Practical Machine Learning

Lecture 8
Convolutional Neural Networks (CNN)

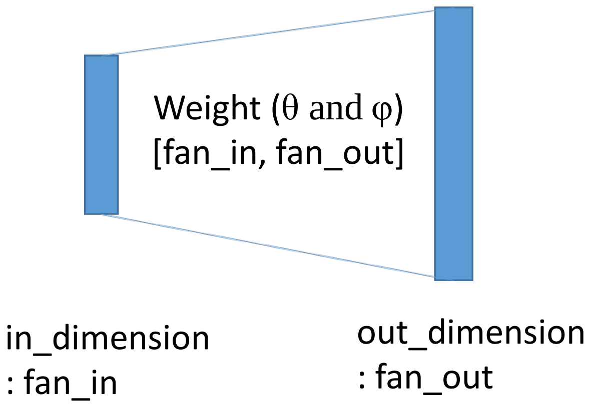
Dr. Suyong Eum

OSAKA UNIVERSITY

A question from the last class: initialization of parameters

- Xavier (2010): http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf
  - Either using an uniform distribution or a normal distribution
  - It is implemented in the current Tensorflow: tf.contrib.layers.xavier_initializer()

$$\boxed{U\left(-\sqrt{\frac{6}{\text{fan\_in}+\text{fan\_out}}},\ \sqrt{\frac{6}{\text{fan\_in}+\text{fan\_out}}}\right)},\quad N\left(0,\frac{2}{\text{fan\_in}+\text{fan\_out}}\right)$$



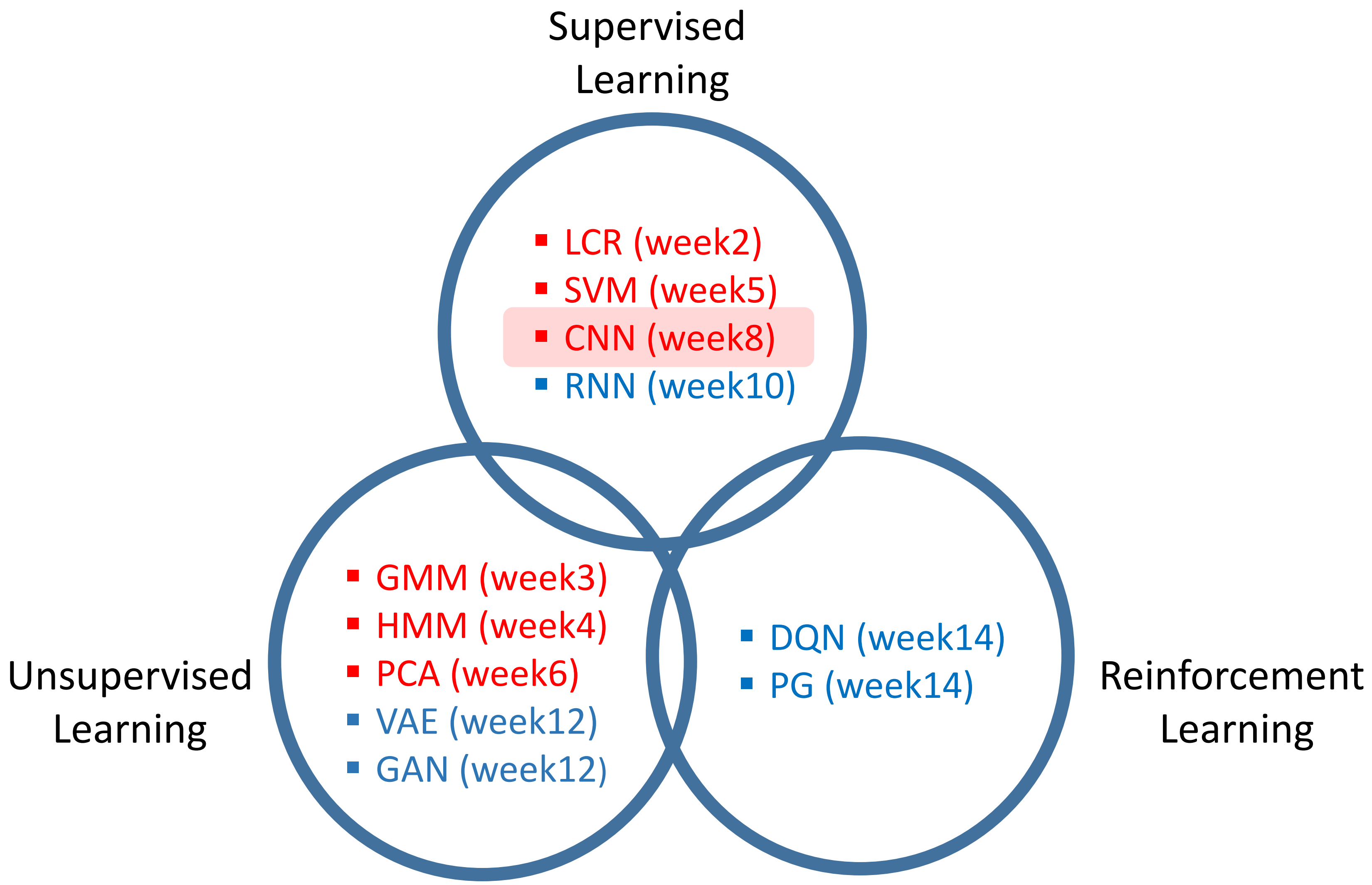
- He (2015): https://arxiv.org/pdf/1502.01852.pdf
  - Using a normal distribution

```python
def xavier_init(fan_in, fan_out, constant=1):
    low = -constant*np.sqrt(6.0/(fan_in + fan_out))
    high = constant*np.sqrt(6.0/(fan_in + fan_out))
    return tf.random_uniform((fan_in, fan_out),
                             minval=low, maxval=high,
                             dtype=tf.float32)
```

$$\boxed{N\left(0,\frac{2}{\text{fan\_in}}\right)}$$

```python
def xavier_init(size):
    in_dim = size[0]
    xavier_stddev = 1. / tf.sqrt(in_dim / 2.)
    return tf.random_normal(shape=size, stddev=xavier_stddev)
```

Where we are

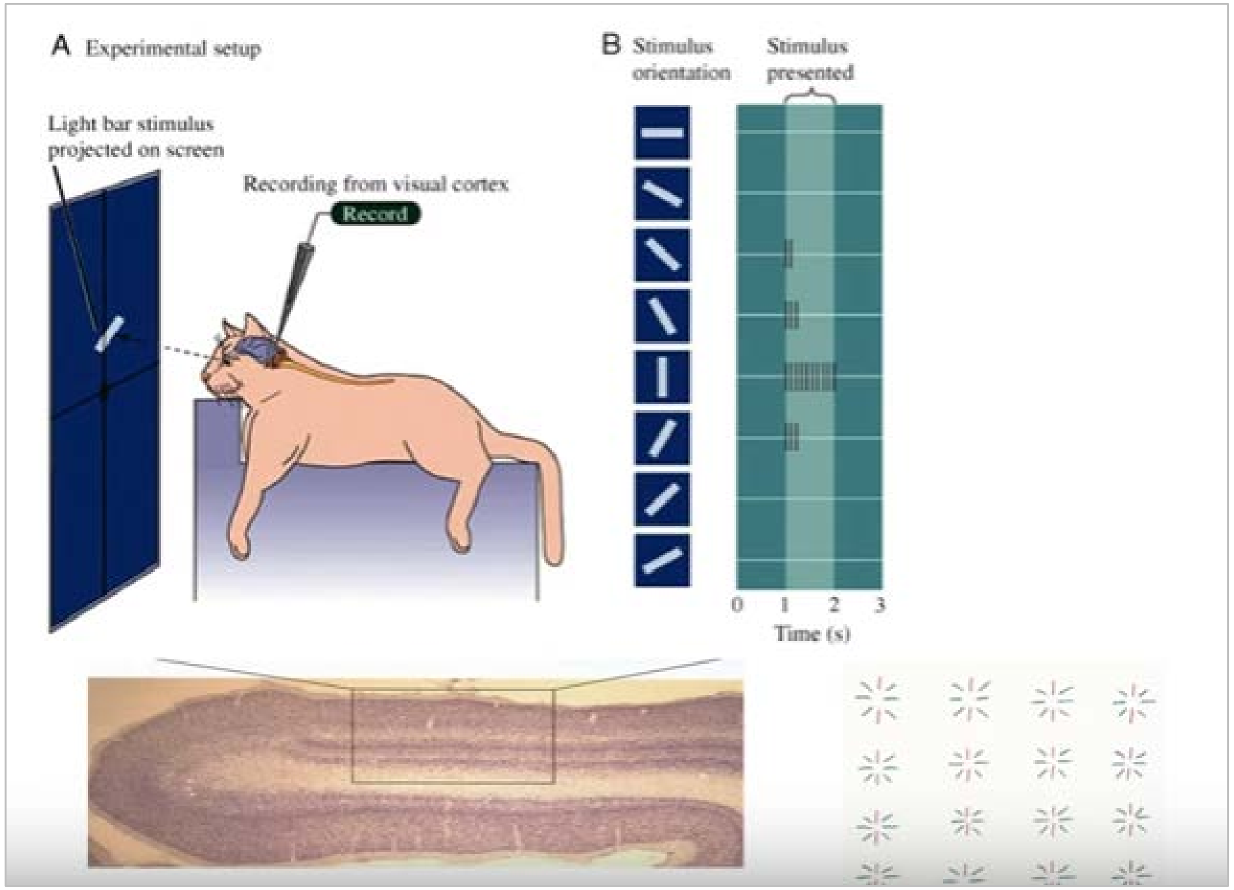

Supervised Learning
LCR (week2)
SVM (week5)
CNN (week8)
RNN (week10)
Unsupervised Learning
GMM (week3)
HMM (week4)
PCA (week6)
VAE (week12)
GAN (week12)
DQN (week14)
PG (week14)
Reinforcement Learning

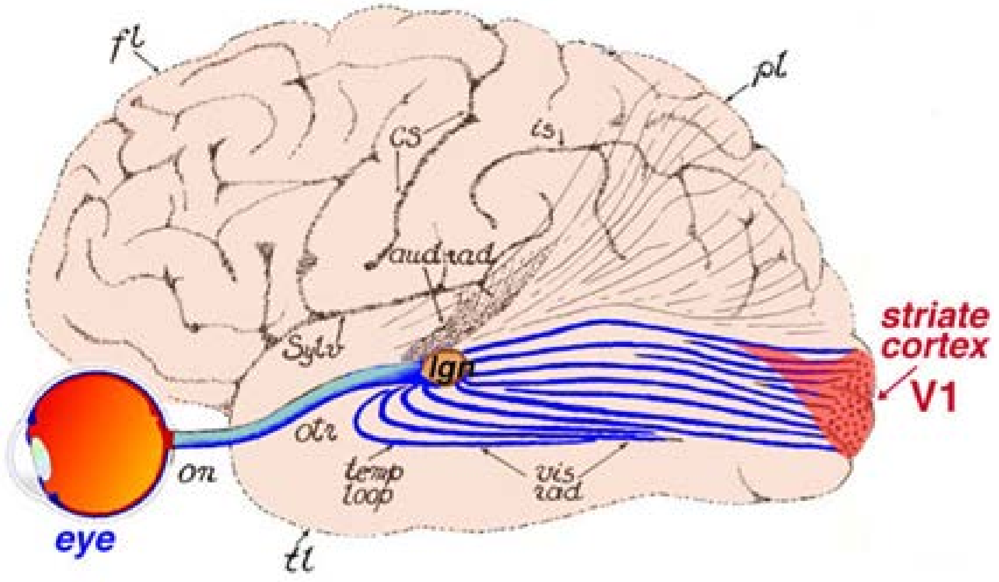
You are going to learn

- What a Convolutional Neural Network (CNN) is,
- How they work,
- How to train CNNs using backpropagation.

History of CNN

- Hubel and Wiesel in 1962

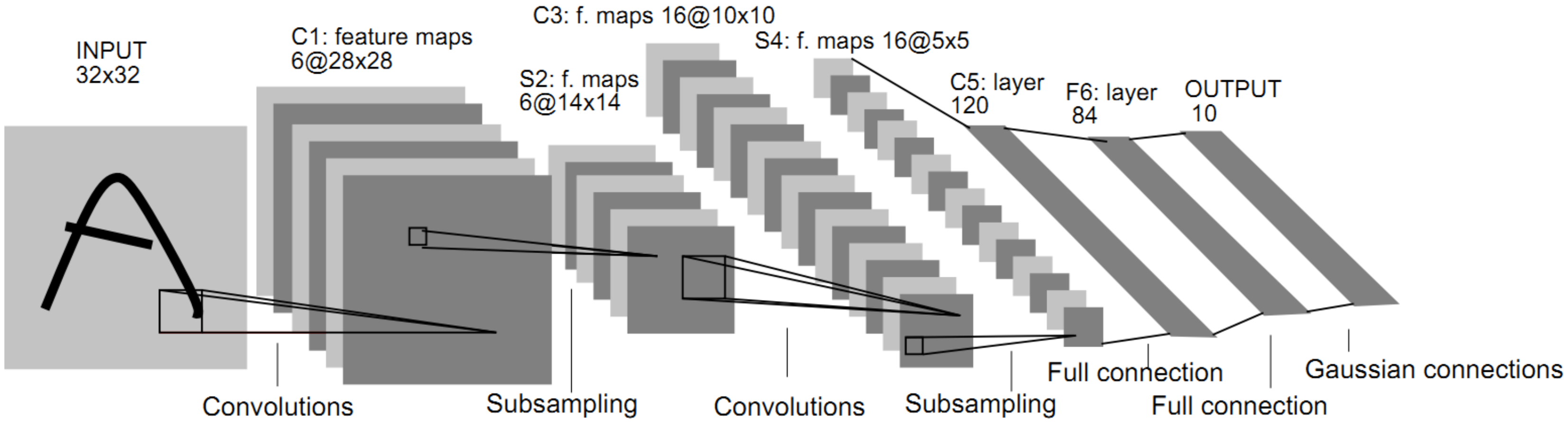




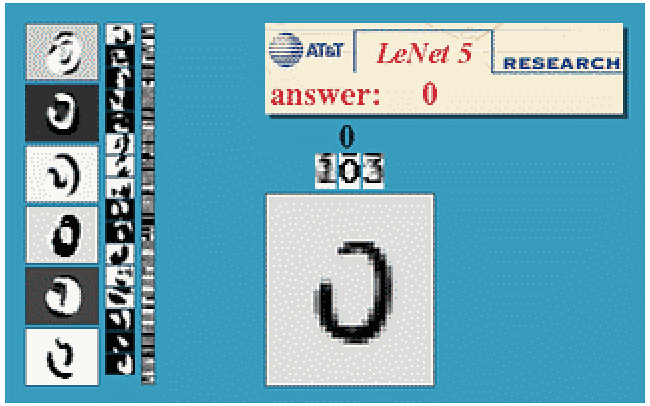
https://www.youtube.com/watch?v=v20-E_2bT2c

History of CNN – 1995

- LeCun et al, "Convolutional Networks for Images, Speech, and Time-Series"
  - http://yann.lecun.com/exdb/publis/pdf/lecun-bengio-95a.pdf



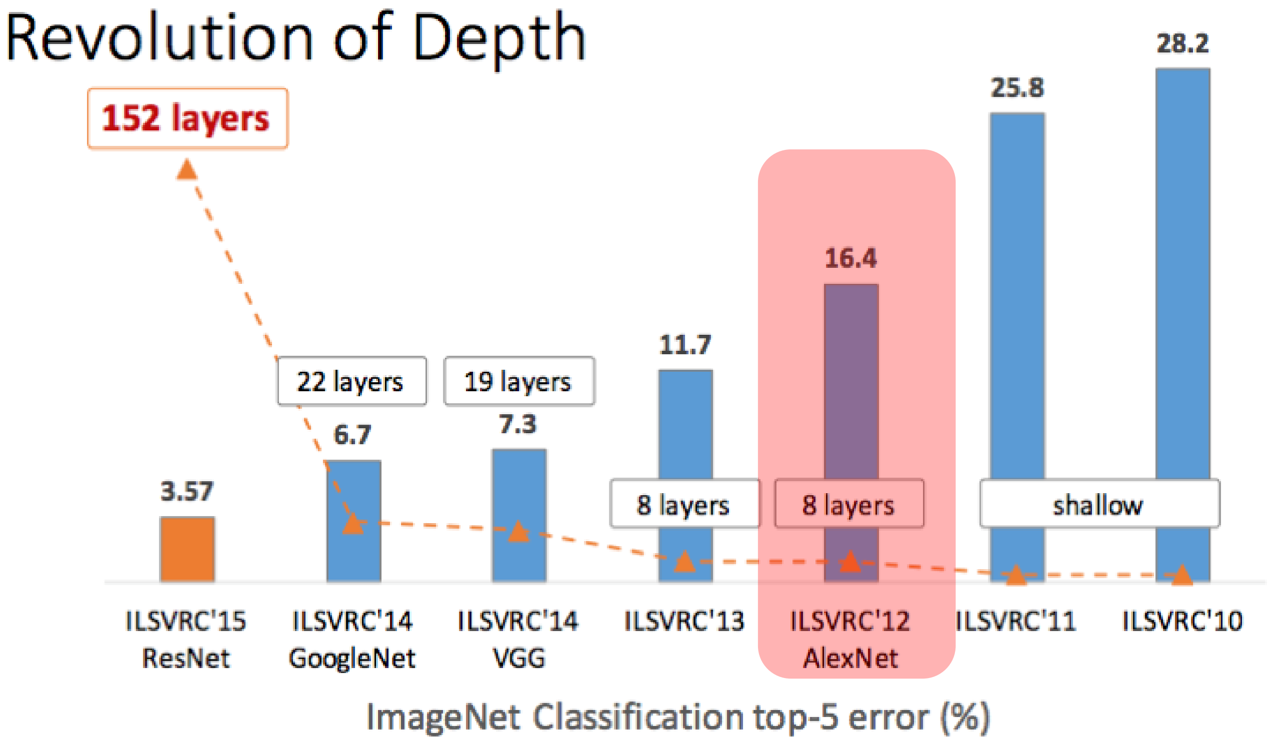
Lecun Yann
Director of AI Research, Facebook

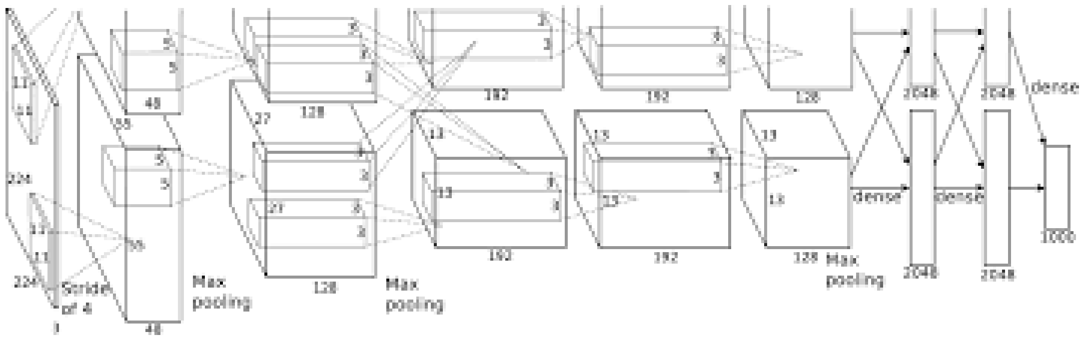


http://yann.lecun.com/

History of CNN – 2012

- Alex Krizhevsky et al, "*ImageNet Classification with Deep Convolutional Neural Networks*" (2012)
  - Win the imageNet competition: annual Olympics of computer vision with astounding results compared to previously existing approaches (26% to 15%).

