Car Classification:
Identify the cars in images using CNNs

2021. 10. 12

Assistant: Jinhee Kim

kaggle

DAVIAN
Data and Visual Analytics Lab

1-1. Image classification

- The goal is to predict a single label for a given image.

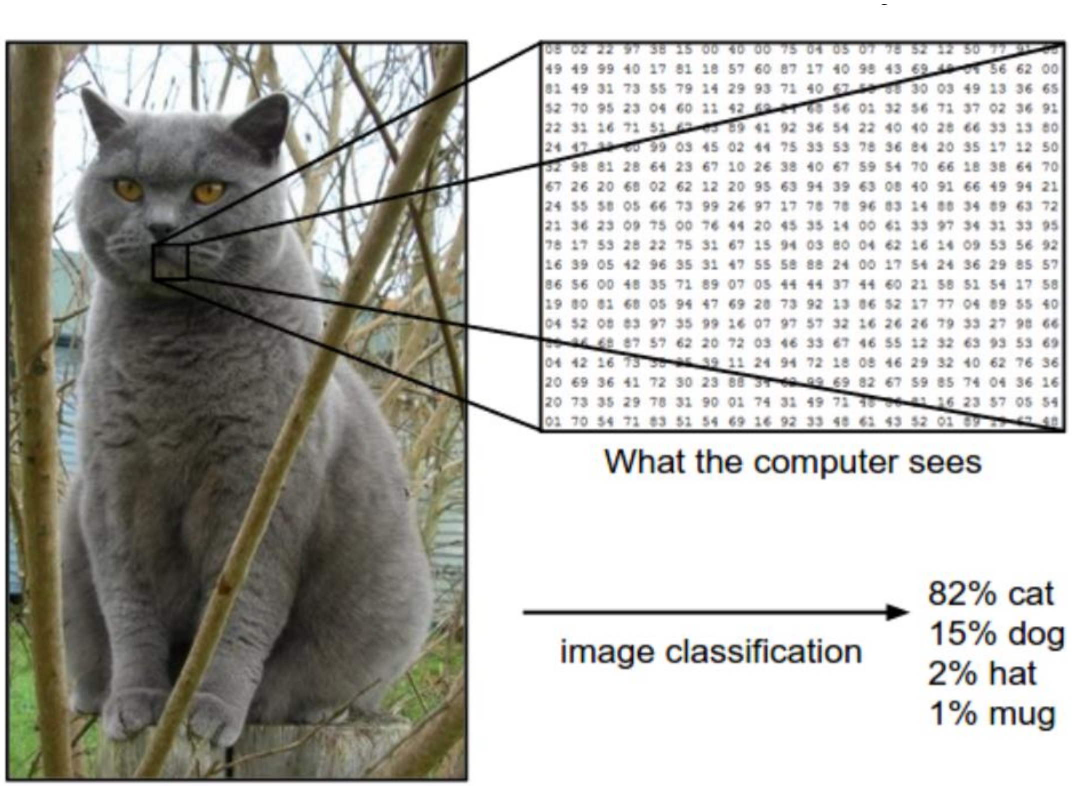
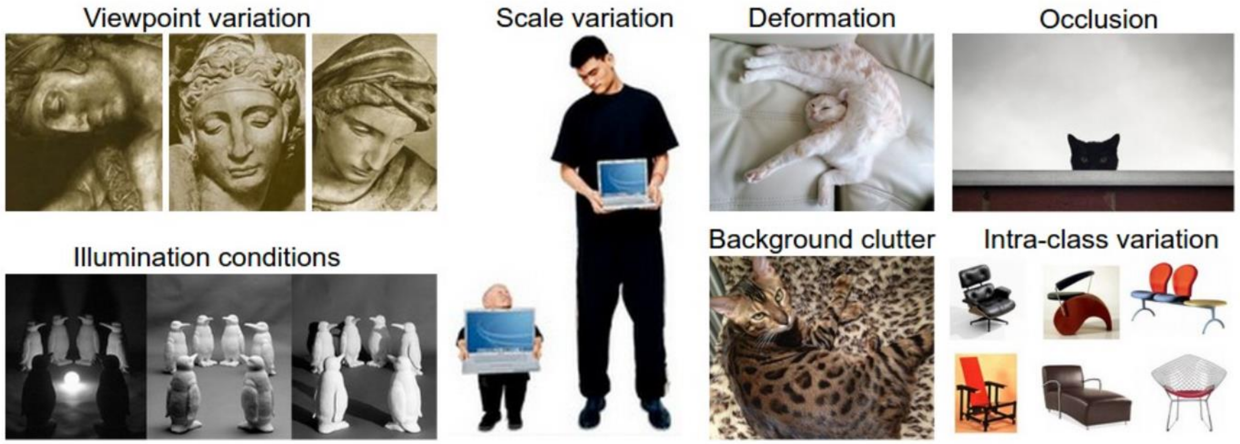


Image size: (3, H, W)

Source: https://cs231n.github.io/classification/

- Images are 3-dimensional arrays of integers from 0 to 255 with three color channels Red, Green, Blue.

- Challenges of image classification:



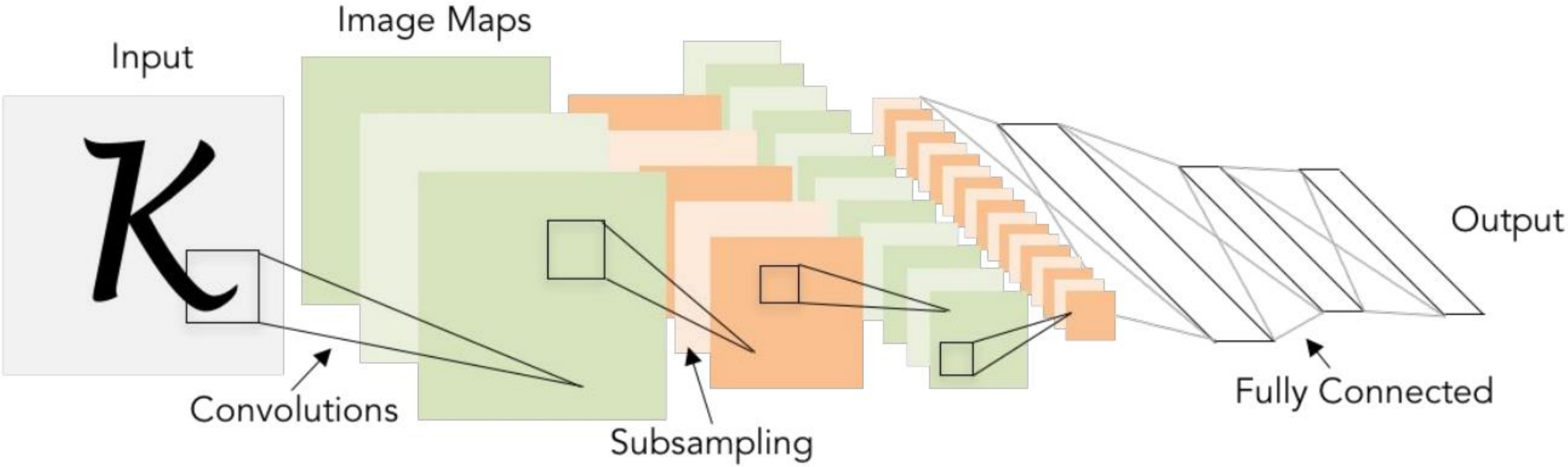
1-1. Image classification

- Data-driven approach in deep learning:

Source: https://cs231n.github.io/classification/

1-2. Image classification using CNN

- Convolutional Neural Network (CNN)

