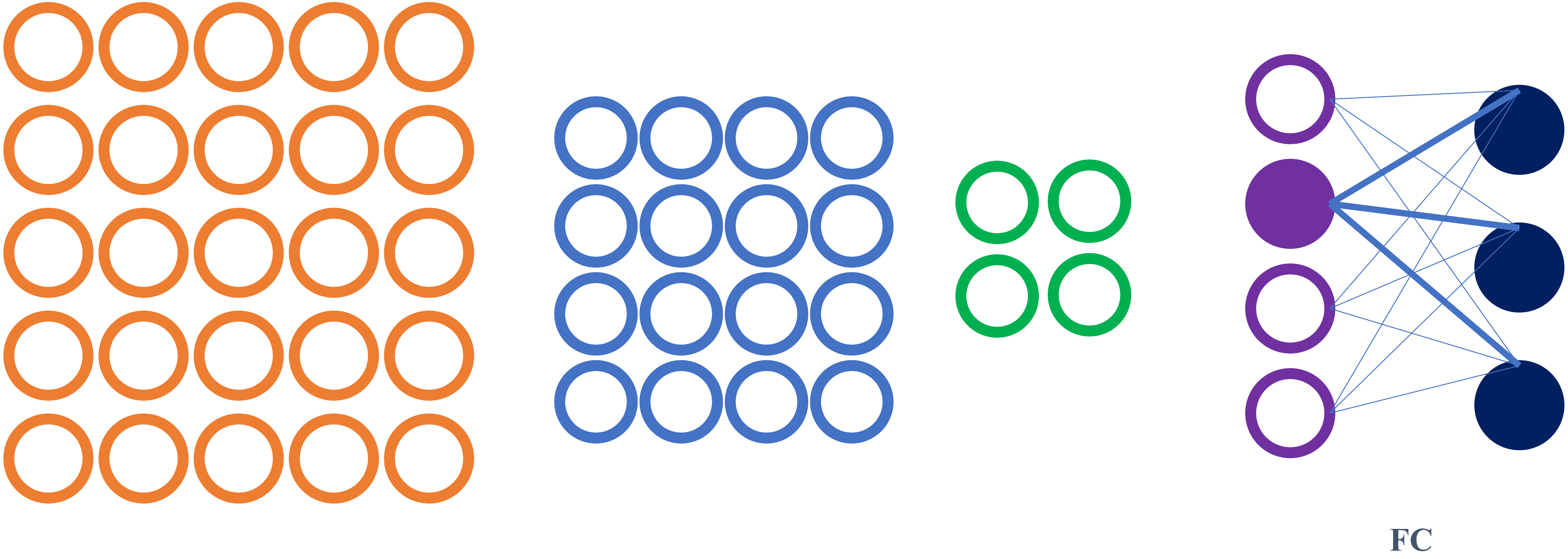


UNIT 02. Layers in CNN

◆ Convolutional Neural Network



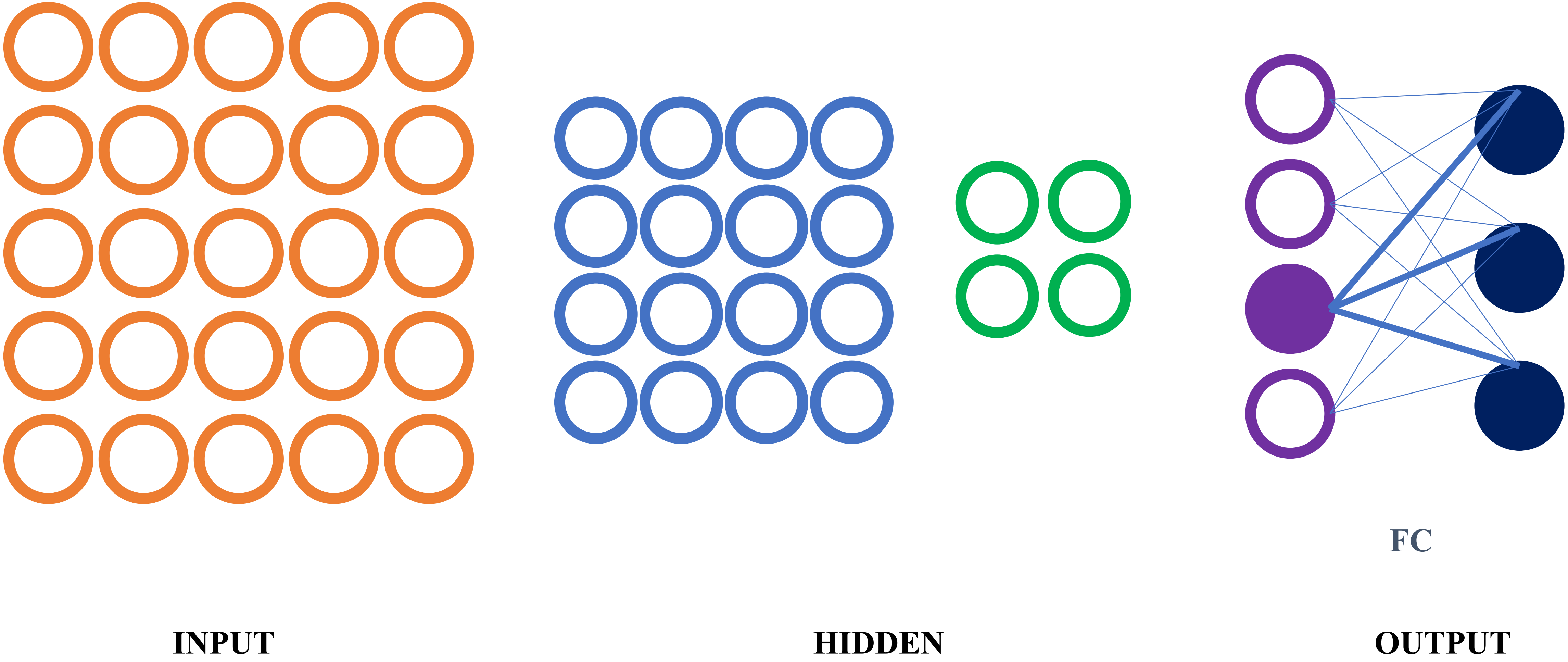
◆ Convolutional Neural Network



FC

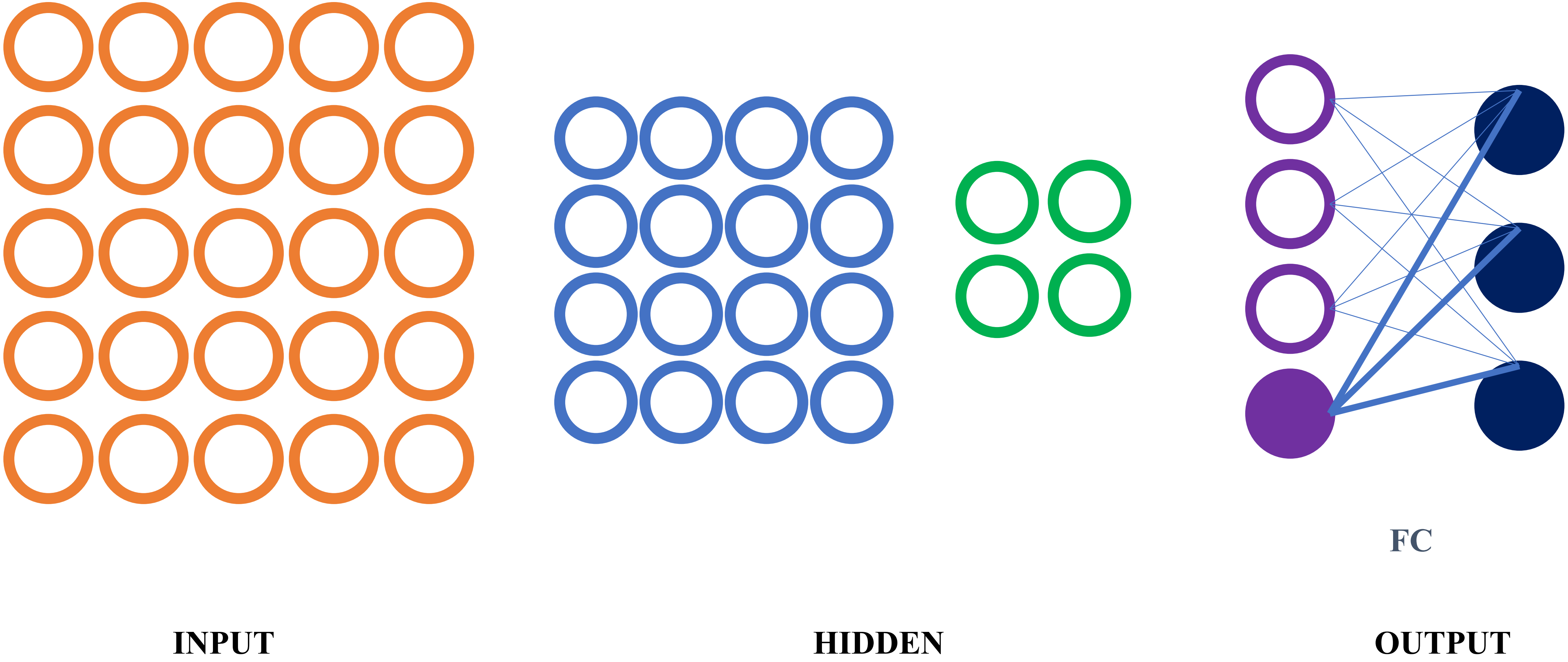
INPUT HIDDEN OUTPUT

◆ Convolutional Neural Network



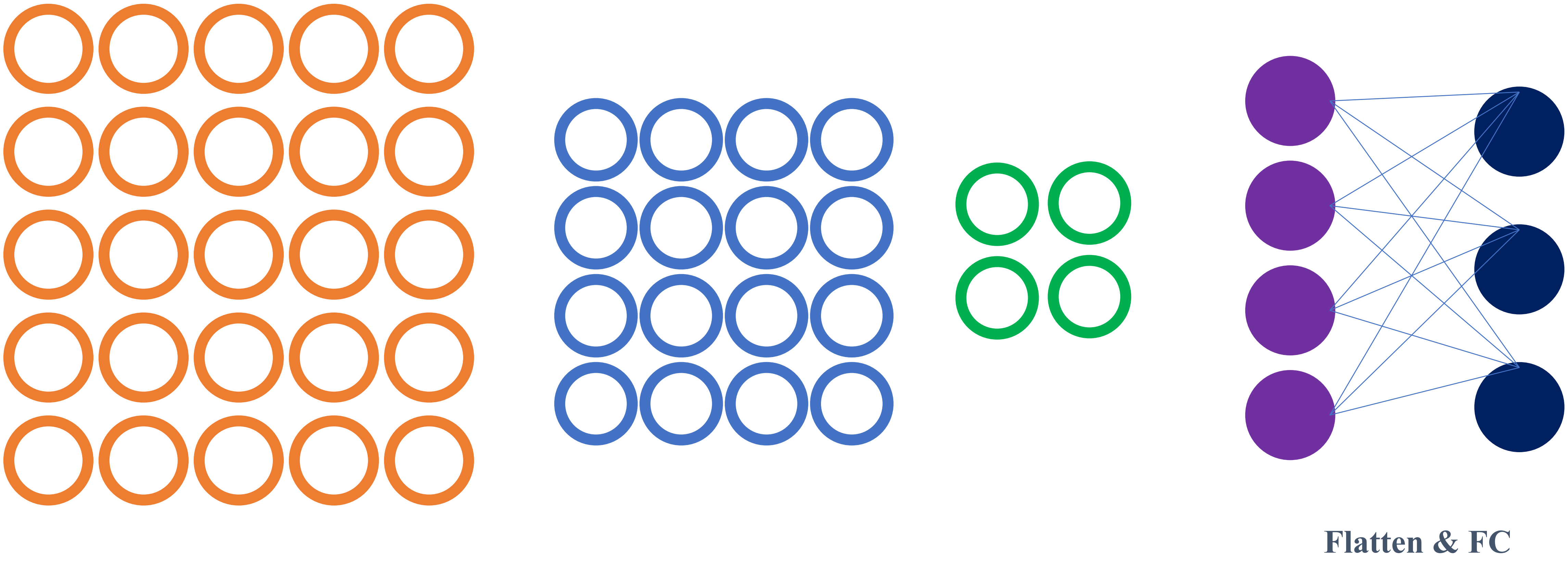
UNIT 02. Layers in CNN

◆ Convolutional Neural Network



UNIT 02. Layers in CNN

◆ Convolutional Neural Network



INPUT HIDDEN OUTPUT

Contents

Unit 01 | Intro

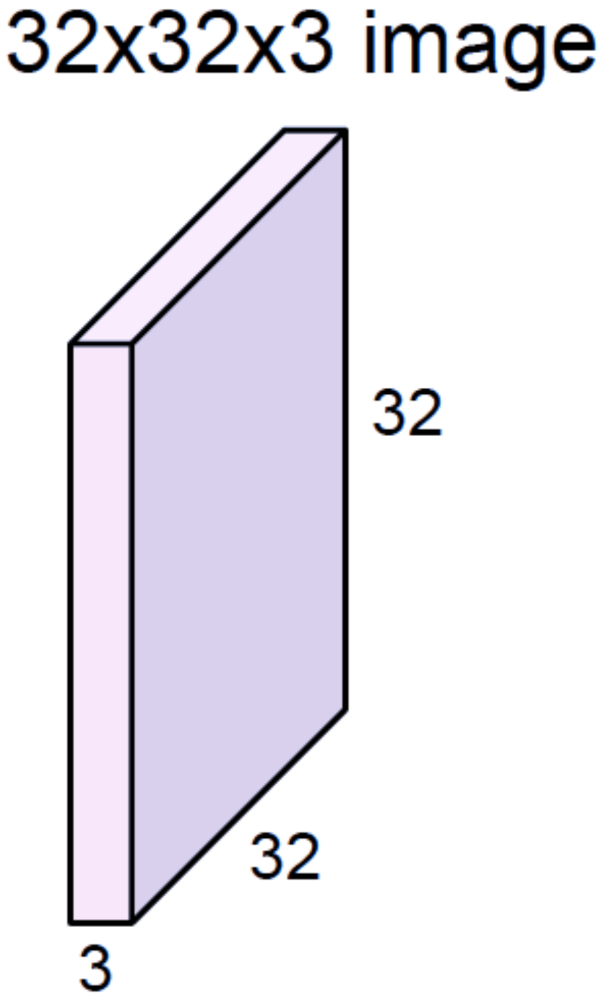
Unit 02 | Layers in CNN

Unit 03 | Convolution

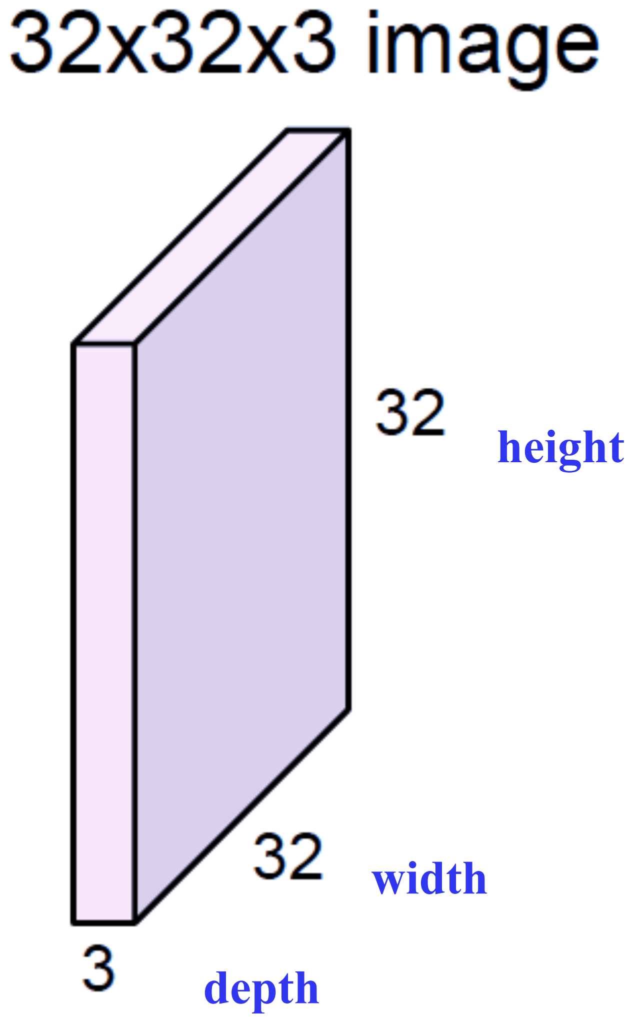
