



Convolutional Neural Network



Sofa?

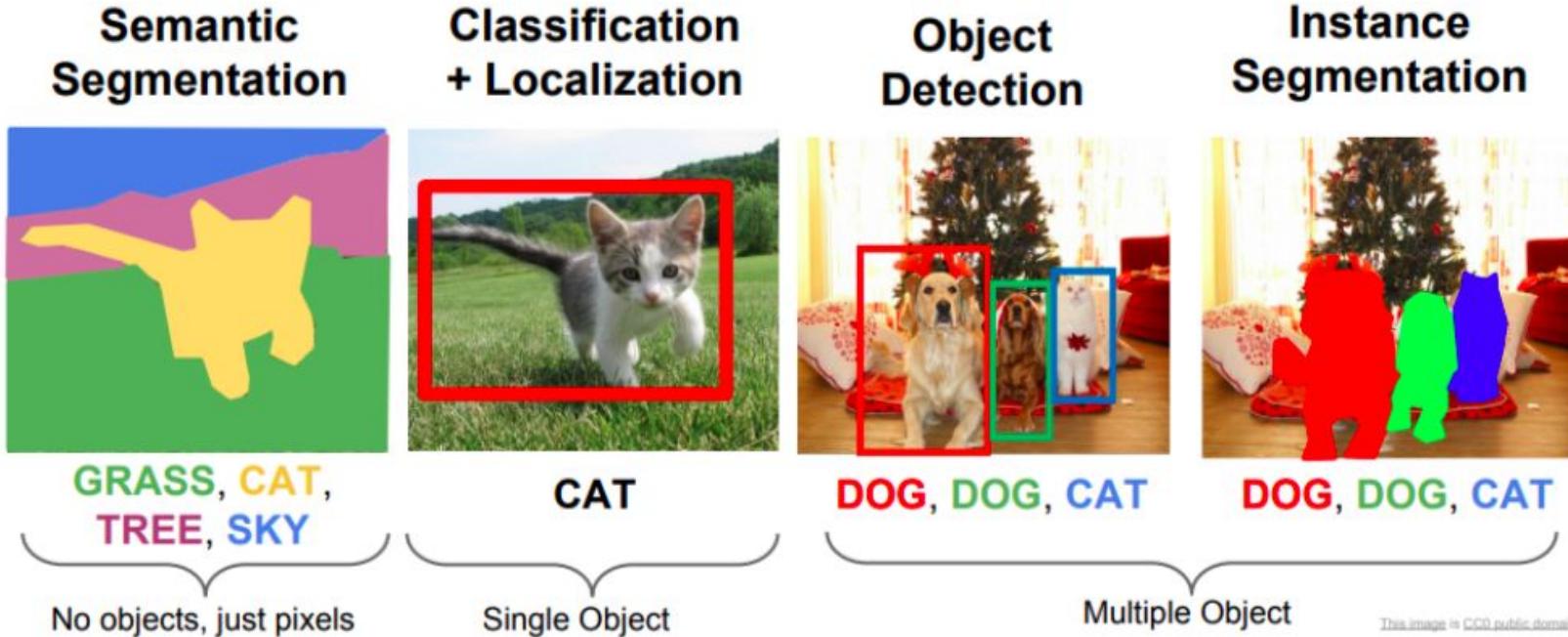
- ❖ **Name:** Teera Laiteerapong
- ❖ **Education:**
 - Computer Engineering (B.Eng.), KMITL
 - Computer Science (M. Eng.), AIT
- ❖ **Research Interests:**
 - Machine Learning in Computer Vision
 - Medical Imaging, Civil Infrastructure Imaging



<https://github.com/mrteera>



Computer Vision Problems



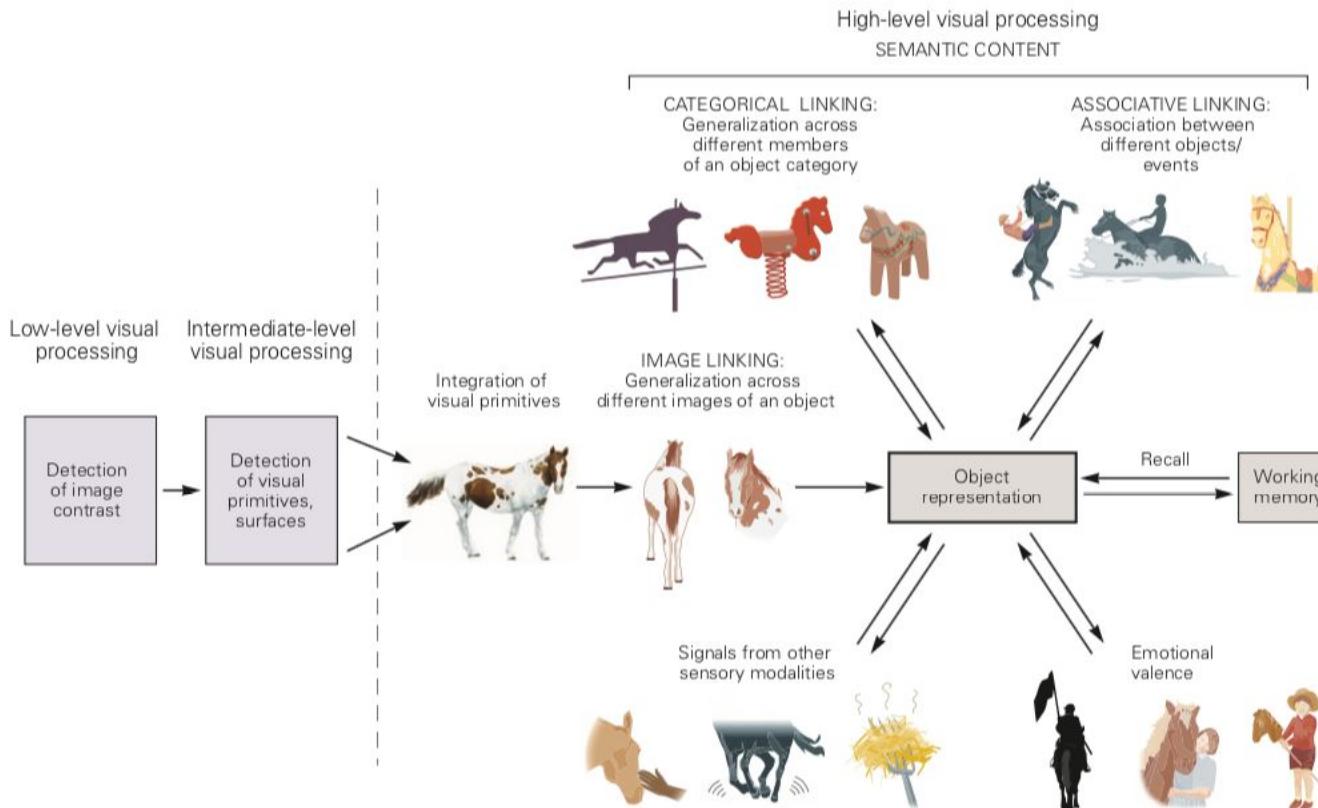
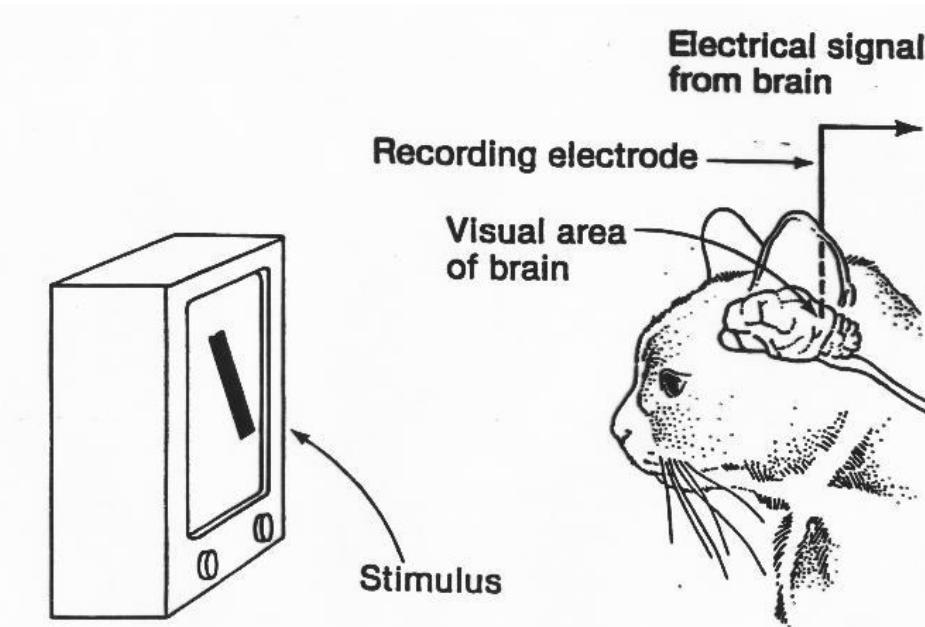


Figure 28–1 The neuronal representation of entire objects is central to high-level visual processing. Object representation involves integration of visual features extracted at earlier stages in the visual pathways. Ideally the resulting representation is a generalization of the numerous retinal images generated by the same object and of different members of an object category.