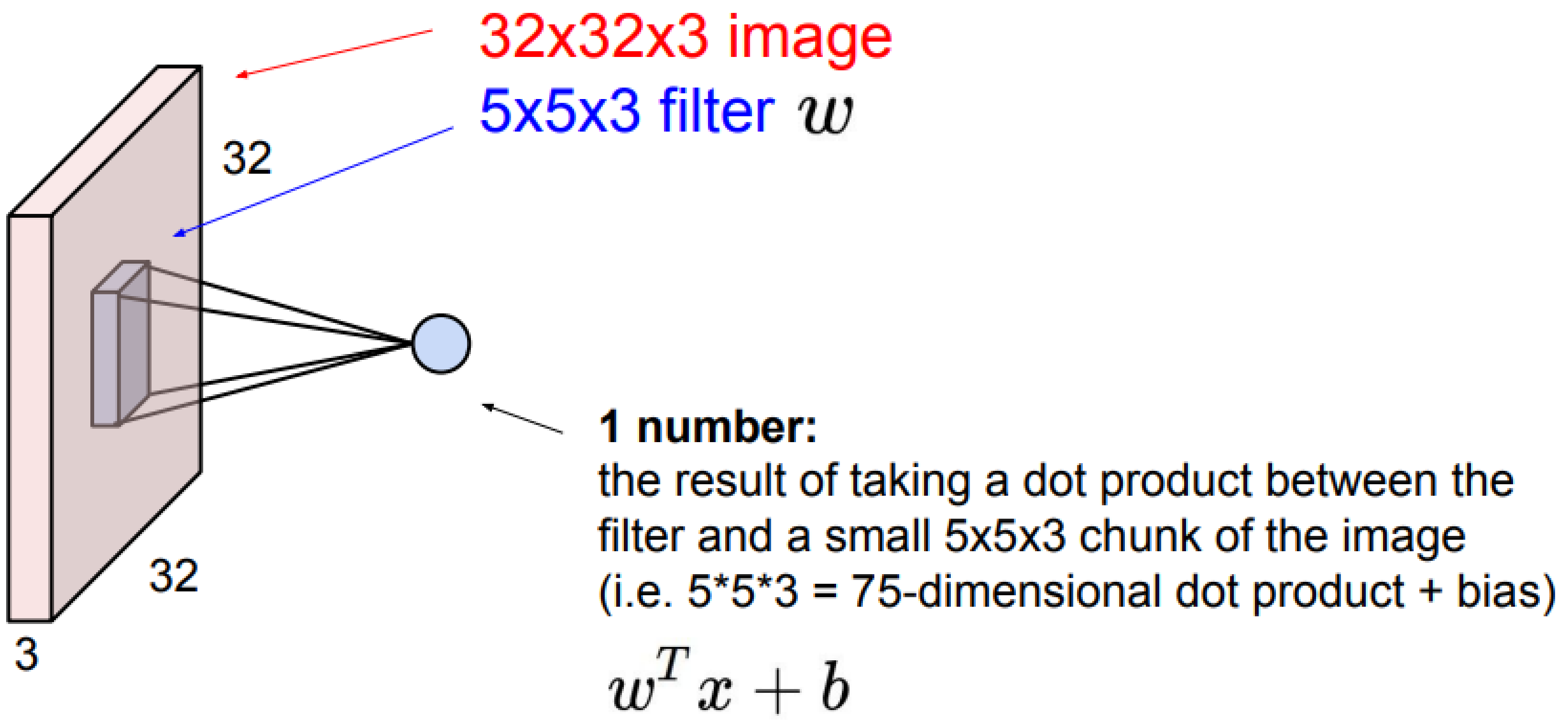


Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2
i.e. architecture is [CONV-POOL-CONV-POOL-FC-FC]