Unit 04 | Sub-Sampling

Unit 05 | Summary

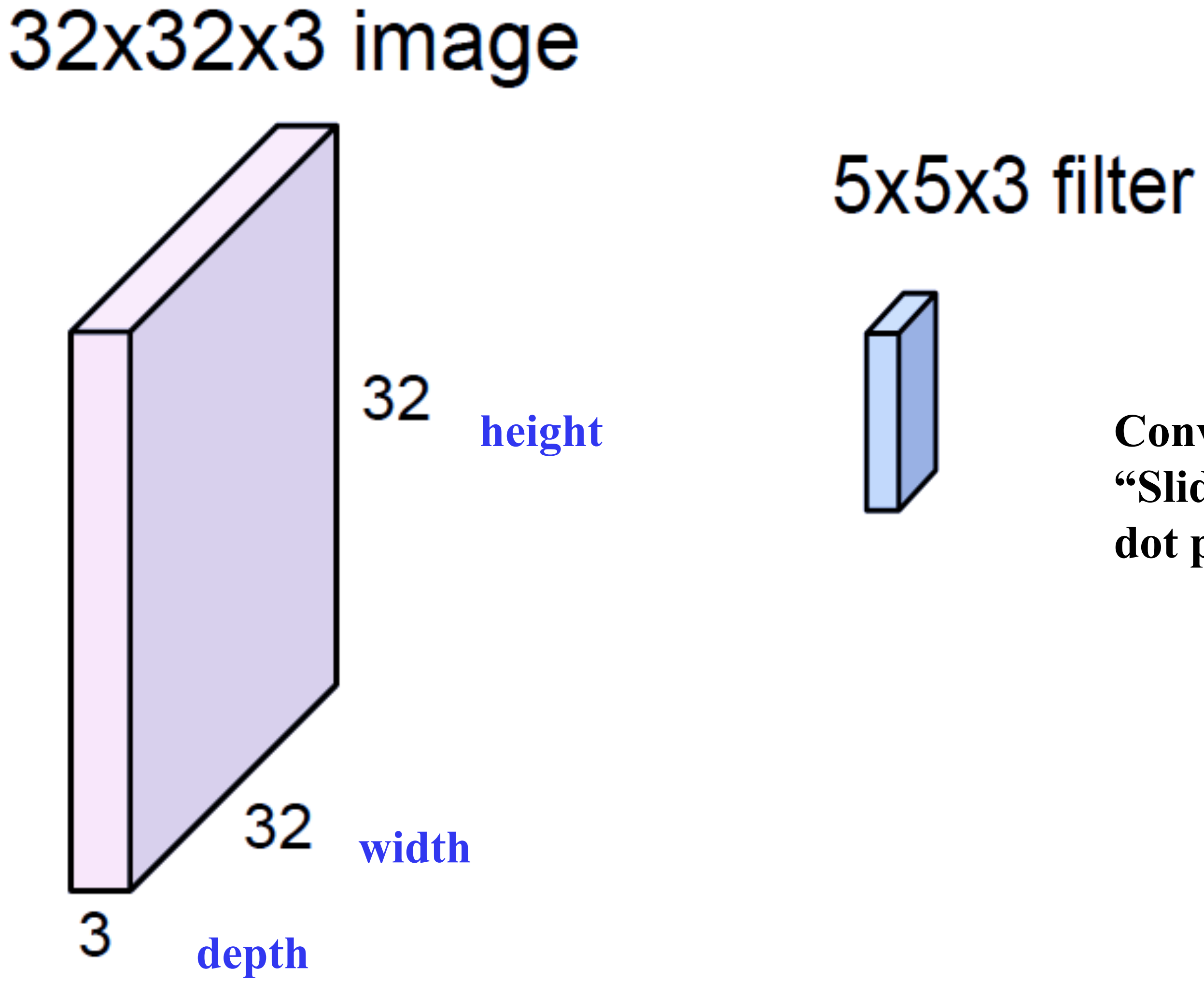
◆ Convolution



◆ Convolution

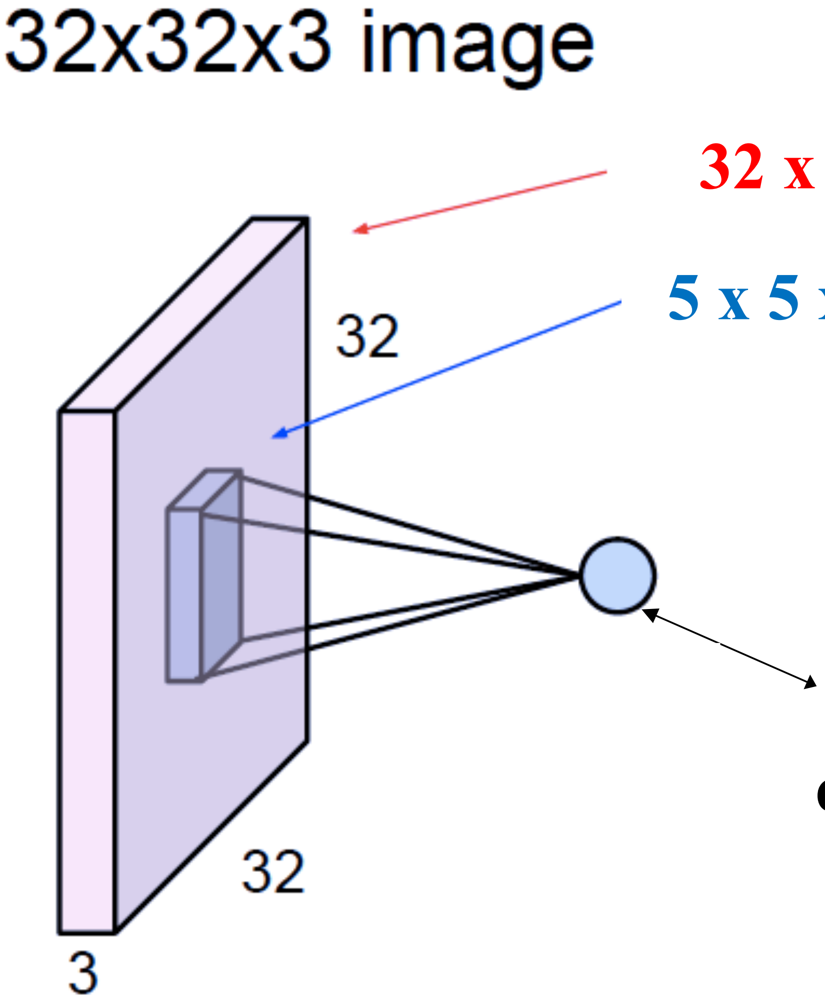


◆ Convolution



Convolve the filter with the image
“Slide over the image spatially, computing dot products”

◆ Convolution



32 x 32 x 3 image

5 x 5 x 3 Filter ; *W*

1 number
the result of taking a dot product
between the filter and a local receptive field

ex) 5*5*3 = 75-dimensional dot product + bias

◆ Convolution

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

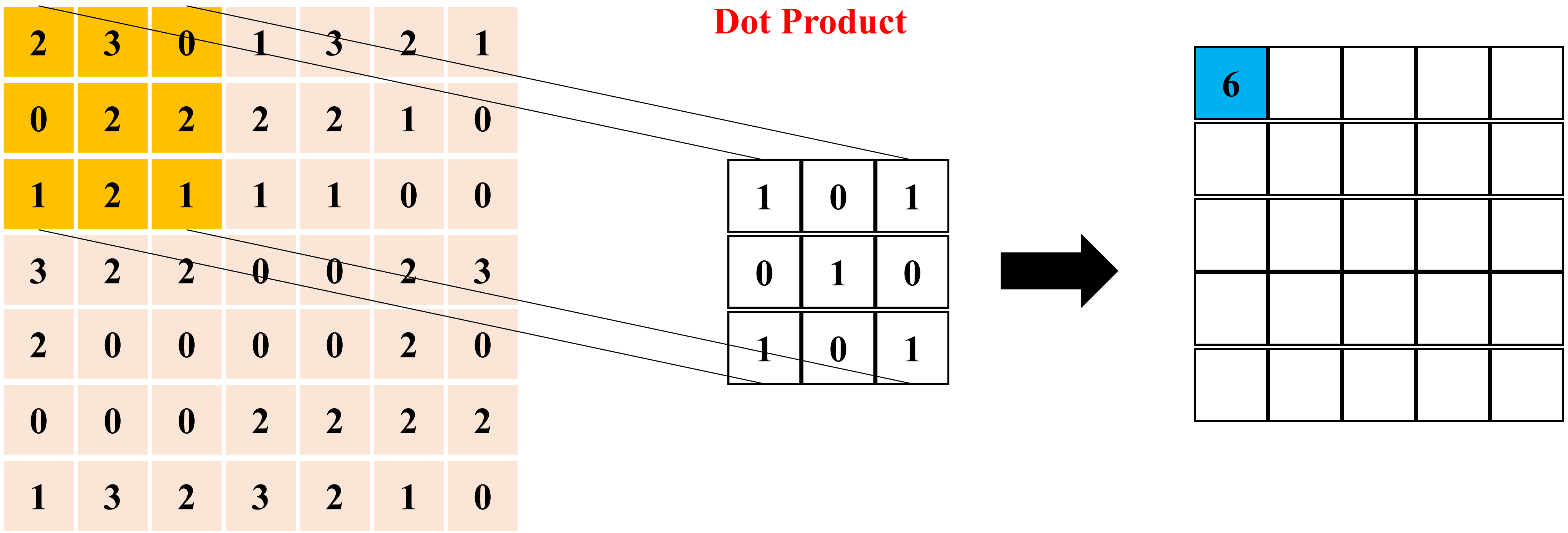
Input Volumn (7x7)

