Convolutional Neural Network

Sofa?

Teera Laiteerapong

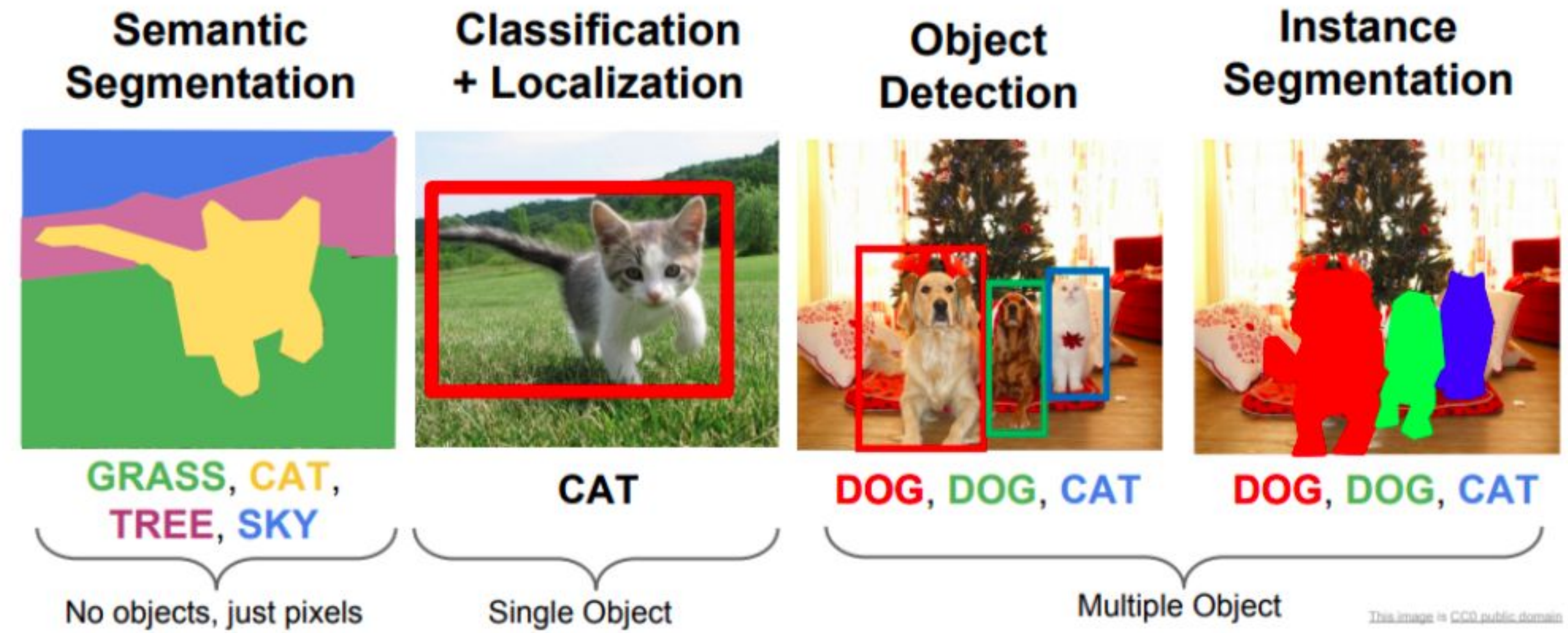
30 March 2019

Image source: http://www.keet.us/portfolio_page/mini-sofa.
More information: http://rocknrollnerd.github.io/ml/2015/05/27/leopard-sofa.html