http://cs231n.stanford.edu/slides/2021/lecture_9.pdf

1-2. Image classification using CNN

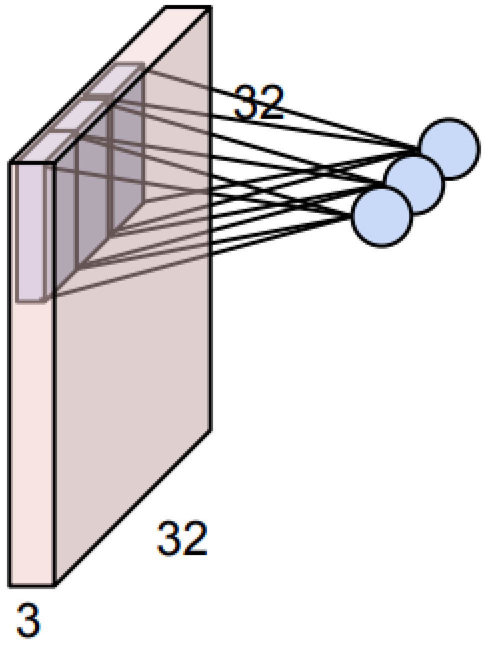
- Convolution



http://cs231n.stanford.edu/slides/2021/lecture_9.pdf

1-2. Image classification using CNN

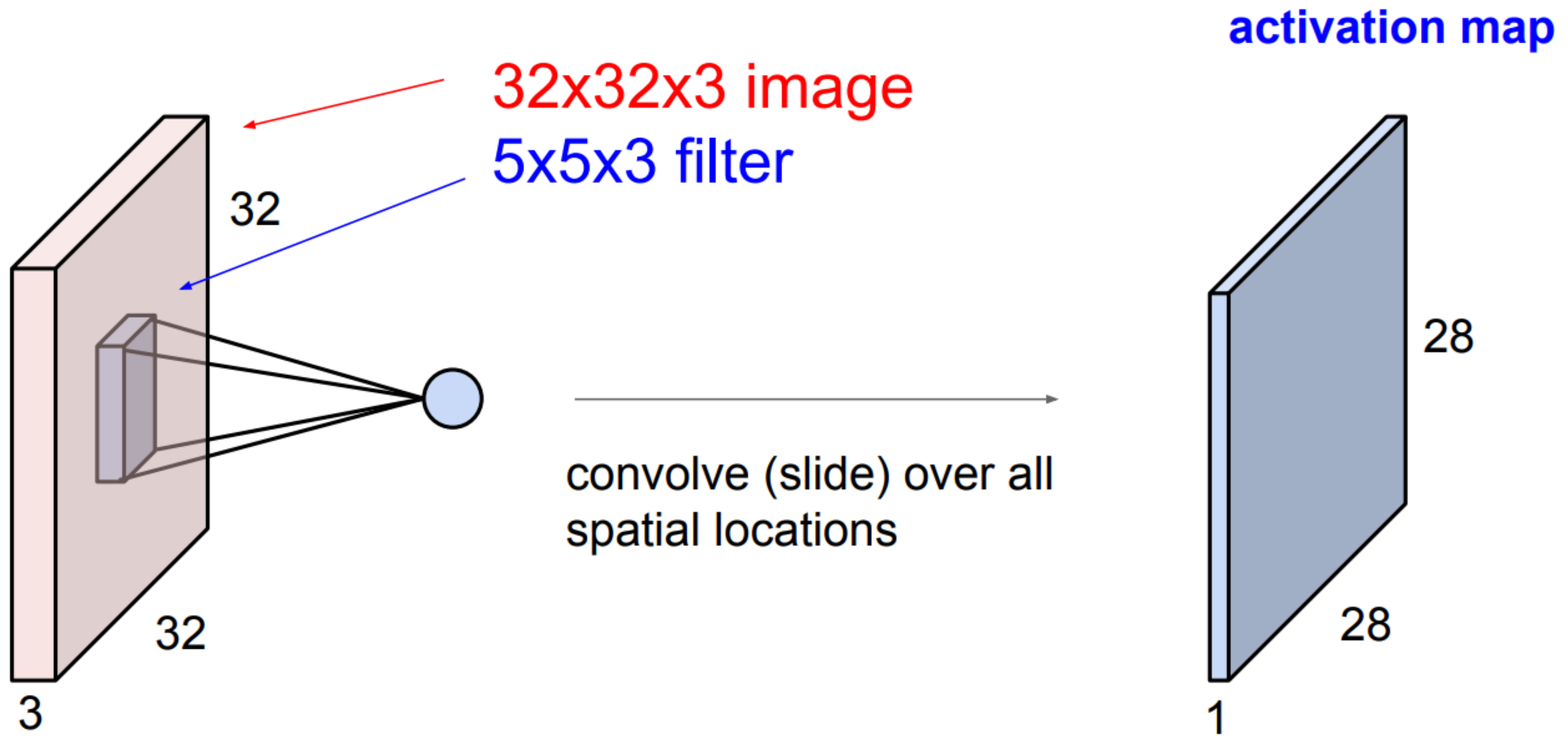
- Convolution



http://cs231n.stanford.edu/slides/2021/lecture_9.pdf

1-2. Image classification using CNN

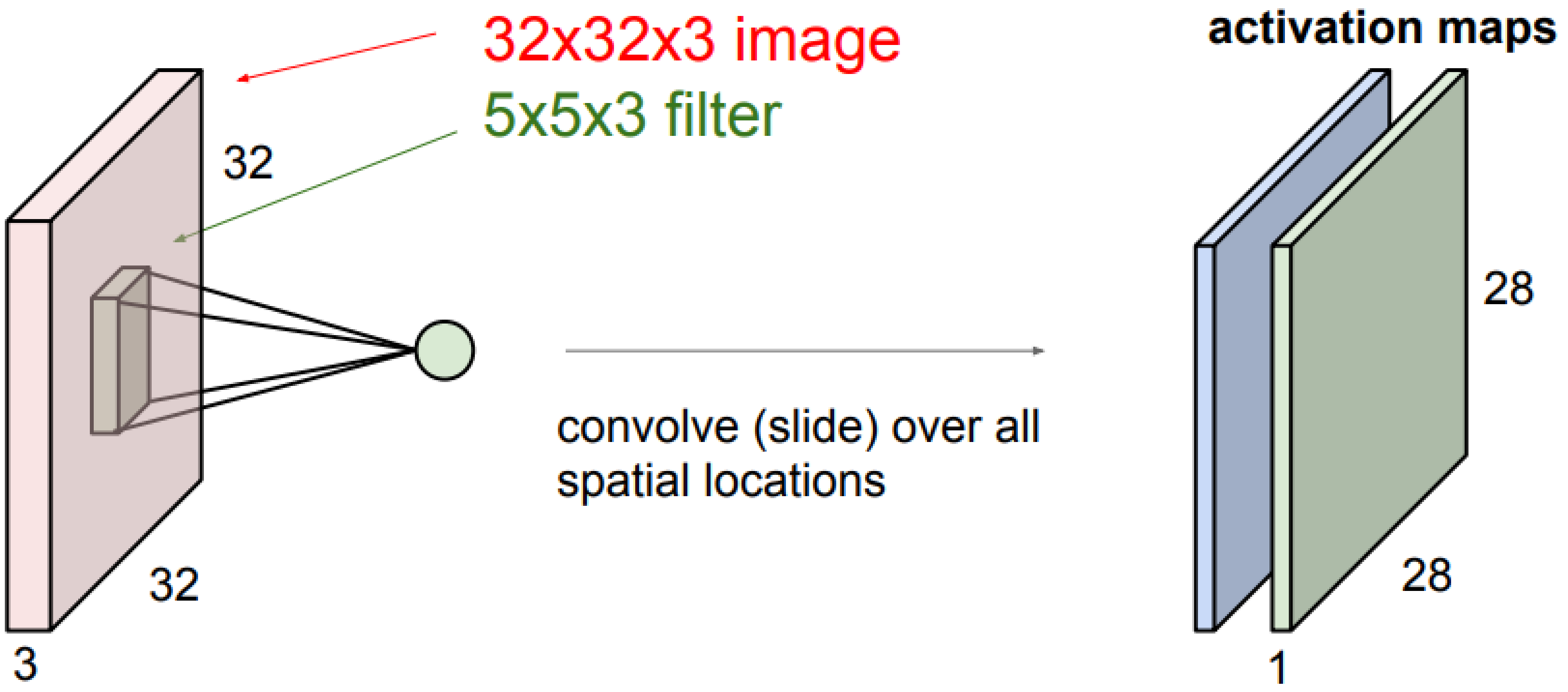
- Convolution



http://cs231n.stanford.edu/slides/2021/lecture_9.pdf

1-2. Image classification using CNN

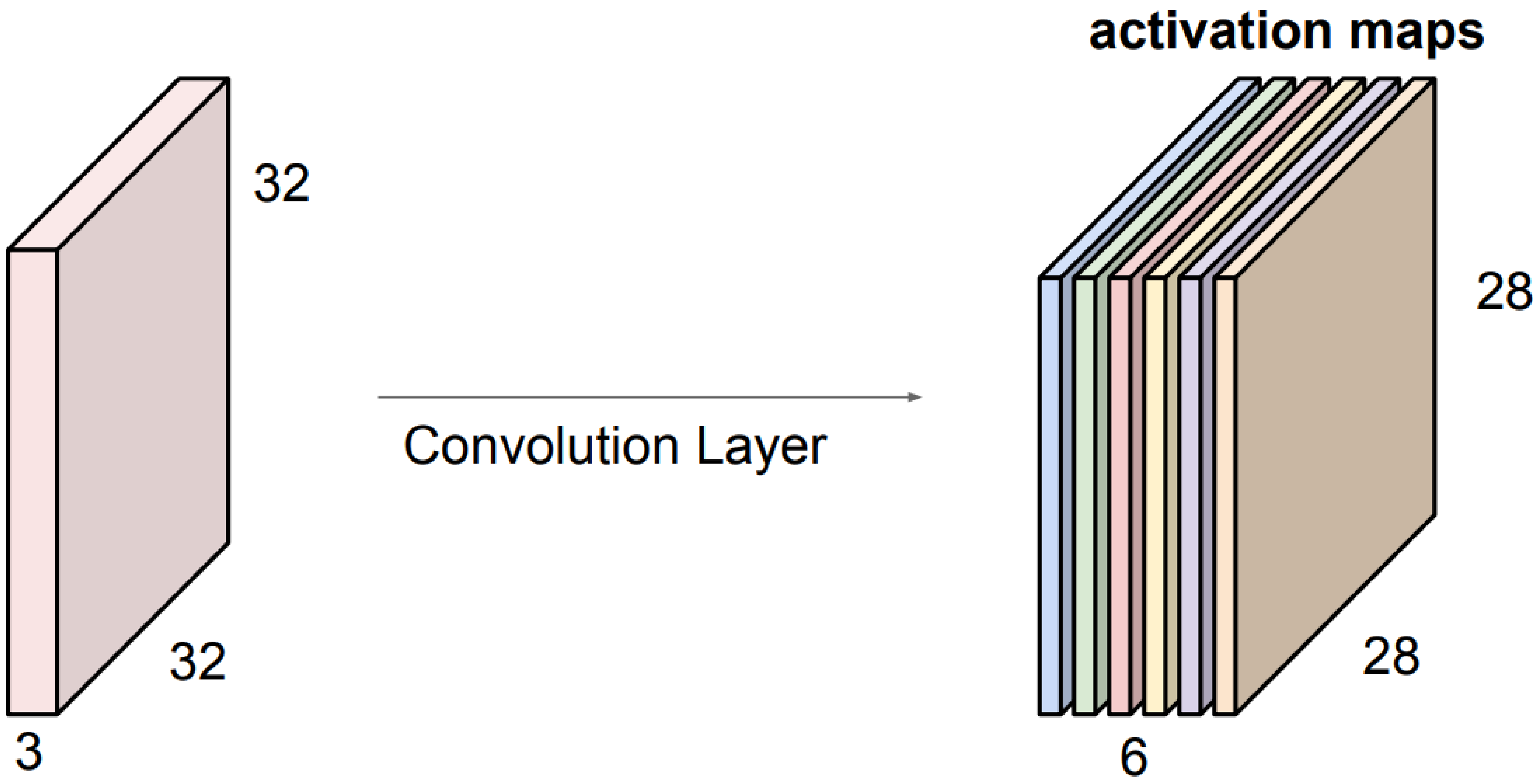
- Convolution



http://cs231n.stanford.edu/slides/2021/lecture_9.pdf

