# Lecture 5.
# Convolutional Neural Network

# Review

1. Limitation of Linear Classifier
   - XOR Gate
2. Perceptron
   - Perceptron = Linear Classifier
   - Analogy to Neurons
   - Building Block of the Neural Network
3. MLP
   - MLP and the Neural Network
   - The Universal Approximation Theorem
   - Where Backpropagation becomes Important
4. Limitations of MLP

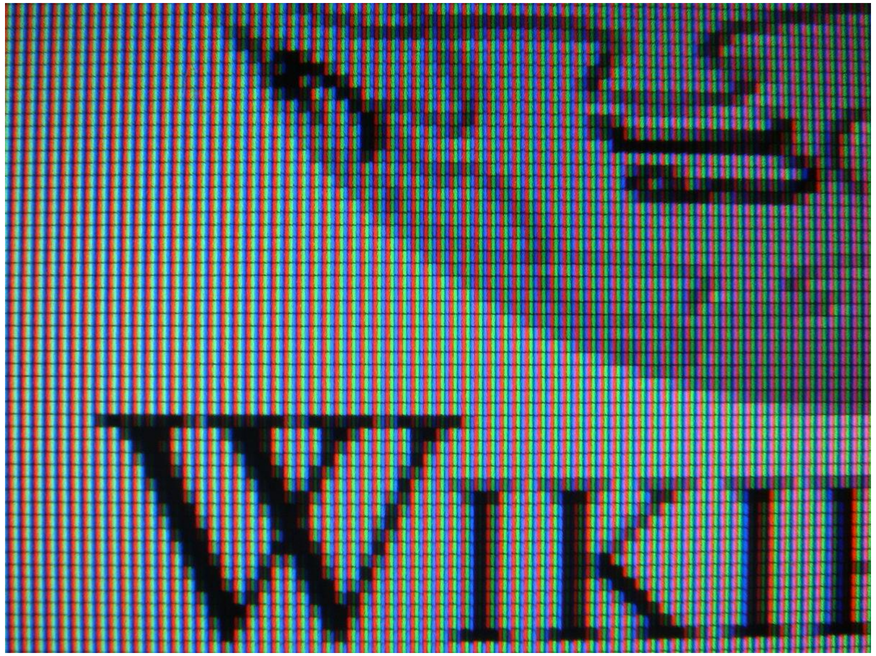
# Today's Contents

1. Back to Image Classification

2. Convolutional Neural Network

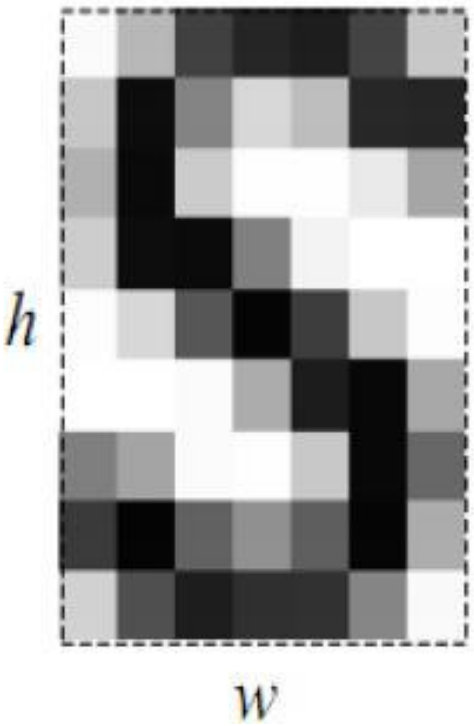
# Back to Image Classification : Pixels

Image = Matrix of Pixels



# Back to Image Classification : Pixels

What we see




What computer sees (grayscale)

$$\mathbf{x} = \begin{pmatrix} 251, 181, 068, 041, 032, 071, 197, \\ 196, 014, 132, 213, 187, 043, 041, \\ 174, 011, 200, 254, 254, 232, 164, \\ 202, 014, 012, 128, 242, 255, 255, \\ 253, 212, 089, 005, 064, 196, 253, \\ 255, 255, 251, 196, 030, 009, 165, \\ 127, 162, 251, 254, 197, 009, 105, \\ 062, 005, 100, 144, 097, 006, 170, \\ 207, 083, 032, 051, 053, 134, 250 \end{pmatrix}$$