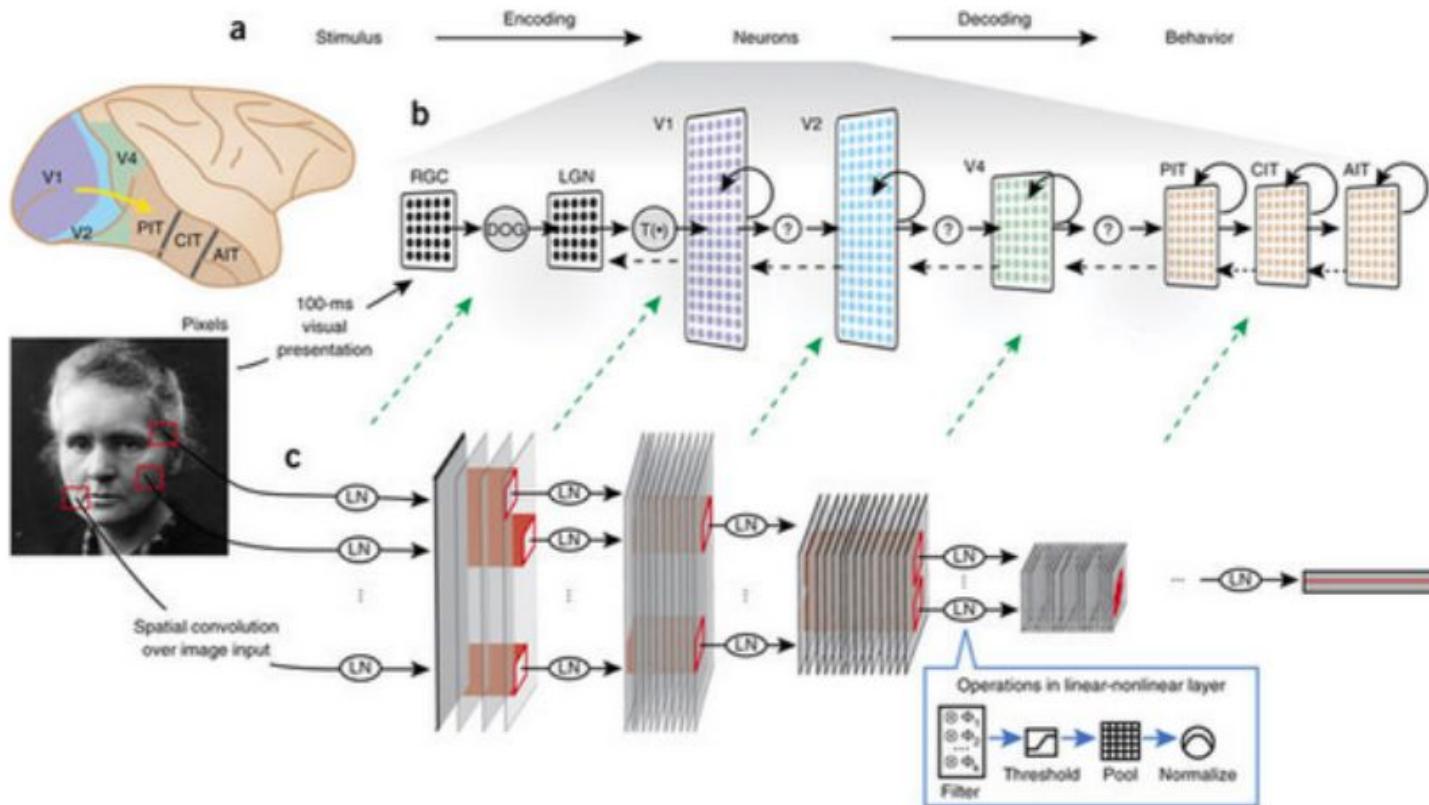
The representation also incorporates information from other sensory modalities, attaches emotional valence, and associates the object with the memory of other objects or events. Object representations can be stored in working memory and recalled in association with other memories.

David Hubel and Torsten Wiesel (1950s)

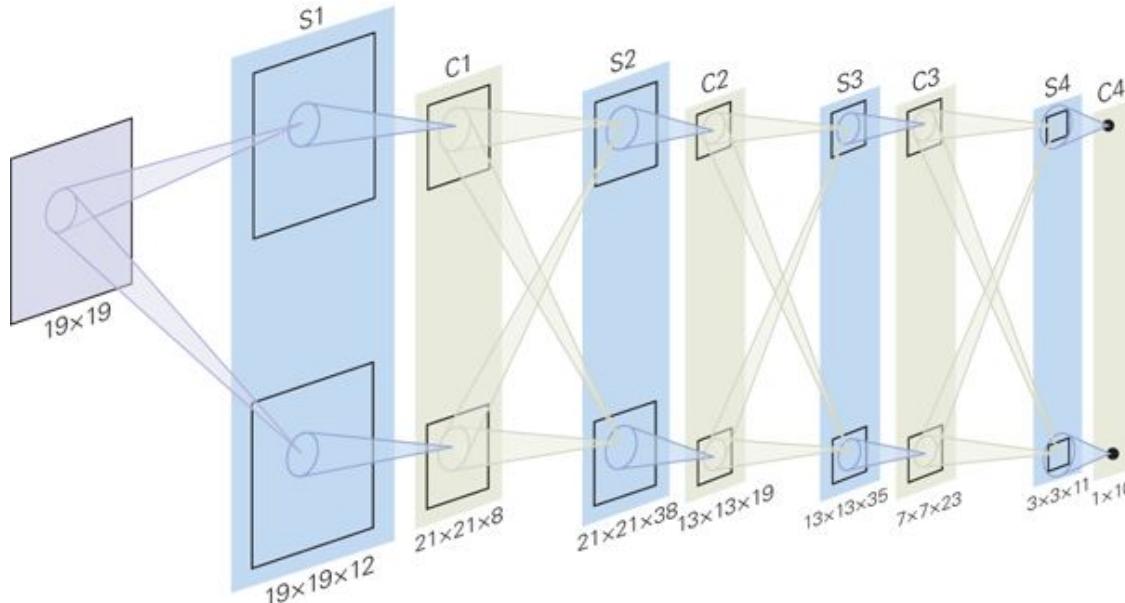


Hubel and Wiesel study how cat's visual cortex react to the different orientation of the line. They earned Nobel Prize for Physiology or Medicine in 1981.

CNN Inspiration from Human Brain



Neocognitron



Kunihiko Fukushima,
1980

แต่ละ layer จะเพิ่มความซับซ้อนในการ detect เส้น

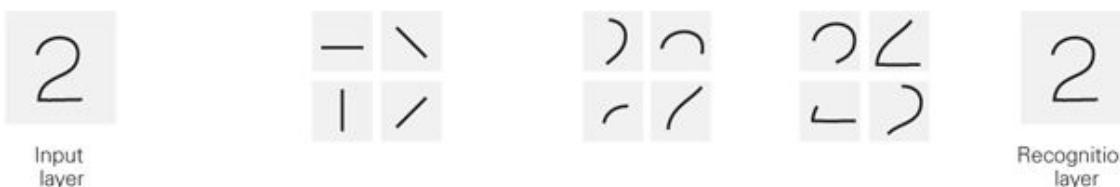
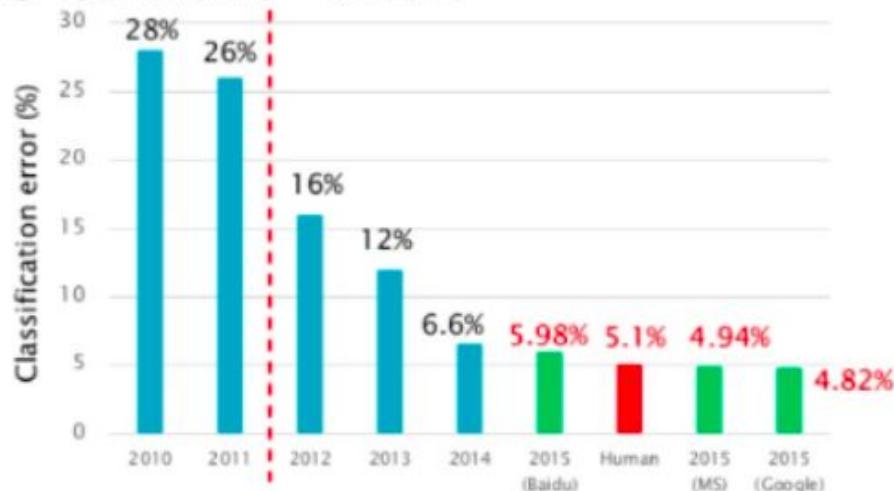


Image Classification Development

ILSVRCにおけるブレークスルー

- エラー率が 16% (2012) → 4.8% (2015)



He et al., "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification", arXiv, 2015.

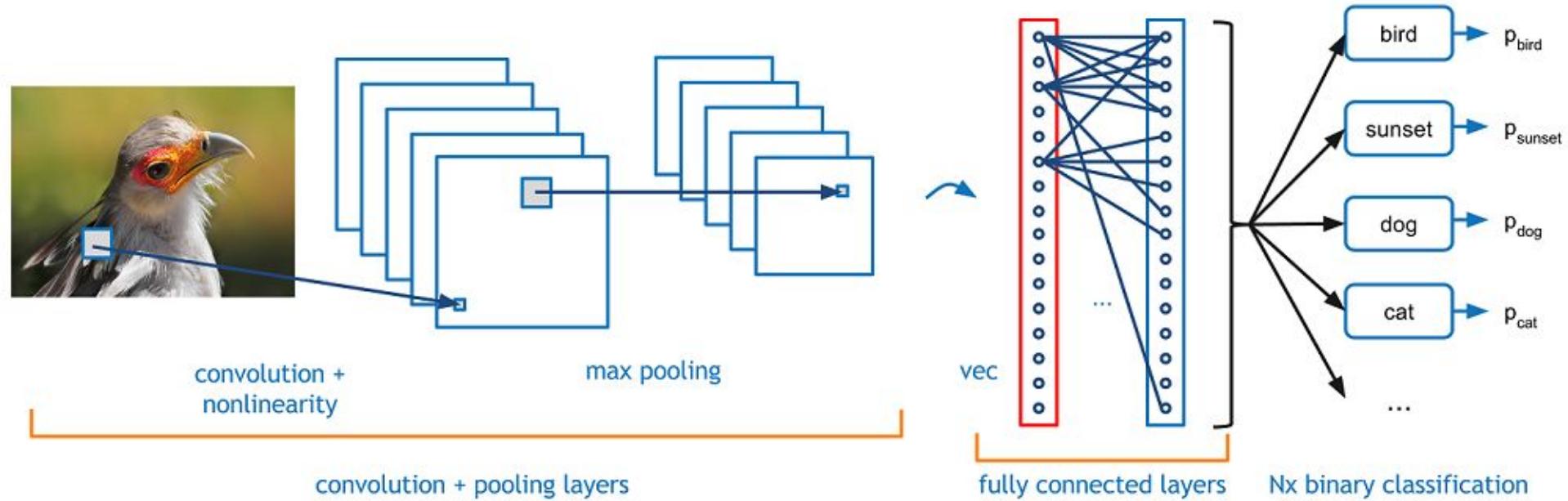
Ioffe et al., "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", arXiv, 2015.

24



Alex Krizhevsky invented AlexNet and won Large Scale Visual Recognition Challenge 2012 (ILSVRC2012).