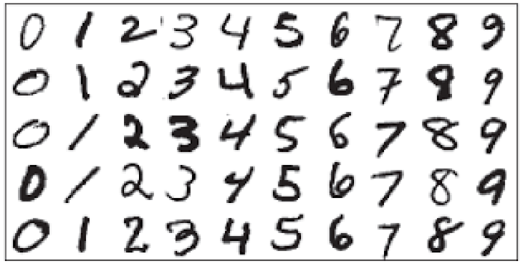


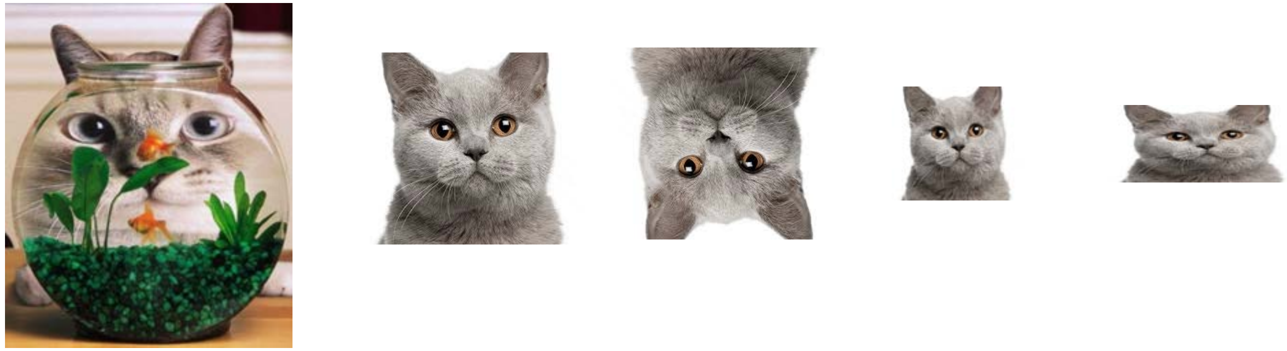


https://medium.com/@Lidinwise/the-revolution-of-depth-facf174924f5

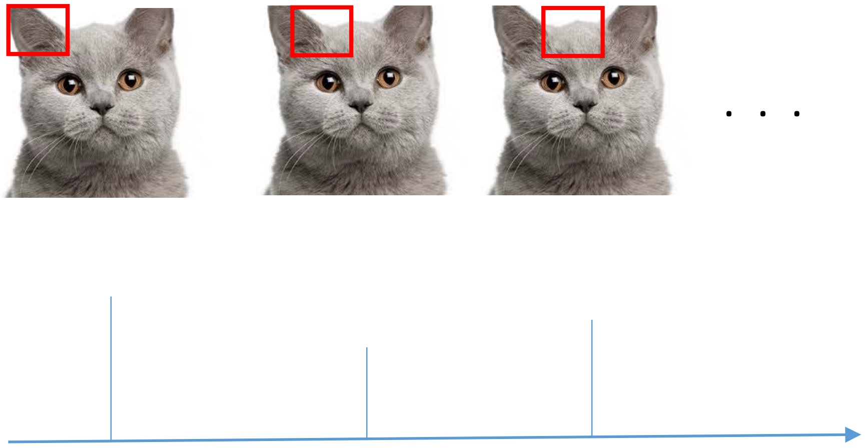
Why Convolutional Neural Networks (CNN)?

- Invariance to rotation, transformation, size, etc.



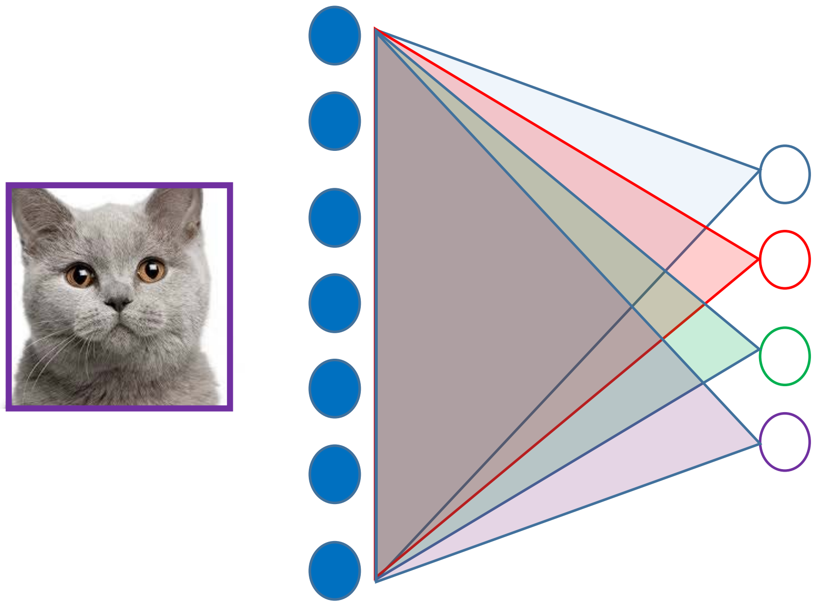


- Three architectural ideas of CNN
  1. Local receptive fields
  2. Weight sharing
  3. Spatial or temporal subsampling



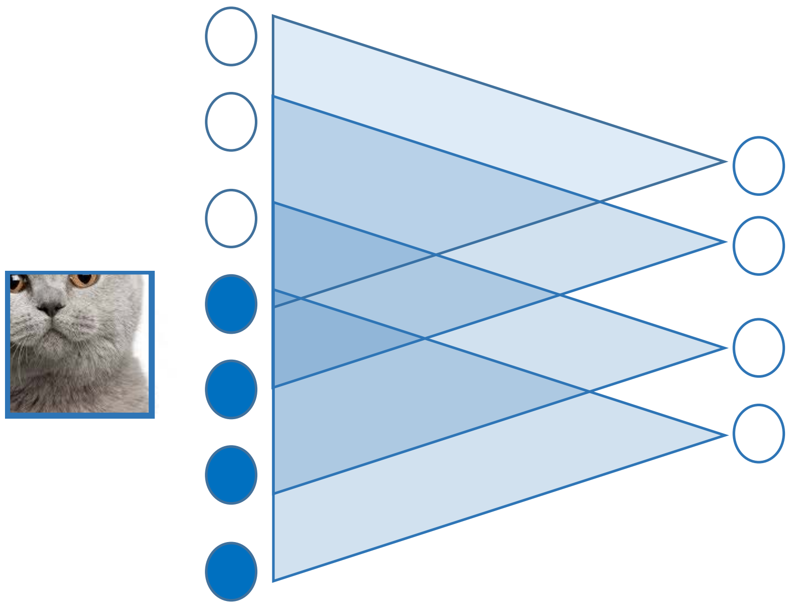
An idea of CNN

- A node takes responsibility for a portion of input data.



Fully connected Layer

- 4 x 7 = 28 weights

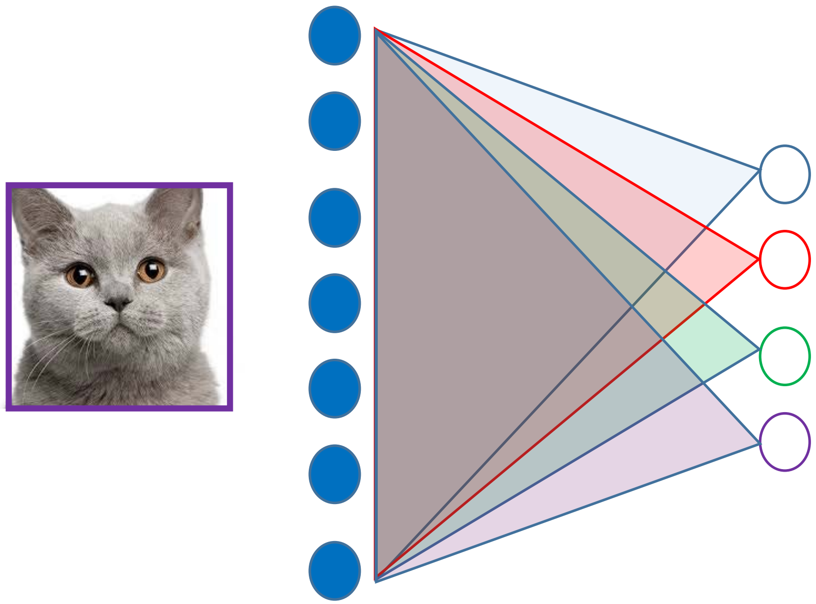


Convolutional Layer

- 4 weights

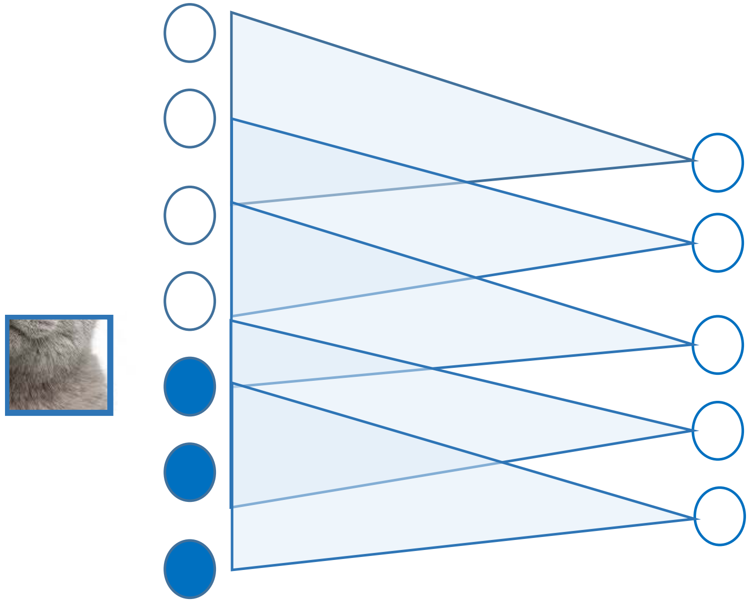
An idea of CNN

- A node takes responsibility for a portion of input data.