1-2. Image classification using CNN

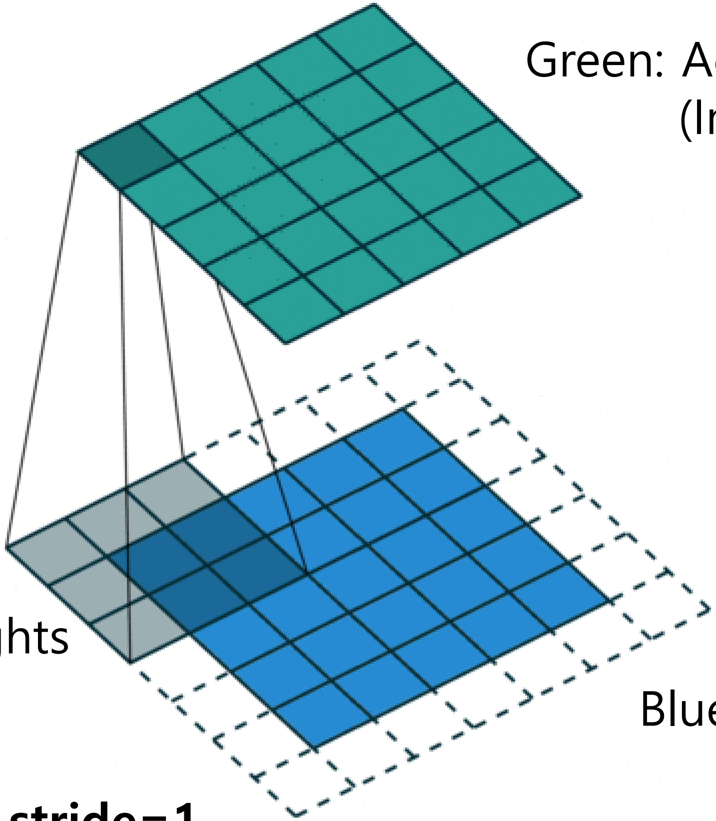
- Convolution



http://cs231n.stanford.edu/slides/2021/lecture_9.pdf

1-2. Image classification using CNN

- Convolution

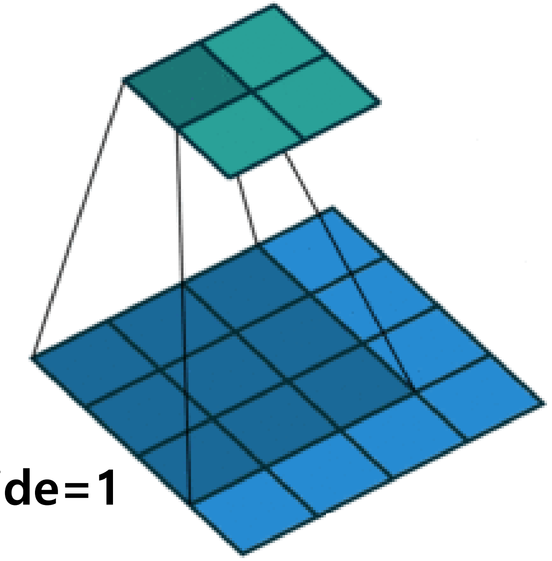


Green: Activation map
(Image features)

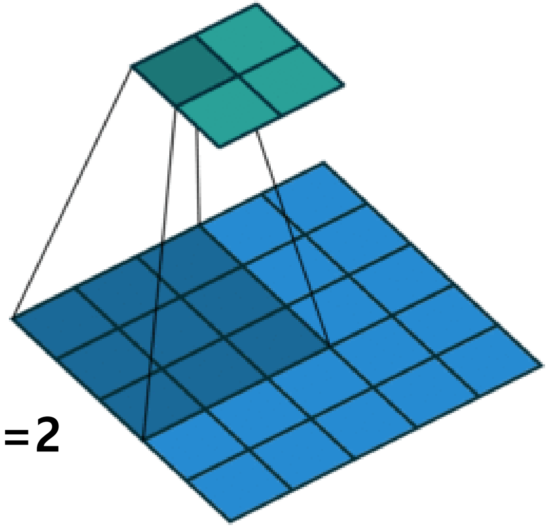
Gray: Model weights

Blue: Image

Padding=1, stride=1



No padding, stride=1



No padding, stride=2

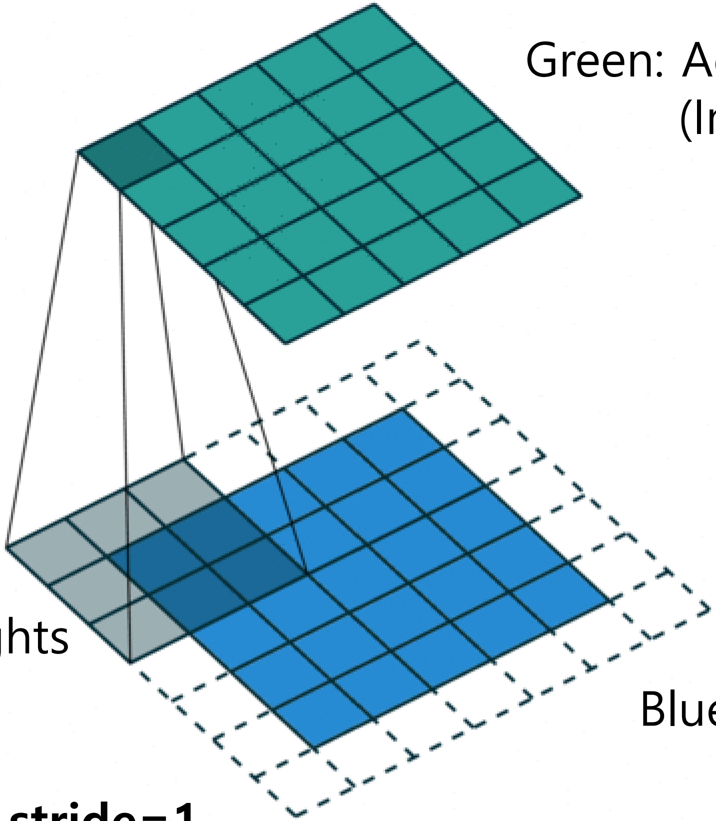
Image source: https://github.com/vdumoulin/conv_arithmetic

1-2. Image classification using CNN

- Convolution

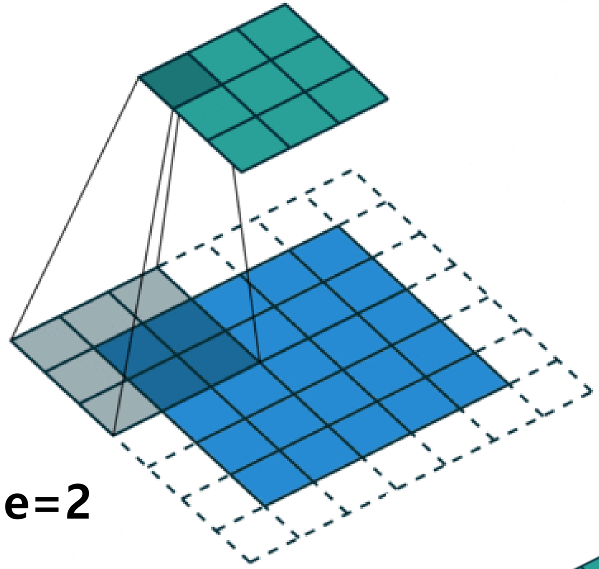
Green: Activation map
(Image features)