Top-1 Accuracy vs Top-5 Accuracy



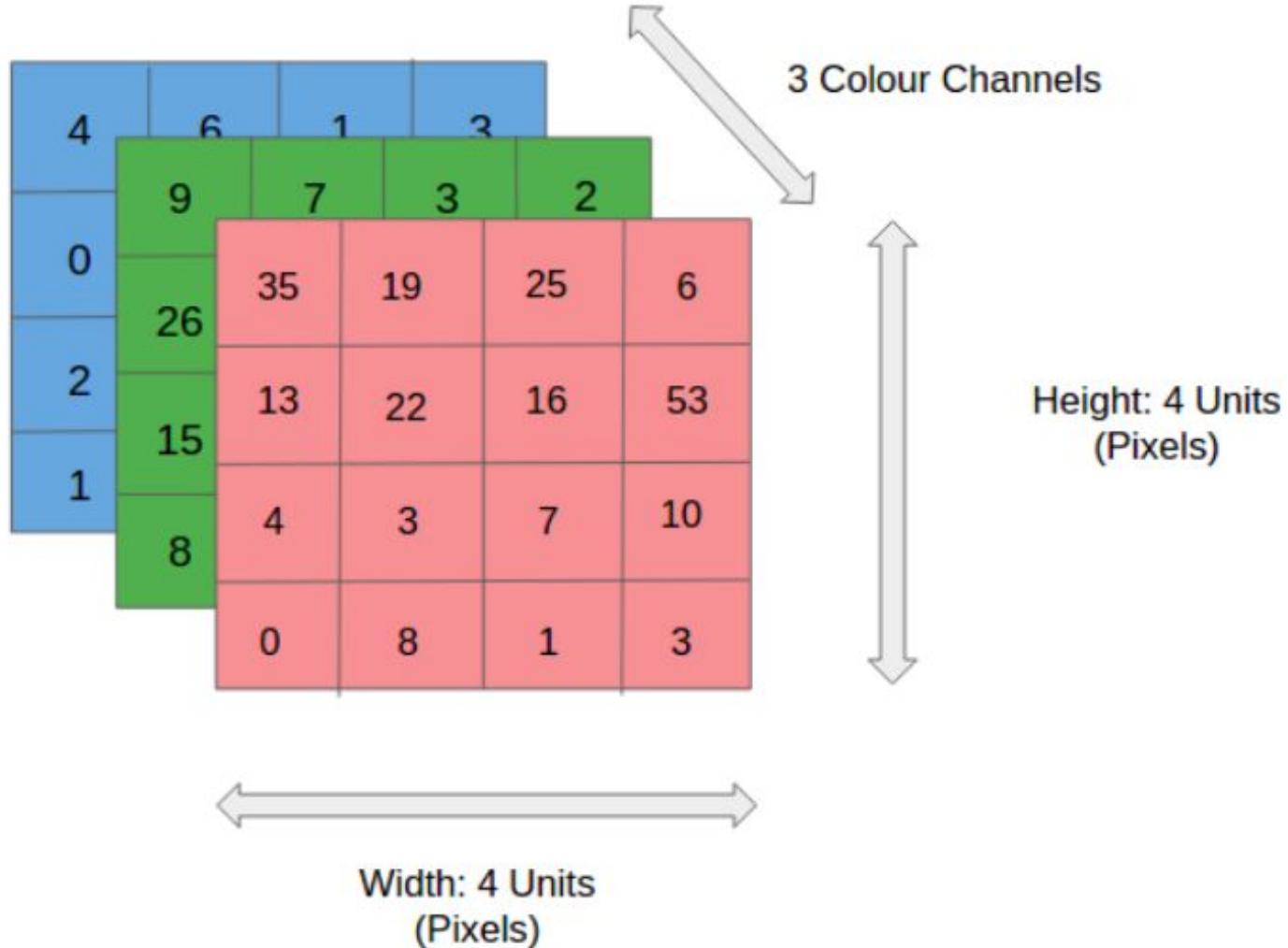
For top-5 accuracy, if top-5 highest prediction probabilities consist of bird, then the accuracy metric counts as correct. But top-1 accuracy checks only if the highest prediction probability matches the label.

Model	Detail	Input size	Top-1 Acc	Top-5 Acc	Param(M)	Mult-Adds	FLOPS(G)	Depth	TF	Keras	Pytorch	Caffe
AmoebaNet-B	(N=6, F=228)	331x331	83.1	96.3	155.3	41.1B			TF			
PNASNet-5_Large_331	(N=4, F=216)	331x331	82.9	96.2	86.1	25.0B	25.169		TF			
AmoebaNet-B	(N=6, F=190)	331x331	82.8	96.1	86.7	23.1B			TF			
SENet-154		320x320	82.7	96.2	145.8	42.3B			TF	Keras	Pytorch	Caffe
NASNet-A_Large_331	(N=6, F=168)	331x331	82.7	96.2	88.9	23.8B	24.021		TF	Keras	Pytorch	
AmoebaNet-B	(N=6, F=190)	331x331	82.3	96.1	84	22.3B			TF			
Dual-Path-Net-131		320x320	81.5	95.8	79.5	32.0B				Keras	Pytorch	Caffe
PolyNet		331x331	81.3	95.8	92	34.7B	34.768			Pytorch		Caffe
SENet-154		224x224	81.16	95.35	115.088		20.742		TF	Keras	Pytorch	
ResNeXt-101	(64x4d)	320x320	80.9	95.6	83.6	31.5B			TF	Keras	Pytorch	Caffe
PyramidNet-200	$\alpha=450$	320x320	80.8	95.3	116.4				TF		Pytorch	Caffe
ResNet152_v1d		224x224	80.61	95.34								
ResNet101_v1d		224x224	80.51	95.12								
PyramidNet-200	$\alpha=300$	320x320	80.5	95.2	62.1				TF		Pytorch	Caffe
Inception-ResNet-v2		299x299	80.4	95.3	55.8		11.75	572	TF	Keras		Caffe
ResNet152_v1d	(no mixup)	224x224	80.26	95								

<https://kobiso.github.io/Computer-Vision-Leaderboard/imagenet>

Image as a Matrix





What is convolution?



Photo editor apps

เรียก kernel หรือ filter

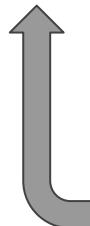
Kernel/Filter



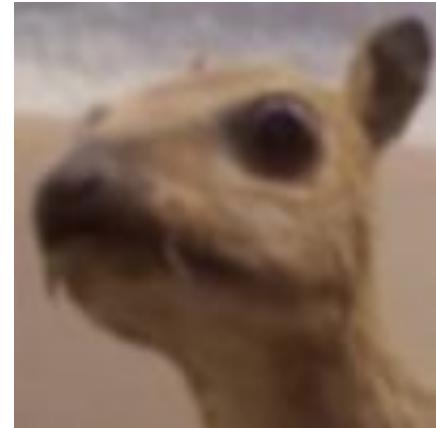
$$\frac{1}{256} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$



Input Image



Convolution



Blurred Image

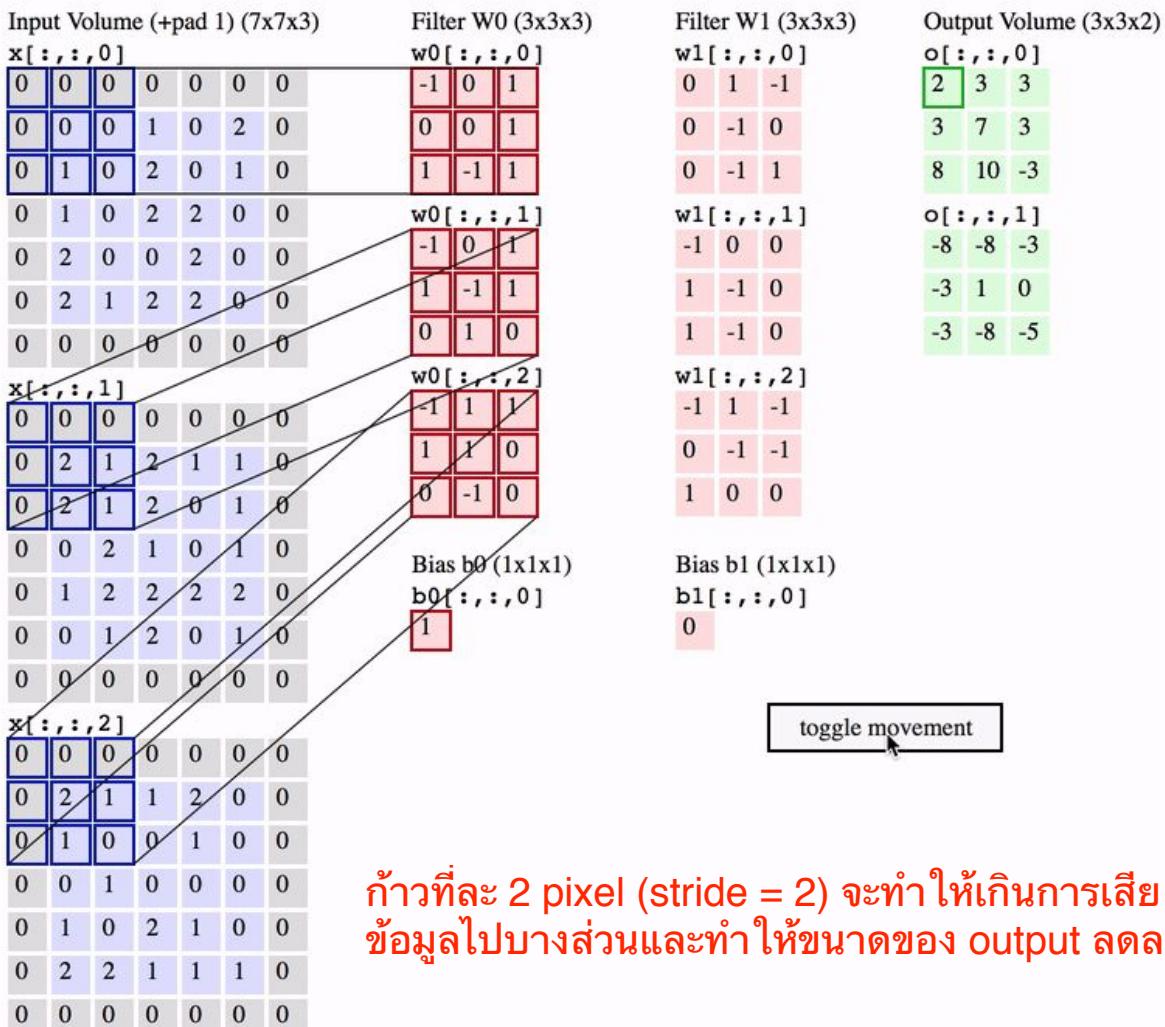
Convolution operation is actually an inner dot product.

See more filters in Gimp:

<https://docs.gimp.org/2.6/en/plug-in-convmatrix.html>

Convolution Operation

padding คือ เลข 0 สีเทา จะใส่หรือไม่ก็ได้
แต่ใส่เพื่อรักษาข้อมูลของภาพ แต่ถ้าไม่ต้องการ
ไม่ต้องใส่ก็ได้



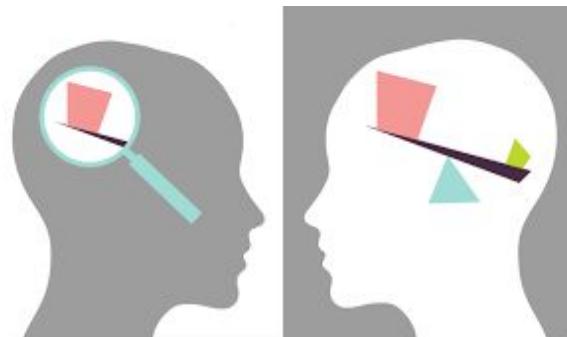
กว้างที่ละ 2 pixel (stride = 2) จะทำให้เกินการเลี้ยง
ข้อมูลไปบางส่วนและทำให้ขนาดของ output ลดลง

Bias

$$Y = \theta x + \text{Thai}$$

If the model want to predict "Which nationality have the most beautiful women?" you say Thai Ladies, we can say its because you are biased.

In CNN, bias is a learnable parameters which help the model assumption minimize an objective function faster.