Fully connected Layer

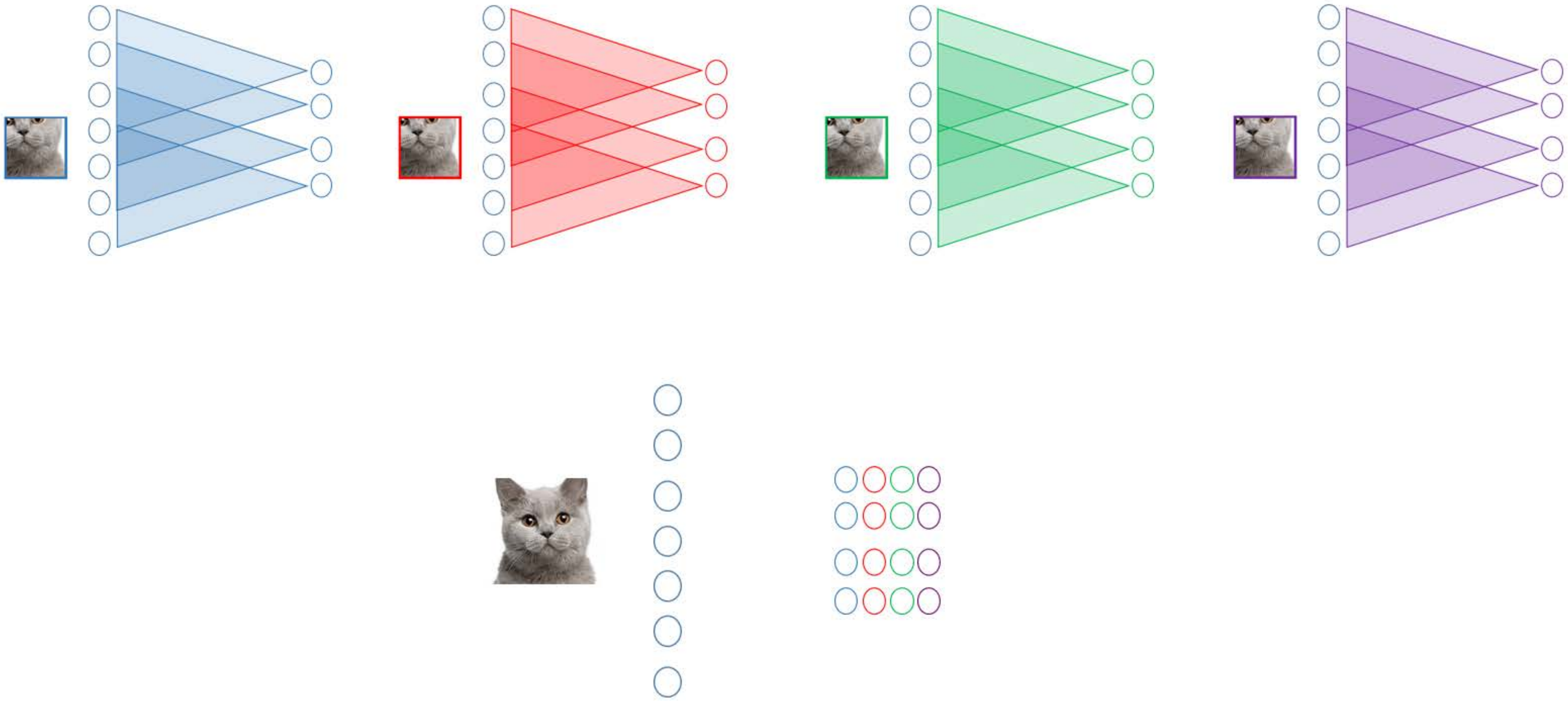
- 4 x 7 = 28 weights



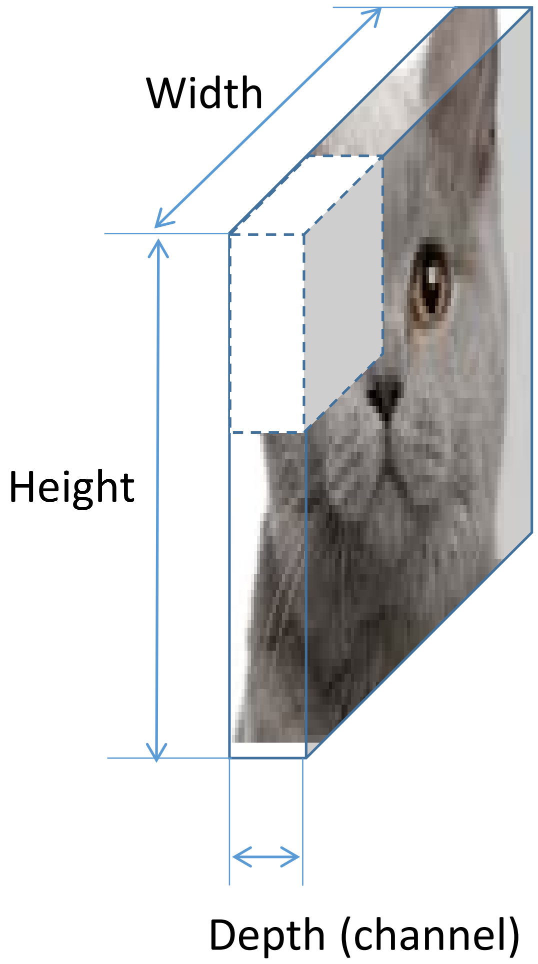
Convolutional Layer

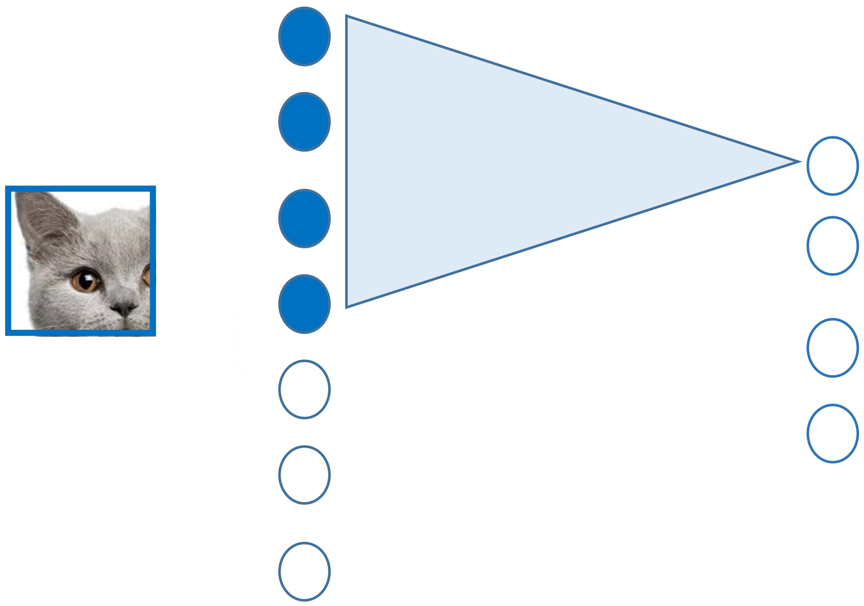
- 3 weights

Convolution Layer tends to produce multiple results

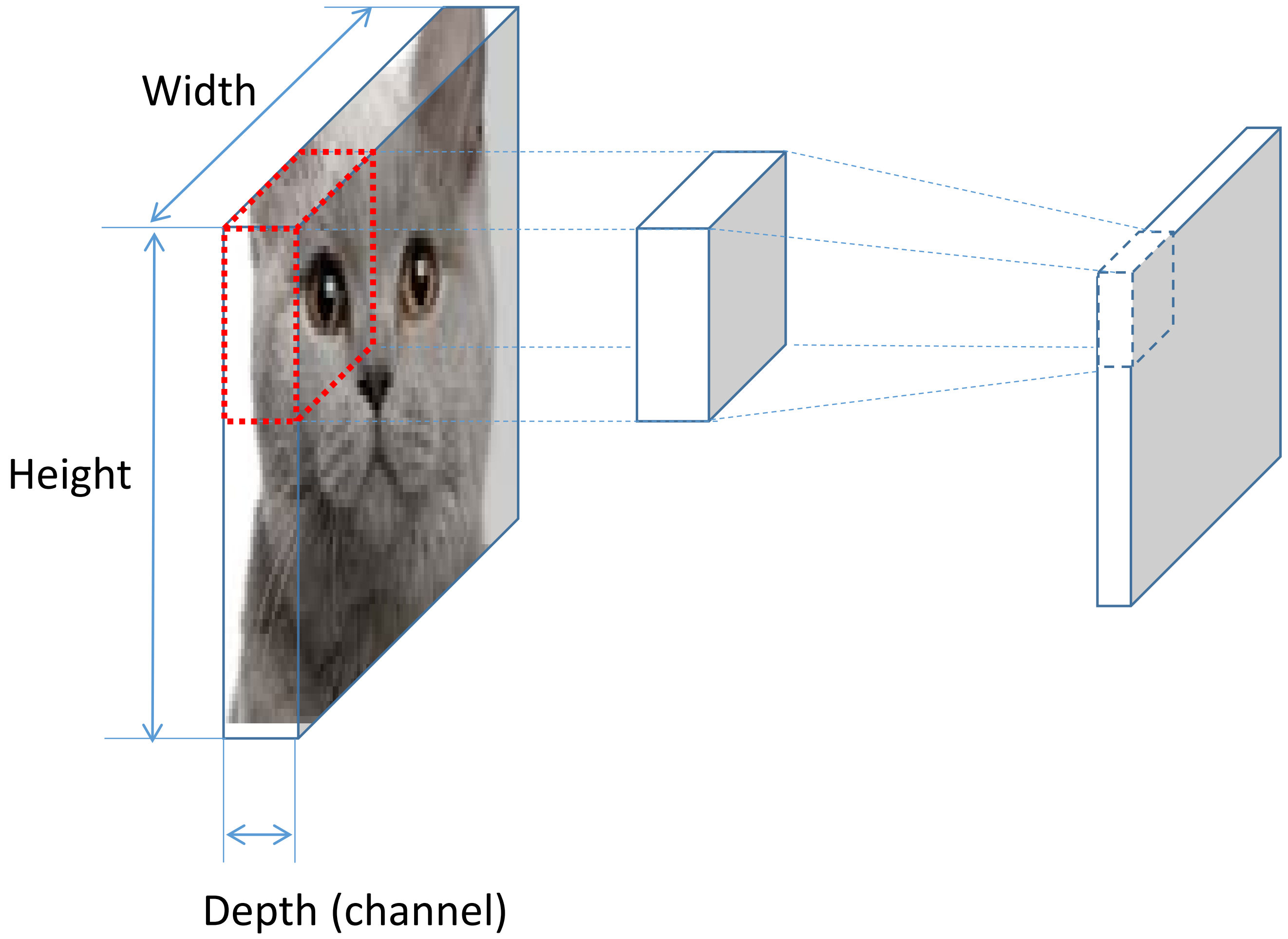


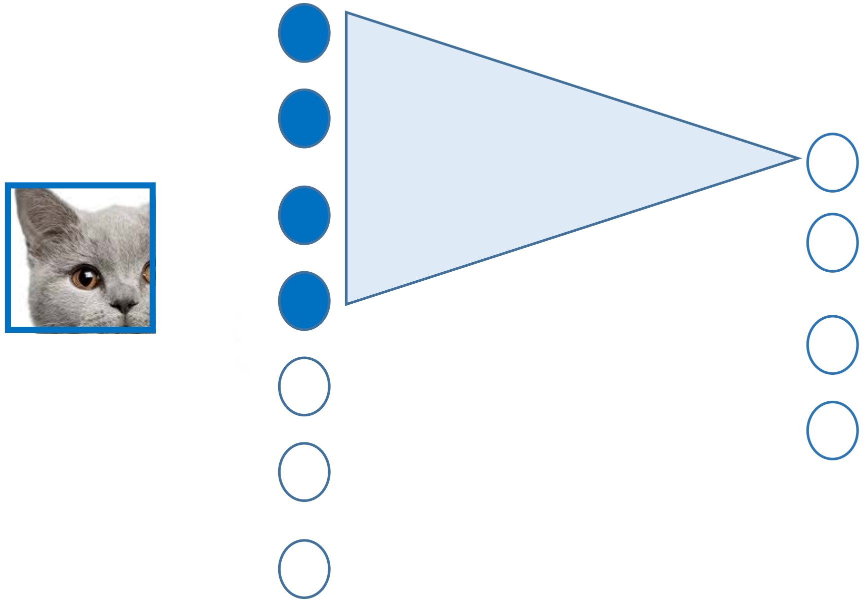
2-D representation vs 1-D representation



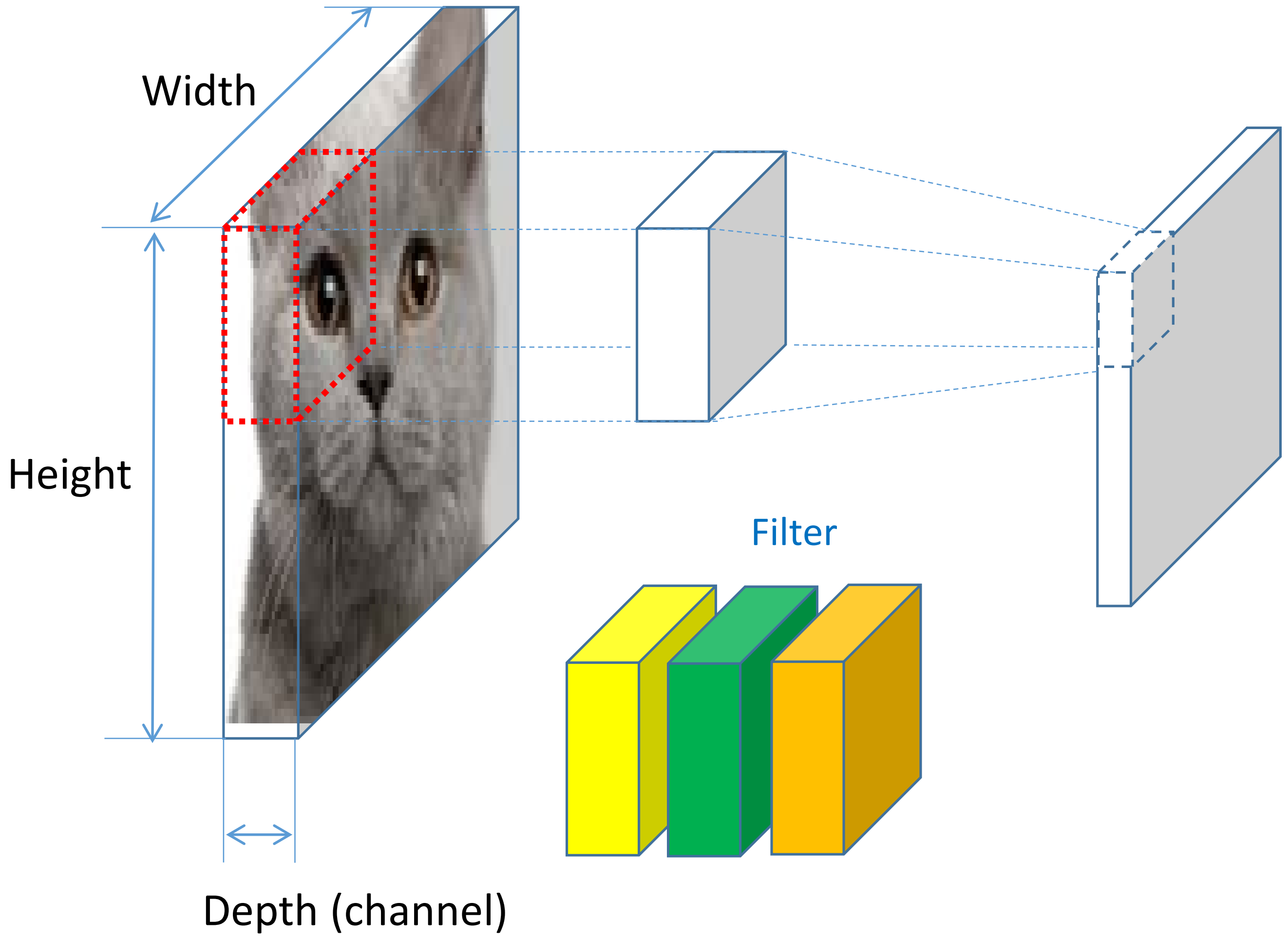


2-D representation vs 1-D representation

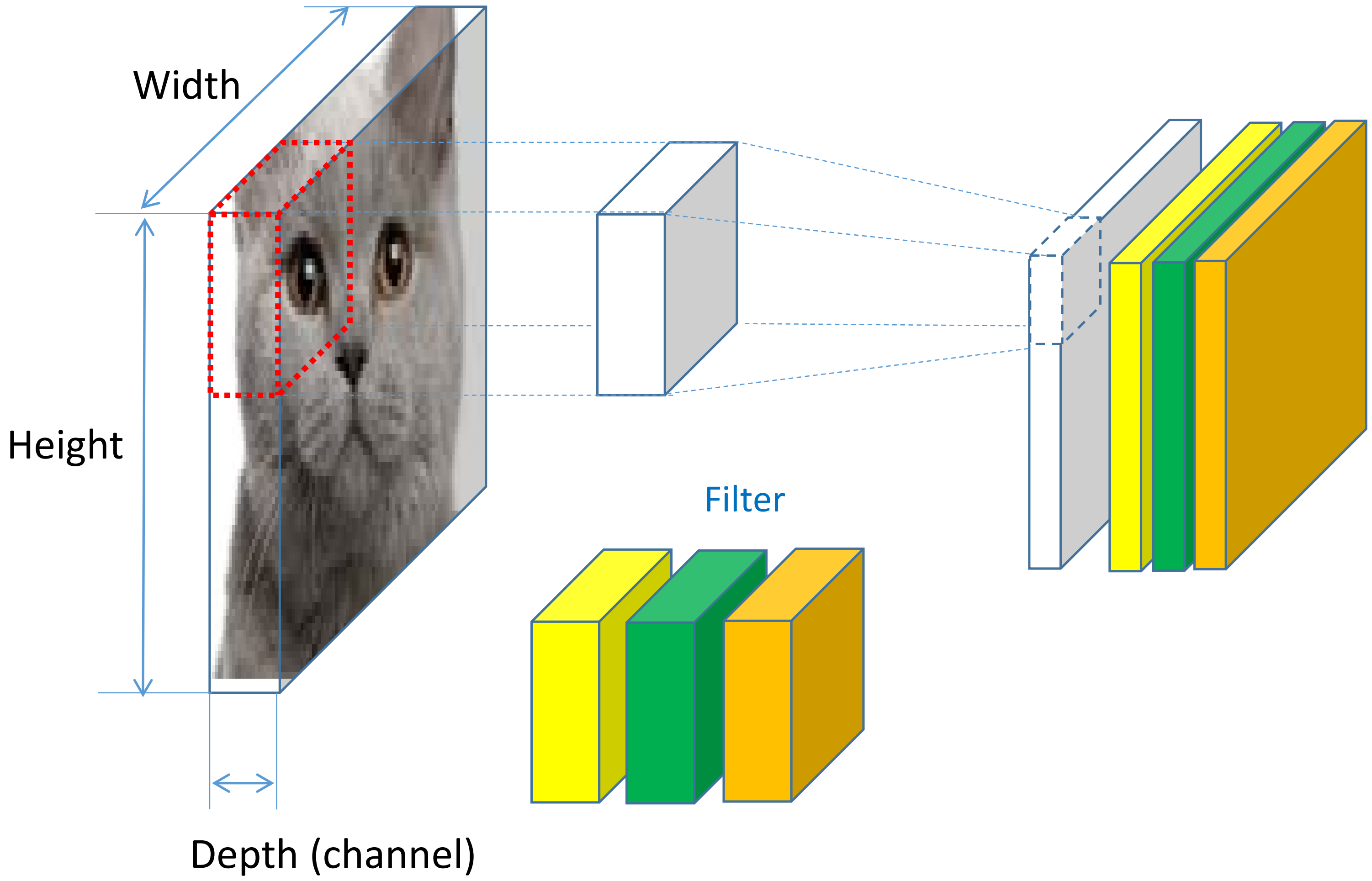




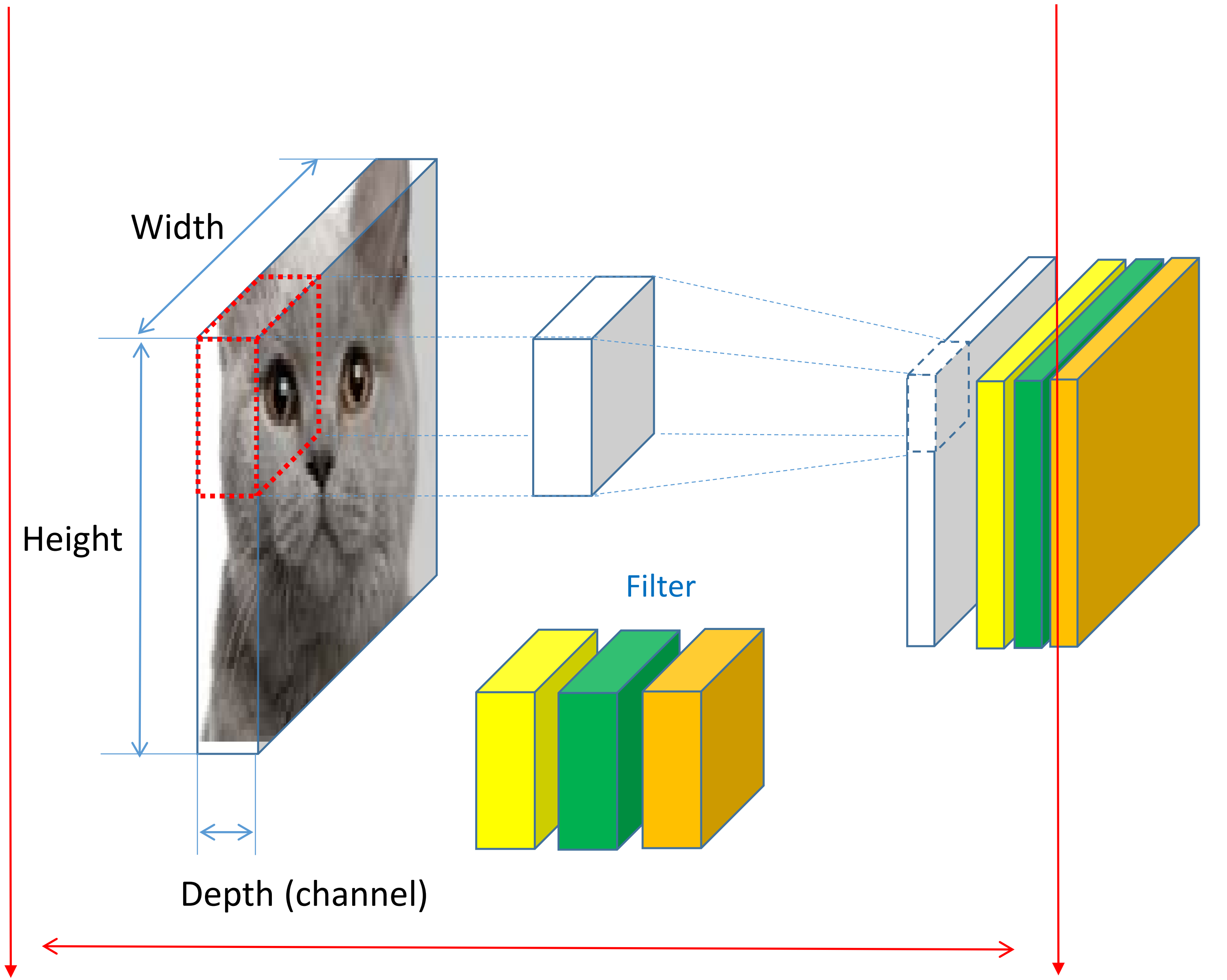
2-D representation of CNN: How does it work?