1	0	1
0	1	0
1	0	1

Filter (3 x 3)

Convolved Feature Volumn (5 x 5)

◆ Convolution



Dot Product

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input Volumn

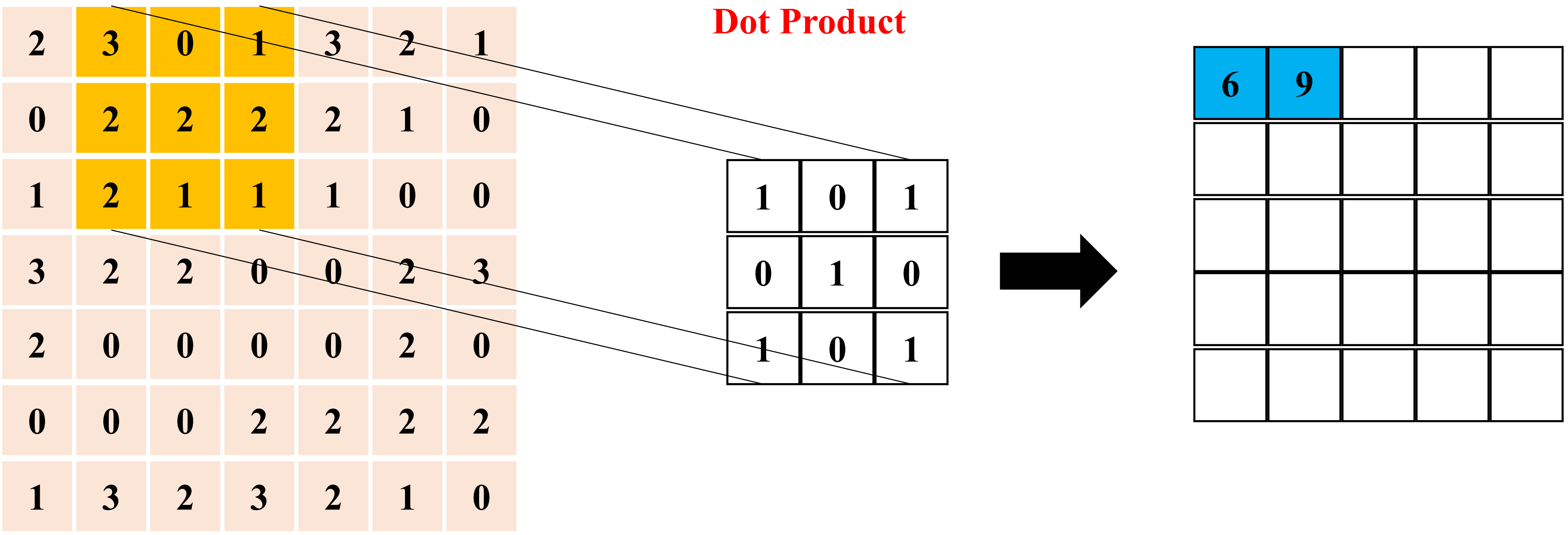
1	0	1
0	1	0
1	0	1

Filter (3 x 3)

6				

Convolved Feature Volumn (5 x 5)

◆ Convolution



2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input Volumn

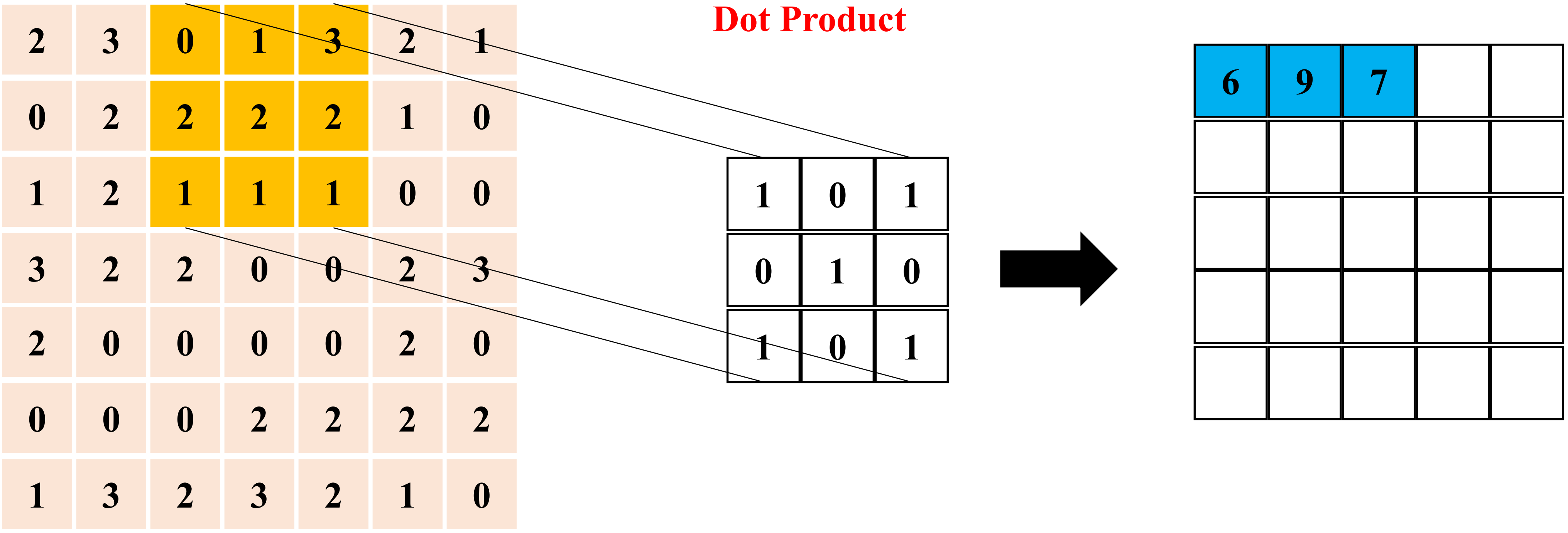
1	0	1
0	1	0
1	0	1

Filter (3 x 3)

6	9			

Convolved Feature Volumn (5 x 5)

◆ Convolution



Dot Product

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input Volumn

1	0	1
0	1	0
1	0	1

Filter (3 x 3)

6	9	7		

Convolved Feature Volumn (5 x 5)

◆ Convolution

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input Volumn

Dot Product

1	0	1
0	1	0
1	0	1

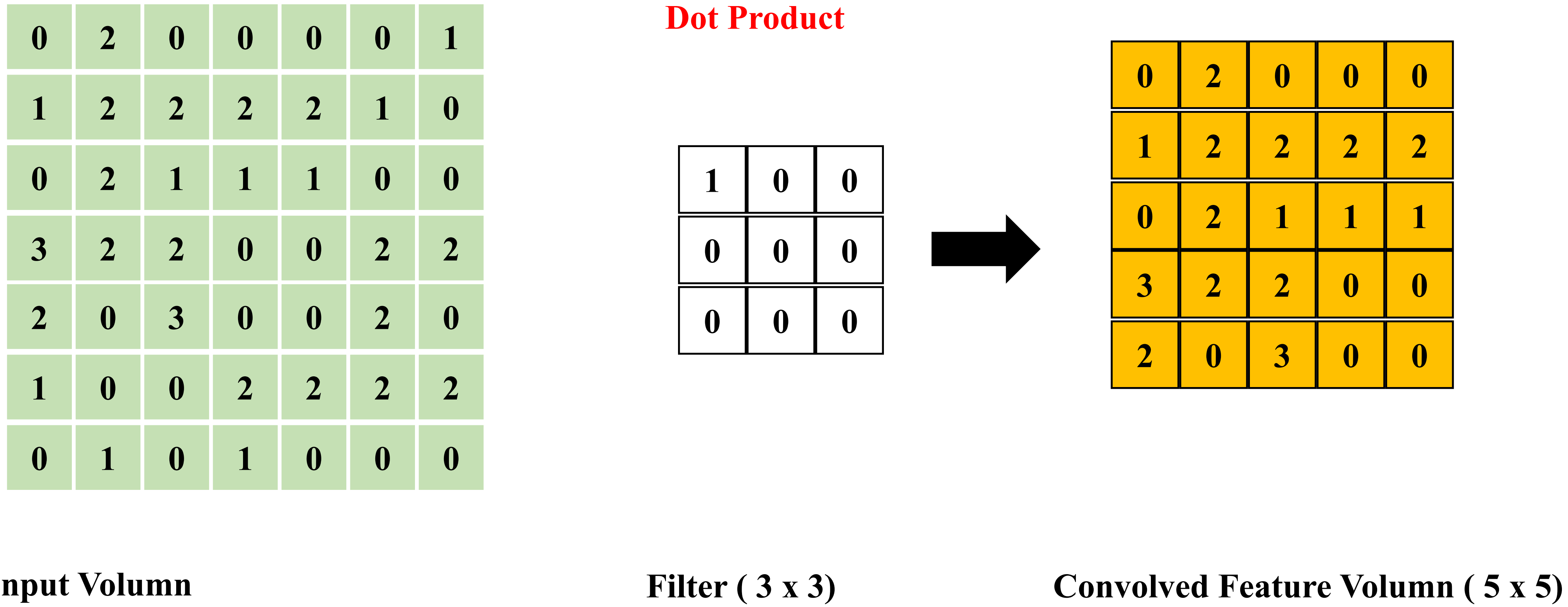
Filter (3 x 3)

➡

6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

Convolved Feature Volumn (5 x 5)

◆ Convolution



Dot Product

0	2	0	0	0	0	1
1	2	2	2	2	1	0
0	2	1	1	1	0	0
3	2	2	0	0	2	2
2	0	3	0	0	2	0
1	0	0	2	2	2	2
0	1	0	1	0	0	0

Input Volumn

1	0	0
0	0	0
0	0	0

Filter (3 x 3)

0	2	0	0	0
1	2	2	2	2
0	2	1	1	1
3	2	2	0	0
2	0	3	0	0

Convolved Feature Volumn (5 x 5)

◆ Convolution

1	0	2	0	1	0	3
0	2	2	2	2	1	0
2	2	1	1	1	0	3
0	2	2	0	0	2	0
0	0	0	0	0	2	2
2	0	0	2	2	2	0
0	0	2	0	2	0	0

Input Volumn

Dot Product

0	0	0
0	1	0
0	0	0

Filter (3 x 3)

→

2	2	2	2	1
2	1	1	1	0
2	2	0	0	2
0	0	0	0	2
0	0	2	2	2

Convolved Feature Volumn (5 x 5)