Calculate Feature Maps Dimension (1)

Output Dimension:

x: $(\text{input_size} + 2 * \text{padding_size} - \text{filter_size}) / \text{stride} + 1$

y: $(\text{input_size} + 2 * \text{padding_size} - \text{filter_size}) / \text{stride} + 1$

input size = 6X6

padding size = 1

filter size = 3x3

stride = 2

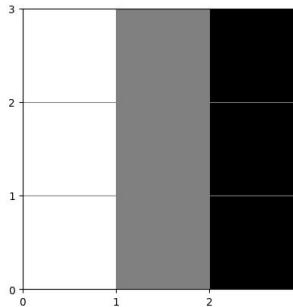
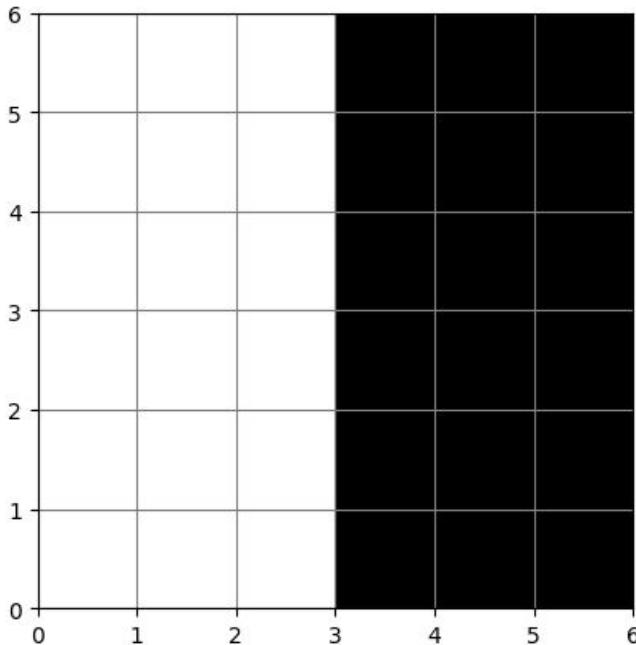
Calculate Feature Maps Dimension (2)

Given an input image size 6×6 , kernel size 3×3 , stride 2, padding 1.

What is the dimension of an output?

Convolution as Edge Detection

Vertical Edge Detection



= ?

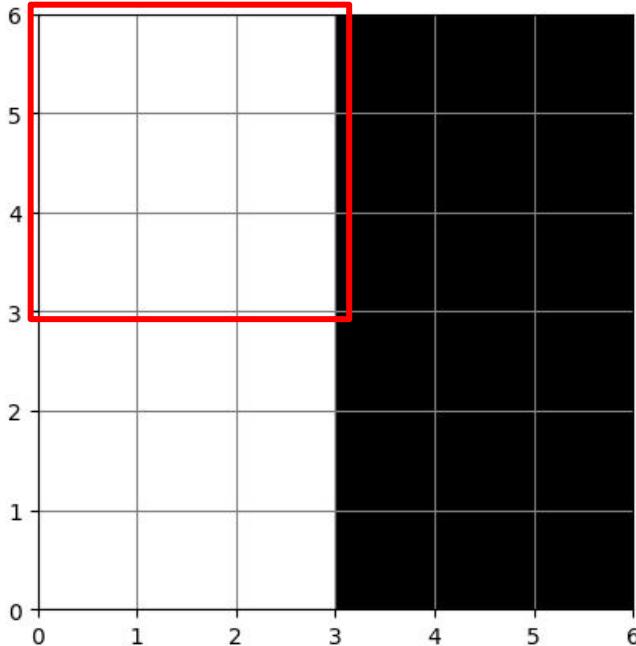
Kernel

```
[[ 1  0 -1]
 [ 1  0 -1]
 [ 1  0 -1]]
```

Input Image

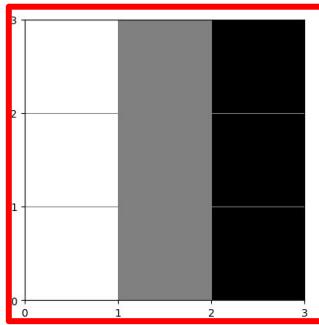
```
[[10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [10 10 10 0 0 0]]
```

Vertical Edge Detection



Input Image

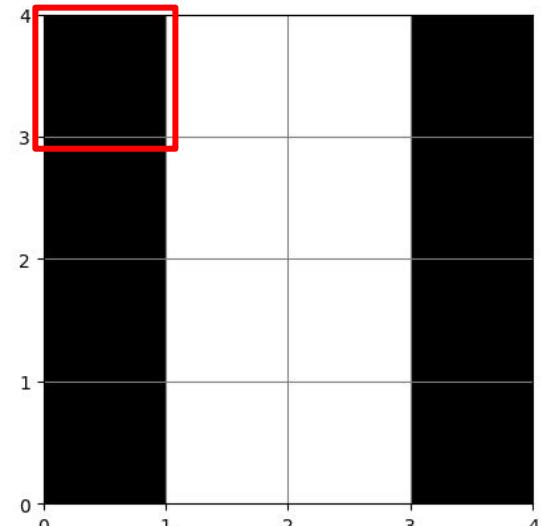
```
[[10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [10 10 10 0 0 0]]
```



Kernel

```
[[ 1  0 -1]
 [ 1  0 -1]
 [ 1  0 -1]]
```

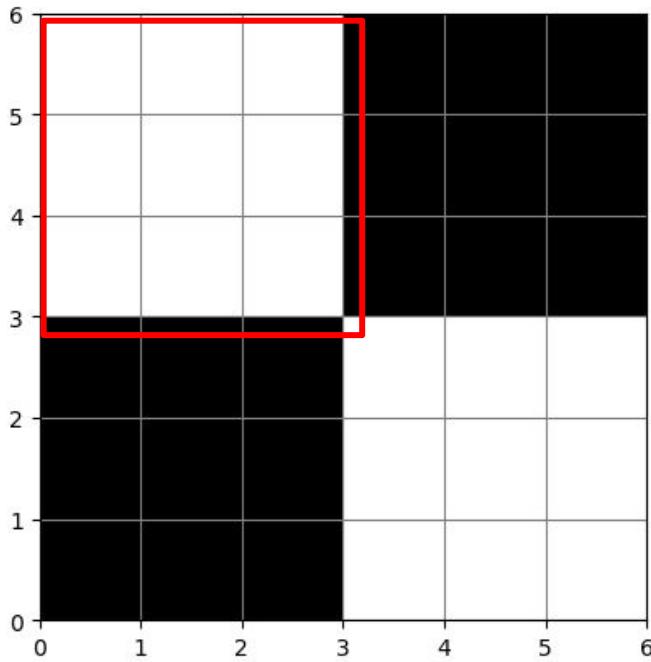
=



Output/Feature maps

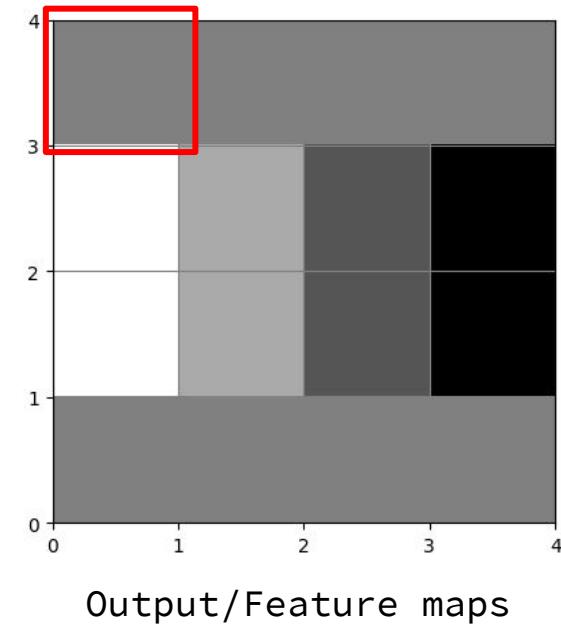
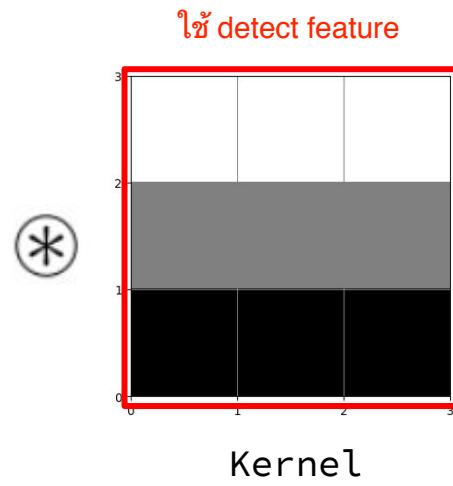
```
[[ 0 30 30 0]
 [ 0 30 30 0]
 [ 0 30 30 0]
 [ 0 30 30 0]]
```

Horizontal Edge Detection



Input Image

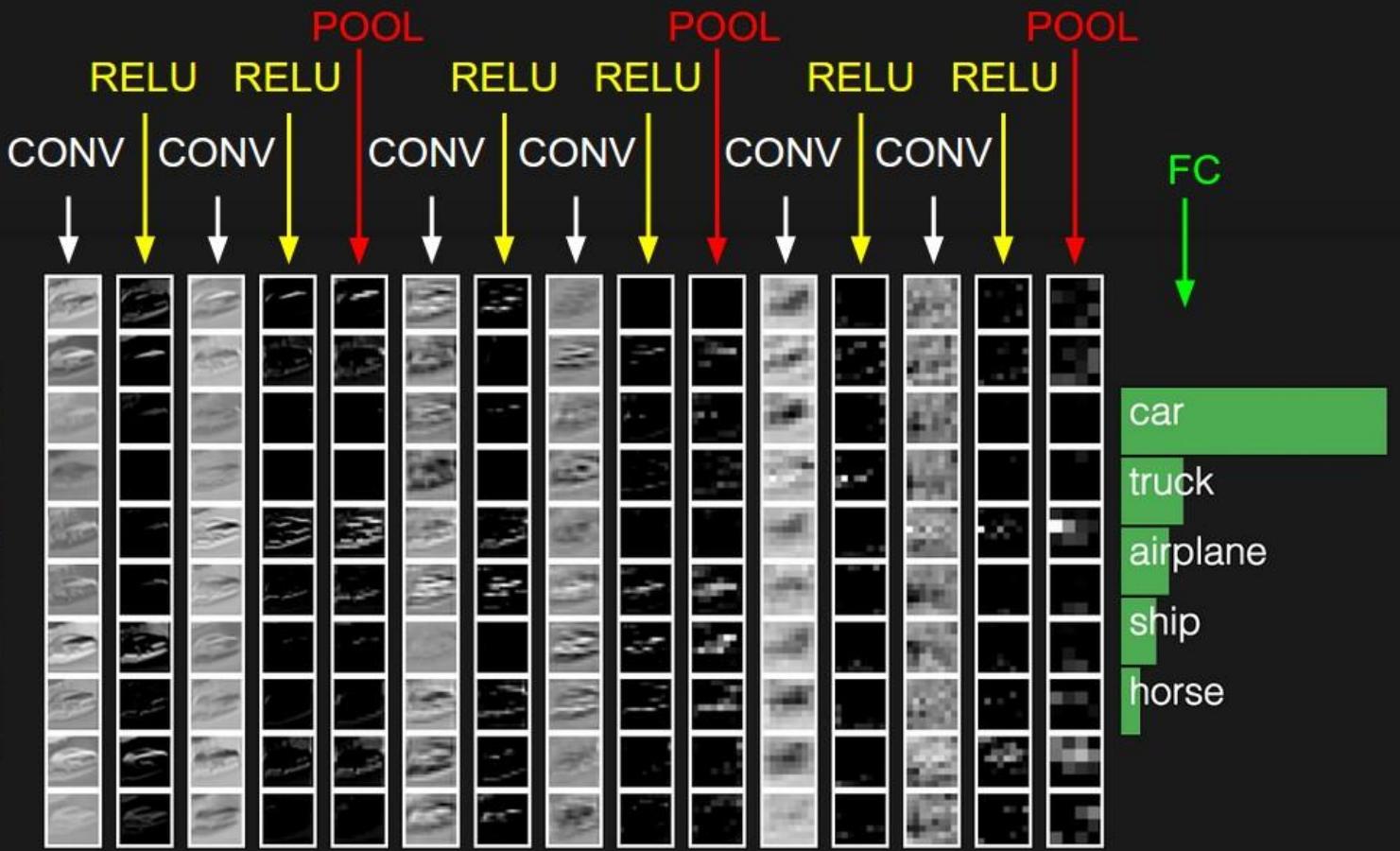
```
[[10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [10 10 10 0 0 0]
 [ 0  0  0 10 10 10]
 [ 0  0  0 10 10 10]
 [ 0  0  0 10 10 10]]
```