Gray: Model weights

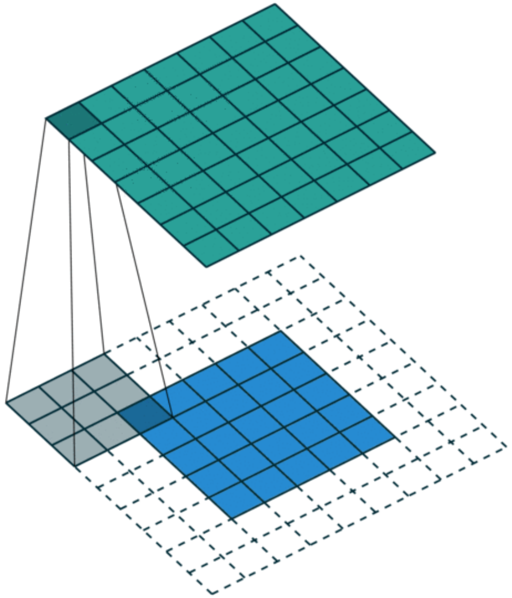


Blue: Image

Padding=1, stride=1



Padding=1, stride=2

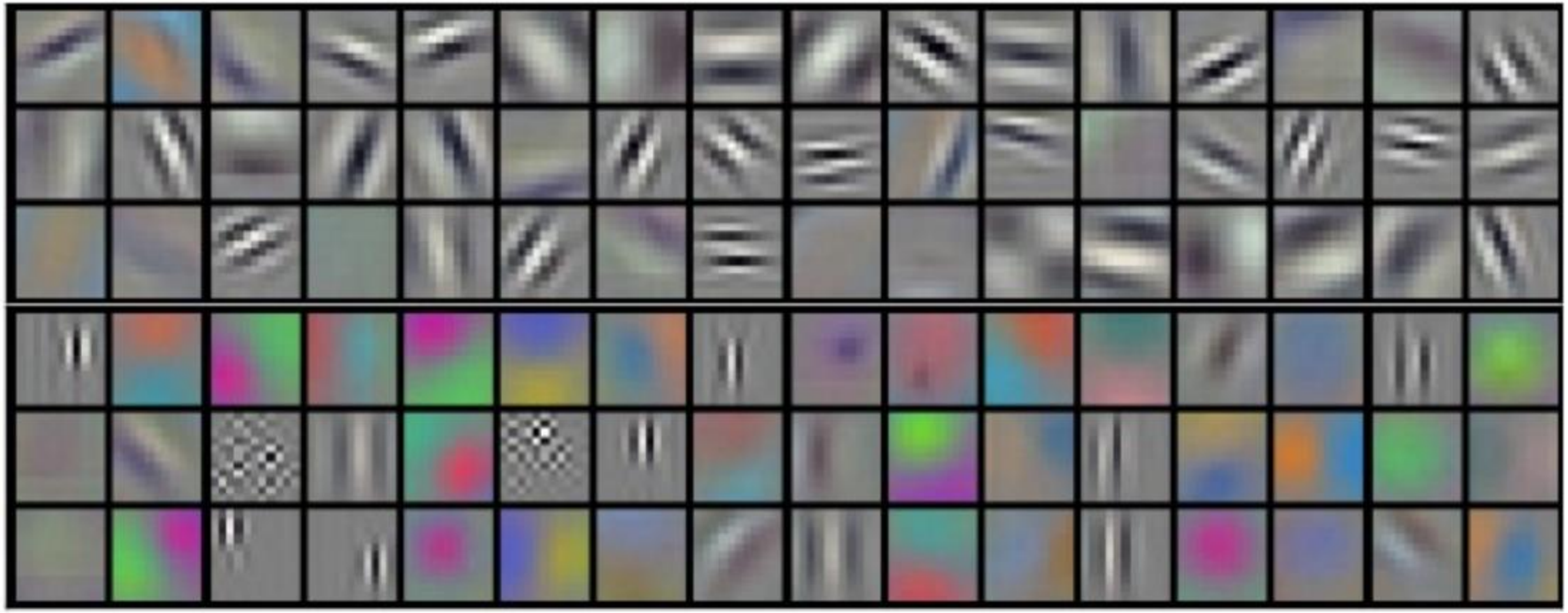


Padding=2, stride=1

Image source: https://github.com/vdumoulin/conv_arithmetic

1-2. Image classification using CNN

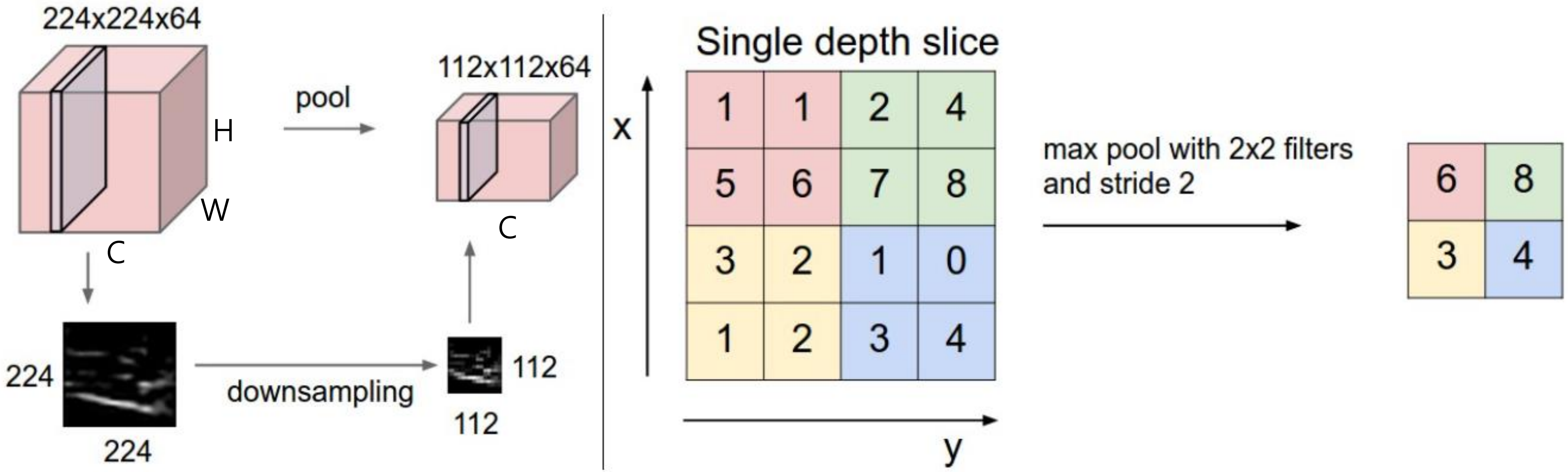
- Convolutional filters learned by Krizhevsky et al.



Source: https://cs231n.github.io/convolutional-networks/

1-2. Image classification using CNN

- Pooling layer downsamples the activation maps *spatially*.