◆ Convolution

6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

+

0	2	0	0	0
1	2	2	2	2
0	2	1	1	1
3	2	2	0	0
2	0	3	0	0

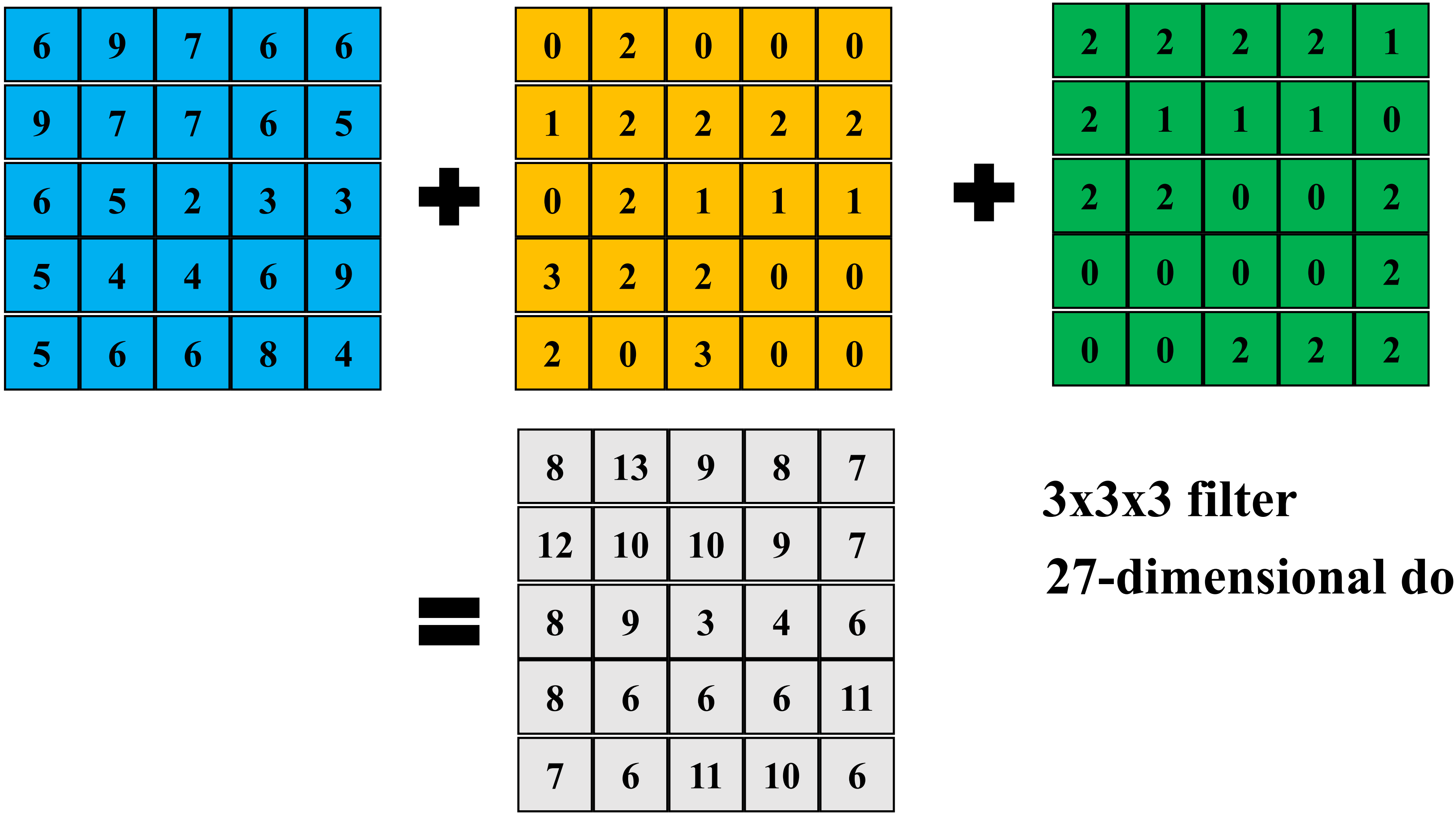
+

2	2	2	2	1
2	1	1	1	0
2	2	0	0	2
0	0	0	0	2
0	0	2	2	2

=

8	13	9	8	7
12	10	10	9	7
8	9	3	4	6
8	6	6	6	11
7	6	11	10	6

◆ Convolution



6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

+

0	2	0	0	0
1	2	2	2	2
0	2	1	1	1
3	2	2	0	0
2	0	3	0	0

+

2	2	2	2	1
2	1	1	1	0
2	2	0	0	2
0	0	0	0	2
0	0	2	2	2

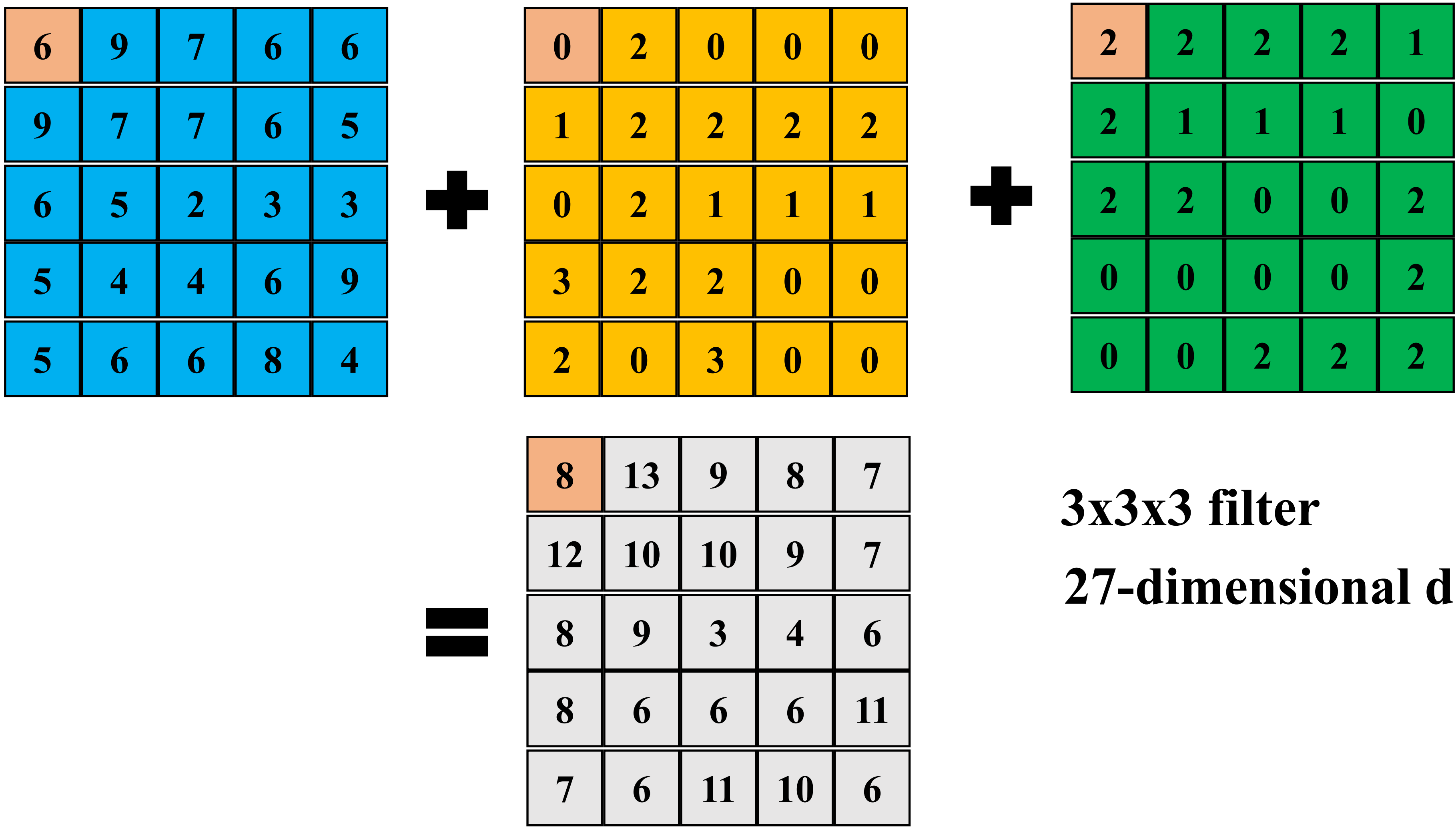
=

8	13	9	8	7
12	10	10	9	7
8	9	3	4	6
8	6	6	6	11
7	6	11	10	6

3x3x3 filter

27-dimensional dot product

◆ Convolution