Image from researchgate.net

# Back to Image Classification : Pixels

RGB

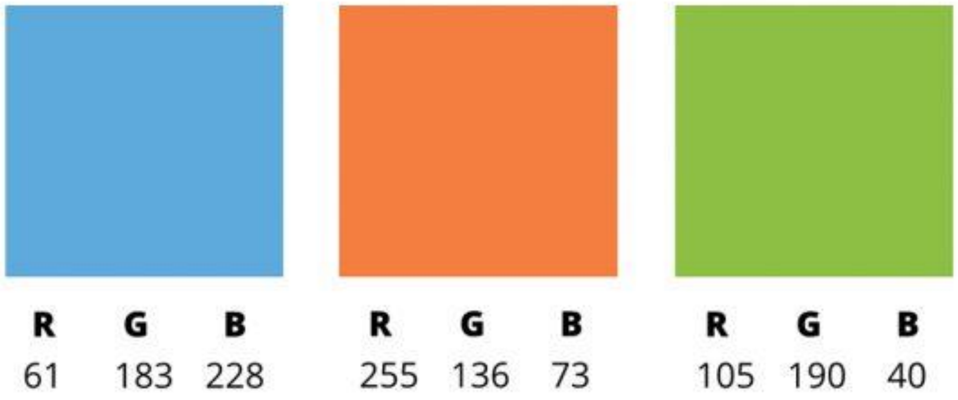



Image from https://negliadesign.com/

CIFAR-10

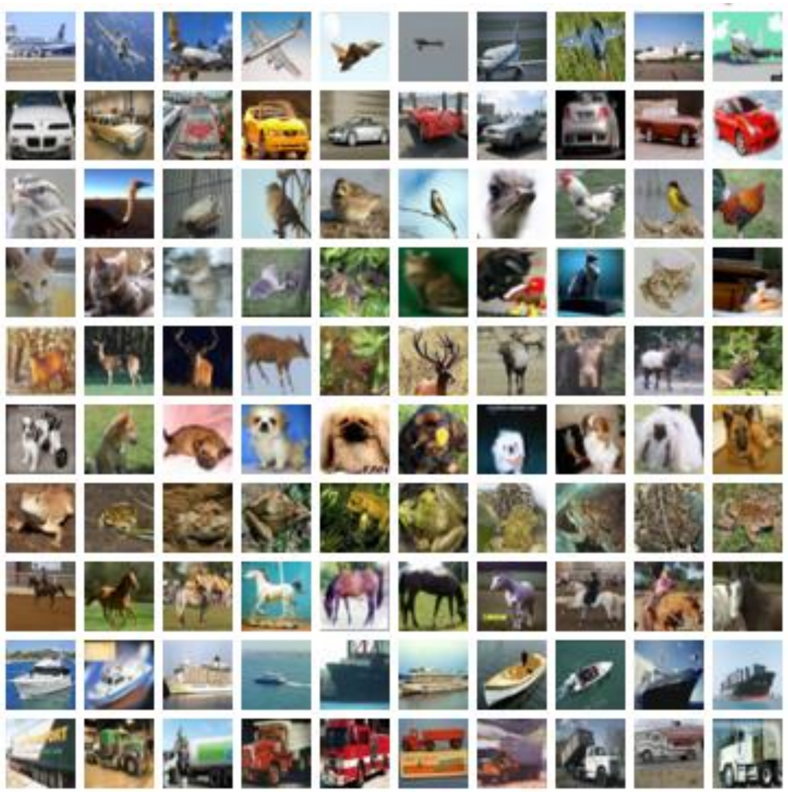
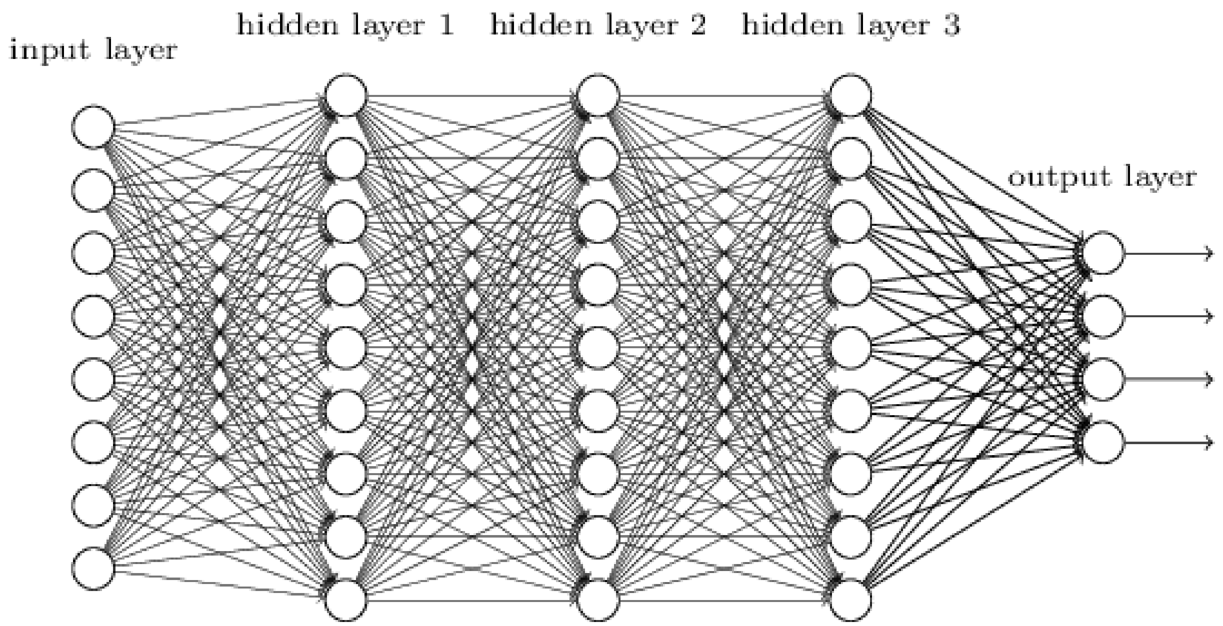


Image from https://www.kaggle.com/

# Back to Image Classification : MLP

MLP



# Back to Image Classification : MLP

Must Stretch Out Tensor into Vector

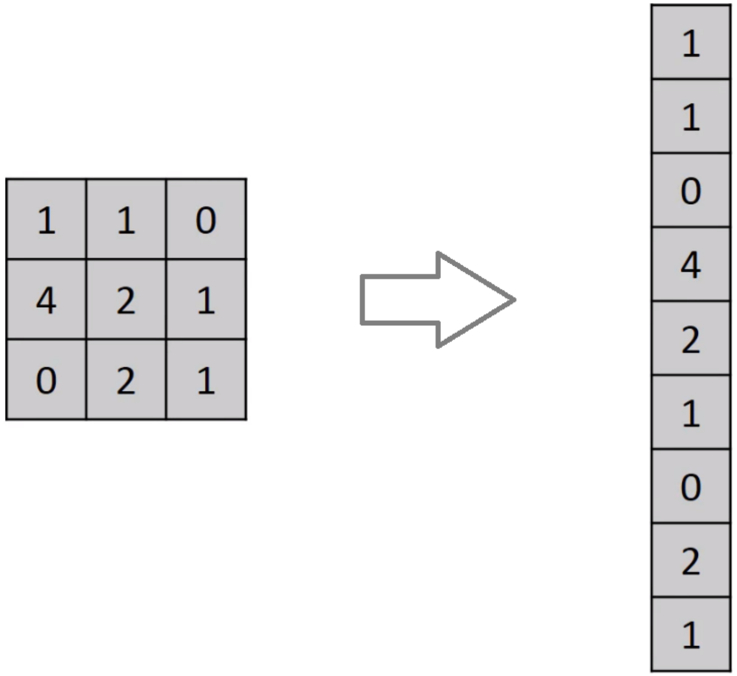



Image from https://towardsdatascience.com

# Back to Image Classification : MLP

Worked well on MNIST

What about on CIFAR-10?

# Back to Image Classification : MLP

However, MLPs are not appropriate for Image Classification…

1. Too many parameters required

2. Spatial structure is lost

# Back to Image Classification : MLP

We want a model that,

1. Uses fewer parameters than MLPs

2. Preserves the spatial structure

# Back to Image Classification : MLP

Recall The Universal Approximation Theorem…

*"1개의 Hidden Layer를 가진 MLP로 어떤 함수도 근사할 수 있다."*

However,

*"A feedforward network with a single layer is sufficient to represent any function,*
*But the layer may be infeasibly large and may fail to learn and generalize correctly."*

# CNN : Idea

We want a model that,

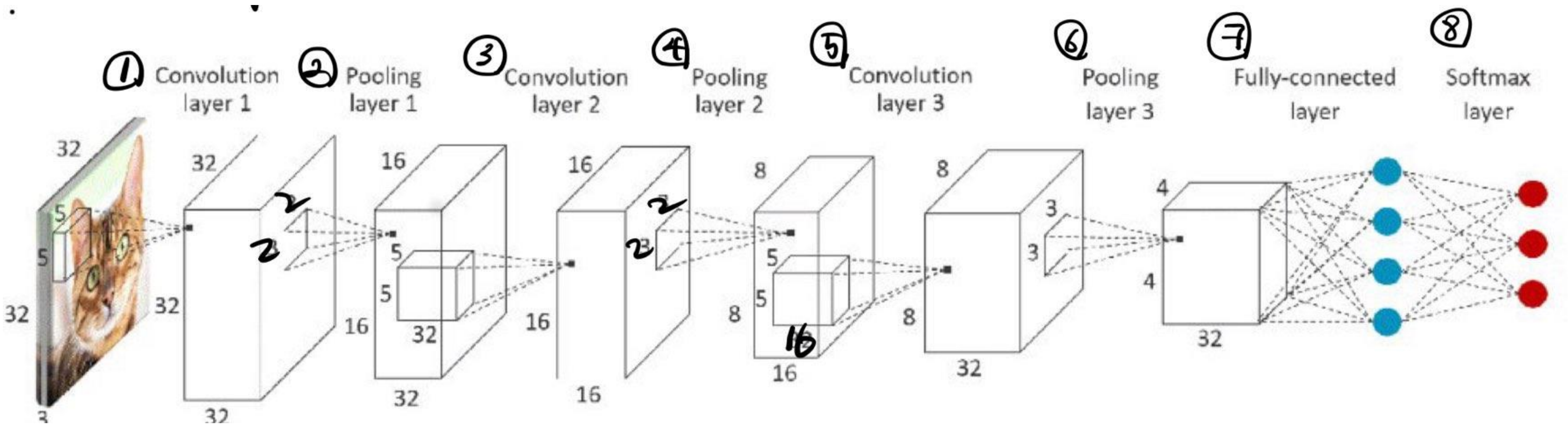
1. Uses fewer parameters than MLPs

2. Preserves the spatial structure

**Convolutional Neural Network** to the rescue!!!

# CNN : Idea

Image from https://community.arm.com/




What happened?!?!?!