- **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



Source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf

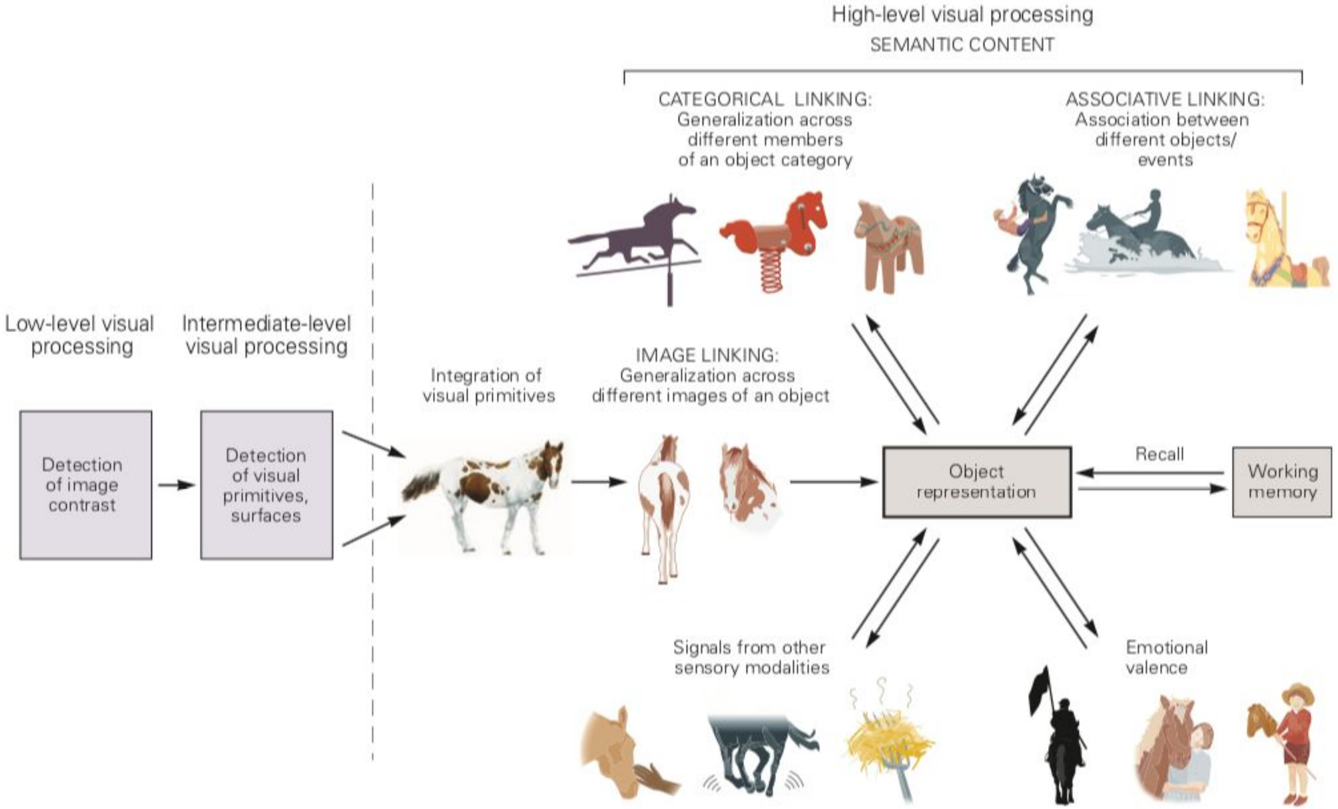
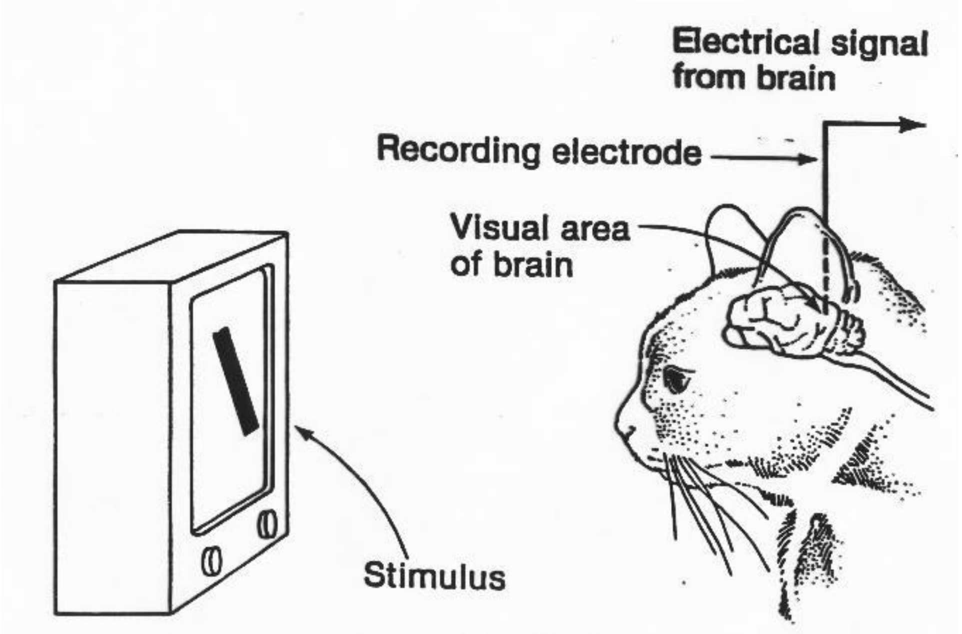