6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

+

0	2	0	0	0
1	2	2	2	2
0	2	1	1	1
3	2	2	0	0
2	0	3	0	0

+

2	2	2	2	1
2	1	1	1	0
2	2	0	0	2
0	0	0	0	2
0	0	2	2	2

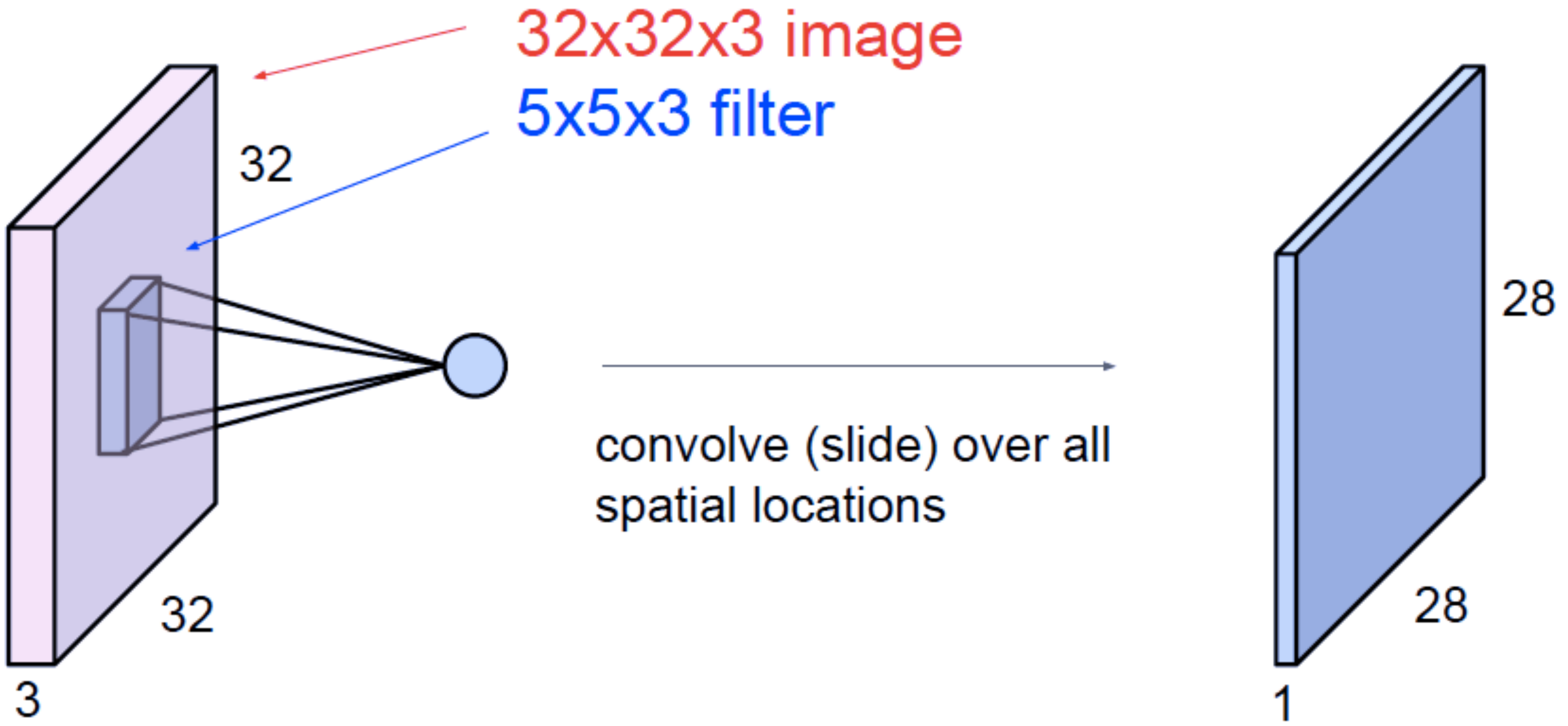
=

8	13	9	8	7
12	10	10	9	7
8	9	3	4	6
8	6	6	6	11
7	6	11	10	6

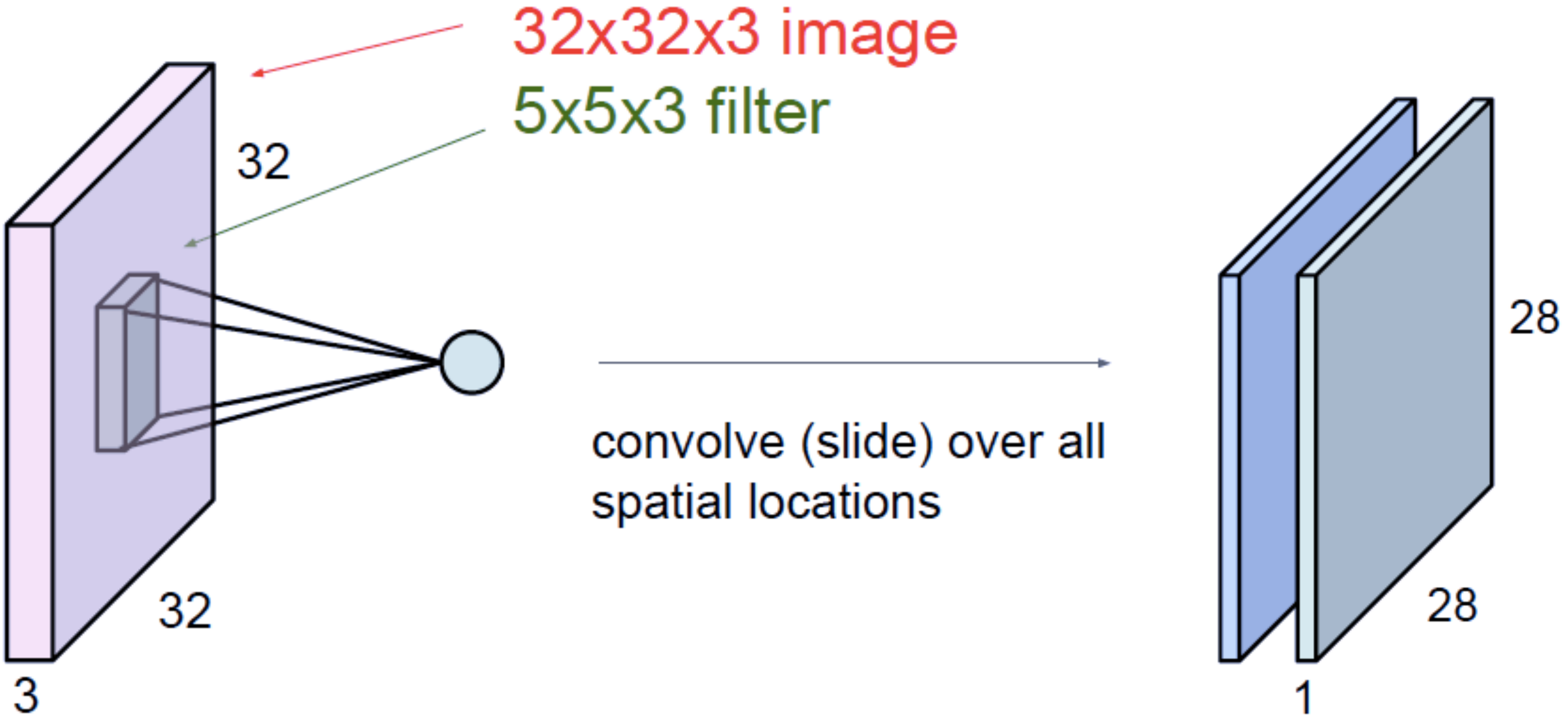
3x3x3 filter

27-dimensional dot product

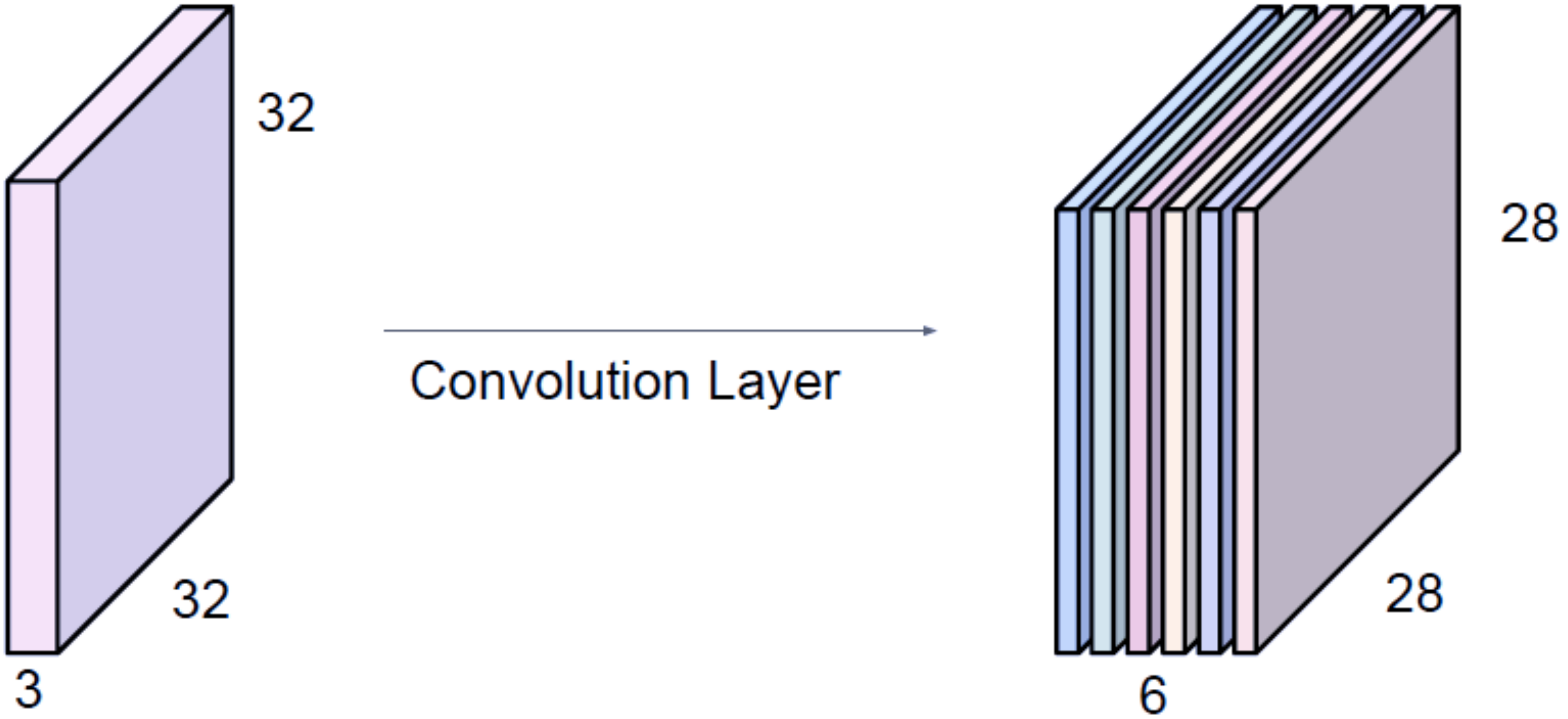
◆ Convolution



◆ Convolution

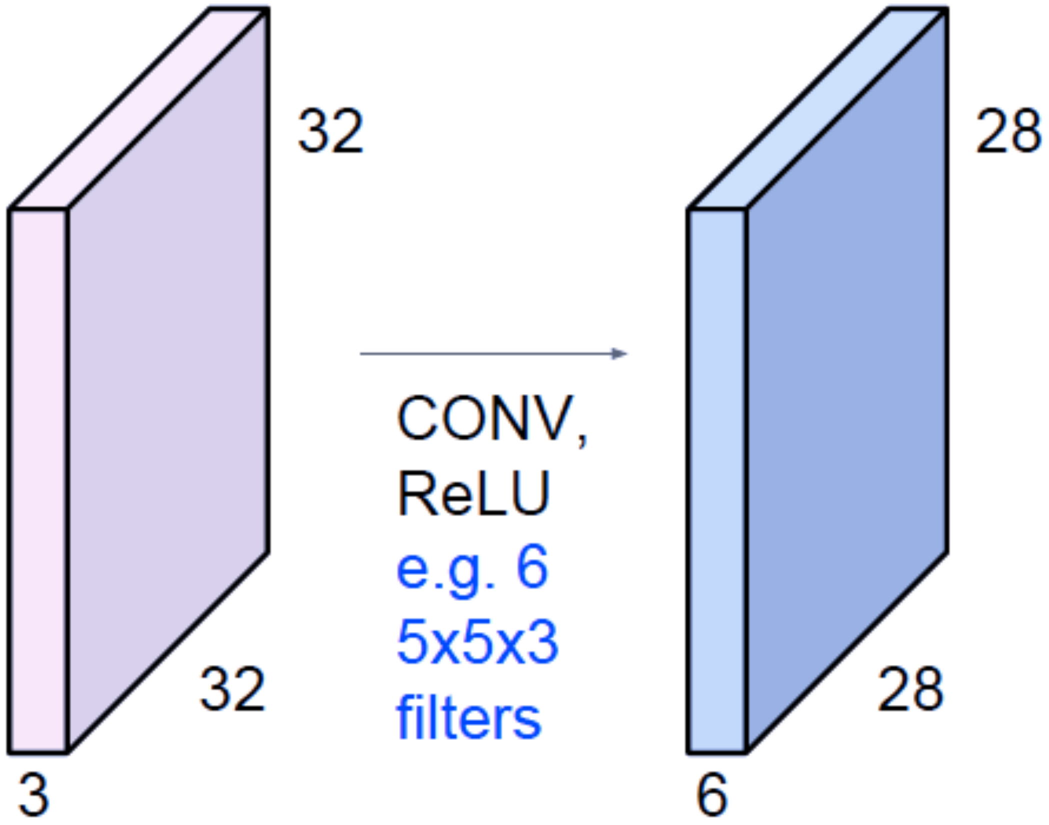


◆ Convolution

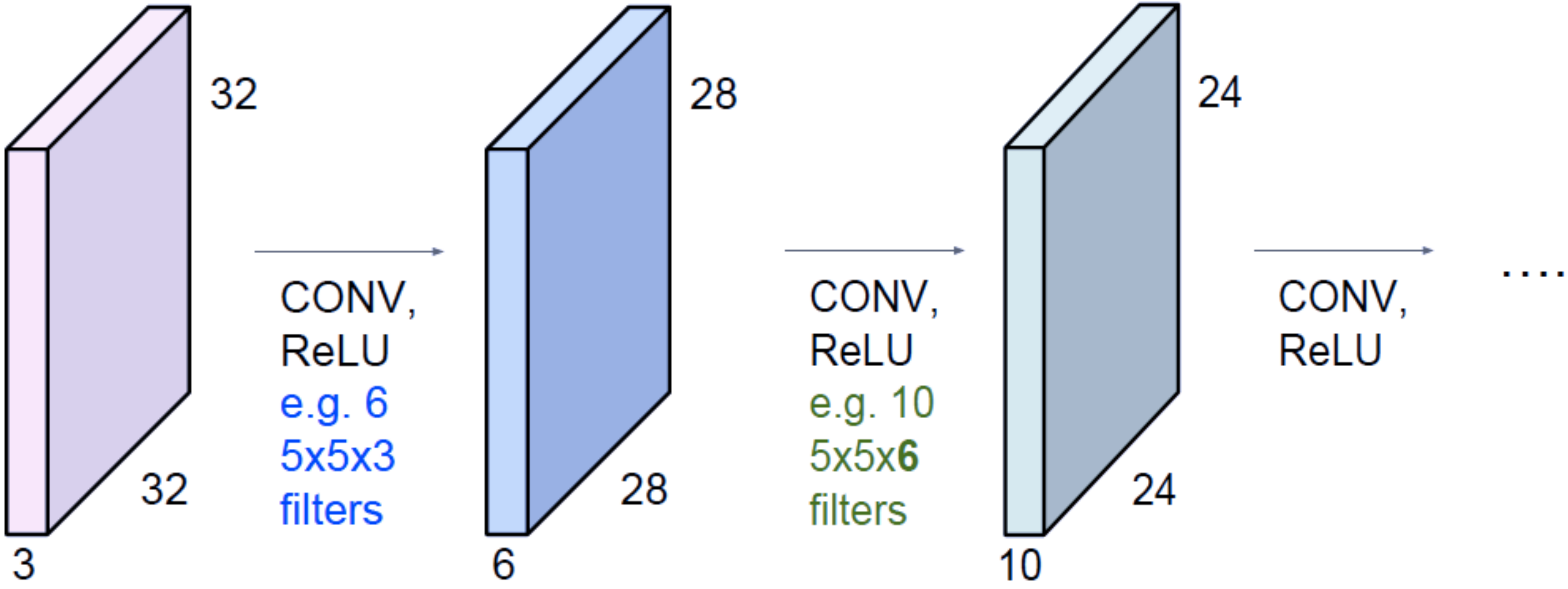


- 예를들어, 6개의 5x5x3 filter가 있다면, 6개의 feature map이 생성됨
- 6개의 feature map을 stack up 하여 28x28x6 의 새로운 size의 이미지를 얻음