# CNN : Idea

What happens during Tensor → Tensor Mapping?

What are Convolution, Pooling, Fully-Connected Layers?

How does this Model extract features from spatial structure?

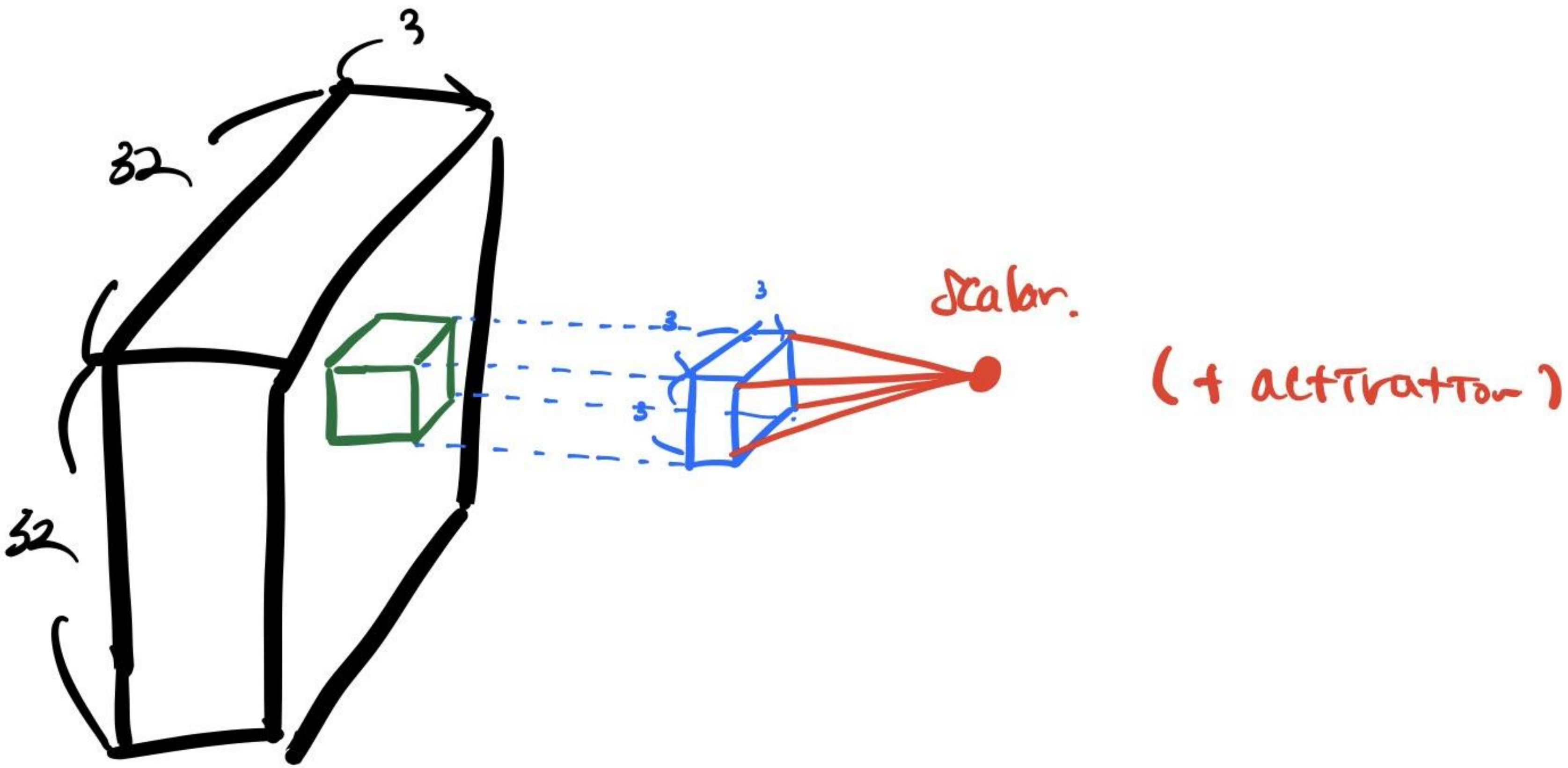
Does it use less parameters than the MLP?

# CNN : Convolution

How do we map Tensor to Tensor, while preserving the spatial structure?

Apply **Filter** to small regions in the Tensor (Image)