Figure 28–1 The neuronal representation of entire objects is central to high-level visual processing. Object representation involves integration of visual features extracted at earlier stages in the visual pathways. Ideally the resulting representation is a generalization of the numerous retinal images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

Source: https://neurology.mhmedical.com/book.aspx?bookID=1049

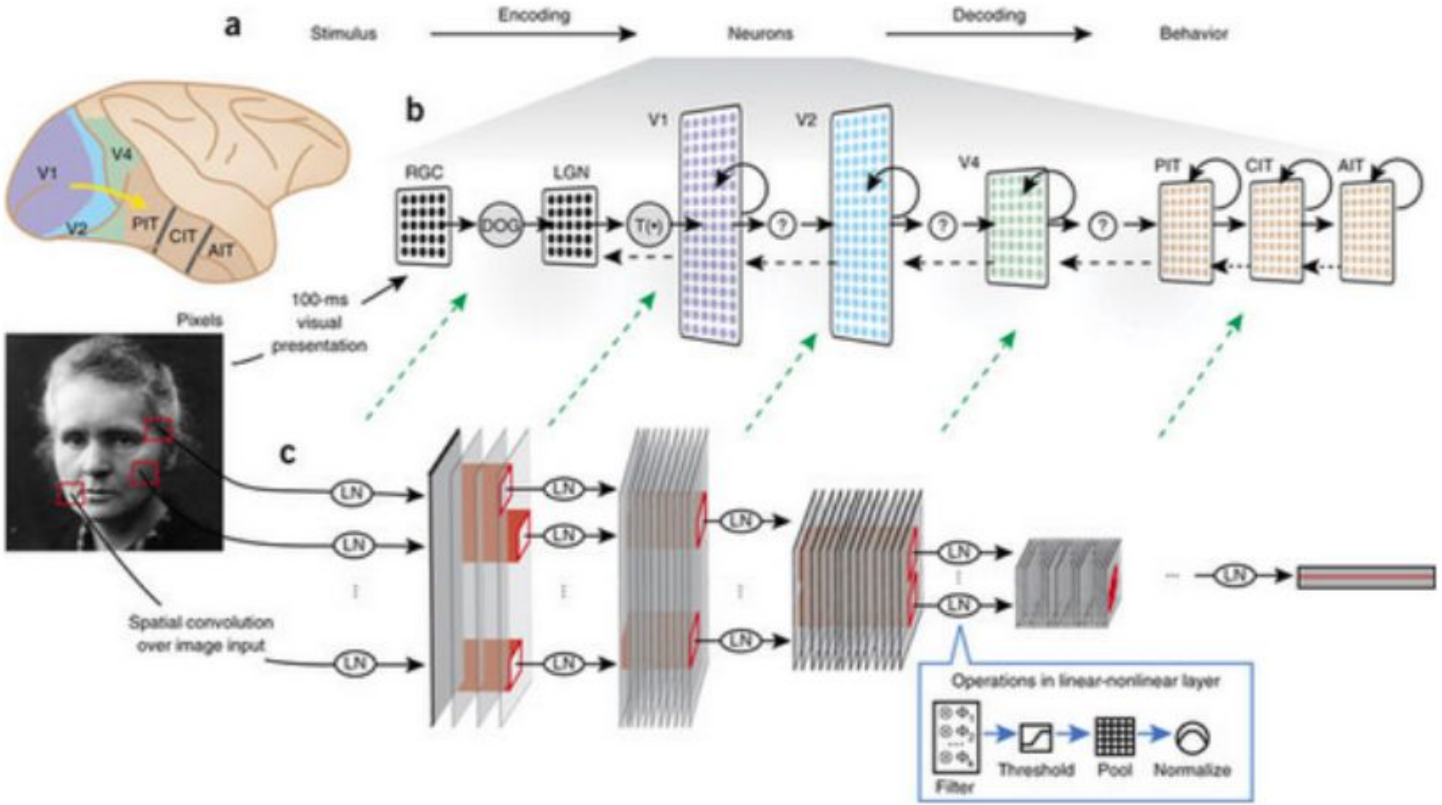
David Hubel and Torsten Wiesel (1950s)