2-D representation of CNN: How does it work?

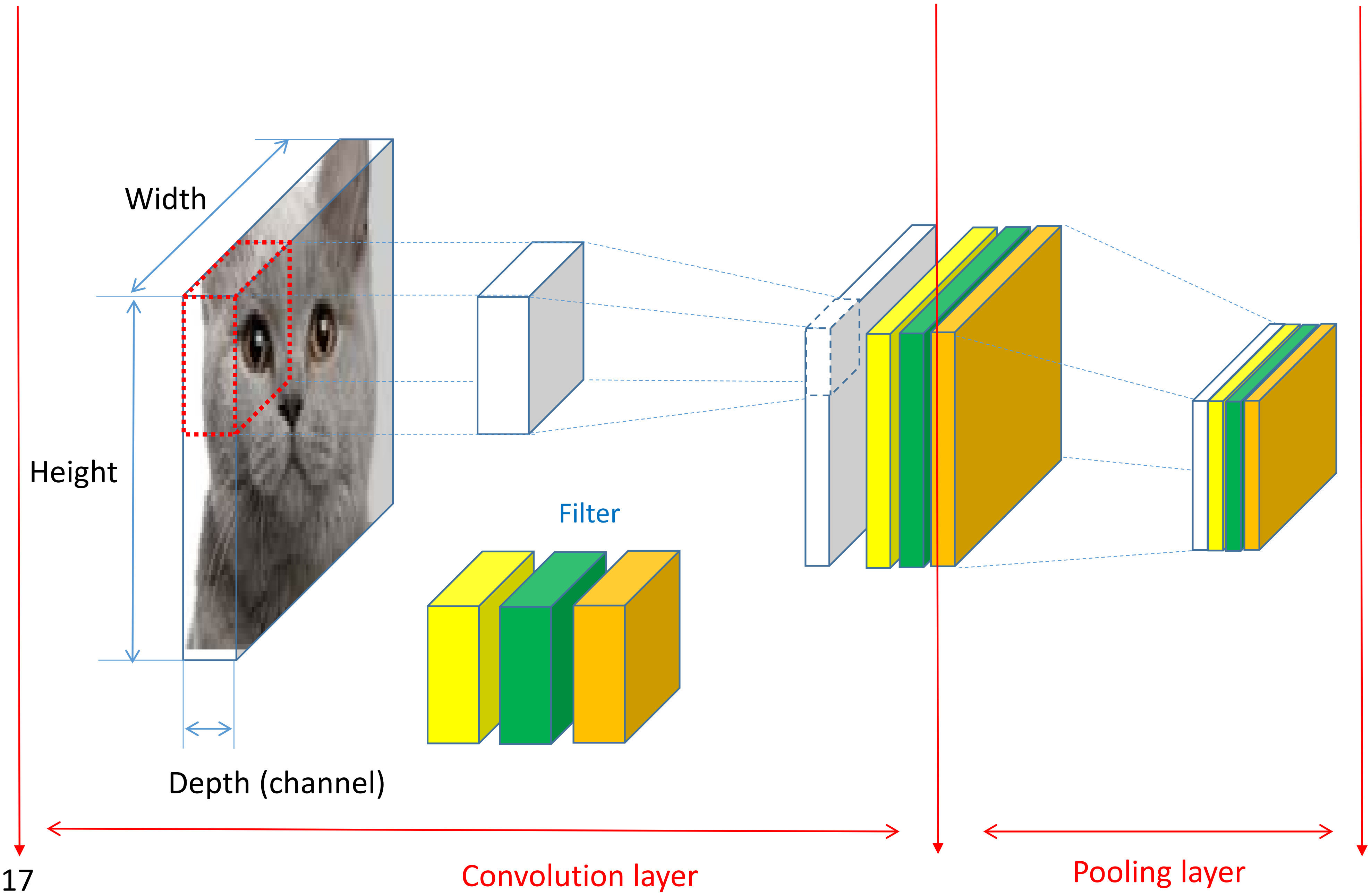


2-D representation of CNN: How does it work?

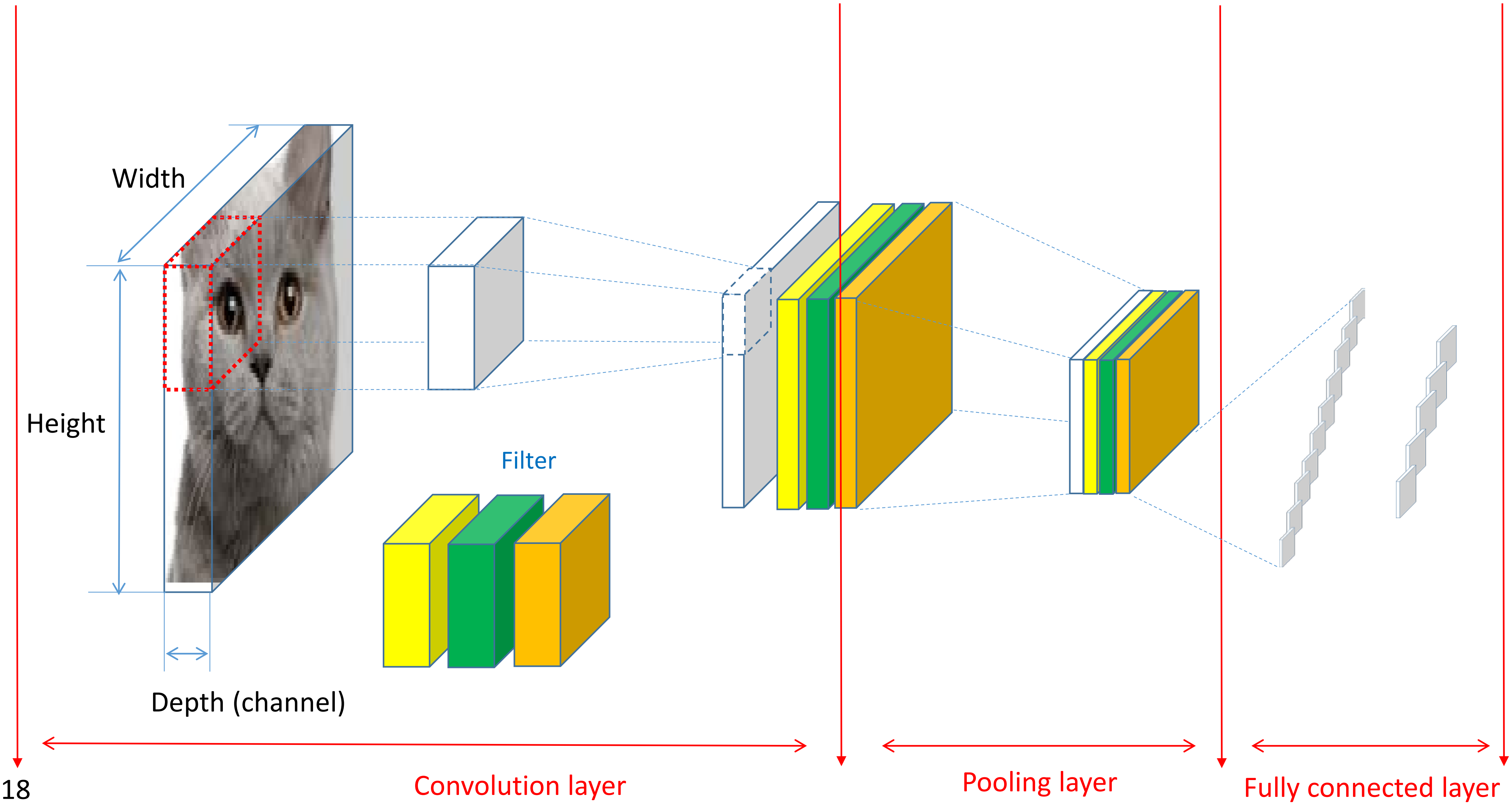


Convolution layer

2-D representation of CNN: How does it work?

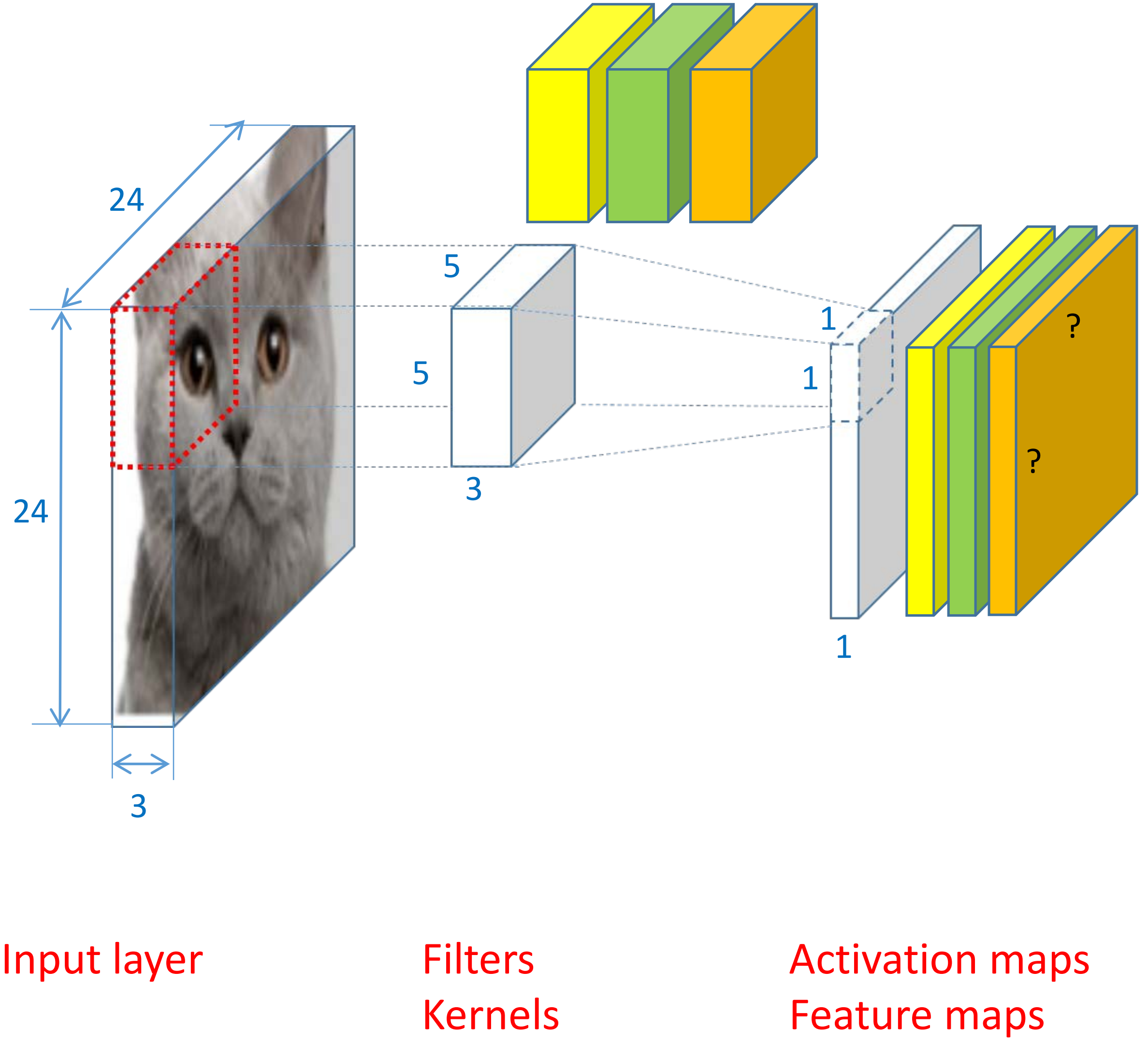


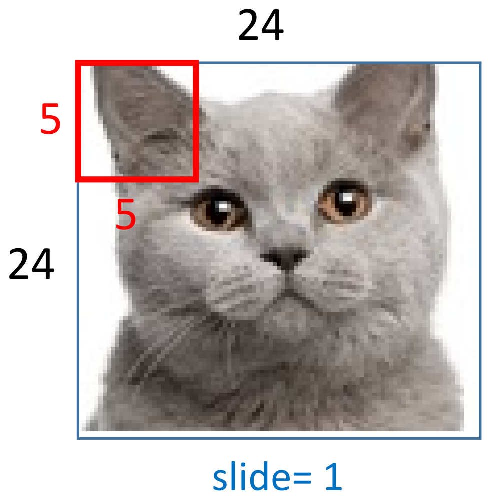
2-D representation of CNN: How does it work?