# CNN : Convolution



# CNN : Convolution

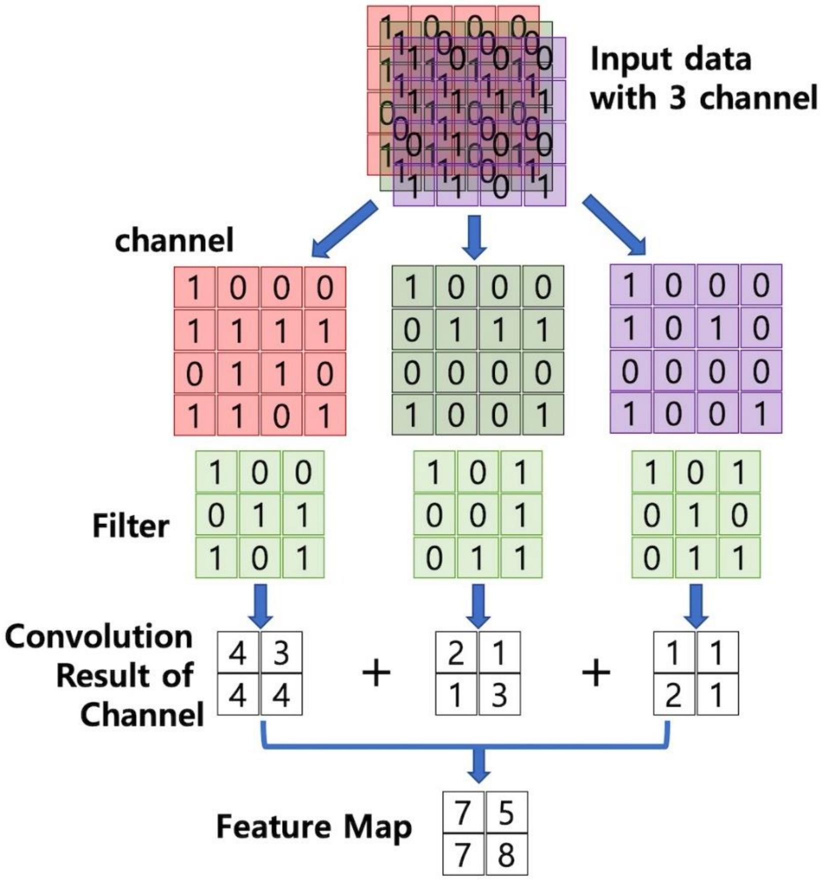



Image from taewan.kim/ppst/cnn

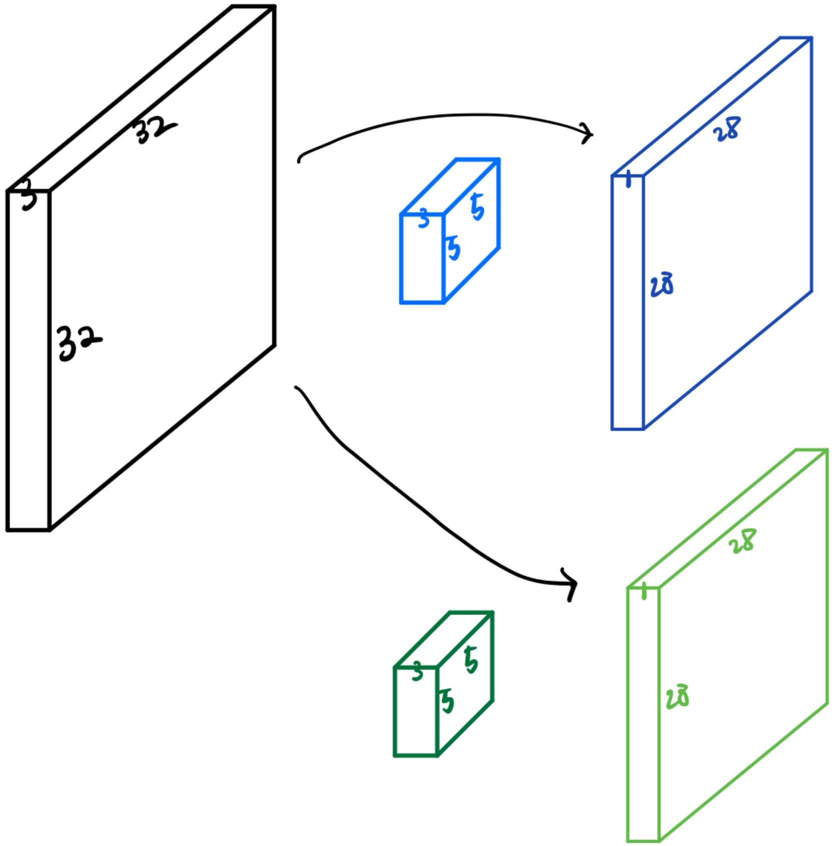
While sliding over the tensor,

calculate the

sum of elementwise multiplication

This is called, **Convolution**

# CNN : Convolution




1 Filter = 1 Activation Map

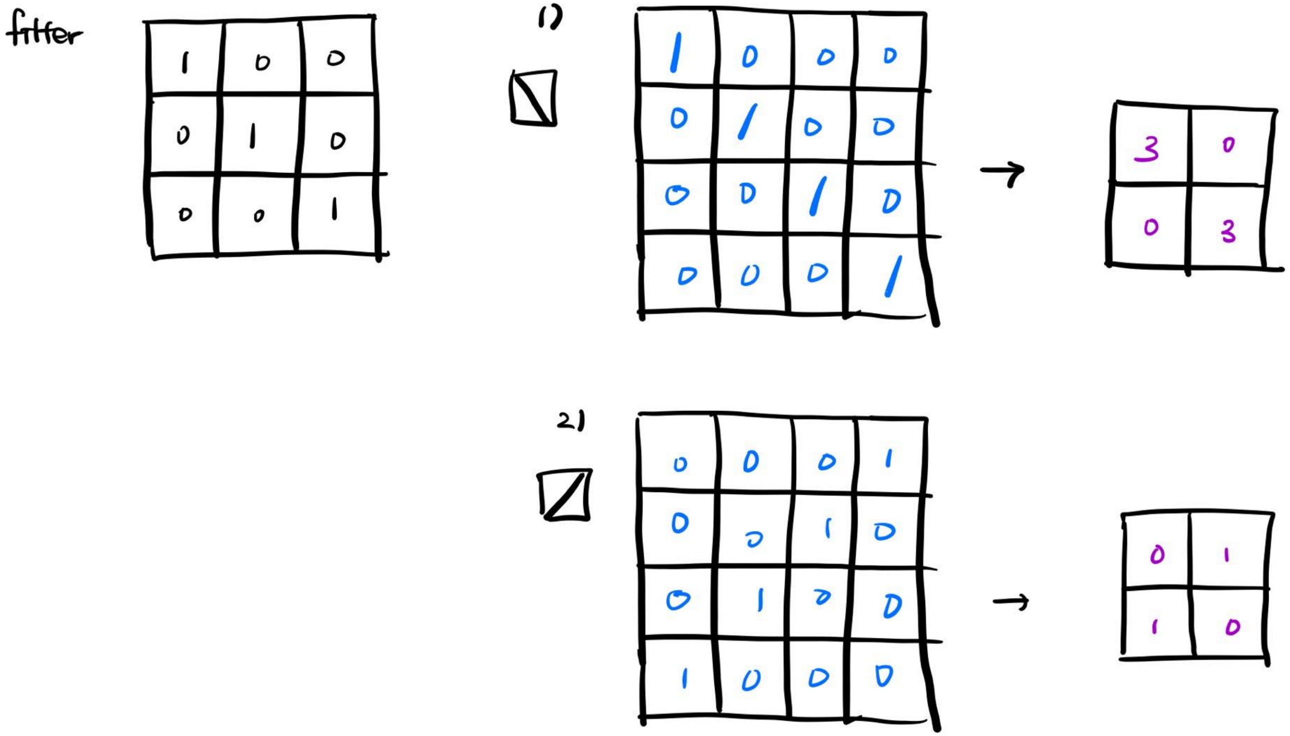
# CNN : Convolution

Then, how do filters extract features from the image,

while preserving the spatial structure?