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.

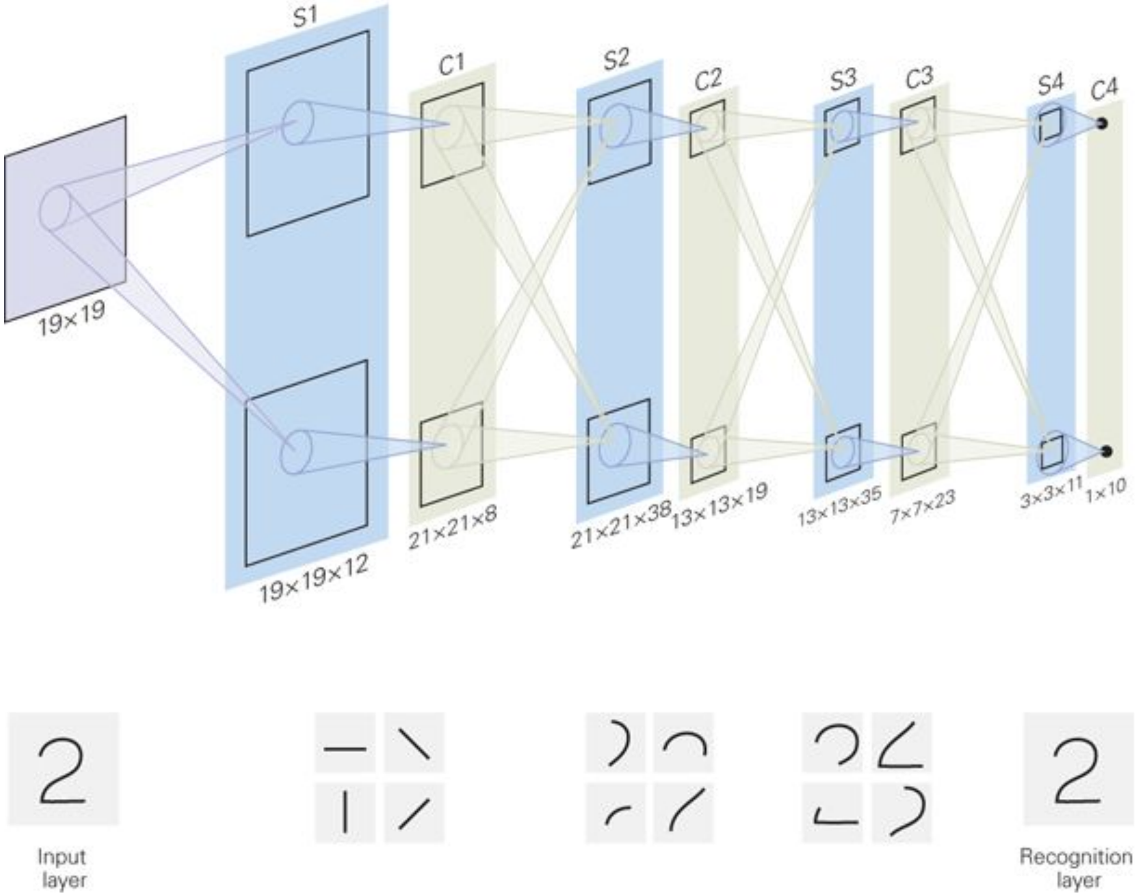
Source: https://goodpsychology.wordpress.com/2013/03/13/235/
http://braintour.harvard.edu/archives/portfolio-items/hubel-and-wiesel