Output/Feature maps

```
[[ 0  0  0  0]
 [ 30 10 -10 -30]
 [ 30 10 -10 -30]
 [ 0  0  0  0]]
```

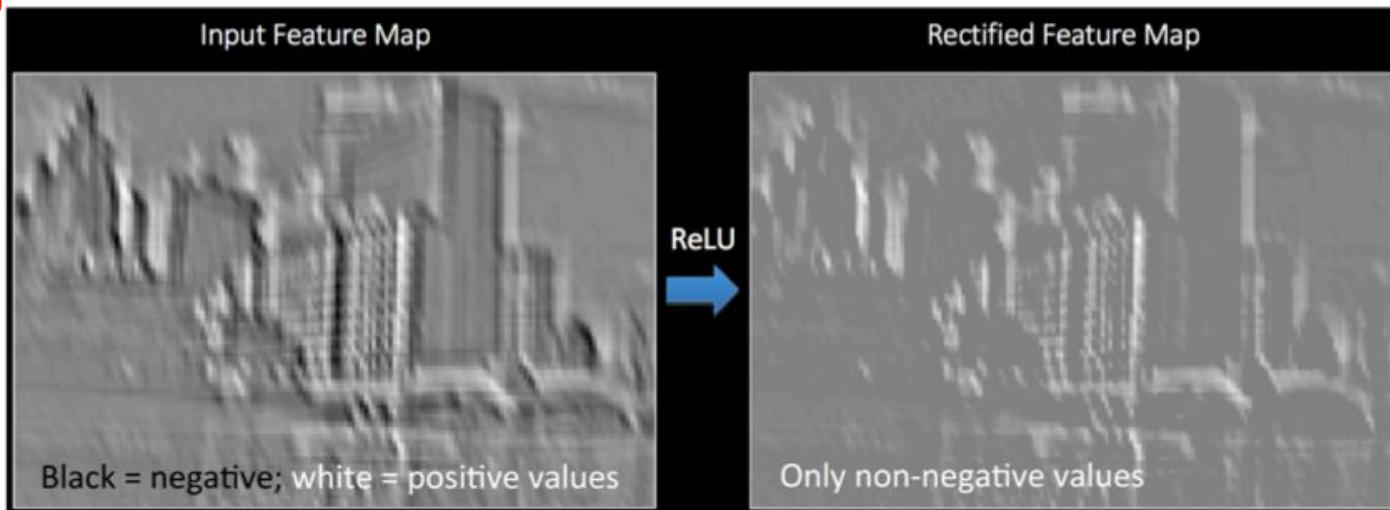
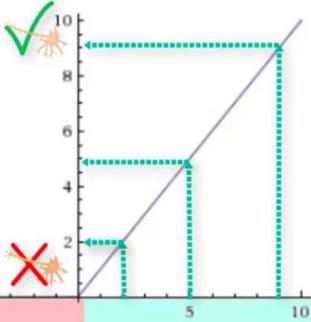
Convolutional Neural Network



ReLU (Rectified Linear Unit)

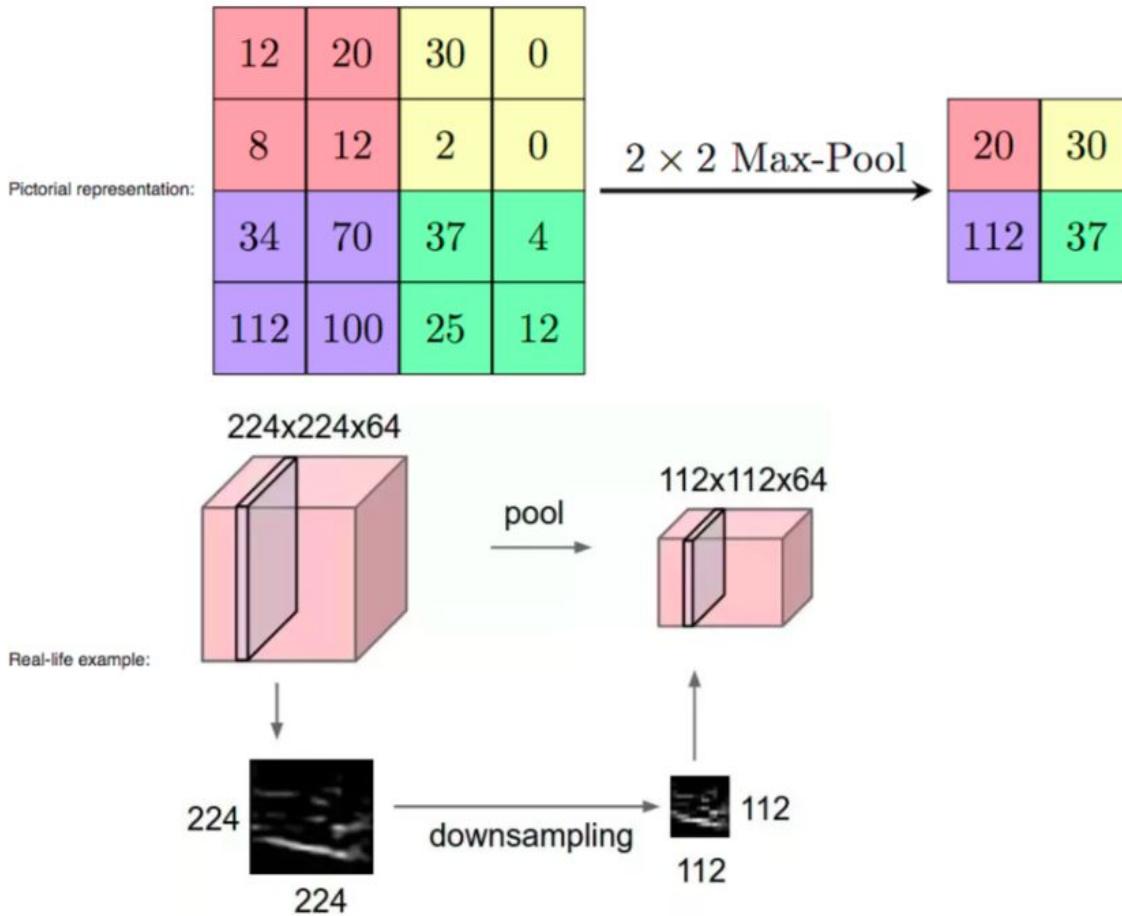
ReLU จะประค่าที่ติดลบเป็น 0

$$f(x) = \max(0, x)$$



ภาพนีผ่านการ convolution มาแล้ว

Pooling Layer



A Simple Convolutional Neural Network

```
simple_model = Sequential() ประการศให้ model เป็นเส้นตรงลงมา (sequential)
simple_model.add(Conv2D(3,
                      activation='relu',
                      kernel_size=3,
                      input_shape = (img_rows, img_cols, 1)))
simple_model.add(Conv2D(3, activation='relu', kernel_size=3))
simple_model.add(Flatten())
simple_model.add(Dense(num_classes, activation='softmax'))
    Dense = class ที่เราต้องการ detect
simple_model.compile(loss='categorical_crossentropy',
                      optimizer='sgd',
                      metrics=['accuracy'])

simple_model.summary()
simple_model.fit(x, y, batch_size=100, epochs=4, validation_split=0.2)
```

A simple model using Keras and TensorFlow.

Input shape: 28 x 28

Layer (type)	Output Shape	Param #
conv2d_7 (Conv2D)	(None, 26, 26, 3)	30

conv2d_8 (Conv2D)	(None, 24, 24, 3)	84
-------------------	-------------------	----

flatten_3 (Flatten)	(None, 1728)	0
---------------------	--------------	---

dense_4 (Dense)	(None, 10)	17290
-----------------	------------	-------

Total params: 17,404

Trainable params: 17,404

Non-trainable params: 0

A simple model summary. None in output shape a batch size, it will be specify when you train a model.

Hyperparameters

hyperparameter คือ parameter ที่เราต้องระบุเอง ซึ่งไม่ได้มาจากการ learn model

Batch Size, Iteration, and Epoch

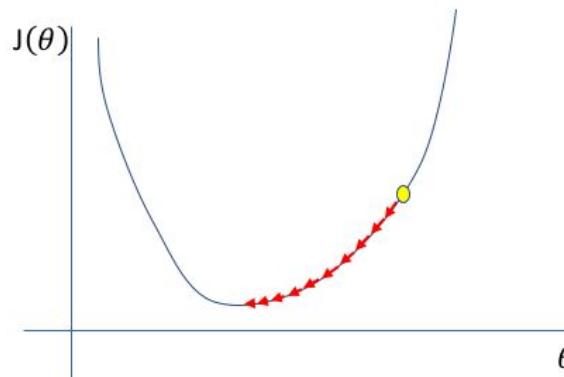
Given a training set with 48,000 samples.

- Batch Gradient Descent. Batch Size = Size of Training Set.
- Stochastic Gradient Descent. Batch Size = 1.
- Mini-Batch Gradient Descent. $1 < \text{Batch Size} < \text{Size of Training Set}$.

We can divide the training set of 48,000 samples into batch size of 100. It will take 480 iterations to complete 1 epoch.