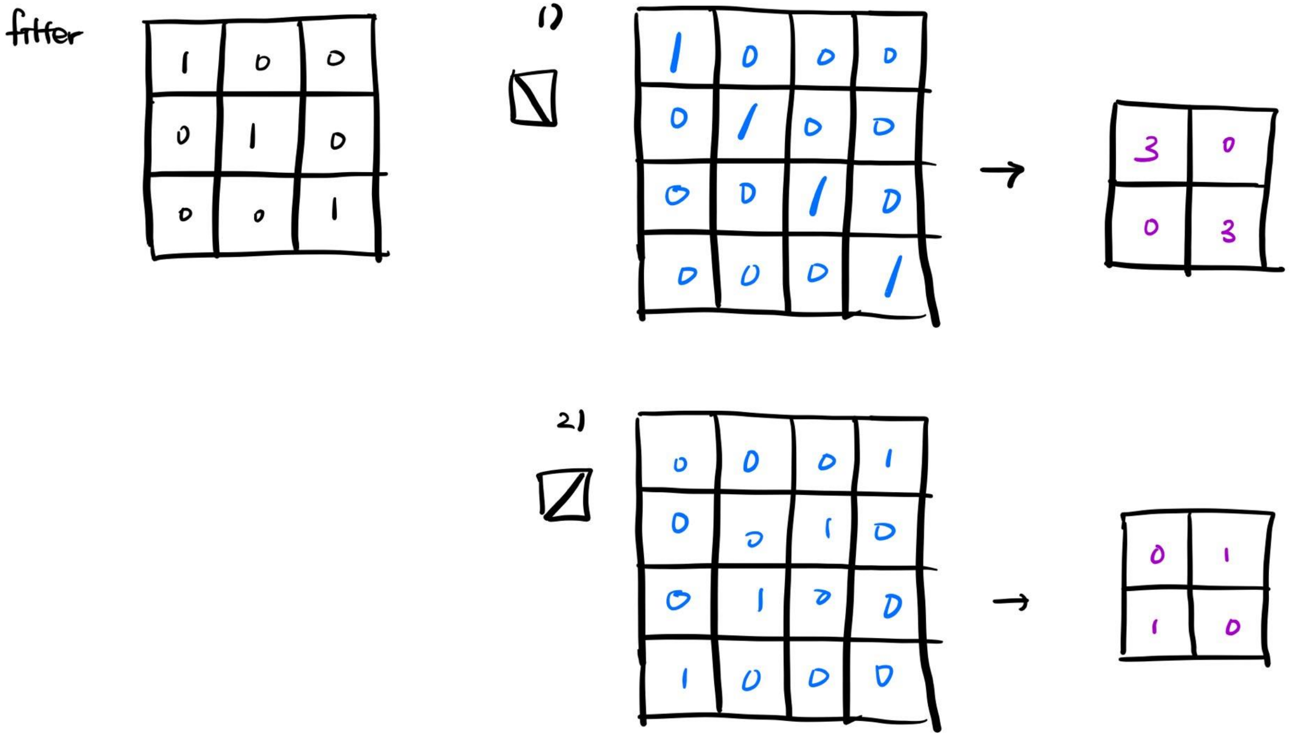
# CNN : Convolution

For simplicity, assume White = 0 , Black = 1, and activation = RELU



# CNN : Convolution

This filter focuses on the major diagonal… Other filters will focus on other features!



# CNN : Convolution

Filters extract features from the spatial structure of the tensor

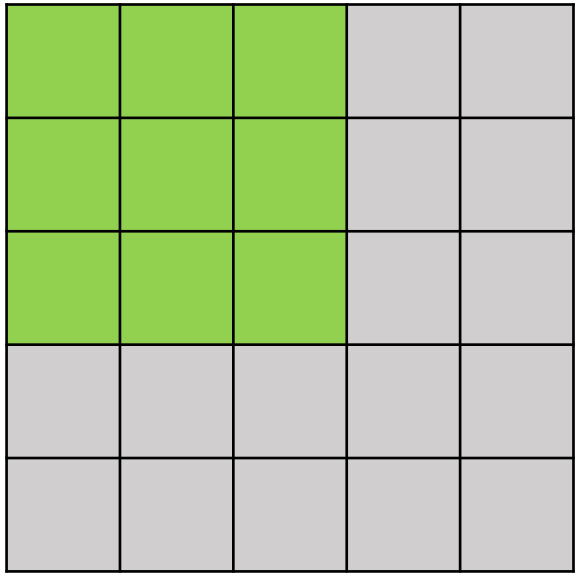
They are Parameters!!!

** So, they must be Updated via Backpropagation.

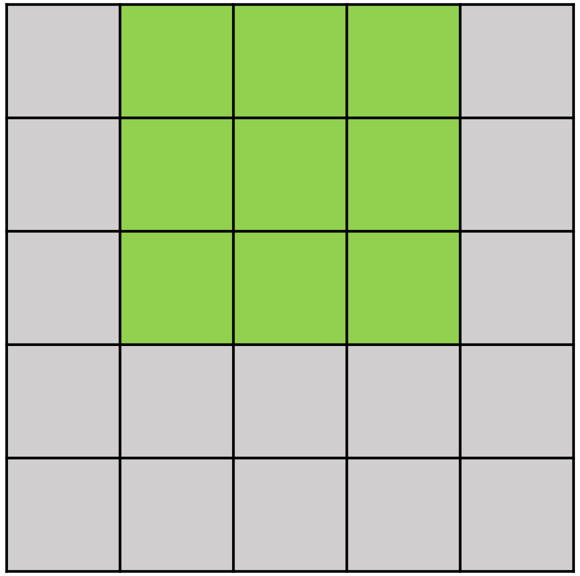
For more detail, read https://becominghuman.ai/back-propagation-in-convolutional-neural-networks-intuition-and-code-714ef1c38199

# CNN : Convolution

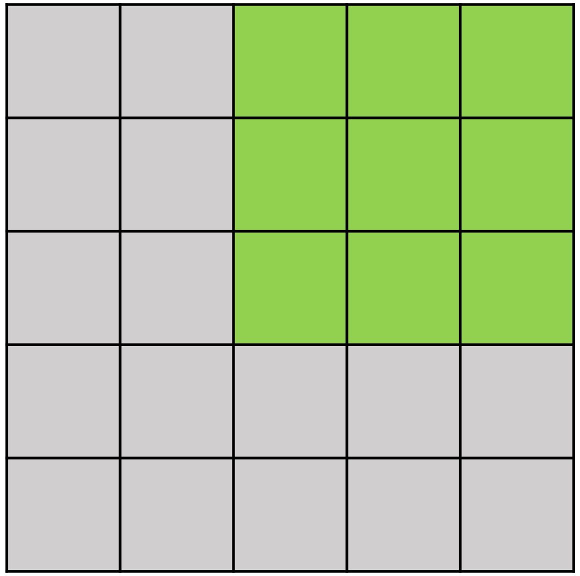
How much does a filter move per slide? **Stride**



**Stride 1**



**Stride 2**

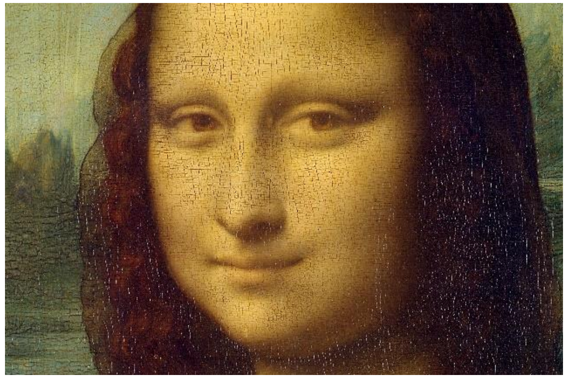


# CNN : Convolution

$$O = \frac{I + 2Pa - F}{S} + 1$$

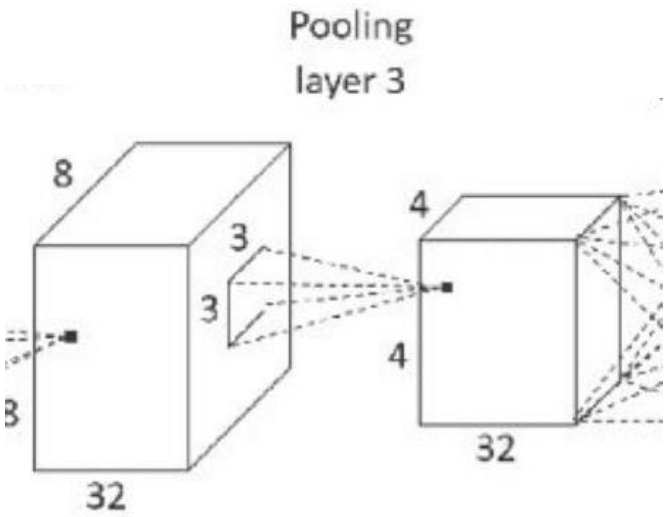
O = output size N = input size F = filter size S = stride Pa = padding size

# CNN : Pooling



We want to reduce the dimension of the feature,

via **Pooling**



# CNN : Pooling

Extract the representative value over a region…! No parameter involved!!!