CNN Inspiration from Human Brain



Source: https://neurdiness.wordpress.com/2018/05/17/deep-convolutional-neural-networks-as-models-of-the-visual-system-qa/

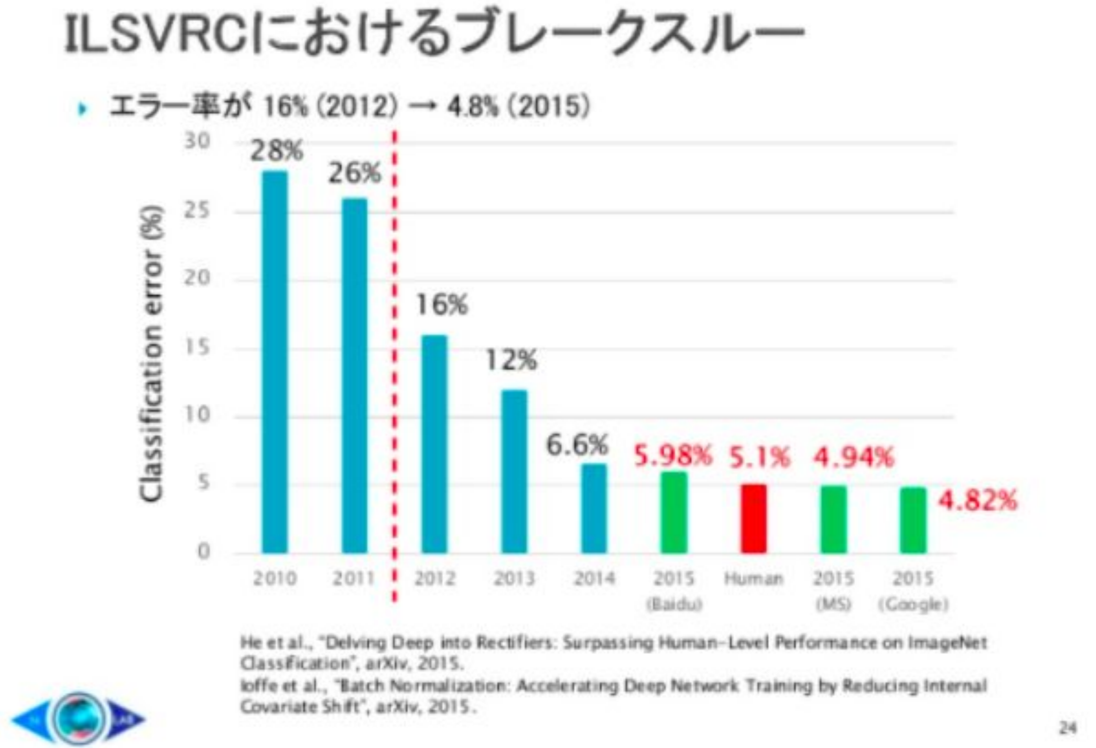
Neocognitron



Kunihiko Fukushima, 1980

Source: https://neurology.mhmedical.com/book.aspx?bookID=1049, https://github.com/vtkuzmin/Neocognitron/blob/master/ClassNeocognitron.py