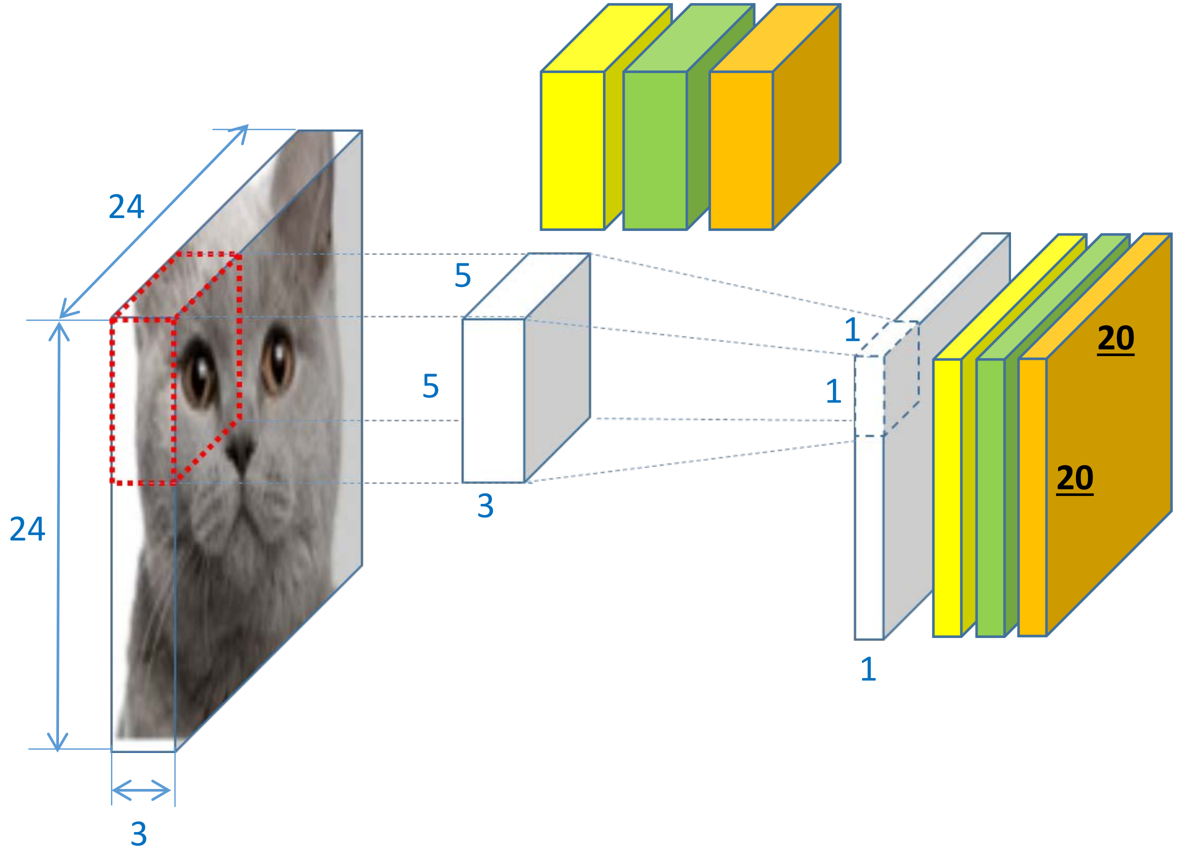
Convolution Layer

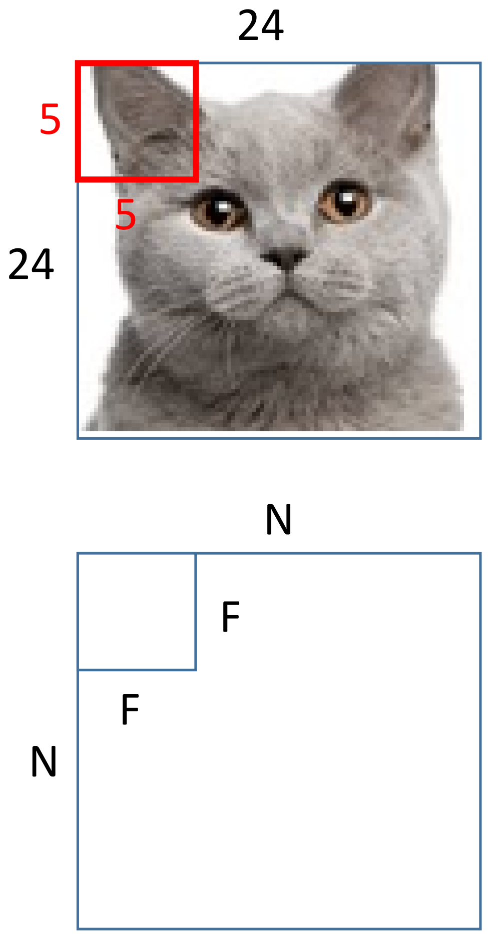
Convolutional Layer





Convolutional Layer





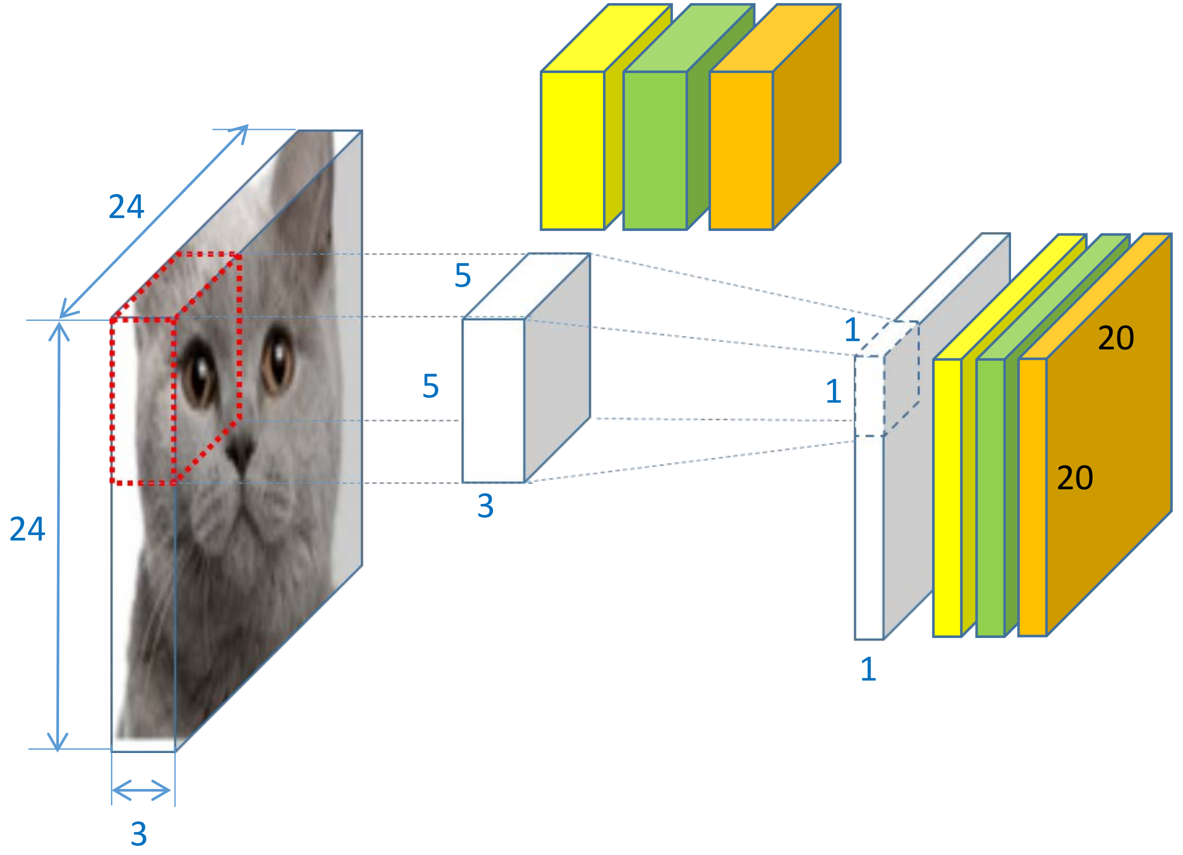
Input layer

Filters
Kernels

Activation maps
Feature maps

Size of activation map $= \dfrac{(N-F)}{stride} + 1$

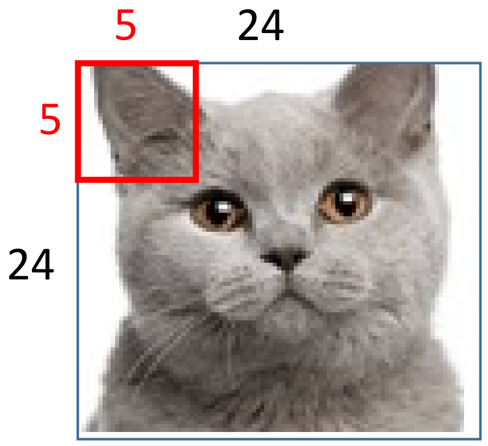
Convolutional Layer

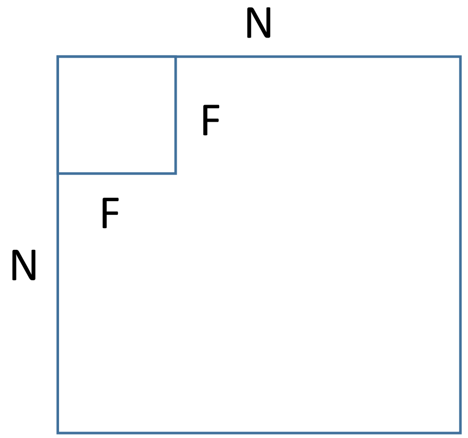


Input layer

Filters
Kernels

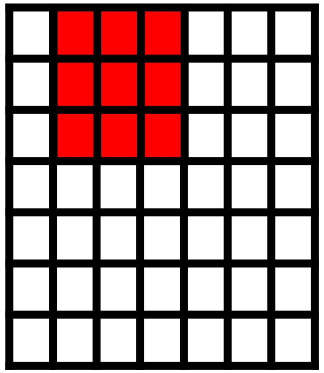
Activation maps
Feature maps





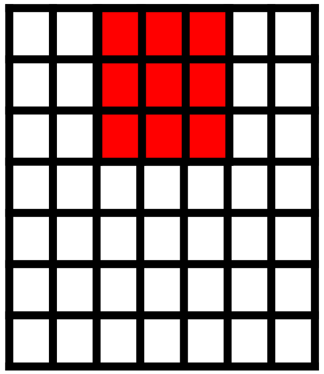
Size of activation map $= \dfrac{(N - F)}{stride} + 1$

stride 1



Activation map
(5 x 5 matrix)

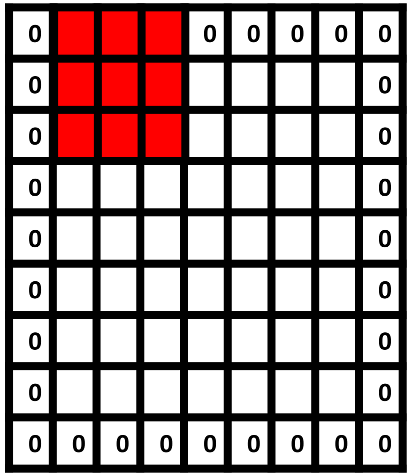
stride 2



Activation map
(3 x 3 matrix)

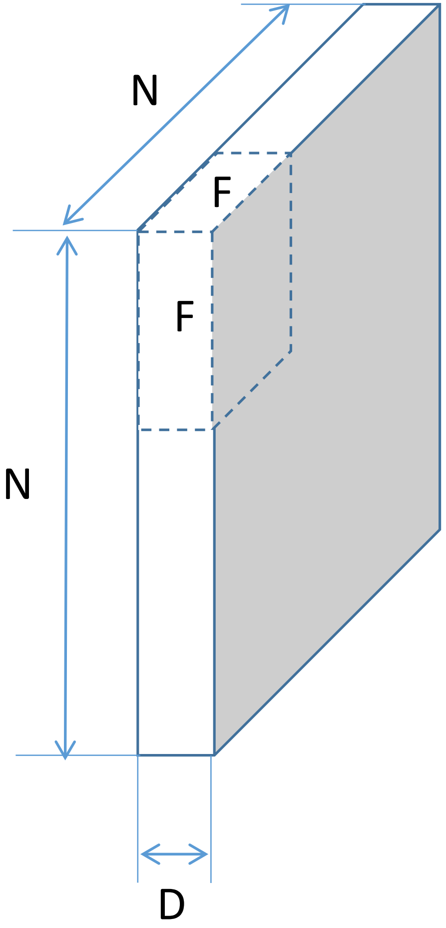
Convolutional Layer

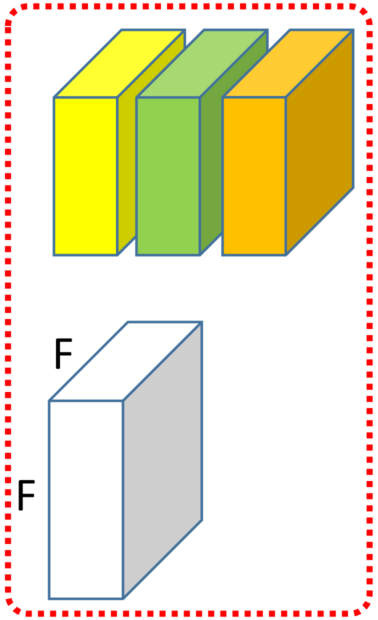
Hyper parameters	Symbols
Number of filters	K
Size of the filter	F
Stride	S
Padding	P



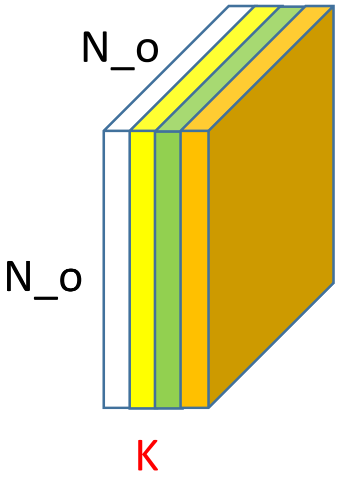
- Padding: P=1
- Stride: S=1
- activation map becomes is 7 x 7 matrix

- Padding aims to maintain the original dimension of the original data.





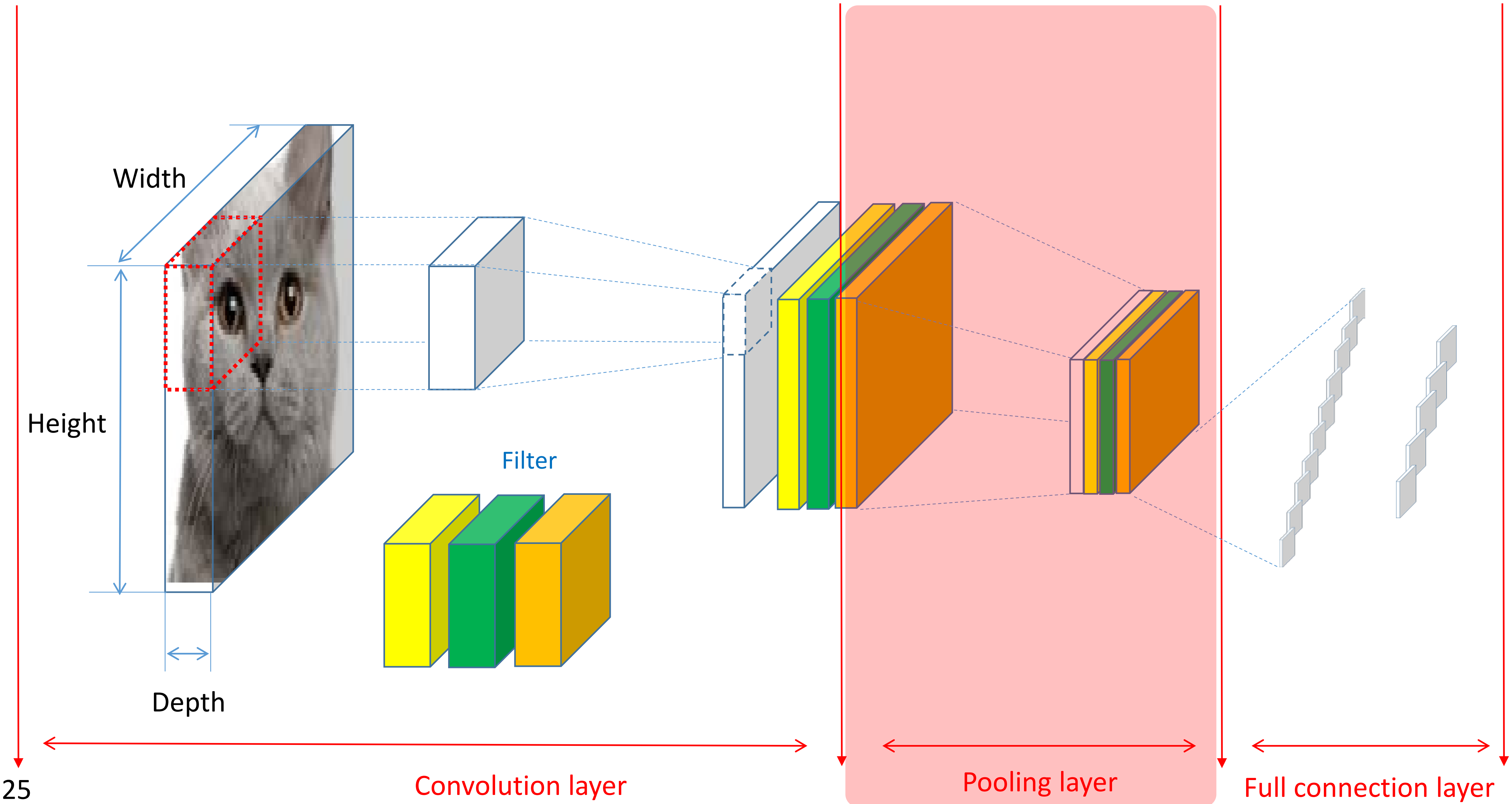
Number of filters: K



$$N\_o = \frac{(N - F + 2P)}{S} + 1$$

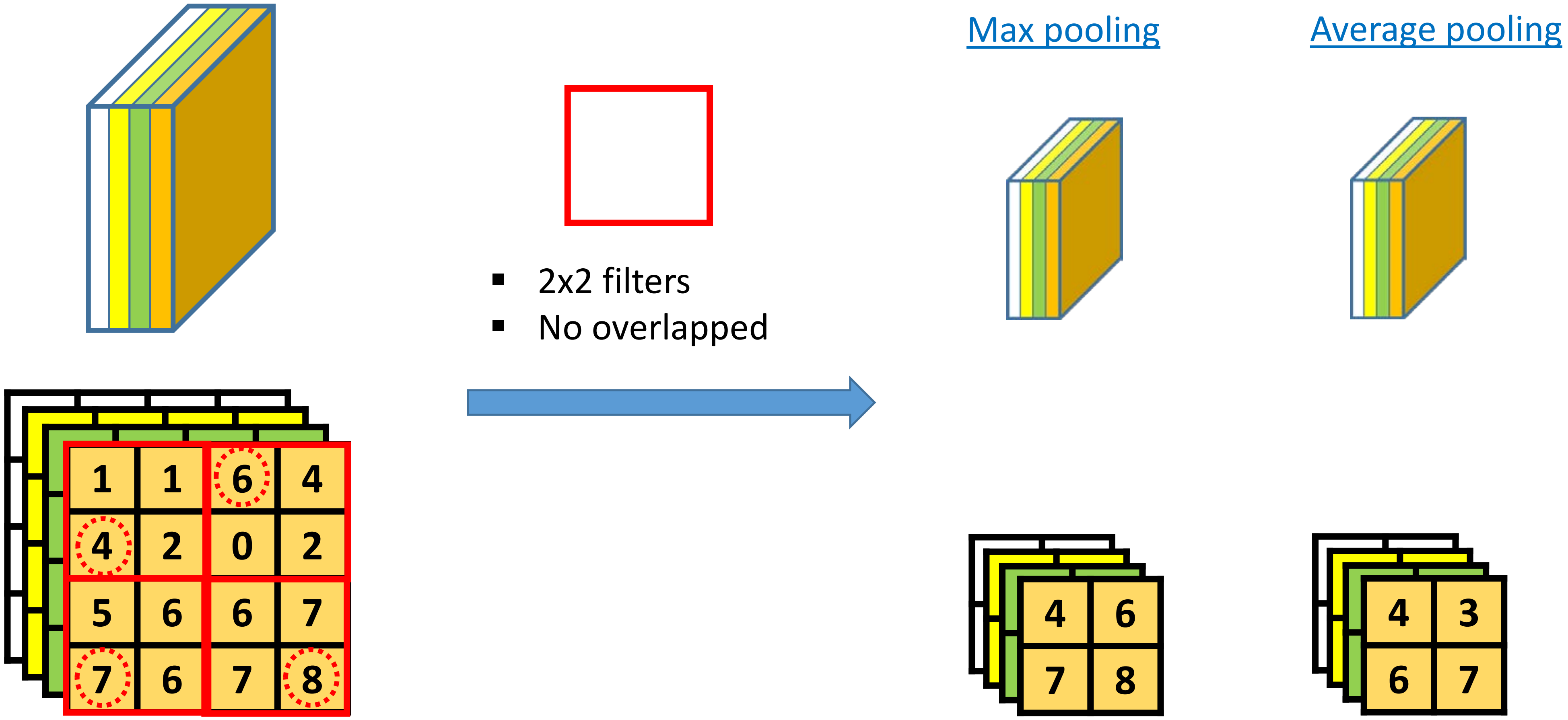
Pooling Layer

2-D representation of CNN: How does it work?