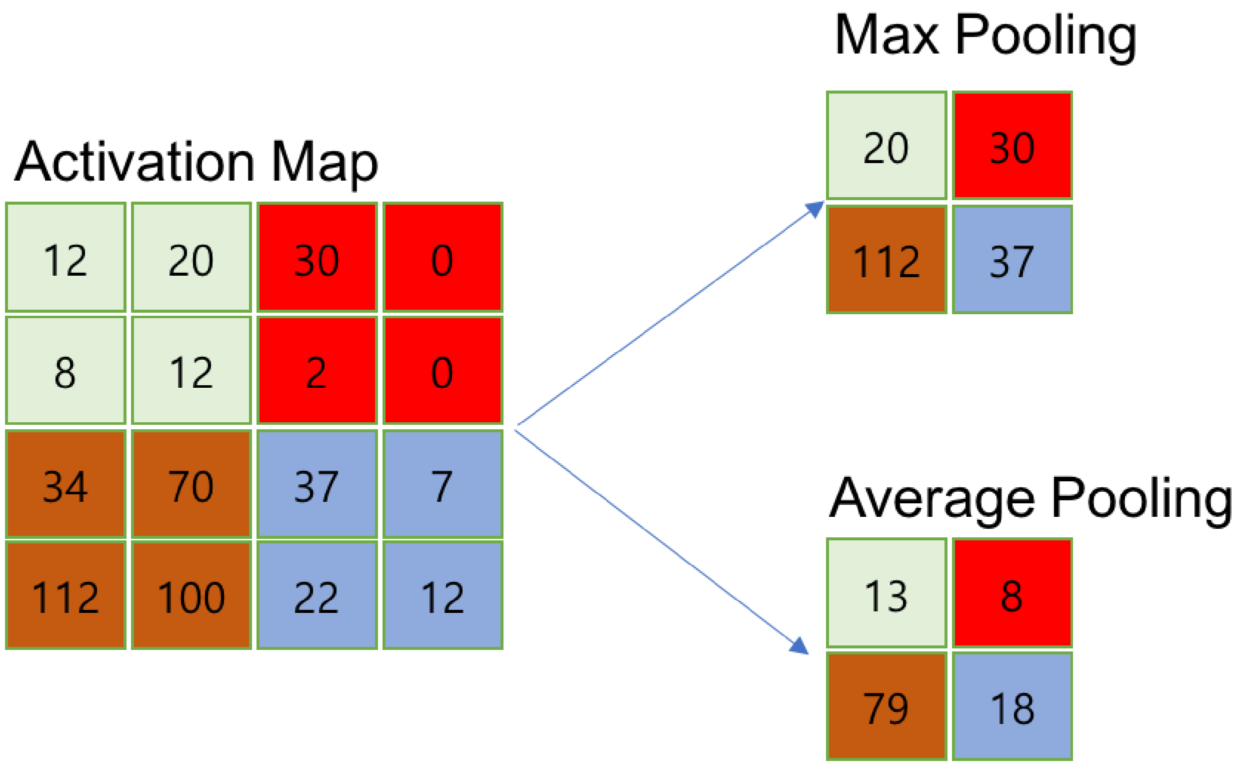



Image from taewan.kim/ppst/cnn

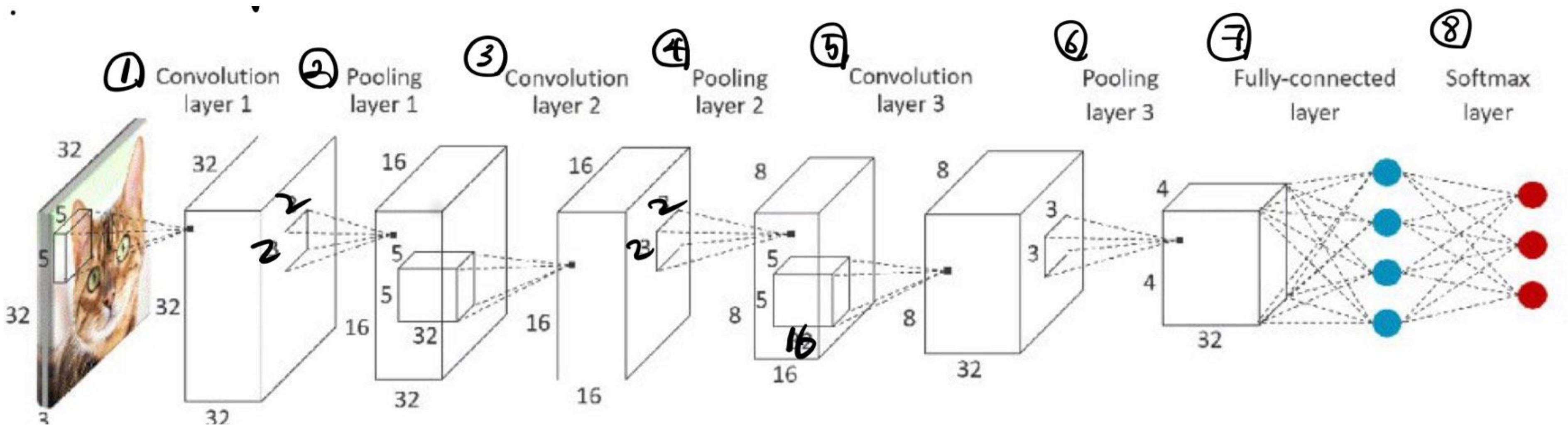
# CNN : Pooling

$$O = \frac{I}{Po}$$

O = output size　　I = input size　　Po = pool size

** Generally, Pool Size = Pooling Stride

# CNN : Fully-Connected Layer

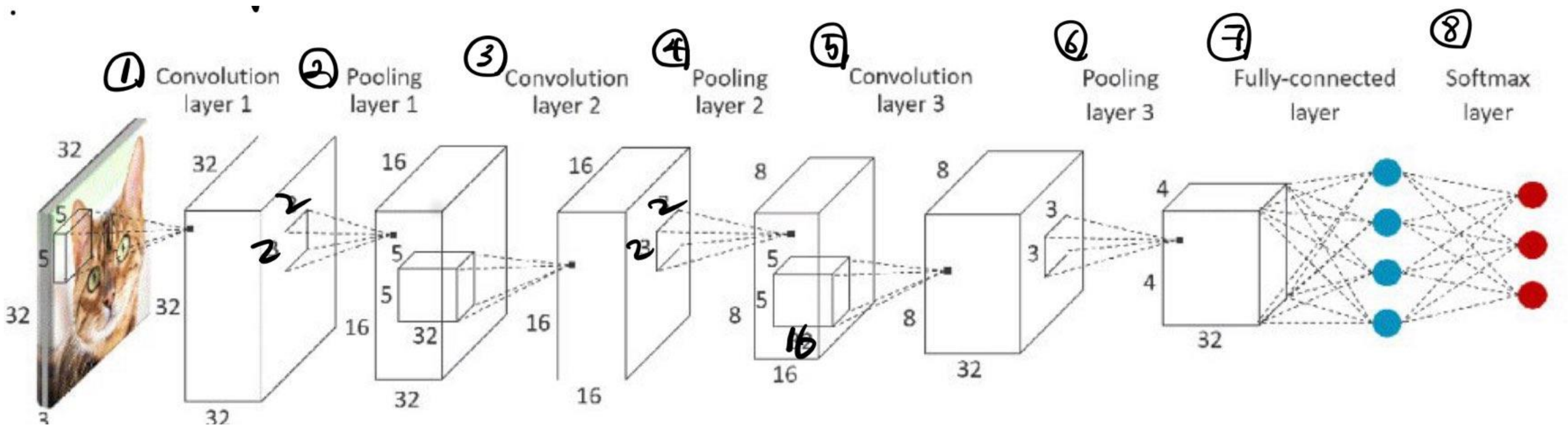



Extract features with **CONV Layers**

Then, reduce dimension with **POOL Layers**

Finally, flatten data and make classification via **FC(Fully-Connected) Layer**

# CNN : Architecture




Extract features with **CONV Layers**

Then, reduce dimension with **POOL Layers**

Finally, flatten data and make classification via **FC(Fully-Connected) Layer**

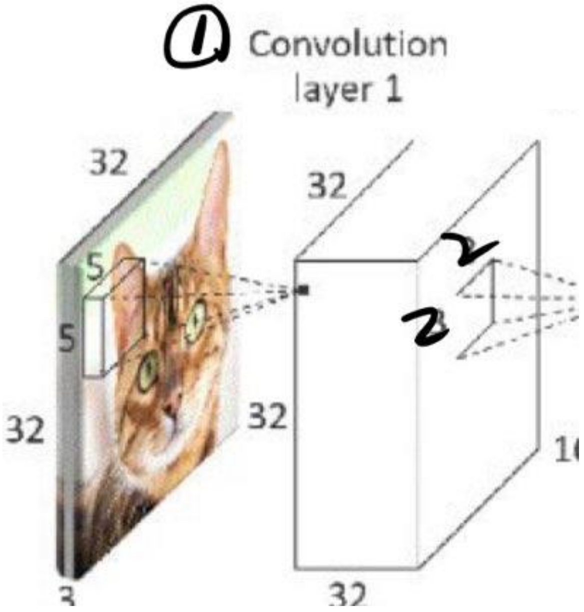
# CNN : Architecture

Does it use less parameters than the MLP?

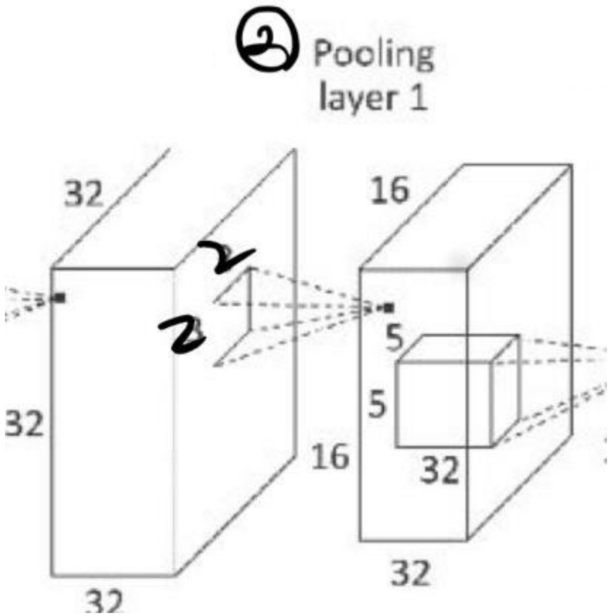
Let's follow the CNN Architecture and figure it out!!!