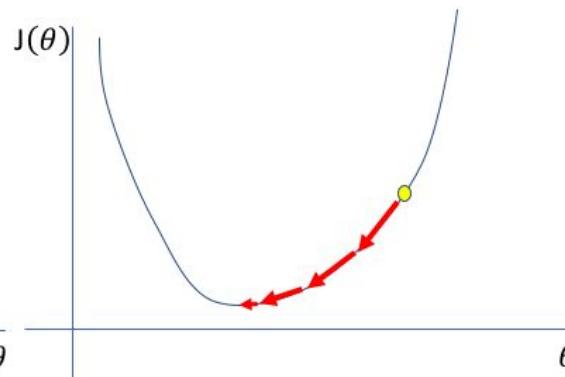
Learning Rate

Too low



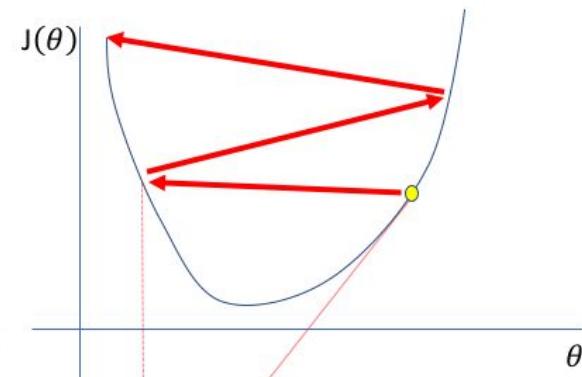
A small learning rate requires many updates before reaching the minimum point

Just right



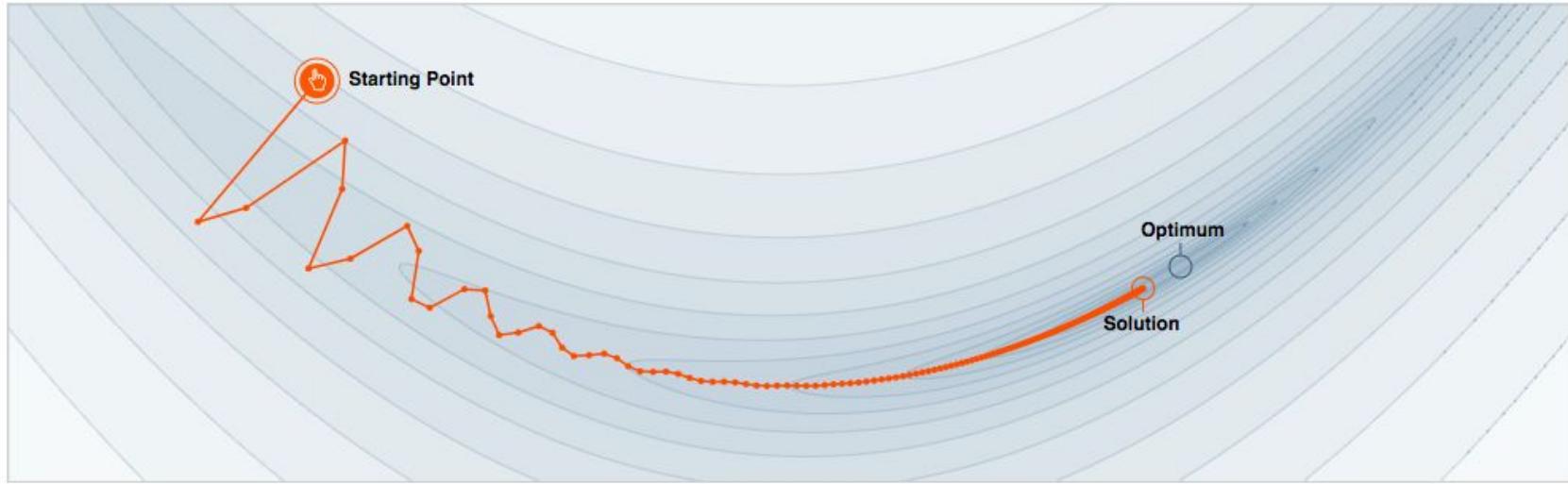
The optimal learning rate swiftly reaches the minimum point

Too high



Too large of a learning rate causes drastic updates which lead to divergent behaviors

Momentum



Step-size $\alpha = 0.02$



Momentum $\beta = 0.99$

We often think of Momentum as a means of dampening oscillations and speeding up the iterations, leading to faster convergence. But it has other interesting behavior. It allows a larger range of step-sizes to be used, and creates its own oscillations. What is going on?

You can try an interactive visualization of learning rate and momentum here: <https://distill.pub/2017/momentum>.

The default learning rate in Keras is 0.01, momentum 0, decay 0.

Weight Decay (Regularization)

$$x = \left(\sum_{j=1}^n \|w^{[j]}\|^2 \right) \frac{\lambda}{2m}$$

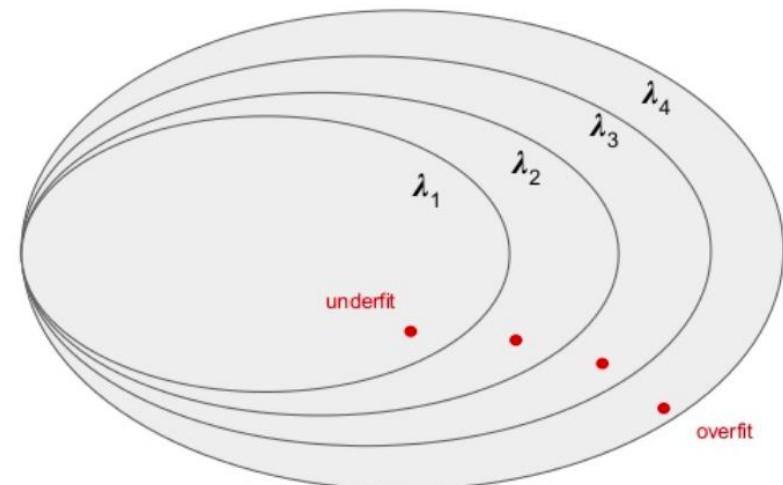
n = the number of layers

$w^{[j]}$ = the weight matrix for the j^{th} layer

m = the number of inputs

λ = the regularization parameter

$$\lambda_1 > \lambda_2 > \lambda_3 > \lambda_4$$



Weight decay penalize the large weights. x will be added to loss function. If some weights are large, the value of loss function will be increased.

Lambda is a weight decay hyperparameter.

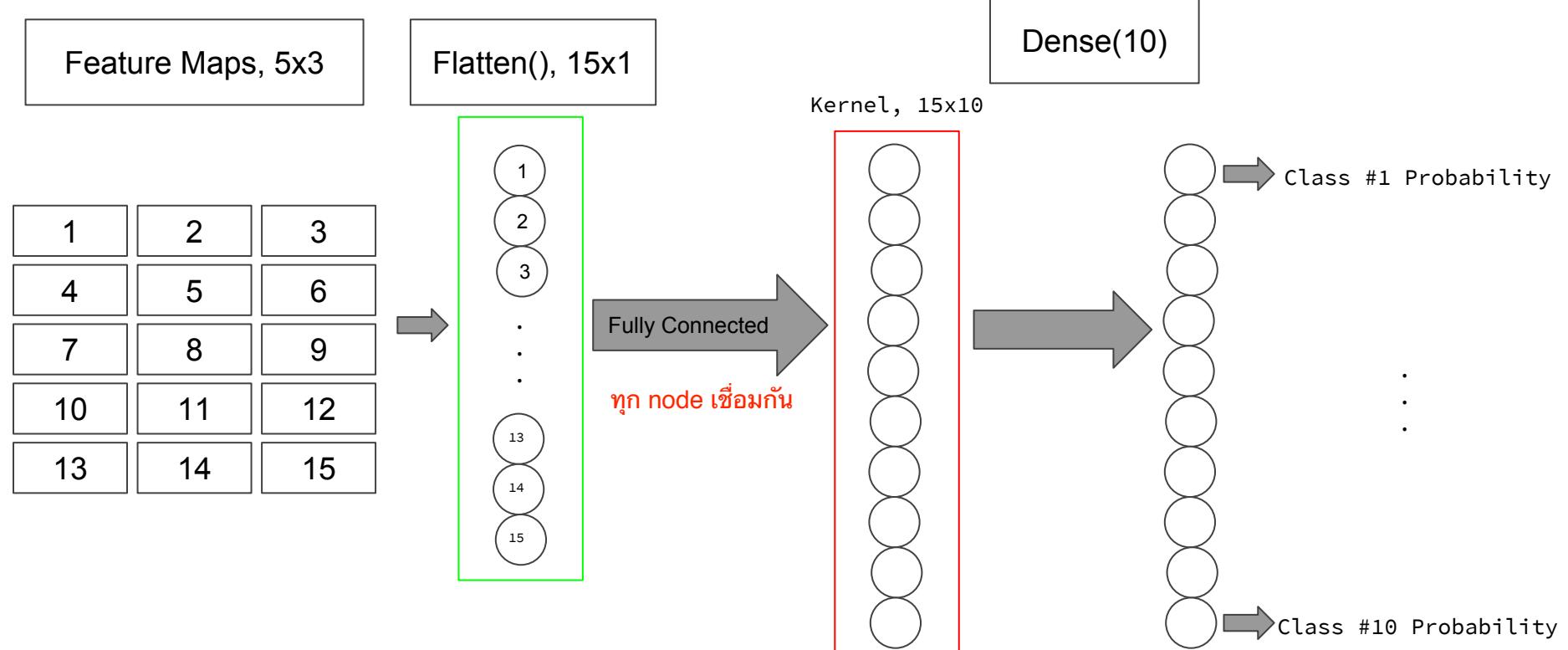
The default value in Keras is 0. AlexNet set it to 0.0005.

Model space

Classification Layers

เมื่อเราทำการ learn เพื่อให้รู้ feature แล้วสุดท้ายต้องทำการ classification

Flatten Layer and Dense Layer



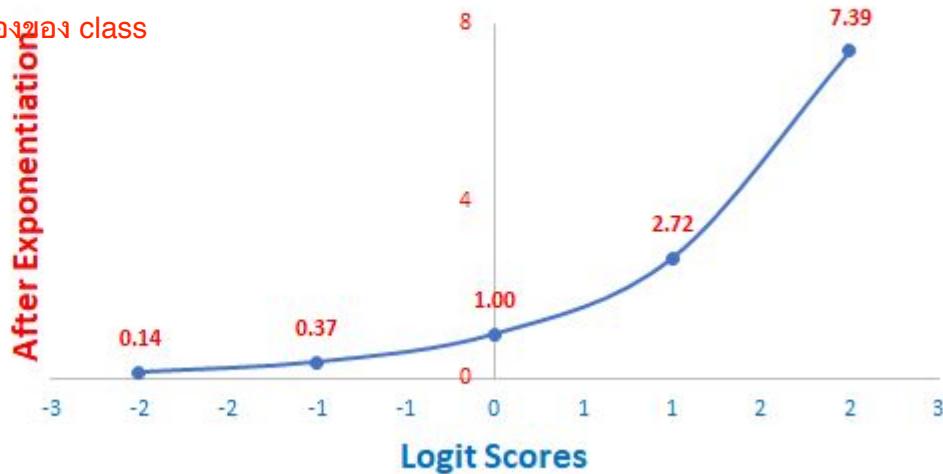
The probability values from Dense layer are raw.
We need to convert them into 0 to 1 range.