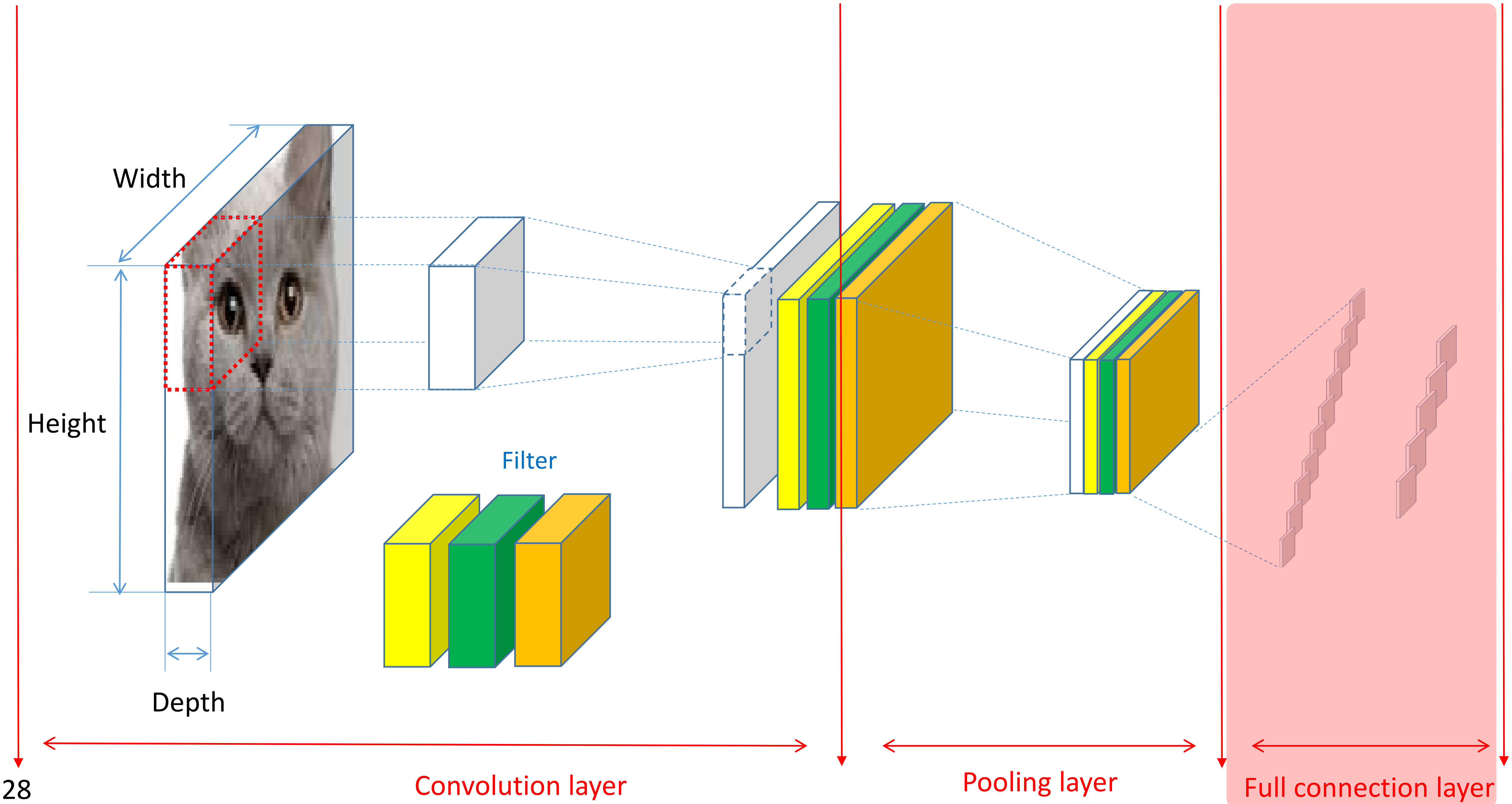
Pooling Layer

- This layer aims to reduce the dimension of each activation map.

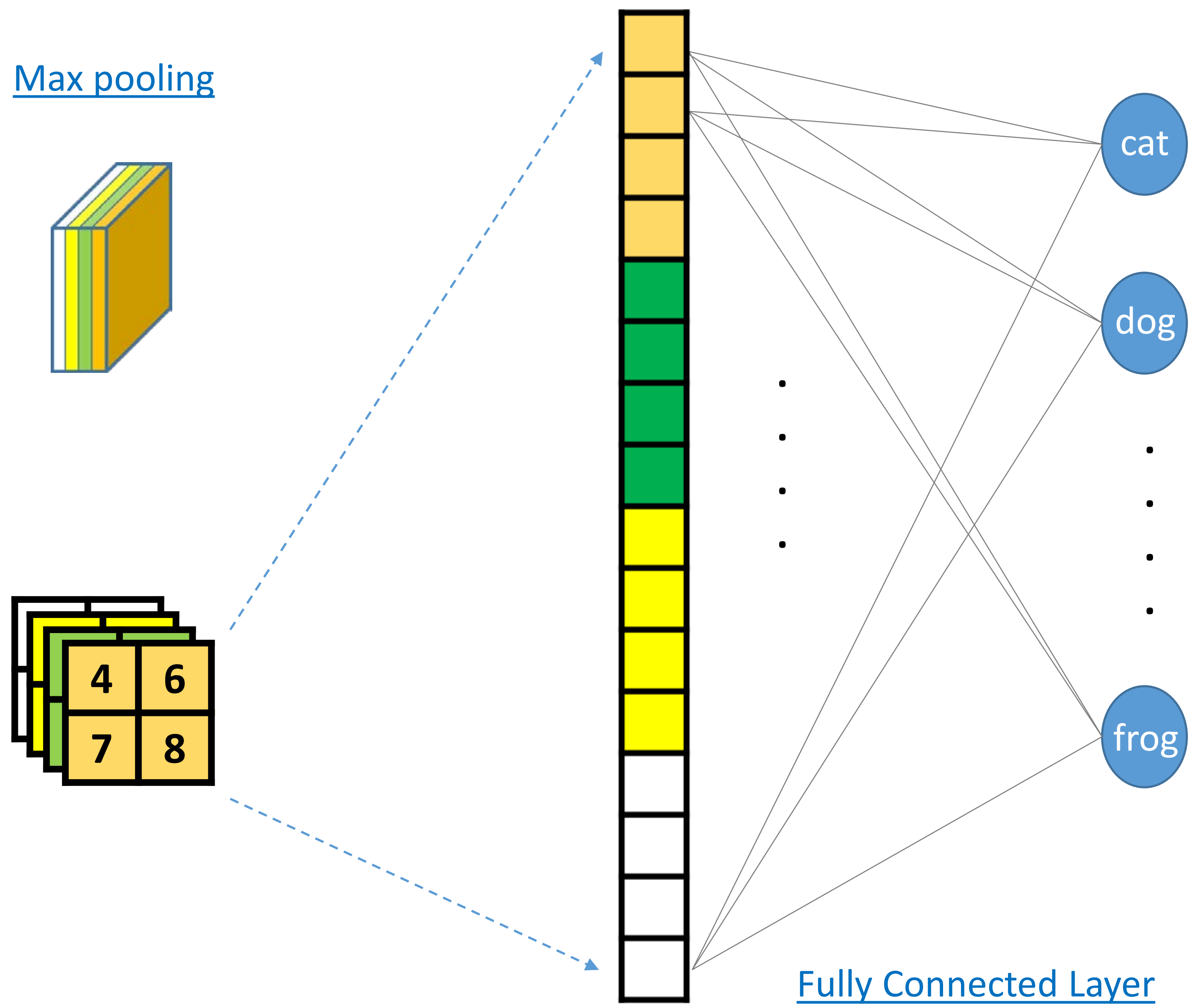


Fully Connected Layer

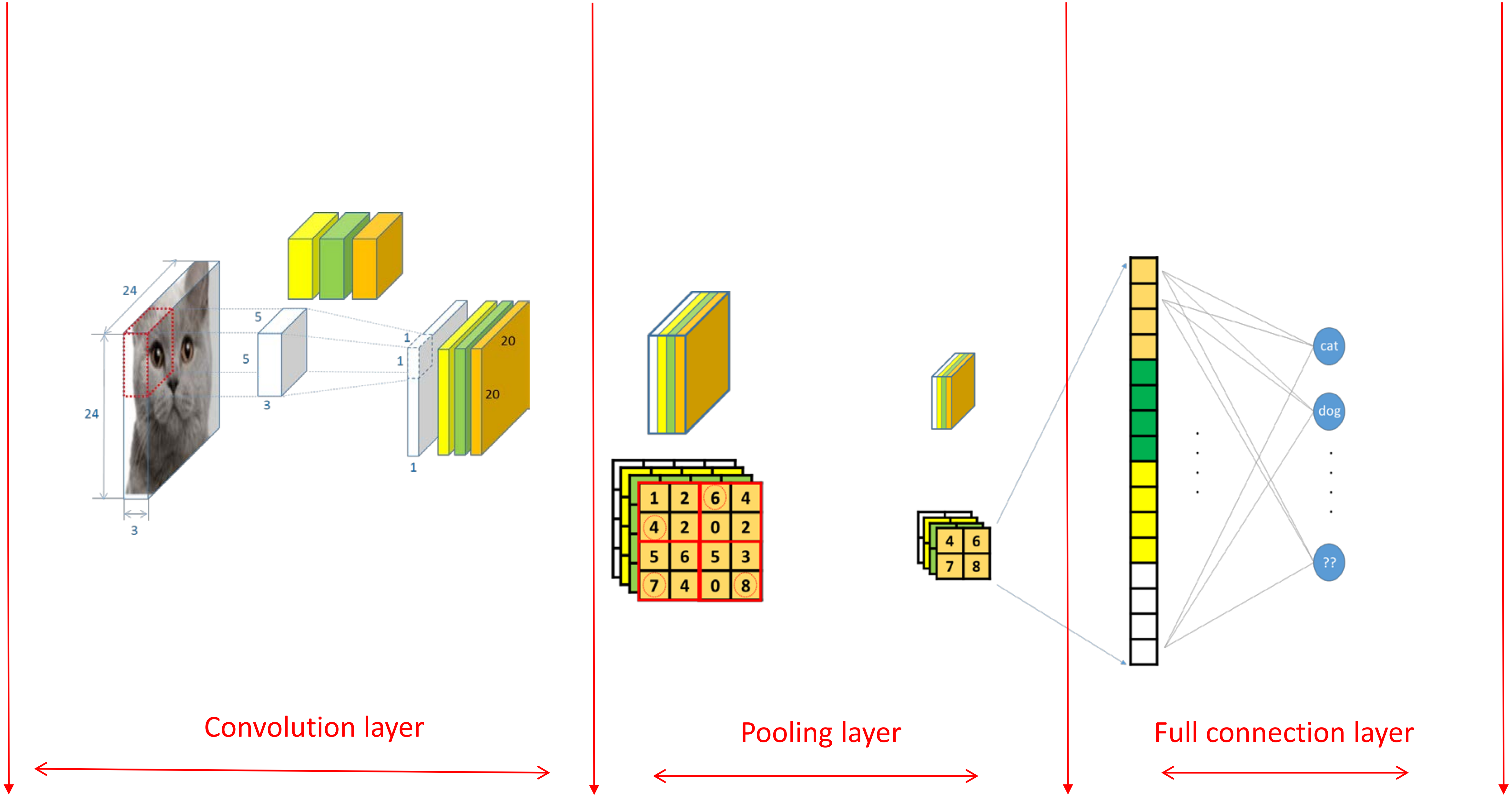
2-D representation of CNN: How does it work?



Fully Connected Layer



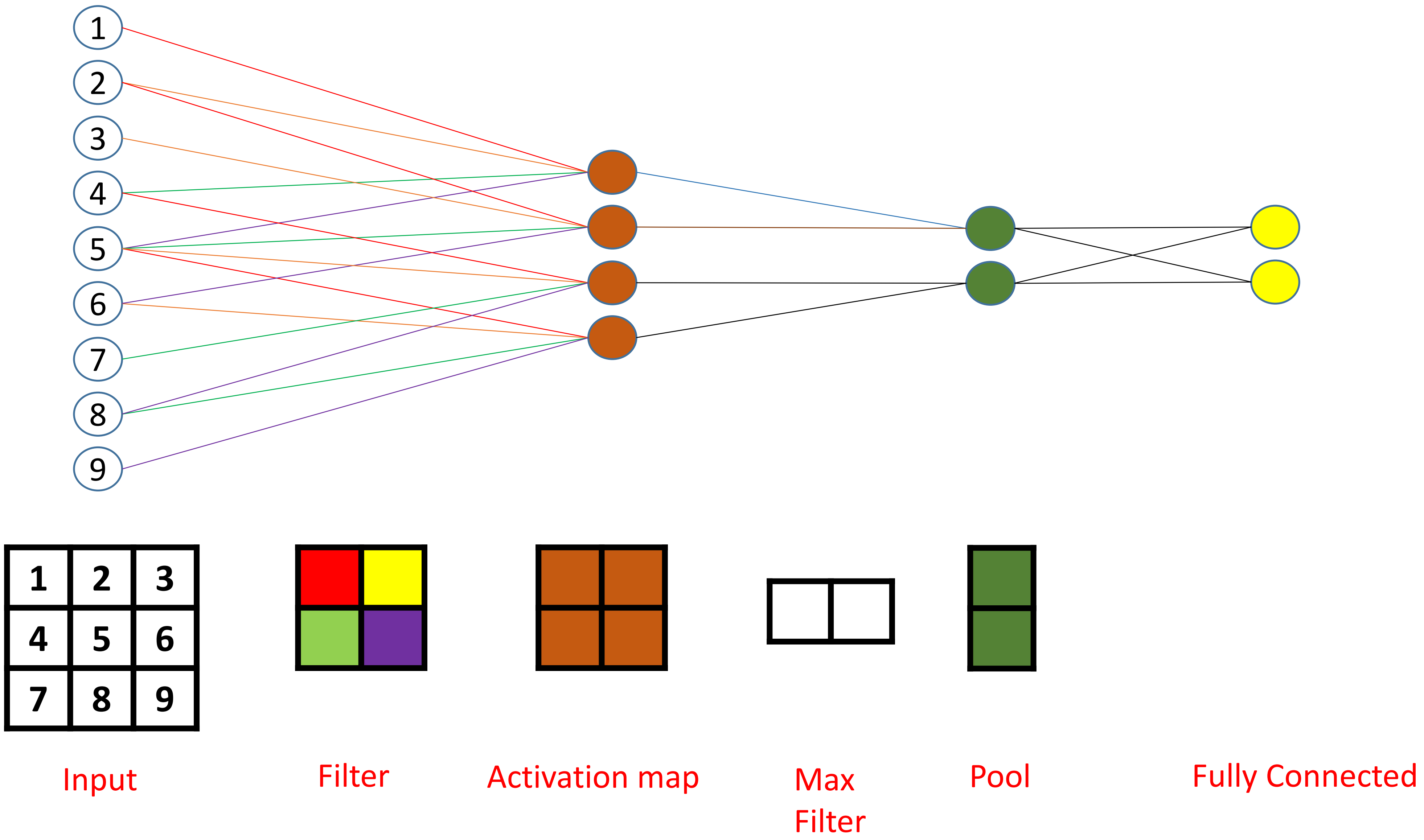
Summary



24
24
3
5
5
3
1
1
1
20
20
1 2 6 4
4 2 0 2
5 6 5 3
7 4 0 8
4 6
7 8
cat
dog
??
Convolution layer
Pooling layer
Full connection layer