Source: https://cs231n.github.io/convolutional-networks/

2. Overview

Preprocessing → CNN-based model → Loss backpropagation or Evaluation

Preprocessing
- Data augmentation
- Normalization

CNN-based model
- Conv layer
- pooling layer
- normalization layer
- linear layer
- Activation function

- Known architecture
  - ResNet
  - Inception v3
  - GoogleNet, …

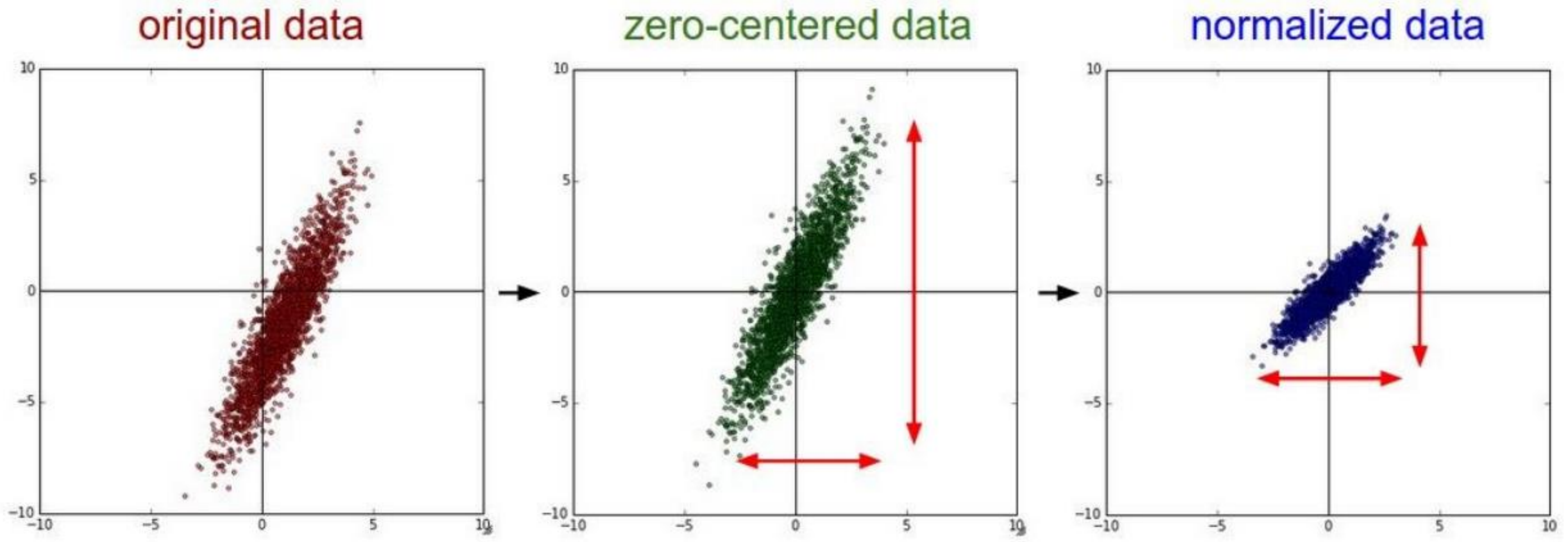
Loss backpropagation or Evaluation
- Compute metric
  - Accuracy, L2 loss, L1 Loss

2-1. Image augmentation

- Purpose of Image augmentation is
  - **to expand the training set size** by generating new samples from the original images
  - to improve the model to **generalize better**

- There are a lot of Python library for image augmentation
  - Torchvision
    - https://pytorch.org/tutorials/beginner/data_loading_tutorial.html
  - Albumentations
    - https://github.com/albumentations-team/albumentations

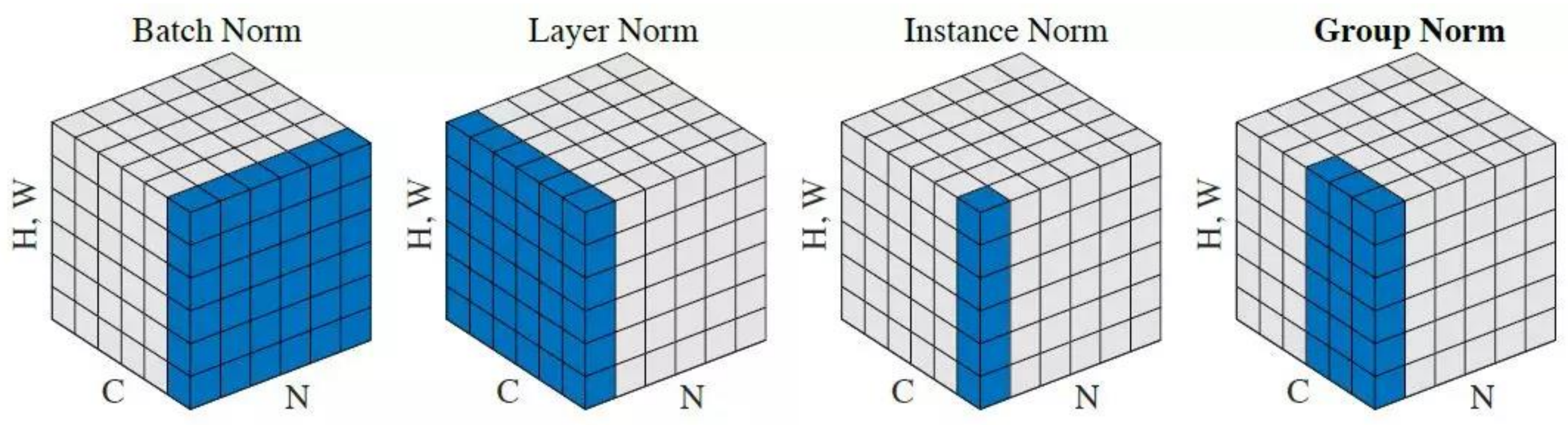
2-1. Data normalization (preprocessing)



http://cs231n.stanford.edu/slides/2021/lecture_8.pdf

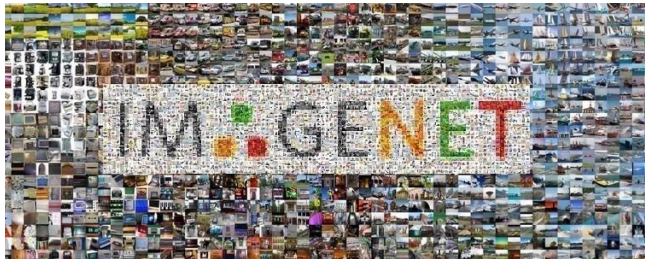
2-2. Data normalization (on activations)

- Different normalization methods

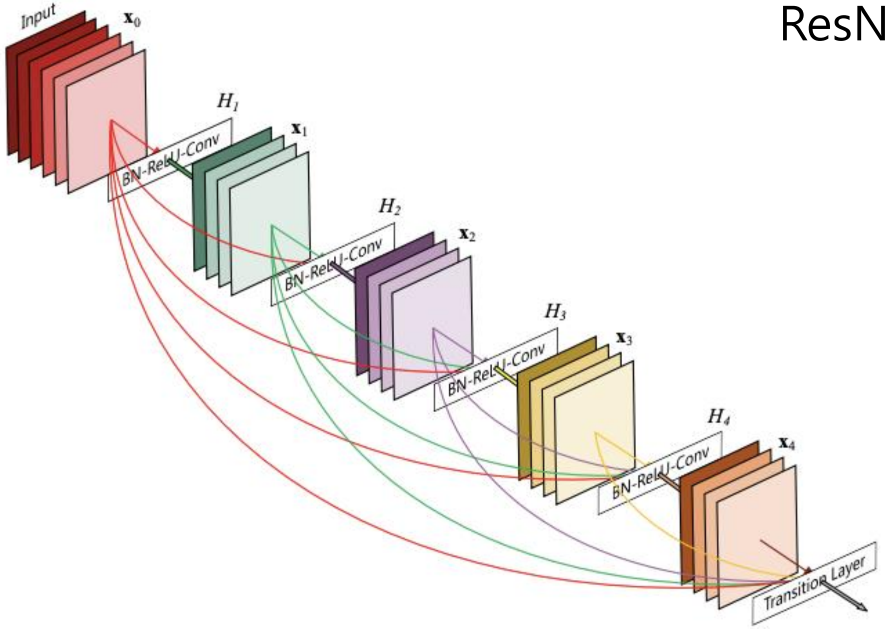


Yuxin Wu and Kaiming He. [Group normalization.](#) ECCV 2018.