# CNN : Architecture

**CONV1**

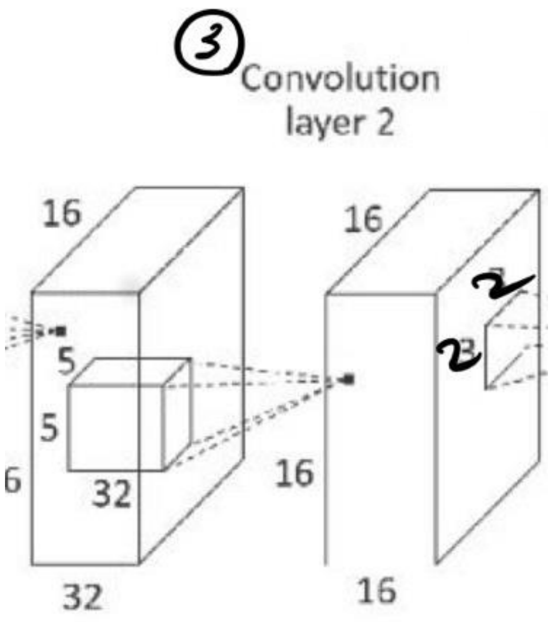



**POOL1**

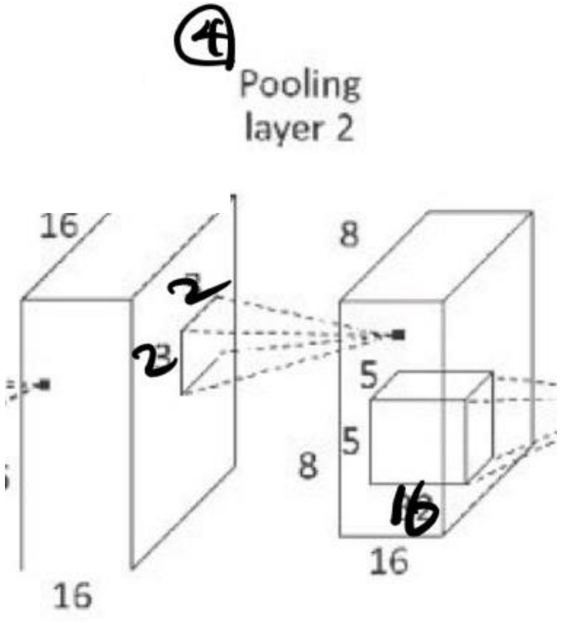


# CNN : Architecture

**CONV2**

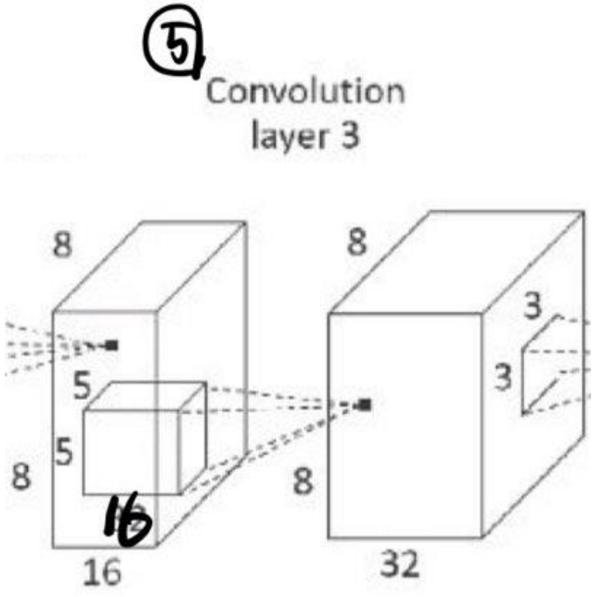



**POOL2**

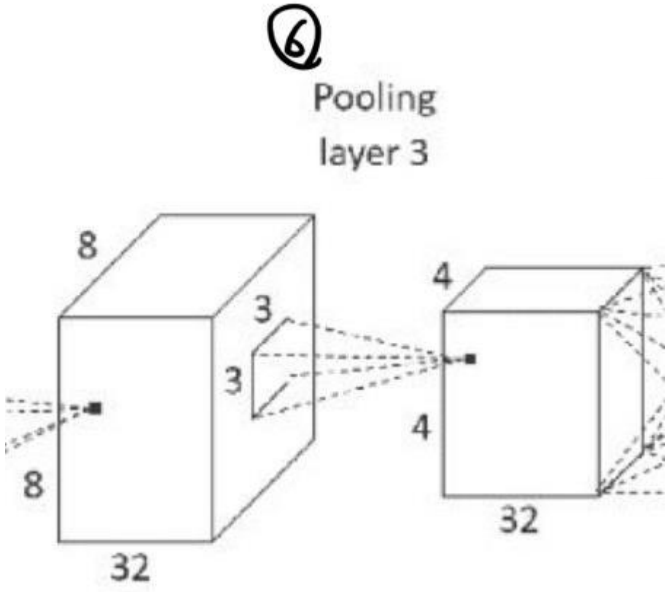


# CNN : Architecture

**CONV3**

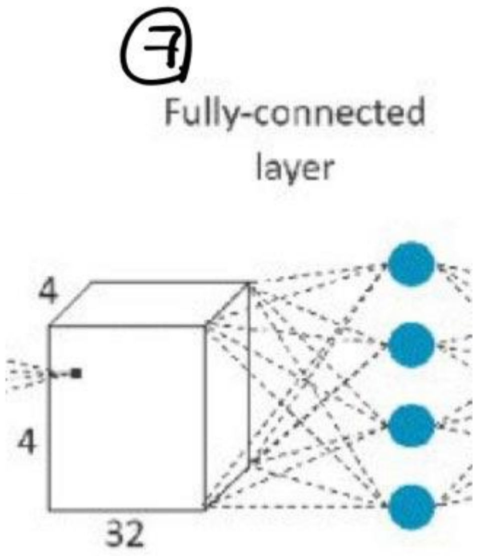



**POOL3**

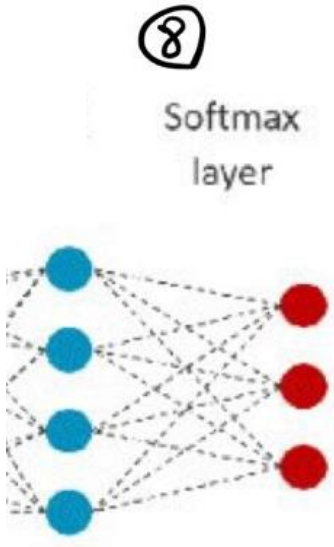


# CNN : Architecture

**FC**




**Softmax**



# CNN : Architecture

Total ( ) Parameters were Used!

# CNN : Architecture

However, if we used MLP instead, assuming CONV + POOL as one Hidden Layer,

32 * 32 * 3 → 16 * 16 * 32 → 8 * 8 * 16 → 4 * 4 * 32 → 10

Total (                                        ) Parameters were Used!

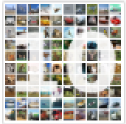
# CNN : Architecture

CNN uses much less Parameters than the MLP!!!

# CNN : Performance

## CIFAR-10

who is the best in CIFAR-10 ?



**CIFAR-10** 49 results collected

Units: accuracy %

Classify 32x32 colour images.

| Result | Method | Venue | Details |
|---|---|---|---|
| 96.53 % | Fractional Max-Pooling | arXiv 2015 | Details |
| 95.59 % | Striving for Simplicity: The All Convolutional Net | ICLR 2015 | Details |
| 94.16 % | All you need is a good init | ICLR 2016 | Details |
| 94% | Lessons learned from manually classifying CIFAR-10 | unpublished 2011 | Details |
| 93.95 % | Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree | AISTATS 2016 | Details |