Backpropagation

Backpropagation

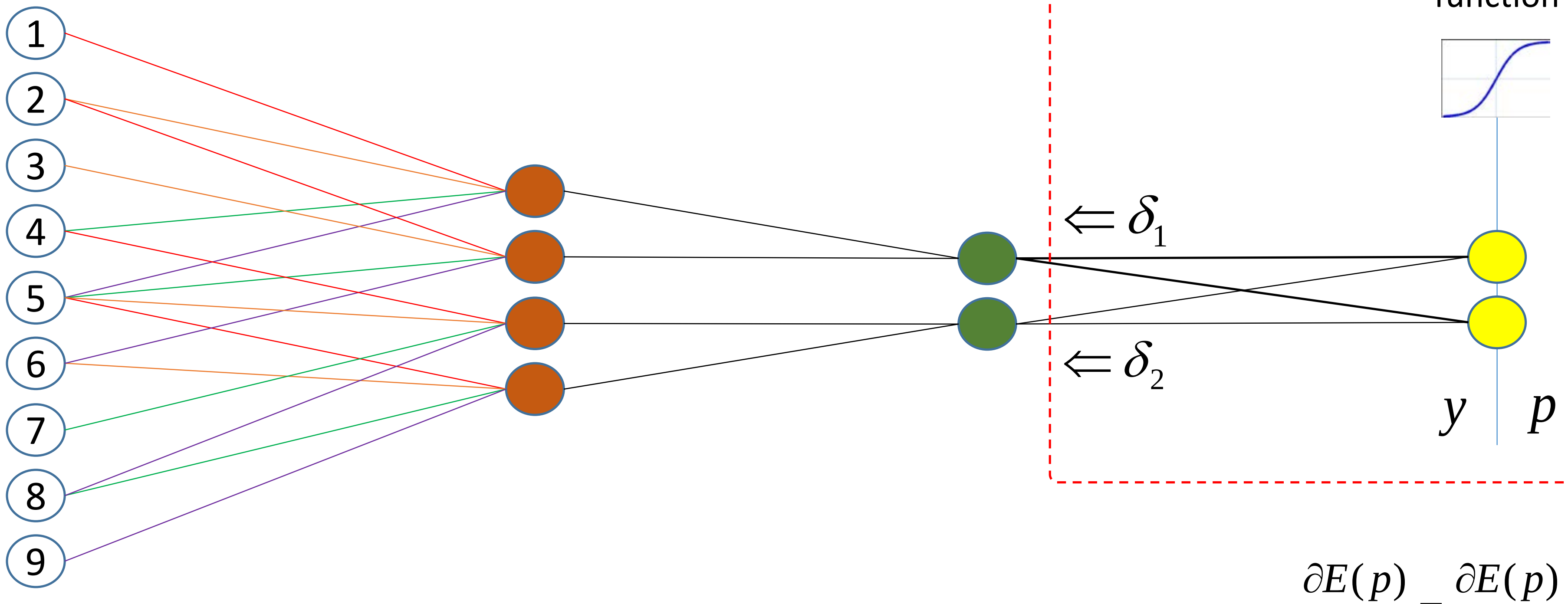


Backpropagation: Fully Connected Layer

Softmax function and Cross Entropy Error

$$p(y_i) = \frac{e^{y_i}}{\sum_k e^{y_k}}$$

SoftMax function



- t_i (Label)=1 or 0

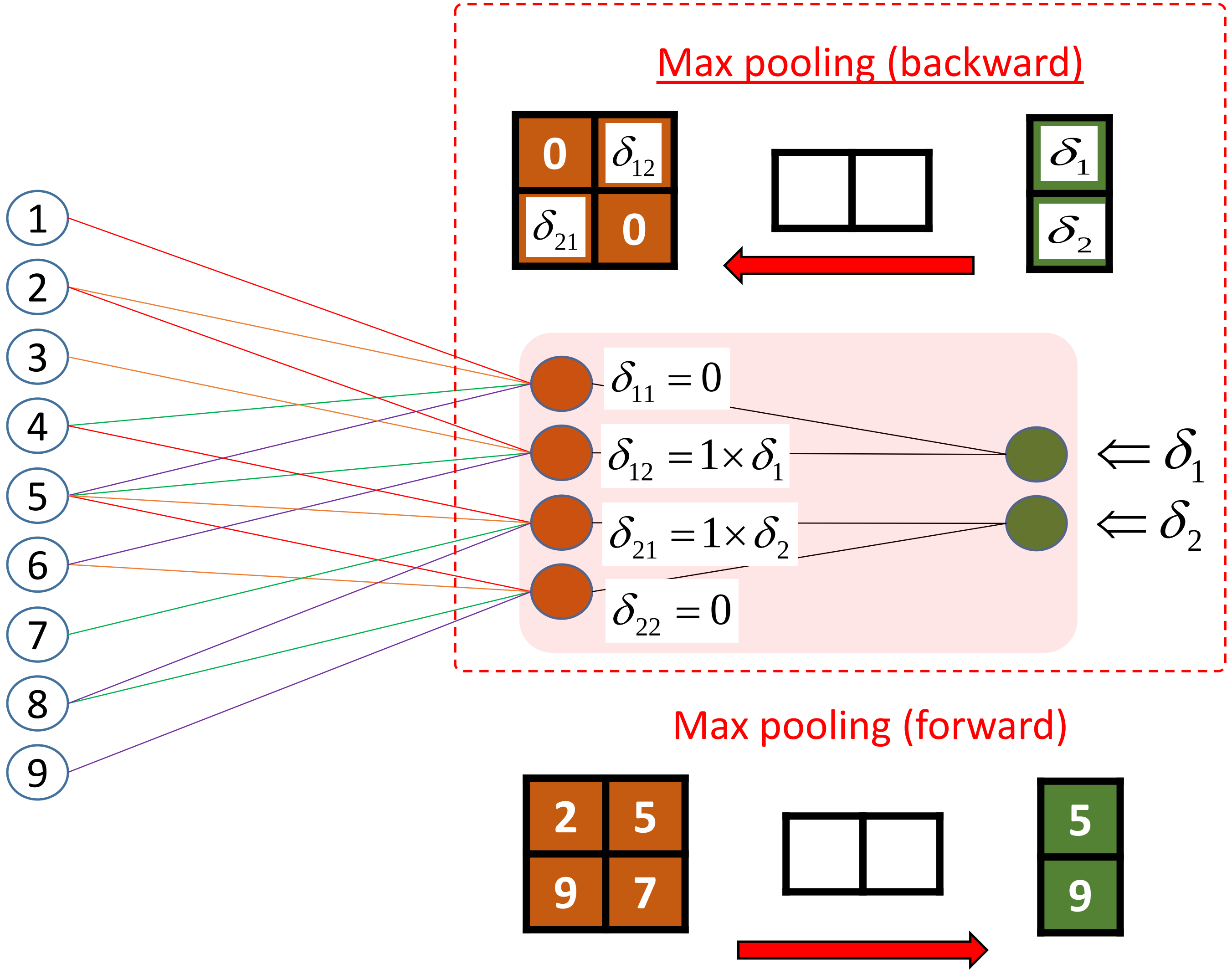
$$E(p) = -\sum_{k}^{\#class} t_k \log p_k$$

Cross entropy Loss function

$$\frac{\partial E(p)}{\partial y_i} = \frac{\partial E(p)}{\partial p} \times \frac{\partial p}{\partial y} = p_i - t_i$$

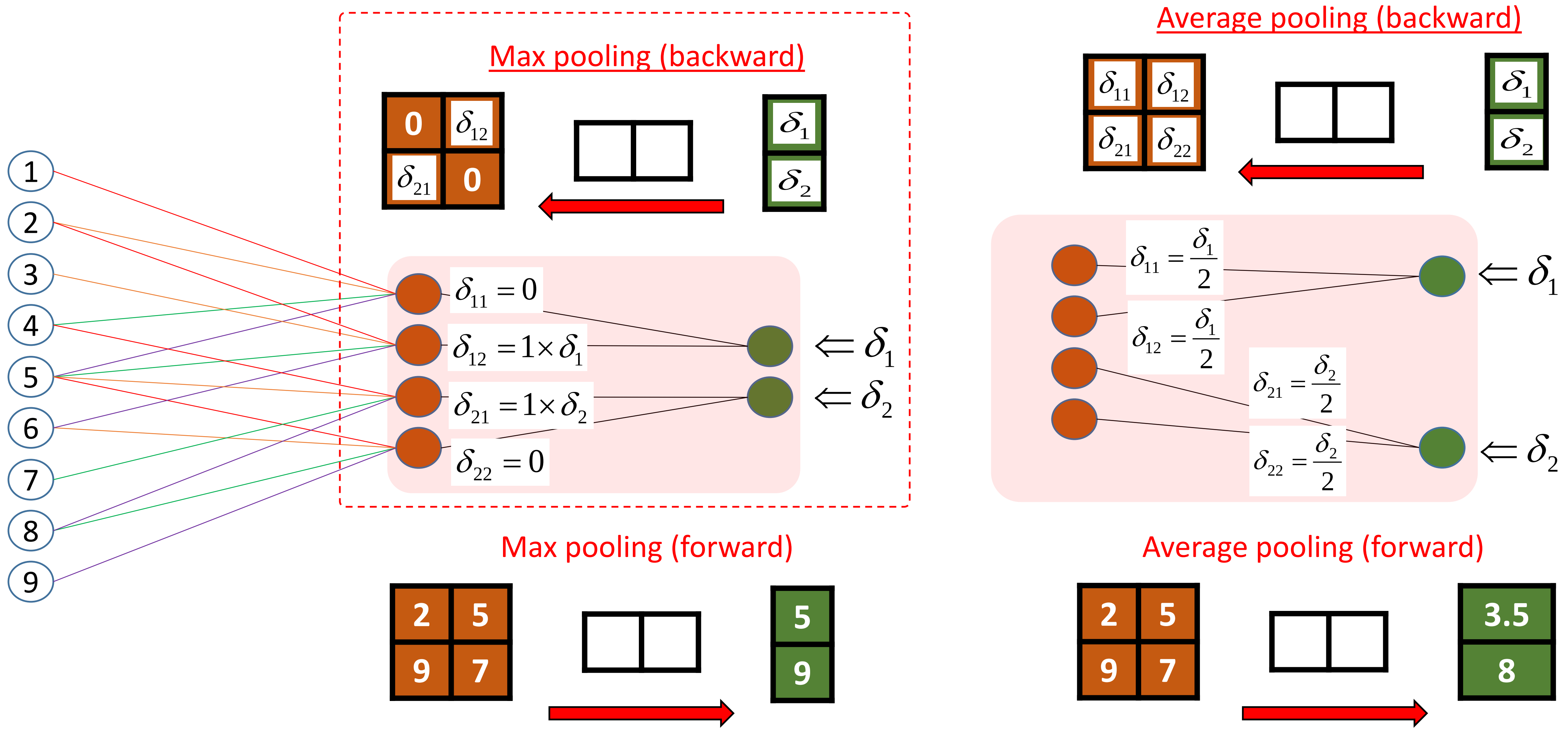
Q12

Backpropagation: Pooling Layer (Max pooling)



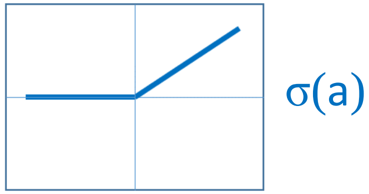
- δ_{12} is set to δ_1
- δ_{21} is set to δ_2

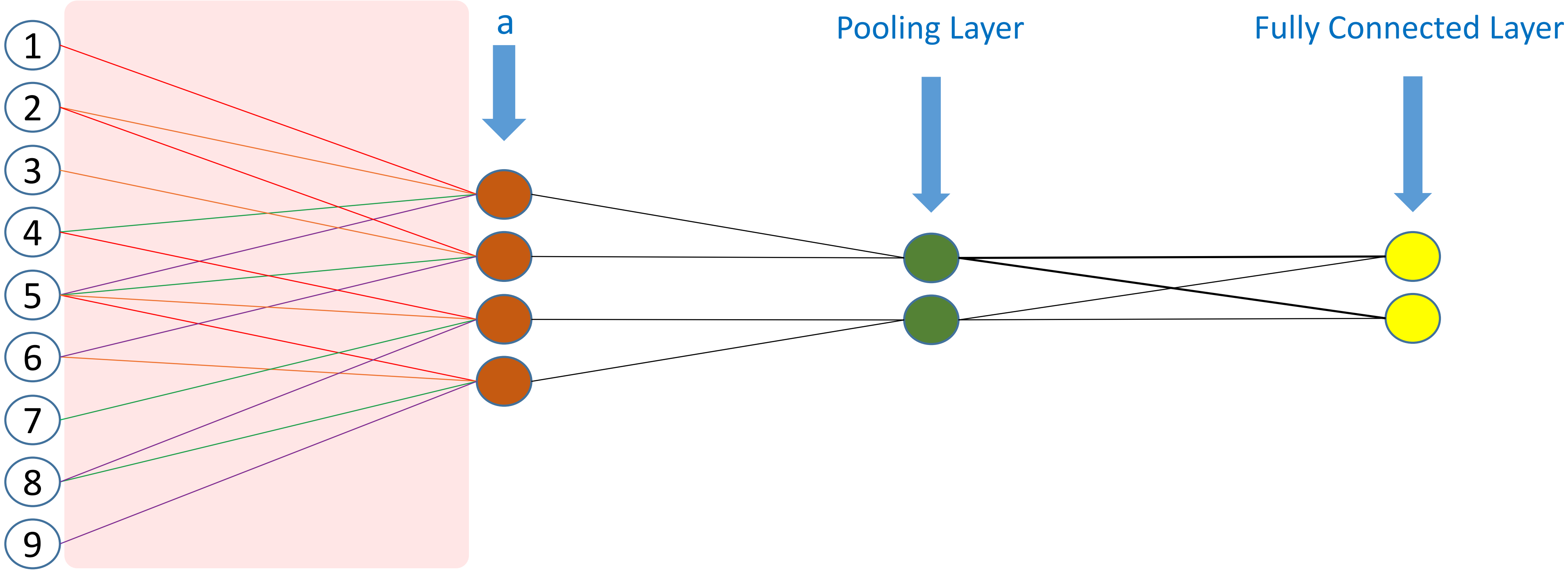
Backpropagation: Pooling Layer (Average pooling)



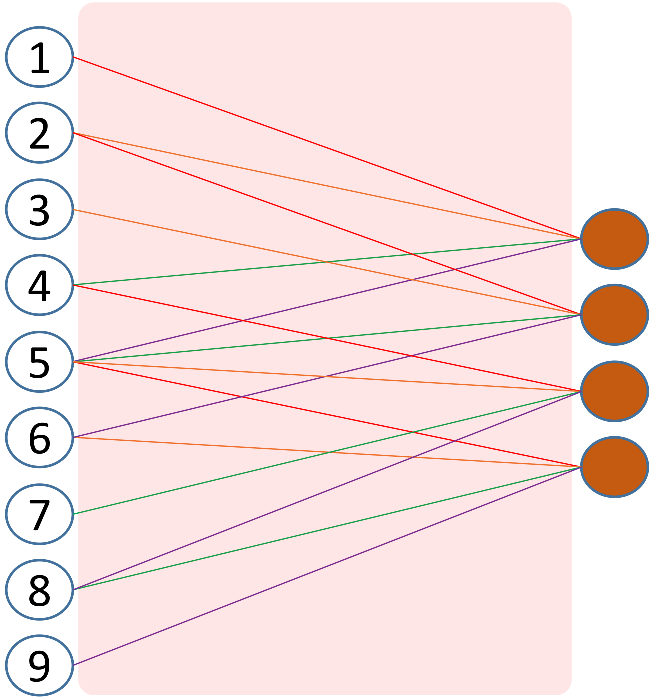
Backpropagation: Convolution Layer

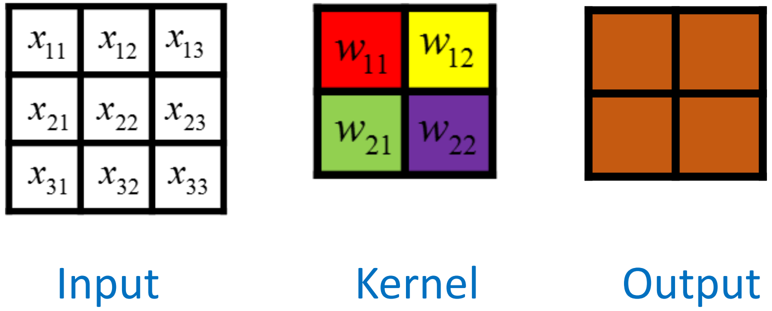
$$\frac{\partial \sigma(a)}{\partial a} = \begin{cases} C & \text{If a > 0} \\ 0 & \text{Otherwise} \end{cases}$$





Backpropagation: Convolution Layer





Backpropagation: Convolution Layer

- Convolution

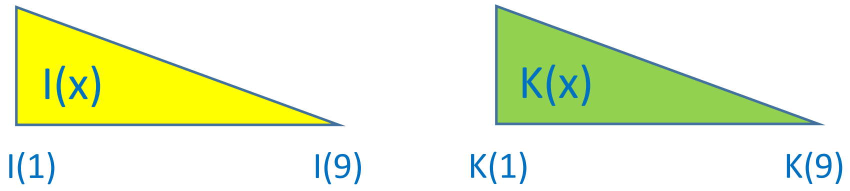
$$(I * K)_{ij} = \sum_{m=0} \sum_{n=0} I(m, n) K(i-m, \ j-n)$$

Individual elements in both functions, located in the opposite index, are multiplied and added.

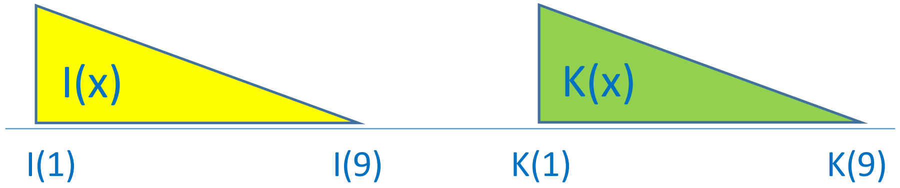
- Cross-correlation

$$(I \circ K)_{ij} = \sum_{m=0} \sum_{n=0} I(i+m, \ j+n) K(m, n)$$

Individual elements in both functions, located in the same index , are multiplied and added.