◆ Convolution



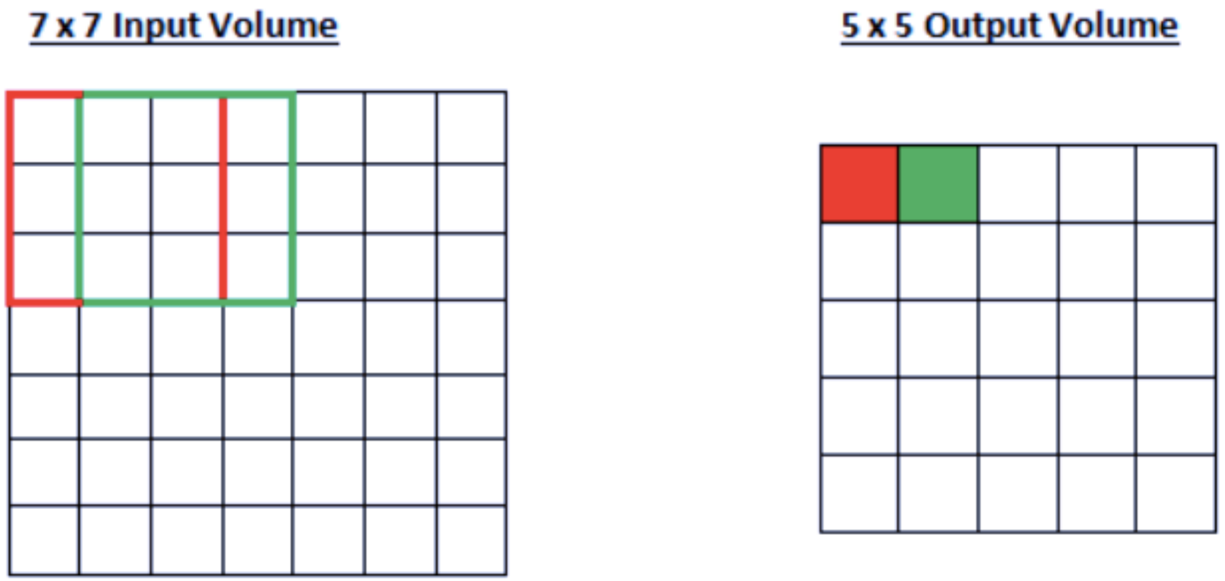
◆ Convolution



◆ Get familiar with this

◆ Stride

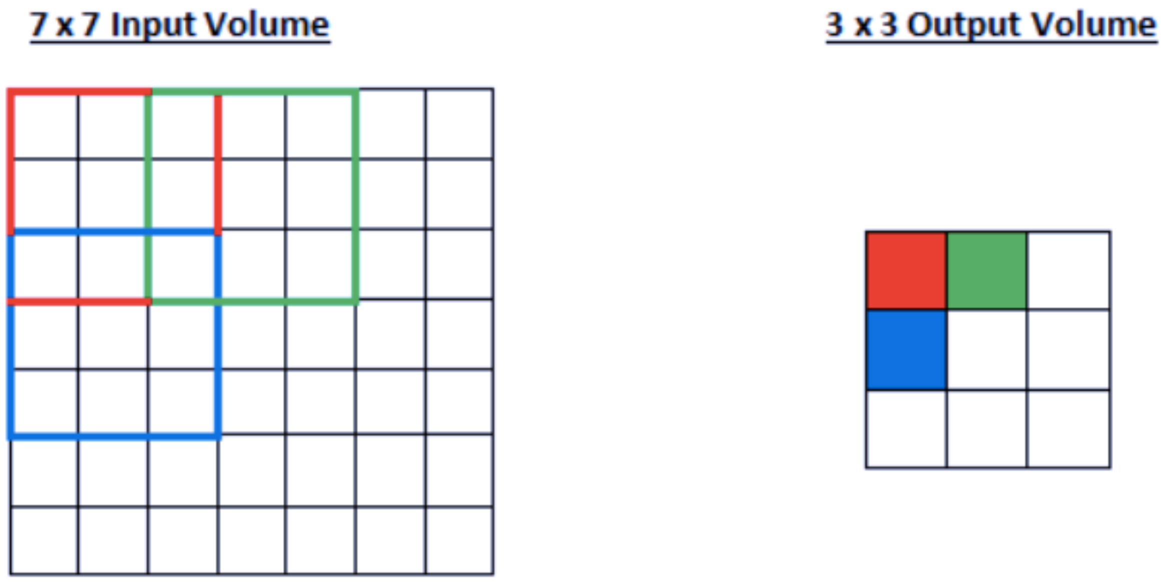
- Stride =The amount by which the filter shifts
- Stride is normally set in a way so that the output volume is an integer and not a fraction



Stride = 1

◆ Stride

- Stride =The amount by which the filter shifts
- Stride is normally set in a way so that the output volume is an integer and not a fraction



Stride = 2

◆ Stride

- **Stride =The amount by which the filter shifts**
- **Stride is normally set in a way so that the output volume is an integer and not a fraction**

?

Stride = 3

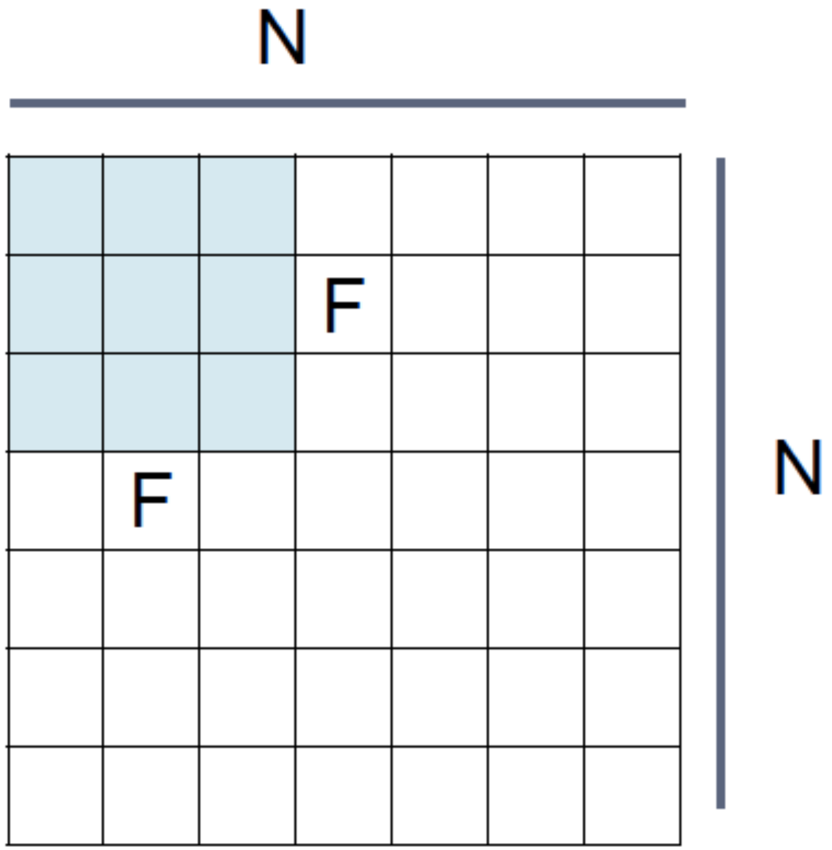
◆ Stride

- Stride =The amount by which the filter shifts
- Stride is normally set in a way so that the output volume is an integer and not a fraction

doesn’t fit!
cannot apply 3x3 filter on
7x7 input with stride 3.

Stride = 3

◆ Stride



- N = Input size
- F = Filter size
- Output size: {(N - F) / stride} + 1

- Ex) N=7, F=3

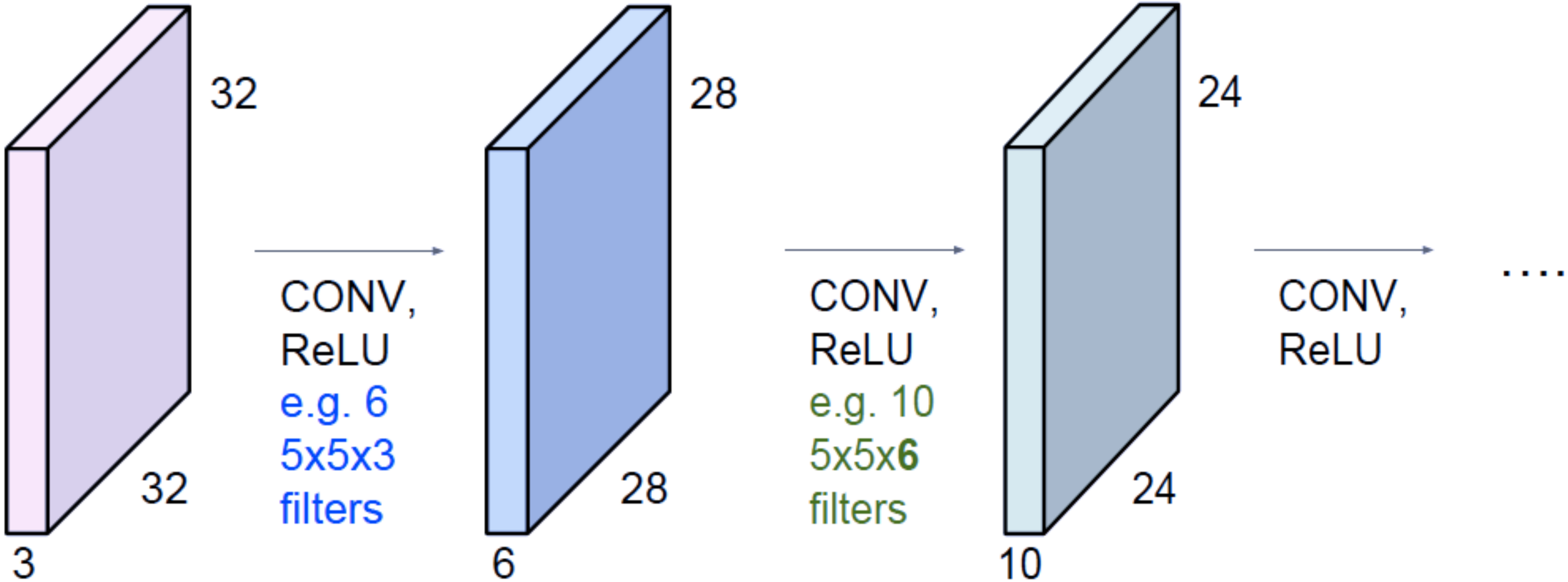
  stride=1 -> {(7-3)/1} + 1 = 5

  Stride=2 -> {(7-3)/2} + 1 = 3

  Stride=3 -> {(7-3)/3} + 1 = 2.33   X

◆ Padding

- **32x32 input convolved repeatedly with 5x5 filters shrinks volumes spatially! (32 -> 28 -> 24 ...). Shrinking too fast is not good, doesn’t work well.**



◆ Padding

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

- **Ex) Input 7x7**

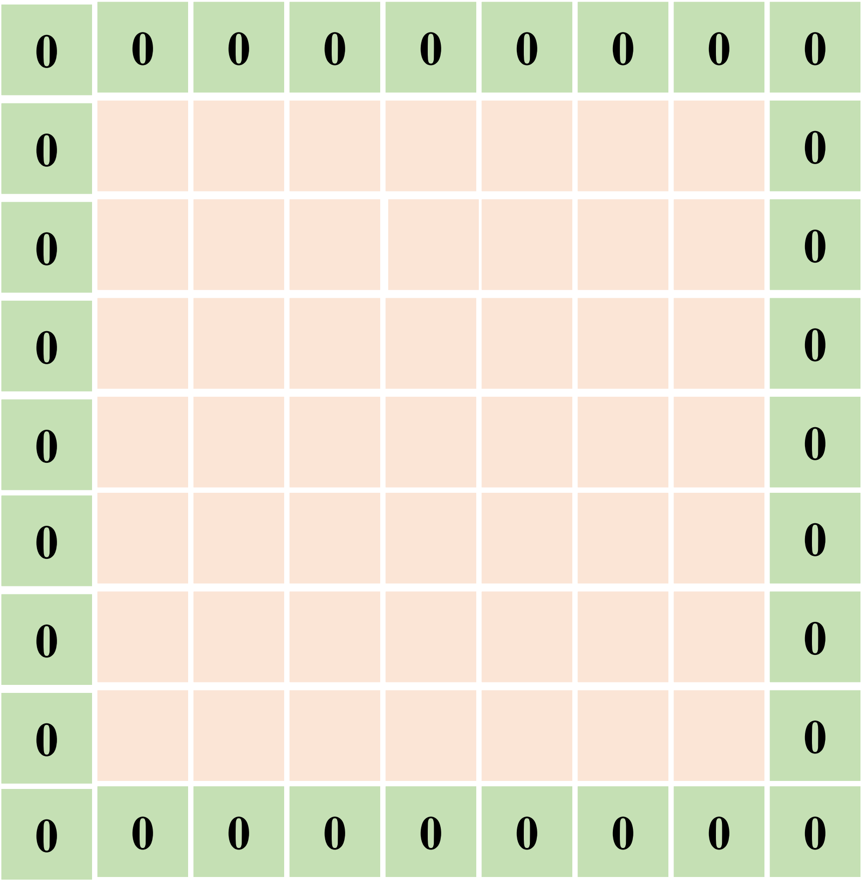
  Filter 3x3, stride = 1

  zero pad with 1 pixel border

  what's the output?