Softmax (1)

มี 5 class

Logit Score	After Exponentiation
-2	0.14
-1	0.37
0	1.00
1	2.72
2	7.39

บวกความถูกต้องของ class



Take exponential of raw probability (logit score) with e^{logit} ensures non-negative value.

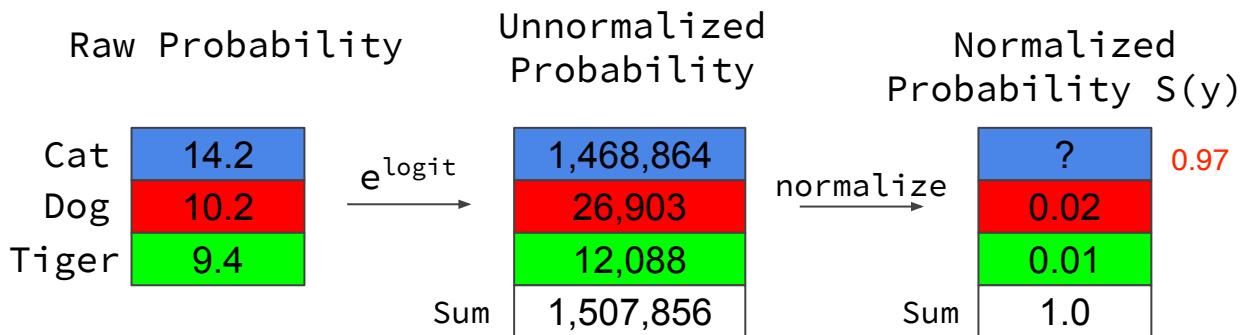
ฐาน e จะให้ค่าที่ไม่ติดลบ

Softmax (2)

Actual Image

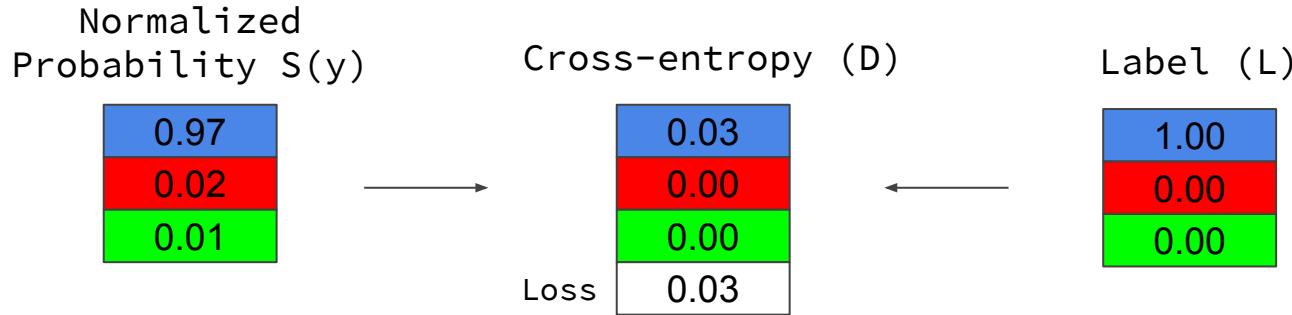


$$s(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$



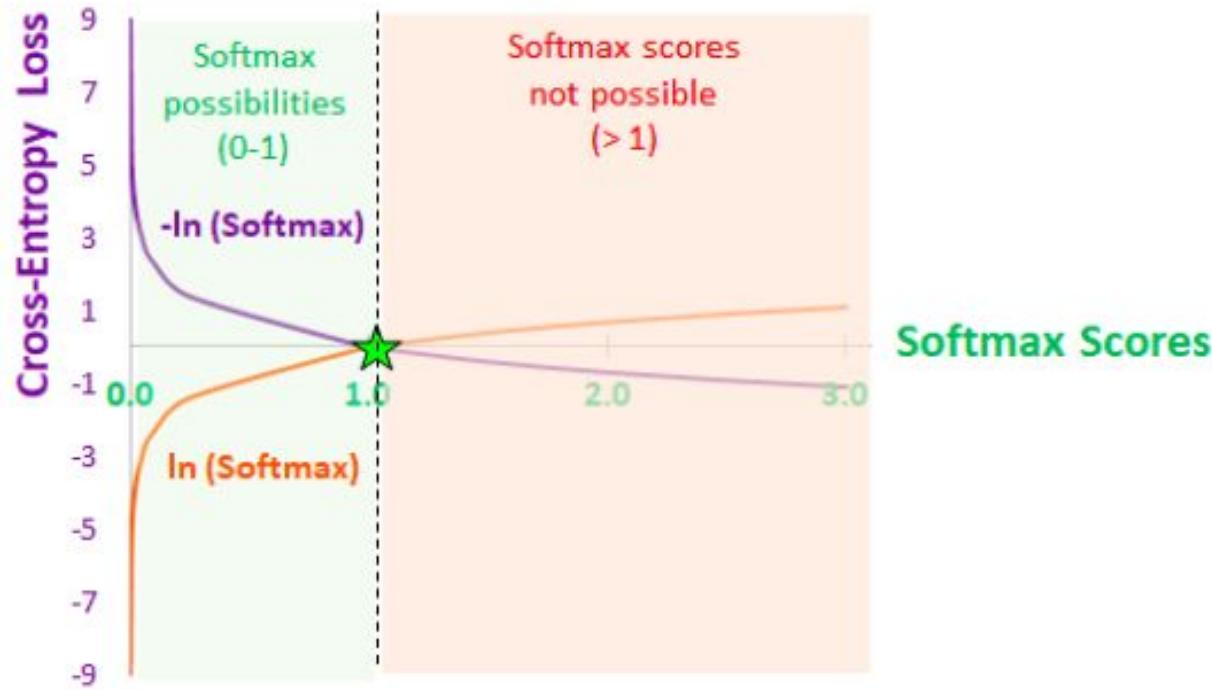
Cross-Entropy Loss

$$D(\mathbf{S}, \mathbf{L}) = - \sum_i L_i \log(S_i)$$



Cross-entropy measures distance (D) between $S(y)$ and label (L) for the correct class.

Softmax Cross-Entropy Loss



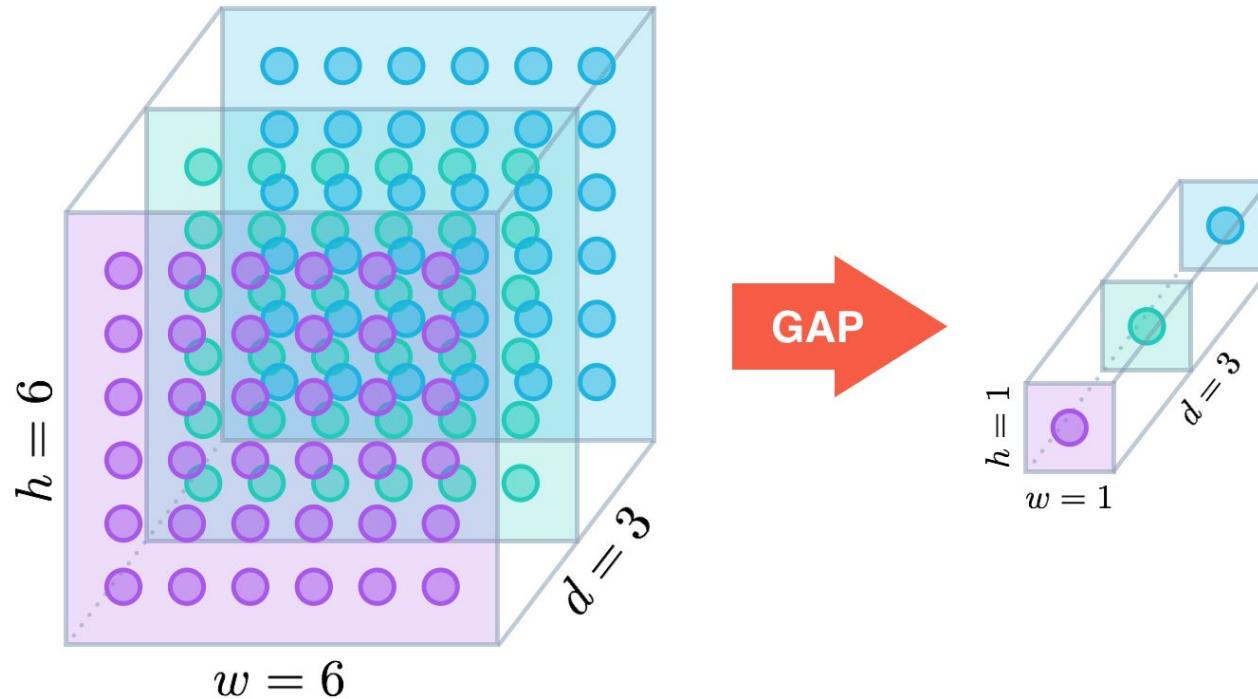
MobileNets

เป็น model สำหรับอุปกรณ์ พกพา javascript

MobileNets Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5 \times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

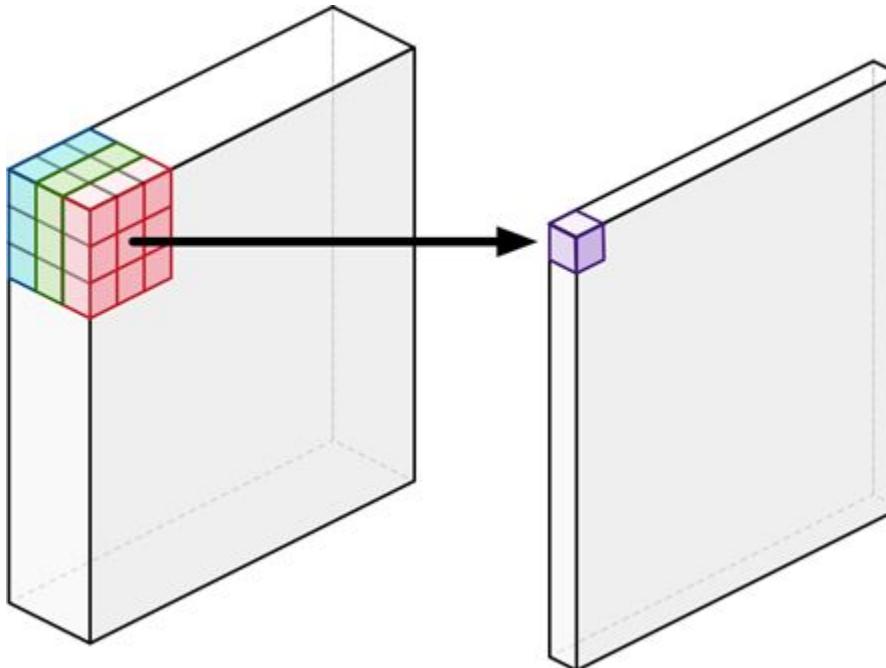
Global Average Pooling (GAP)



Global average pooling layer is similar to max pooling layer but it reduced dimensions more than max pooling layer.