- Convolution



- Cross-correlation

Convolution $(I * K)_{ij} = (I \circ flip(K))_{ij}$ Correlation

http://www.cs.umd.edu/~djacobs/CMSC426/Convolution.pdf

Backpropagation: Convolution Layer

- Convolution

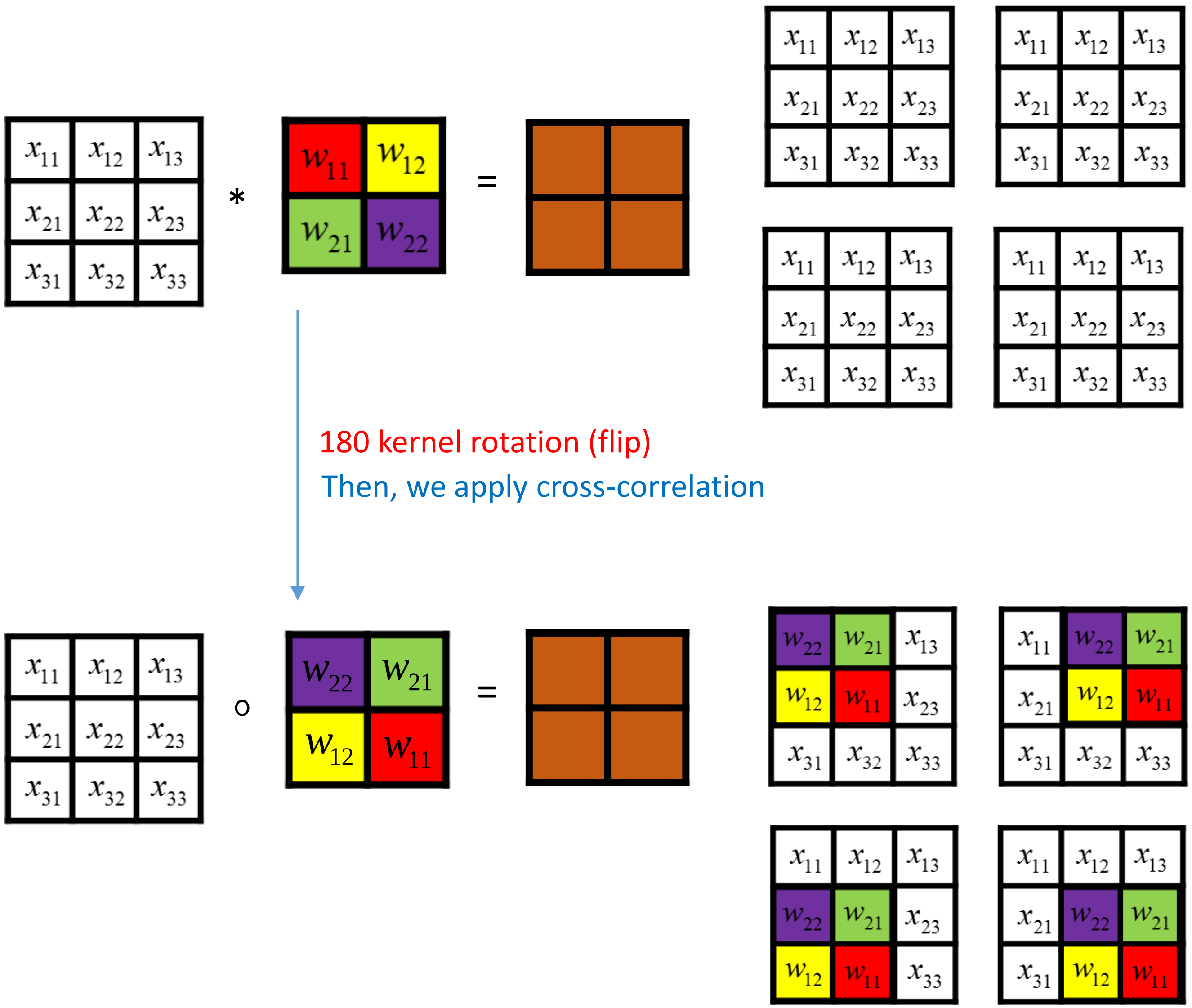
$$(I * K)_{ij} = \sum_{m=0} \sum_{n=0} I(m, n) K(i-m,\ j-n)$$

Individual elements in both functions, located in the opposite index, are multiplied and added.

- Cross-correlation

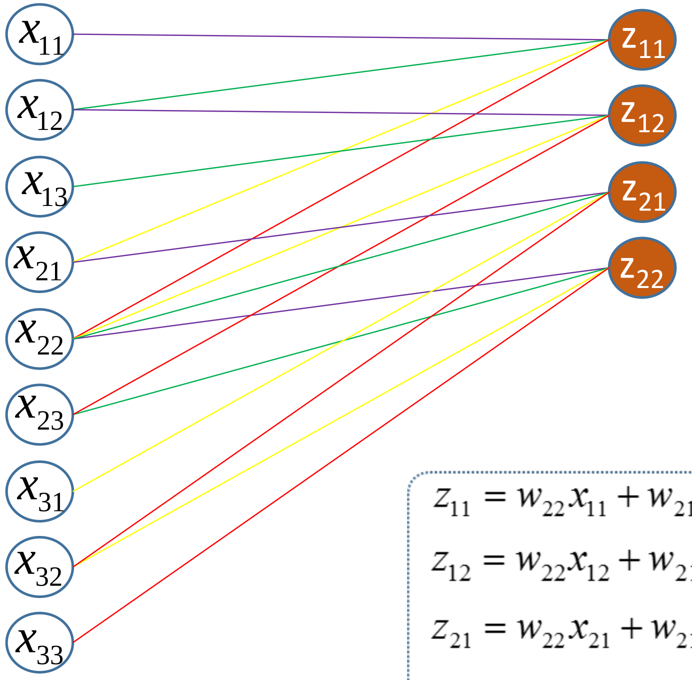
$$(I \circ K)_{ij} = \sum_{m=0} \sum_{n=0} I(i+m,\ j+n) K(m, n)$$

Individual elements in both functions, located in the same index , are multiplied and added.



http://www.cs.umd.edu/~djacobs/CMSC426/Convolution.pdf

Backpropagation: Convolution Layer - forwarding

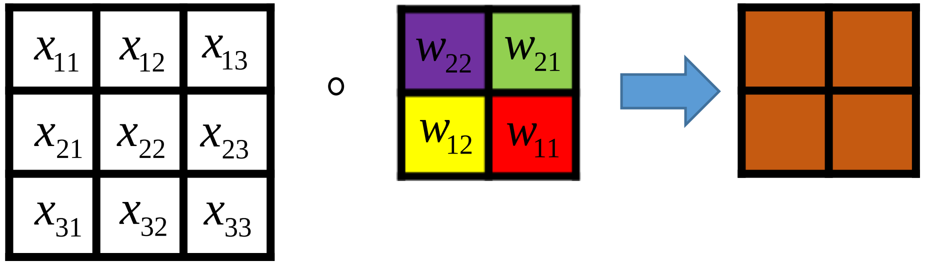


$$z_{11} = w_{22}x_{11} + w_{21}x_{12} + w_{12}x_{21} + w_{11}x_{22}$$

$$z_{12} = w_{22}x_{12} + w_{21}x_{13} + w_{12}x_{22} + w_{11}x_{23}$$

$$z_{21} = w_{22}x_{21} + w_{21}x_{22} + w_{12}x_{31} + w_{11}x_{32}$$

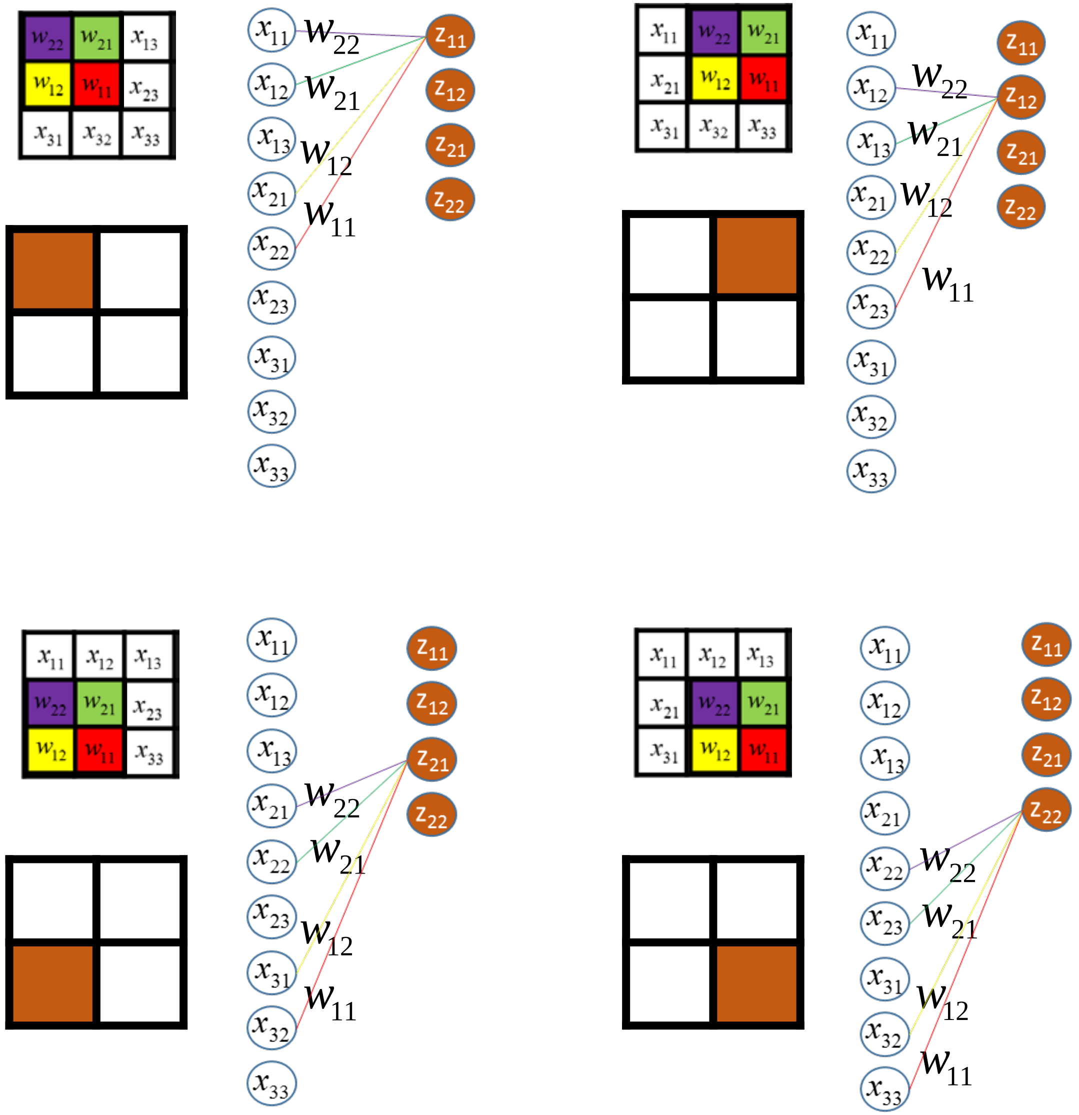
$$z_{22} = w_{22}x_{22} + w_{21}x_{23} + w_{12}x_{32} + w_{11}x_{33}$$



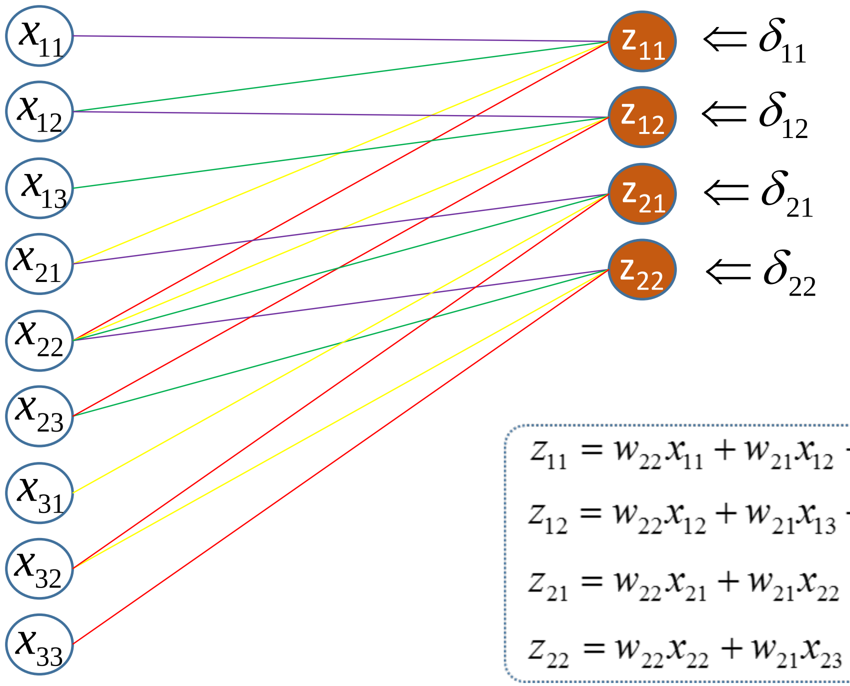
Input

Flipped kernel

Activation map



Backpropagation: Convolution Layer – backward: correlation between "w" and "δ"



$$z_{11} = w_{22}x_{11} + w_{21}x_{12} + w_{12}x_{21} + w_{11}x_{22}$$
$$z_{12} = w_{22}x_{12} + w_{21}x_{13} + w_{12}x_{22} + w_{11}x_{23}$$
$$z_{21} = w_{22}x_{21} + w_{21}x_{22} + w_{12}x_{31} + w_{11}x_{32}$$
$$z_{22} = w_{22}x_{22} + w_{21}x_{23} + w_{12}x_{32} + w_{11}x_{33}$$

Input　　Activation map　　kernel

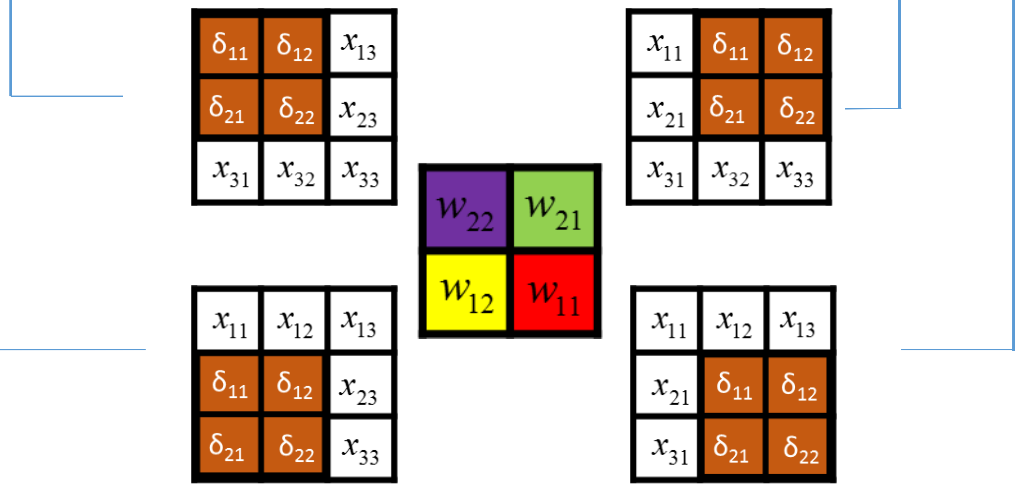
$$(1)\ \frac{\partial E}{\partial w_{mn}} = \sum_{i=1}^{N-F+1}\sum_{j=1}^{N-F+1} \delta_{ij}\frac{\partial z_{ij}}{\partial w_{mn}}$$

$$\frac{\partial E}{\partial w_{11}} = \delta_{11}x_{22} + \delta_{12}x_{23} + \delta_{21}x_{32} + \delta_{22}x_{33}$$

$$\frac{\partial E}{\partial w_{12}} = \delta_{11}x_{21} + \delta_{12}x_{22} + \delta_{21}x_{31} + \delta_{22}x_{32}$$

$$\frac{\partial E}{\partial w_{21}} = \delta_{11}x_{12} + \delta_{12}x_{13} + \delta_{21}x_{22} + \delta_{22}x_{23}$$

$$\frac{\partial E}{\partial w_{22}} = \delta_{11}x_{11} + \delta_{12}x_{12} + \delta_{21}x_{21} + \delta_{22}x_{22}$$



Backup Slides

Backpropagation: Convolution Layer – backward: why correlation here - proof?

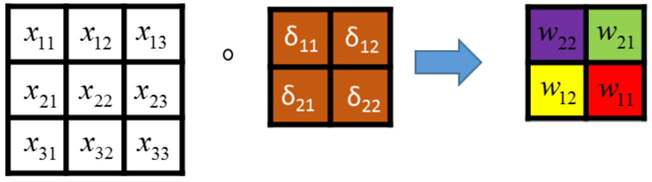
$$(1)\ \frac{\partial E}{\partial w_{mn}} = \sum_{i=1}^{N-F+1}\sum_{j=1}^{N-F+1}\delta_{ij}\frac{\partial z_{ij}}{\partial w_{mn}}$$

One in the right side was obtained using convolution previously.

$$z_{11} = w_{22}x_{11} + w_{21}x_{12} + w_{12}x_{21} + w_{11}x_{22}$$
$$z_{12} = w_{22}x_{12} + w_{21}x_{13} + w_{12}x_{22} + w_{11}x_{23}$$
$$z_{21} = w_{22}x_{21} + w_{21}x_{22} + w_{12}x_{31} + w_{11}x_{32}$$
$$z_{22} = w_{22}x_{22} + w_{21}x_{23} + w_{12}x_{32} + w_{11}x_{33}$$

$$= \sum_{i=1}^{N-F+1}\sum_{j=1}^{N-F+1}\delta_{ij}\frac{\partial}{\partial w_{mn}}\left(\sum_k\sum_q w_{kq}x_{i+k,\,j+q}\right)$$

We only consider when "$w_{mn}= w_{kq}$", in other words "m=k, n=q".

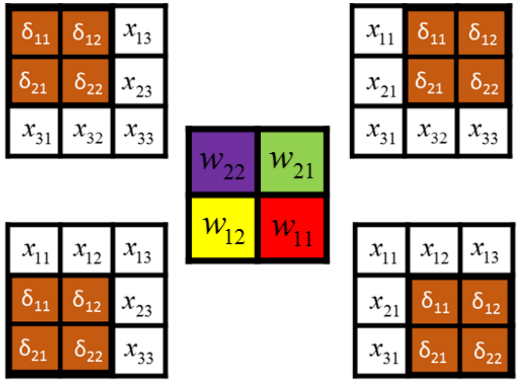


Input Activation map kernel

$$= \sum_{i=1}^{N-F+1}\sum_{j=1}^{N-F+1}\delta_{ij}\frac{\partial}{\partial w_{mn}}\left(\sum_k\sum_q w_{mn}x_{i+m,\,j+n}\right)$$

It is correlation

$$= \sum_{i=1}^{N-F+1}\sum_{j=1}^{N-F+1}\delta_{ij}x_{m+i,n+j}$$



Finally, "əw" is the result of cross correlation between "x" and "δ" previously.