2-2. Pretrained CNN models

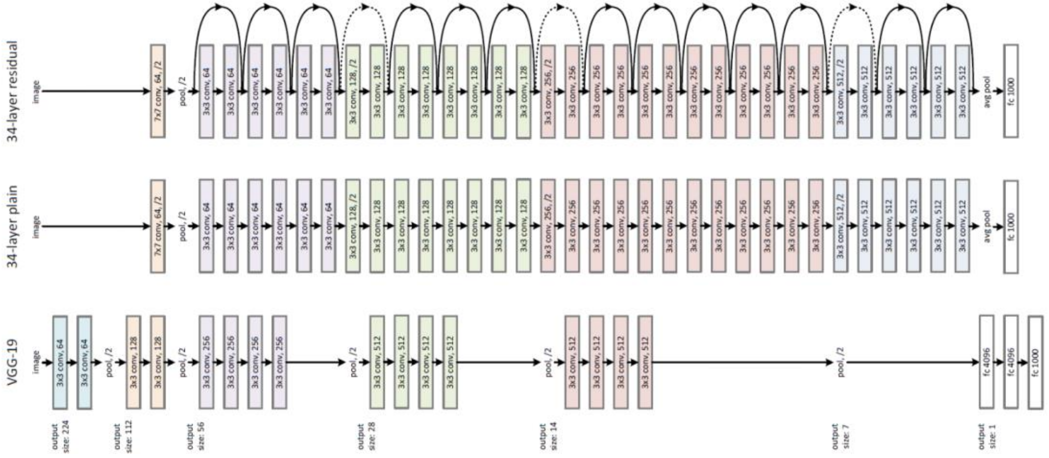


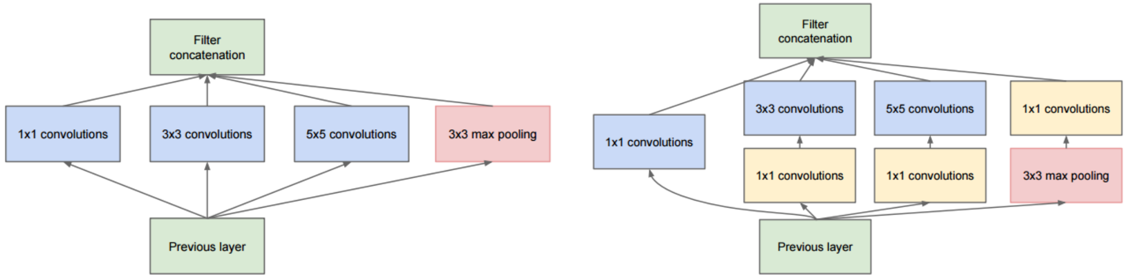
- https://pytorch.org/vision/stable/models.html



DenseNet

ResNet





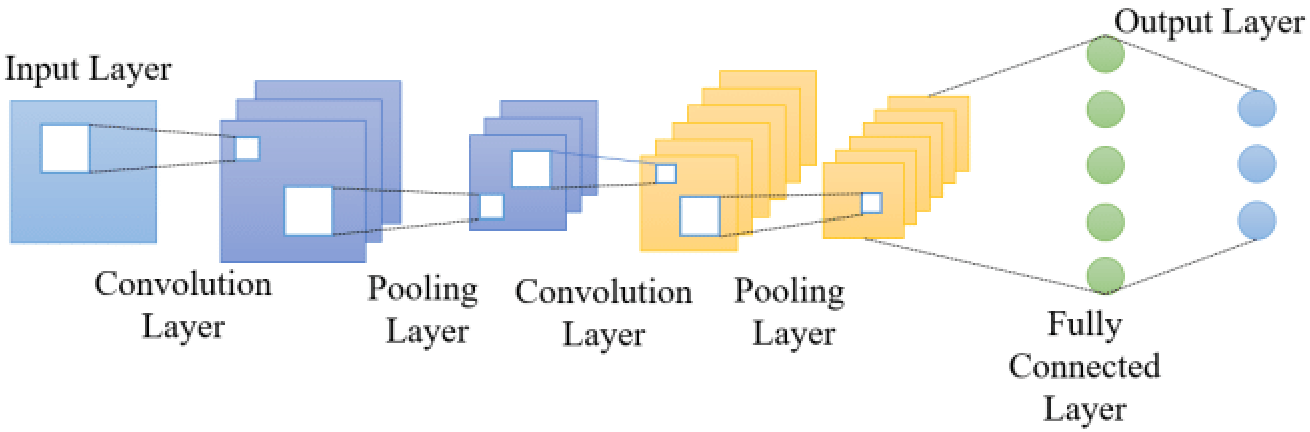
(a) Inception module, naïve version (b) Inception module with dimension reductions

Inception

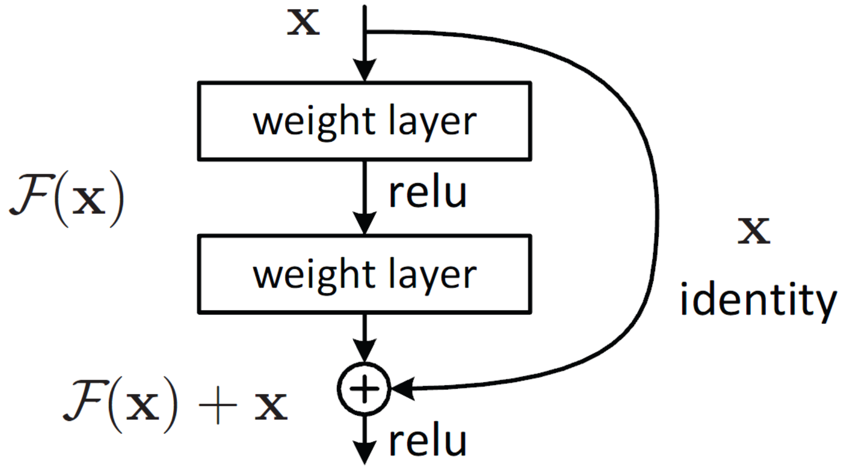
2-2. Make your own model

- https://pytorch.org/vision/stable/models.html

Basic architecture of CNN model



Residual block

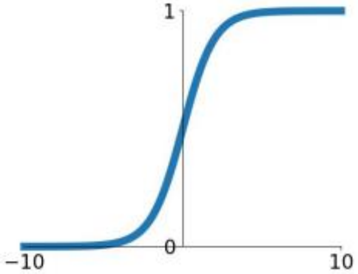


https://www.researchgate.net/figure/Basic-architecture-of-CNN_fig3_335086346

2-2. Activation functions

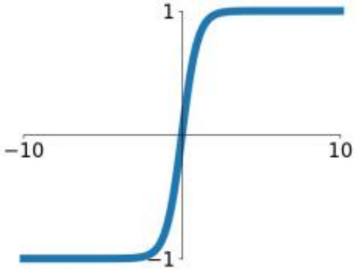
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



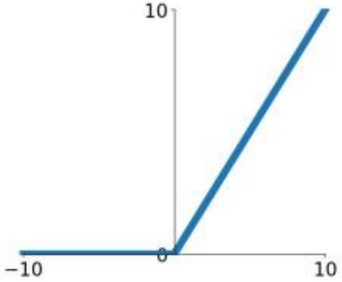
tanh

$\tanh(x)$



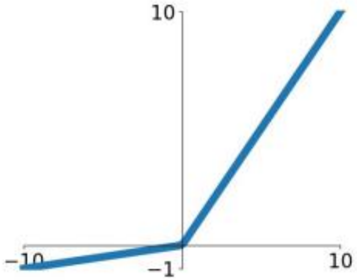
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