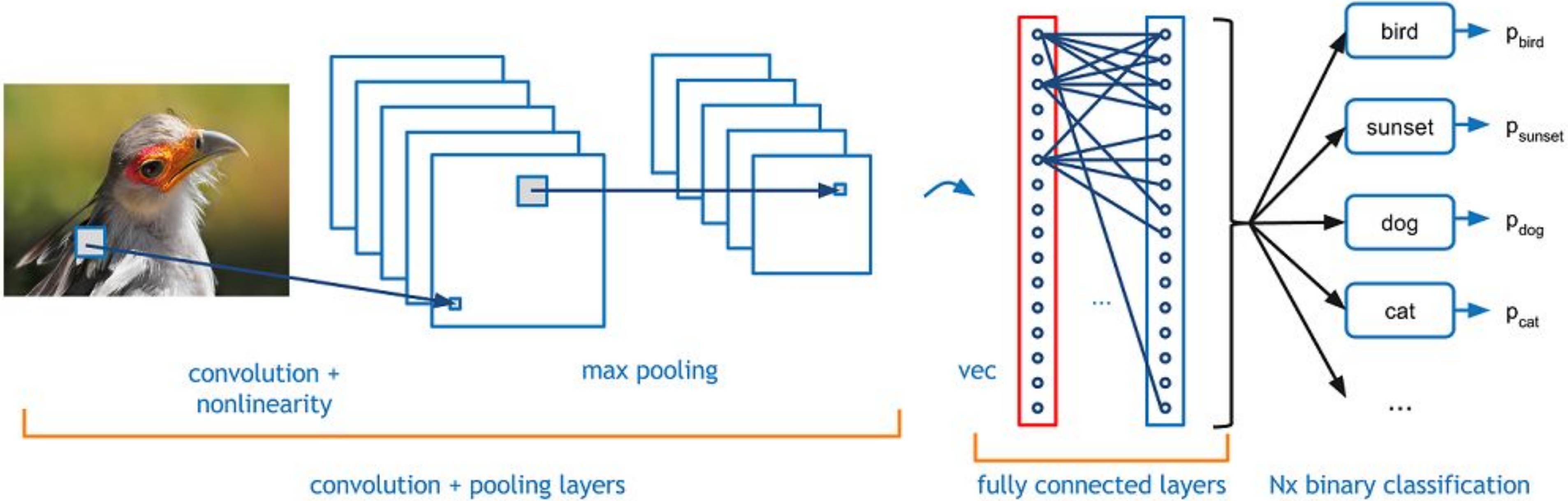
Image Classification Development



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

Source: https://shift.newco.co/there-is-one-thing-that-computers-will-never-beat-us-at-f66af30565f0
https://qz.com/1307091/the-inside-story-of-how-ai-got-good-enough-to-dominate-silicon-valley/