- **Recall**
  Output size: {(N - F) / stride} + 1

◆ Padding



0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

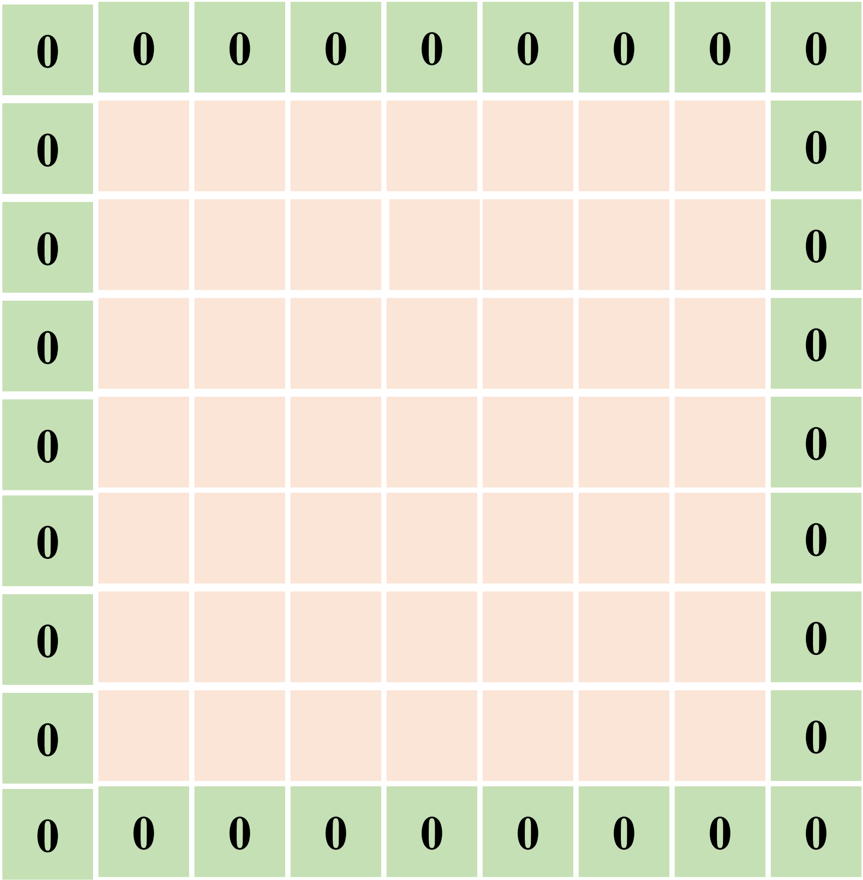
- **Ex) Input 7x7**

  Filter 3x3

  pad with 1 pixel border -> what’s the output?

- **7x7 output!!**
- **output = {(9-3)/1} + 1**

◆ Padding



0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

- Ex) Input 7x7

  Filter 3x3

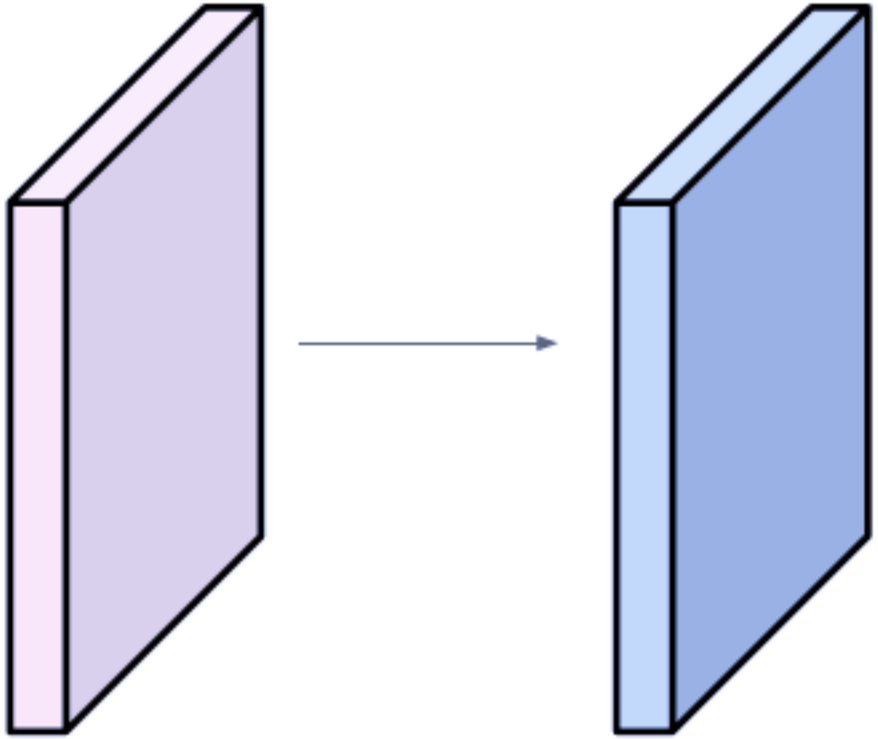
  pad with 1 pixel border -> what's the output?

- 7x7 output!!
- 일반적으로 stride =1 을 주며, Filter size 가 3인 경우 zero-padding = 1 을 주면
- 공식에 따라, input size 를 보존하는 output이 생성

◆ Example

- Input volume: 32x32x3

  10 5x5 filters with stride 1, pad 2

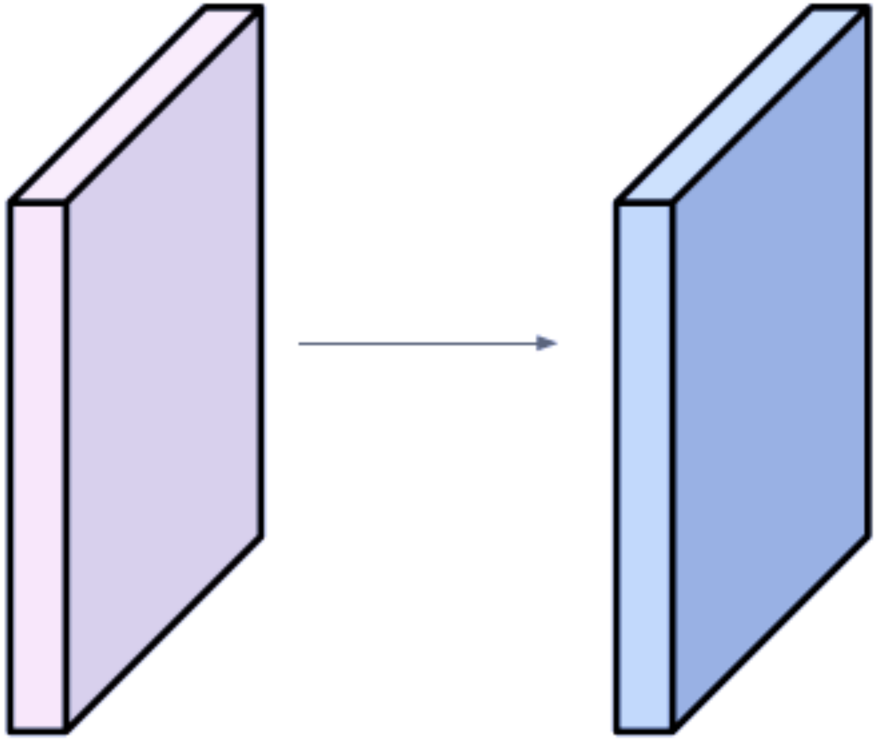


- Number of parameters in this layer?

◆ Example

- Input volume: 32x32x3

  10 5x5 filters with stride 1, pad 2



- Number of parameters in this layer?
- Each filter has 5*5*3 + 1 = 76 params (+1 for bias)
  -> 76 * 10 = 760