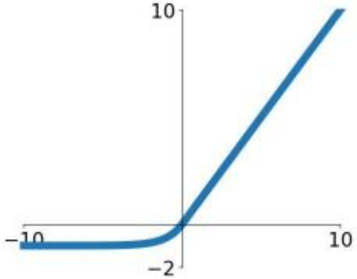


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

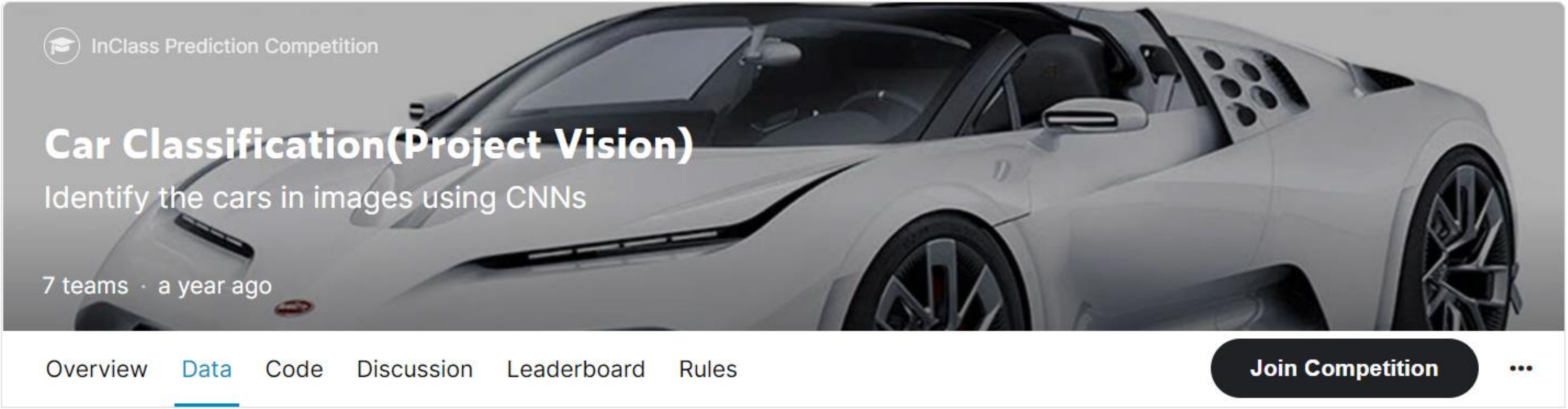
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



http://cs231n.stanford.edu/slides/2021/lecture_8.pdf

2. Car classification using Kaggle data

- https://www.kaggle.com/c/car-classificationproject-vision/data



2. Car classification using Kaggle data

- Dataset configuration
  - number of class: **45** different cars
  - number of images:
    - **training set: 100 images for each car**
    - test set: 450 images of different cars

	A	B	C
1	Cars	Class Numbers	
2	Alfa Romeo Stelvio	0	
3	Aston Martin DB11	1	
4	Aston Martin DBS	2	
5	Aston Martin Valkyrie	3	
6	Aston Martin Vantage	4	
7	Aston Martin Vulcan	5	
8	Audi A3	6	
9	Audi A6	7	
10	Audi E-tron GT	8	
11	Audi R8	9	
12	Bentley Bentayga	10	
13	Bentley Continental	11	
14	BMW 3-series	12	
15	BMW 7-series	13	
16	BMW x7	14	
17	Bugatti Centidieci	15	
18	Bugatti Chiron	16	
19	Bugatti Divo	17	
20	Bugatti La Voiture Noire	18	
21	Buggati Veyron	19	
22	Cadillac Escalade	20	
23	Corvette ZR	21	
24	Ferrari 458	22	
25	Ferrari FF	23	
26	Ferrari Pininfarina	24	
27	Jaguar F-type	25	
28	Jaguar XJ	26	
29	Koenigsegg CC8S	27	
30	Koenigsegg CCX	28	
31	La Ferrari	29	
32	Lamborghini Gallardo	30	
33	Lamborghini Murceilago	31	
34	Lamborghini Veneno	32	
35	Mustang GT	33	
36	Pagani Zonda	34	
37	Porsche 911	35	
38	Porsche Cayenne	36	
39	Range Rover Discovery	37	
40	Renault Duster	38	
41	Rolls Royce Ghost	39	
42	Rolls Royce Phantom	40	
43	Tata Tiago	41	
44	Toyota Fortuner	42	
45	Volkswagen Polo	43	
46	Volkswagen Vento	44	

2. Car classification using Kaggle data

- Submission on Kaggle
  - The model predictions on the 450 images of the test set
  - The prediction results should be stored in a csv file as example

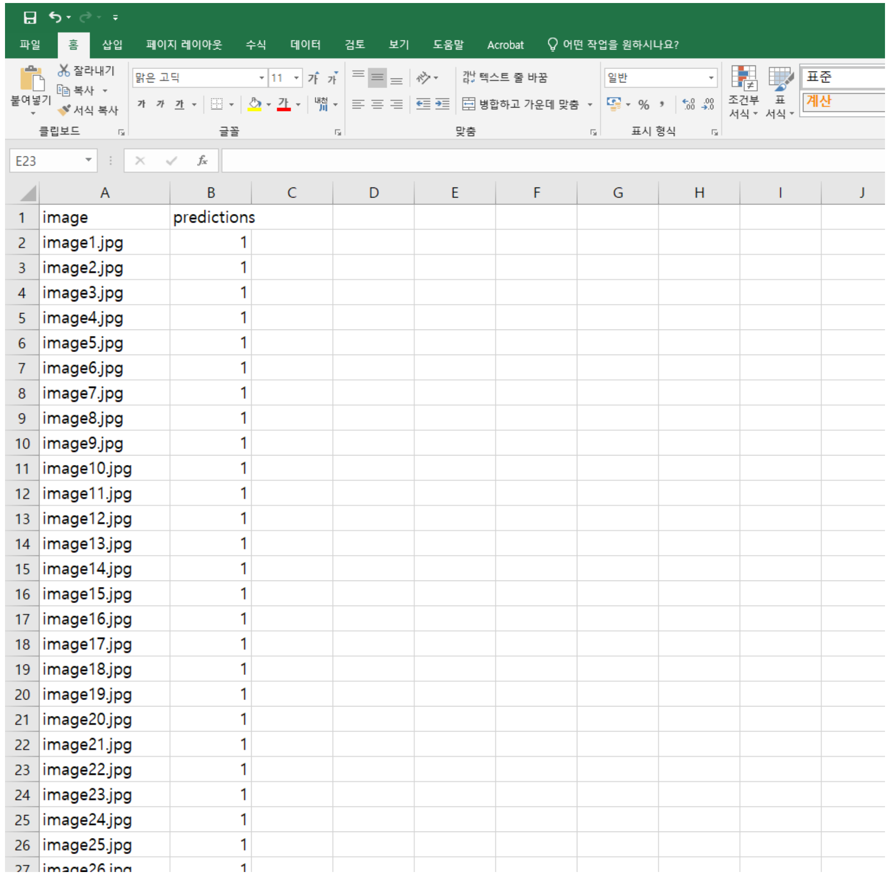


image	predictions
image1.jpg	1
image2.jpg	1
image3.jpg	1
image4.jpg	1
image5.jpg	1
image6.jpg	1
image7.jpg	1
image8.jpg	1
image9.jpg	1
image10.jpg	1
image11.jpg	1
image12.jpg	1
image13.jpg	1
image14.jpg	1
image15.jpg	1
image16.jpg	1
image17.jpg	1
image18.jpg	1
image19.jpg	1
image20.jpg	1
image21.jpg	1
image22.jpg	1
image23.jpg	1
image24.jpg	1
image25.jpg	1
image26.jpg	1