| 93.72 % | Spatially-sparse convolutional neural networks | arXiv 2014 | |
| 93.63 % | Scalable Bayesian Optimization Using Deep Neural Networks | ICML 2015 | |
| 93.57 % | Deep Residual Learning for Image Recognition | arXiv 2015 | Details |
| 93.45 % | Fast and Accurate Deep Network Learning by Exponential Linear Units | arXiv 2015 | Details |
| 93.34 % | Universum Prescription: Regularization using Unlabeled Data | arXiv 2015 | |
| 93.25 % | Batch-normalized Maxout Network in Network | arXiv 2015 | Details |
| 93.13 % | Competitive Multi-scale Convolution | arXiv 2015 | |
| 92.91 % | Recurrent Convolutional Neural Network for Object Recognition | CVPR 2015 | Details |
| 92.49 % | Learning Activation Functions to Improve Deep Neural Networks | ICLR 2015 | Details |

Classification Rankings :
https://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html

# CNN : Performance

# Review

We want a model that,

1. Uses fewer parameters than MLPs

2. Preserves the spatial information

**Convolutional Neural Network** to the rescue!!!

# Preview on Next Lecture(s)

## Limitations of MLP

1. Needs a lot of Labeled Data
2. Vanishing Gradient
3. Overfitting
4. Gets stuck in the Local Minima / Saddle Point

이재현, 김동주 DiriDiri Lab. Lecture Vision - CS231n 52

# Preview on Next Lecture(s)

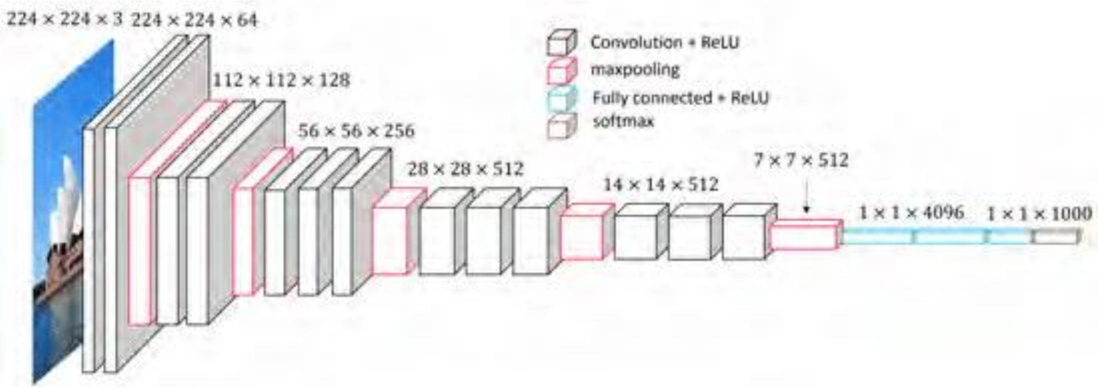



Image from researchgate.net

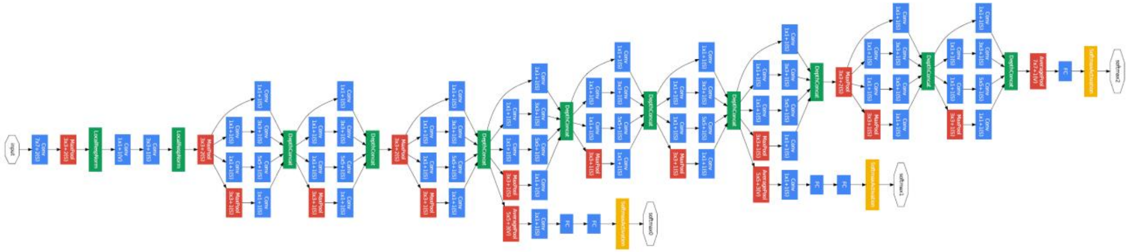


Image from medium.com