Contents

Unit 01 | Intro

Unit 02 | Layers in CNN

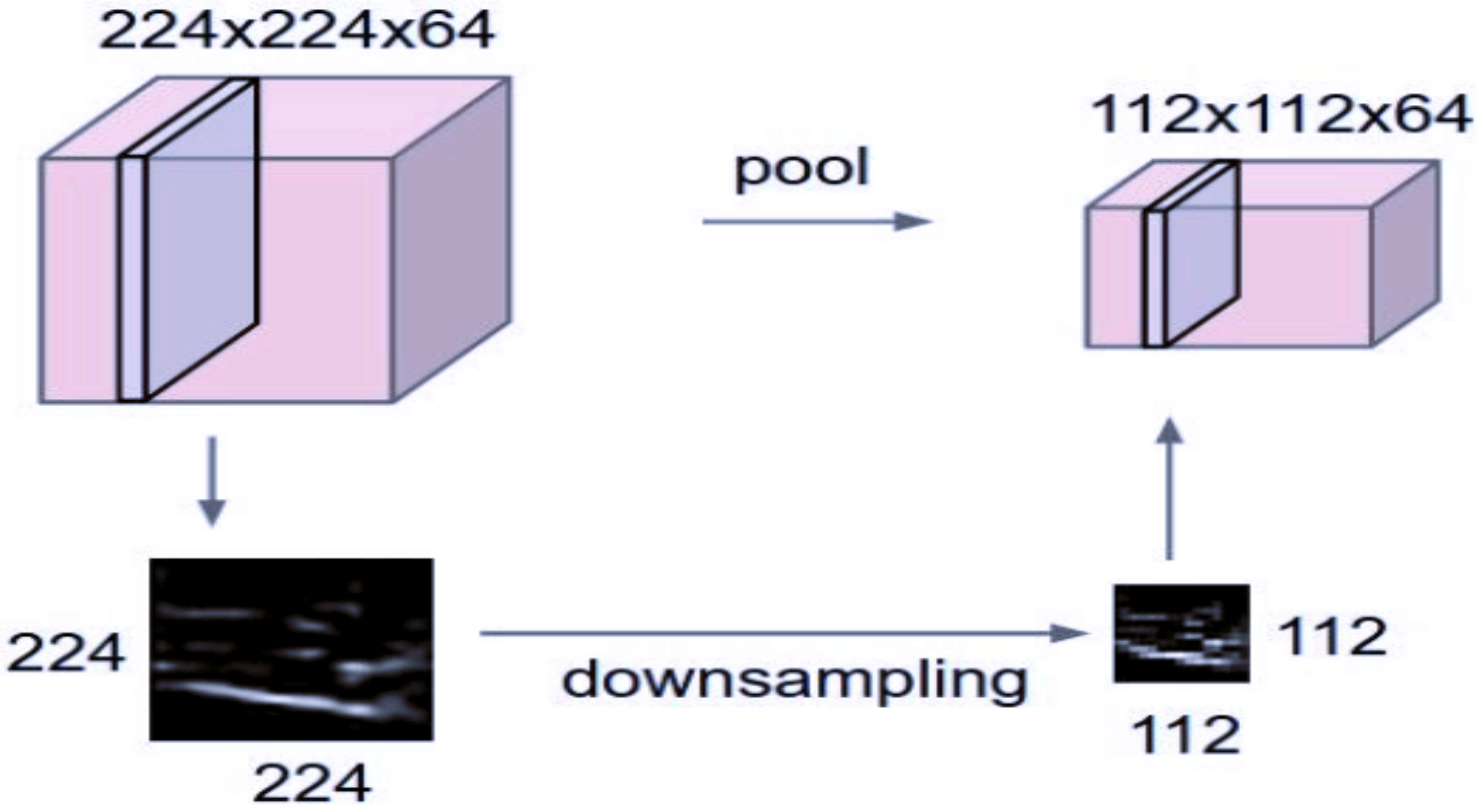
Unit 03 | Convolution

Unit 04 | Sub-Sampling

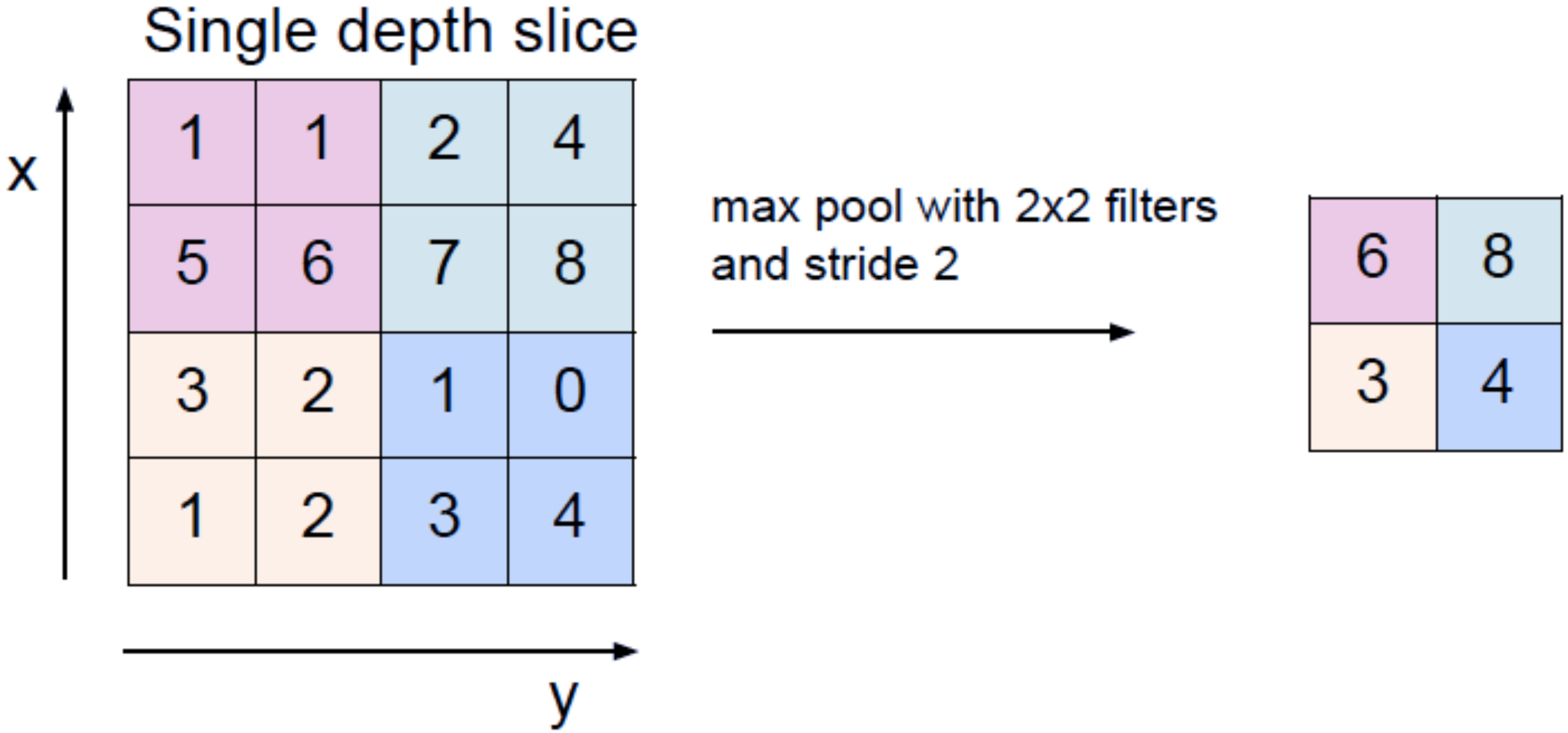
Unit 05 | Summary

◆ Subsampling

- **makes the representations smaller and more manageable**
- **operates over each activation map independently:**



◆ Max-pooling



Single depth slice

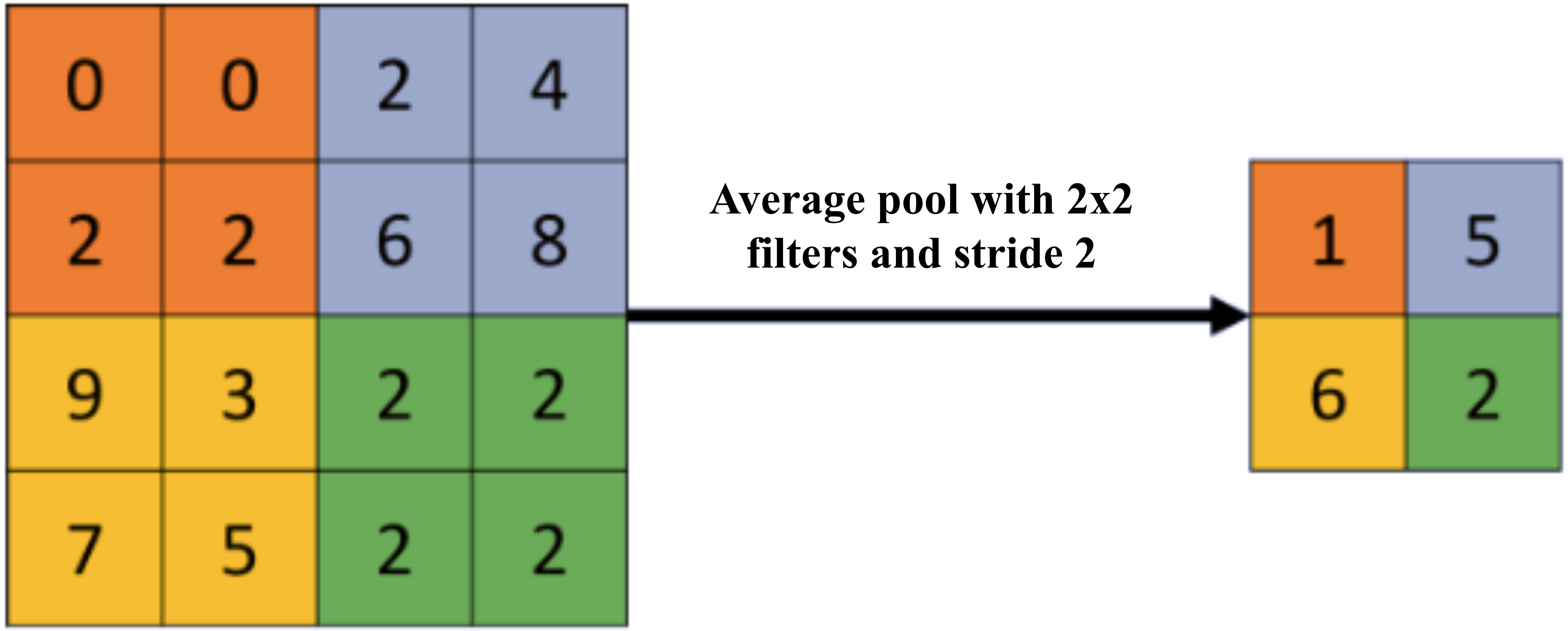
1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

max pool with 2x2 filters and stride 2

6	8
3	4

◆ Average Pooling

Single depth slice



0	0	2	4
2	2	6	8
9	3	2	2
7	5	2	2

Average pool with 2x2 filters and stride 2

1	5
6	2

Contents

Unit 01 | Intro

Unit 02 | Layers in CNN

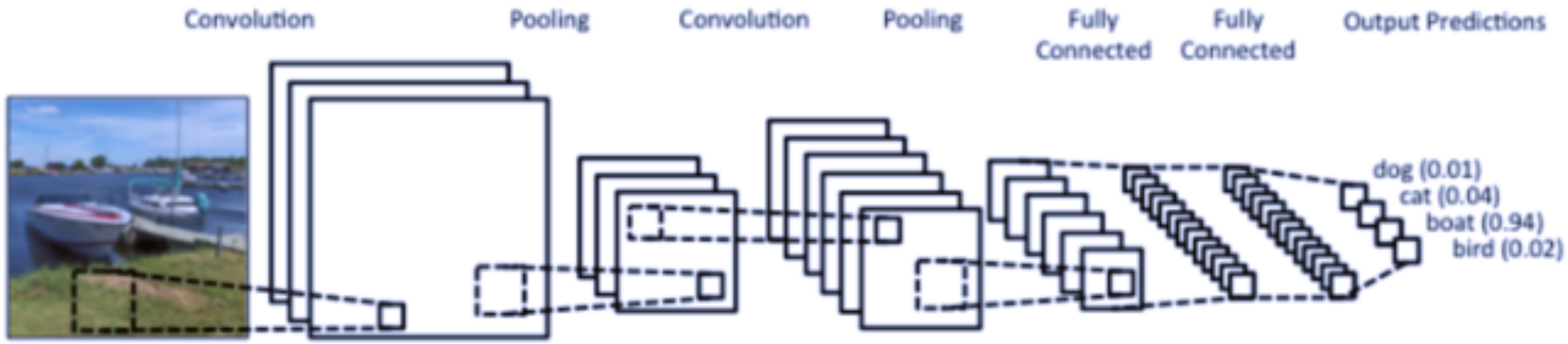
Unit 03 | Convolution

Unit 04 | Sub-Sampling

Unit 05 | Summary

UNIT 05. Summary

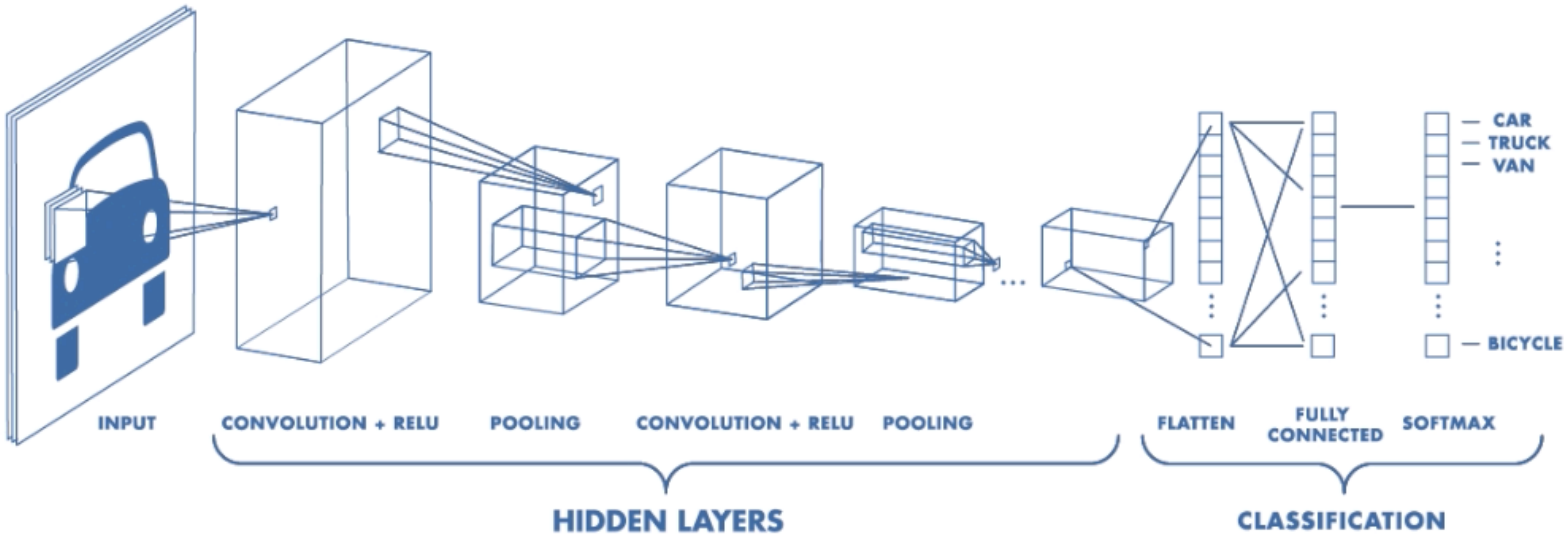
◆ Layers in CNN



- **CNN = Convolutions followed by subsampling and fully connected layer**
- **Feature Extraction <- Convolution + subsampling layers**
- **Classifier <- Fully connected layer**

UNIT 02. Layers in CNN

◆ Layers in CNN



- This is Pretty much everything about the convolutional neural network
- Convolution + Subsampling + Full Connection

◆ Layers in CNN

Why is so powerful?

- **Local connectivity(**receptive field**)**
- **Shared Weights and Biases**
- **Compositionality**

Q & A

Reference

https://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html

http://cs231n.github.io/convolutional-networks/

https://medium.com/fbdevclagos/applications-of-deep-learning-4f725f9147ac

imagenet-classification-with-deep-convolutional-neural-networks(Alex Krizhevsky 2012)

https://laonple.blog.me/220643128255