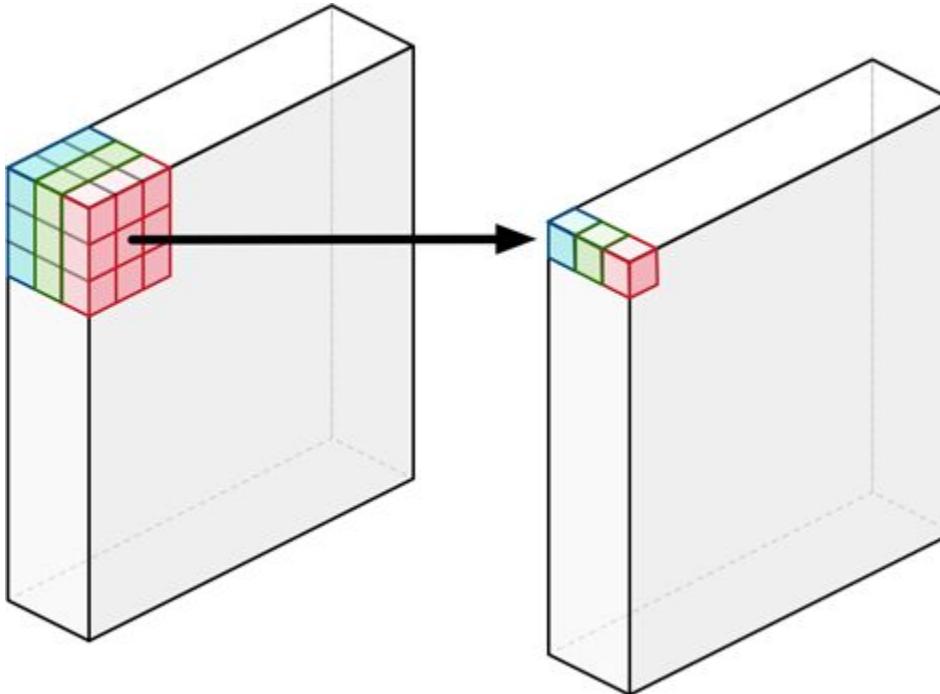
Depthwise Separable Convolution

Regular Convolutional Layer



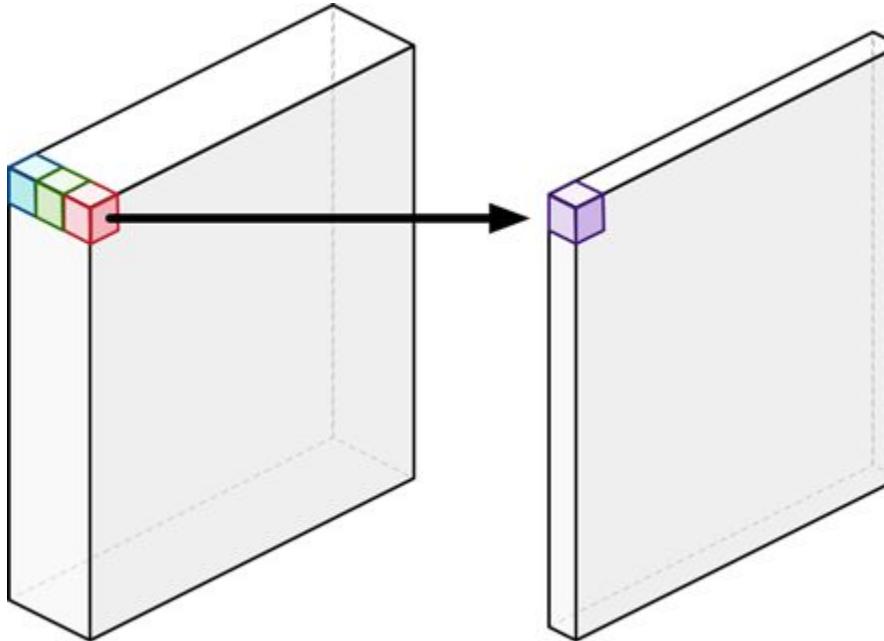
For example, each channel of $3 \times 3 \times 3$ kernel slided through each channel of an input image, and sum all values into the feature maps.

Depthwise Convolutional Layer



Depthwise Convolution performs convolution on each channel separately. For an image with 3 channels, a depthwise convolution creates an output feature maps that also has 3 channels. Each channel gets its own set of weights.

Pointwise Convolutional Layer



The depthwise convolution is followed by a pointwise convolution. Pointwise convolution is actually a regular convolution with 1×1 kernel size. So, pointwise convolution convert 3 channels input into one channel.

Depthwise Separable Convolution is Fast!

Input Image: 224x224x3

Regular Convolution (64 filters)

$$(224-3+1)^2 \times (3 \times 3 \times 3) \times 64 = 85,162,752 \text{ multiplication operations}$$

output size: filter size number of filters
(input size - filter size + 1)²

Depthwise Separable Convolution (64 filters)

Depthwise Convolution

$$(224-3+1)^2 \times (3 \times 3 \times 1) \times 3 = 1,330,668$$

+ 3 kernels is number of channels from input image

Pointwise Convolution

$$(224-3+1)^2 \times (1 \times 1 \times 3) \times 64 = 9,462,528$$

Total: ? multiplication operations

filter 64

Depthwise Separable Convolution is Fast!

Input Image: 224x224x3

Regular Convolution (64 filters)

$(224-3+1)^2 \times (3 \times 3 \times 3) \times 64 = 85,162,752$ multiplication operations
output size: filter size number of filters
input size - filter size

Depthwise Separable Convolution (64 filters)

Depthwise Convolution

$(224-3+1)^2 \times (3 \times 3 \times 1) \times 3 = 1,330,668$

+ 3 kernels is number of channels from input image

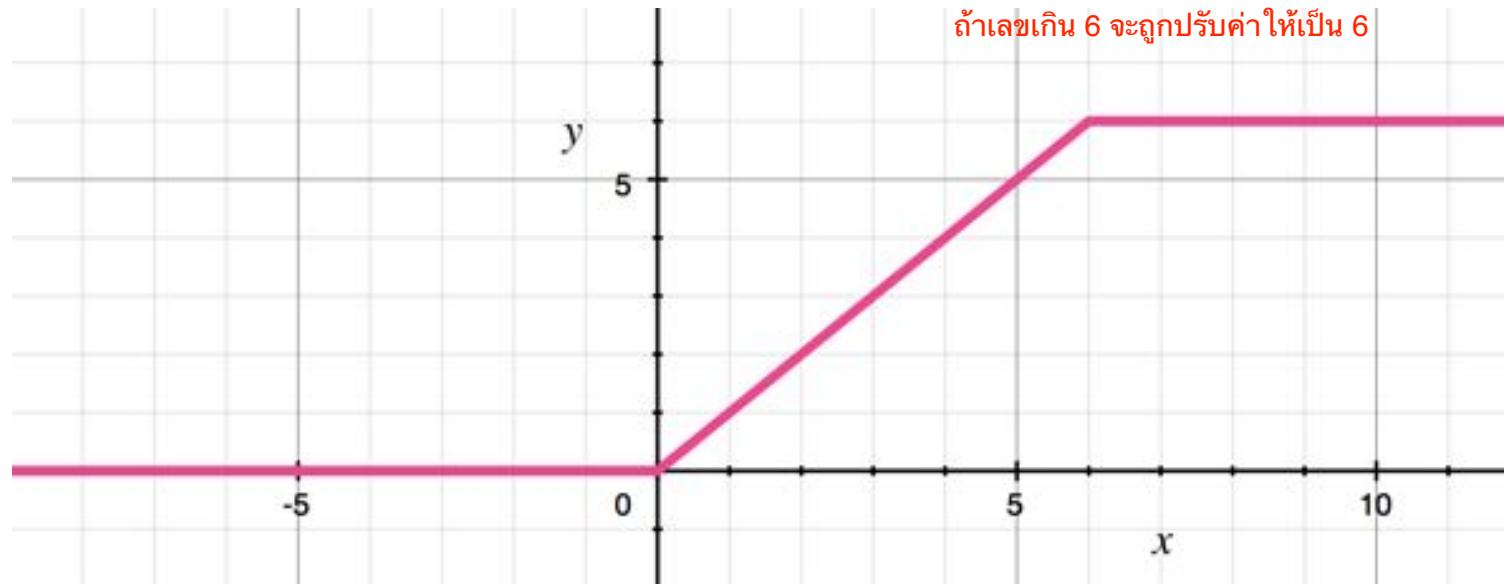
Pointwise Convolution

$(224-3+1)^2 \times (1 \times 1 \times 3) \times 64 = 9,462,528$

Total: 10,793,196 multiplication operations

7.9 Times Faster!

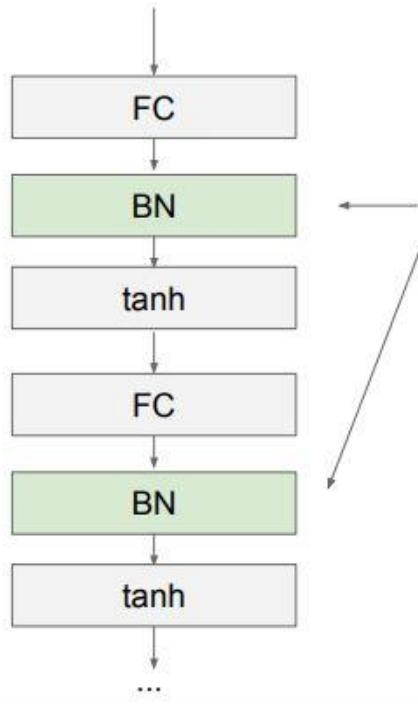
ReLU6



In MobileNet paper, they found that ReLU6 is more robust than regular ReLU when using low-precision computation such as mobile devices.

Batch Normalization

[Ioffe and Szegedy, 2015]



Usually inserted after Fully Connected or Convolutional layers, and before nonlinearity.

$$\mu = \frac{1}{m} \sum_{i=1}^m z^{(i)}$$
$$\sigma^2 = \frac{1}{m} \sum_{i=1}^m (z^{(i)} - \mu)^2$$
$$z_{norm}^{(i)} = \frac{z^{(i)} - \mu}{\sqrt{\sigma^2 + \epsilon}}$$
$$z_{bn}^{(i)} = \gamma z_{norm}^{(i)} + \beta$$

feature scaling គឺ ការធានាដែលមូលដ្ឋានរវាង 0-1 ដើម្បី ធានាដែលមូលដ្ឋាន ក្នុងតាមគ្រប់គ្រង និង train ថាវា error ខ្សោយកាល 0 ។

Transfer Learning



ImageNet dataset contains the 1000 categories and 1.2 million images.

Case: Tiny Dataset

For example, less than 500 labels for each class.

Transferred weights from ImageNet. All these weights are frozen.

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Train these classification layers on your dataset.



Case: Small Dataset

For example, 500-1,000 labels for each class.

Transferred weights from ImageNet. All these weights are frozen.

Train your dataset on the last few convolutional layers and classification layers.

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Case: Large Dataset

For example, more than 1,000 labels for each class.

Train the entire networks on your dataset. But it is still good to initialize with ImageNet.

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$