Top-1 Accuracy vs Top-5 Accuracy



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

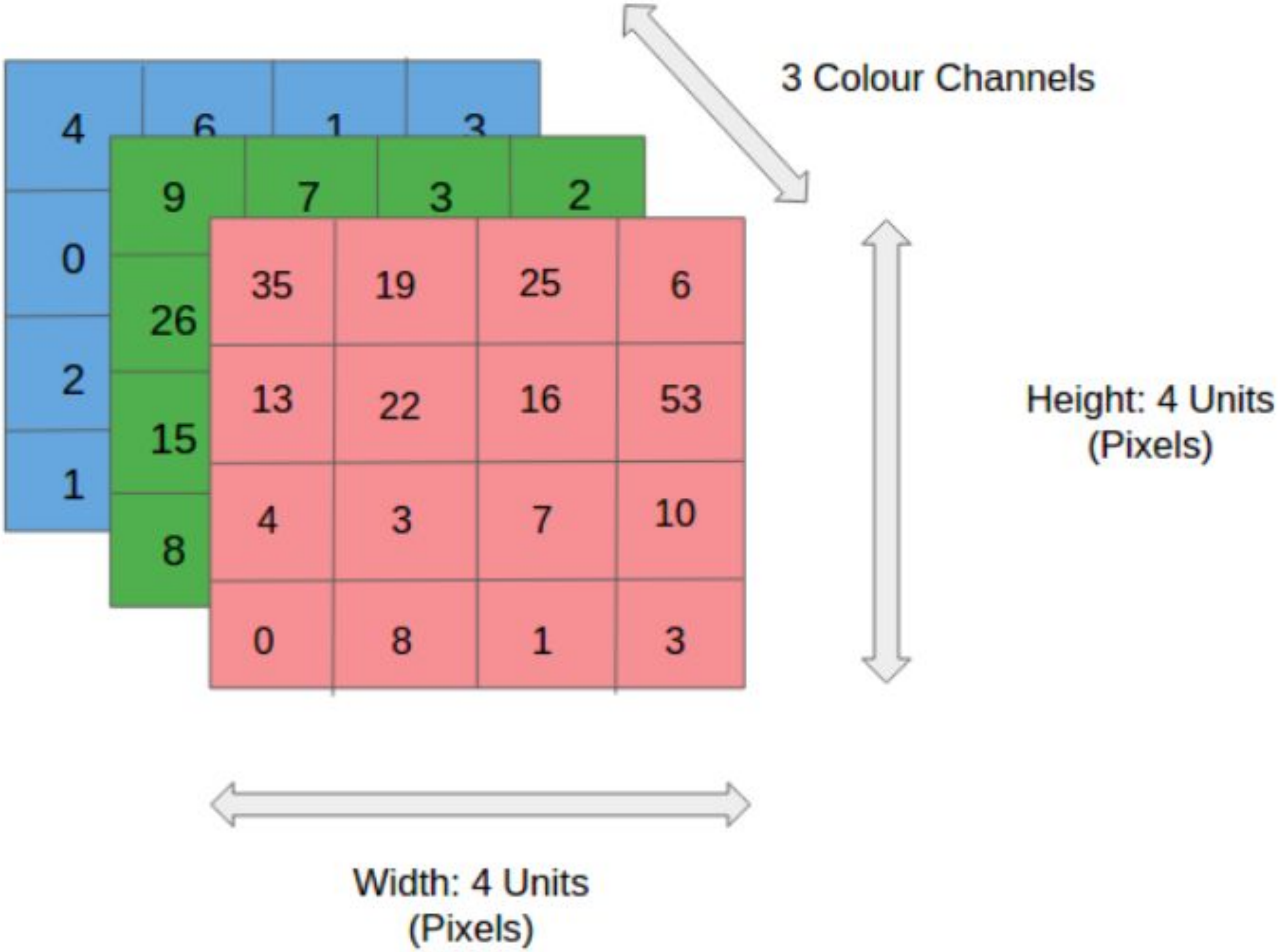
Source: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

1 - 189 / 189

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	α=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	α=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


3 Colour Channels
Height: 4 Units
(Pixels)
Width: 4 Units
(Pixels)
4 6 1 3
9 7 3 2
0
26
2
15
1
8
35 19 25 6
13 22 16 53
4 3 7 10
0 8 1 3

What is convolution?

Photo editor apps

Source: https://www.macthai.com/2016/01/08/15-necessary-apps-editor-photo-for-ios/

Input Image

Kernel/Filter

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

Convolution

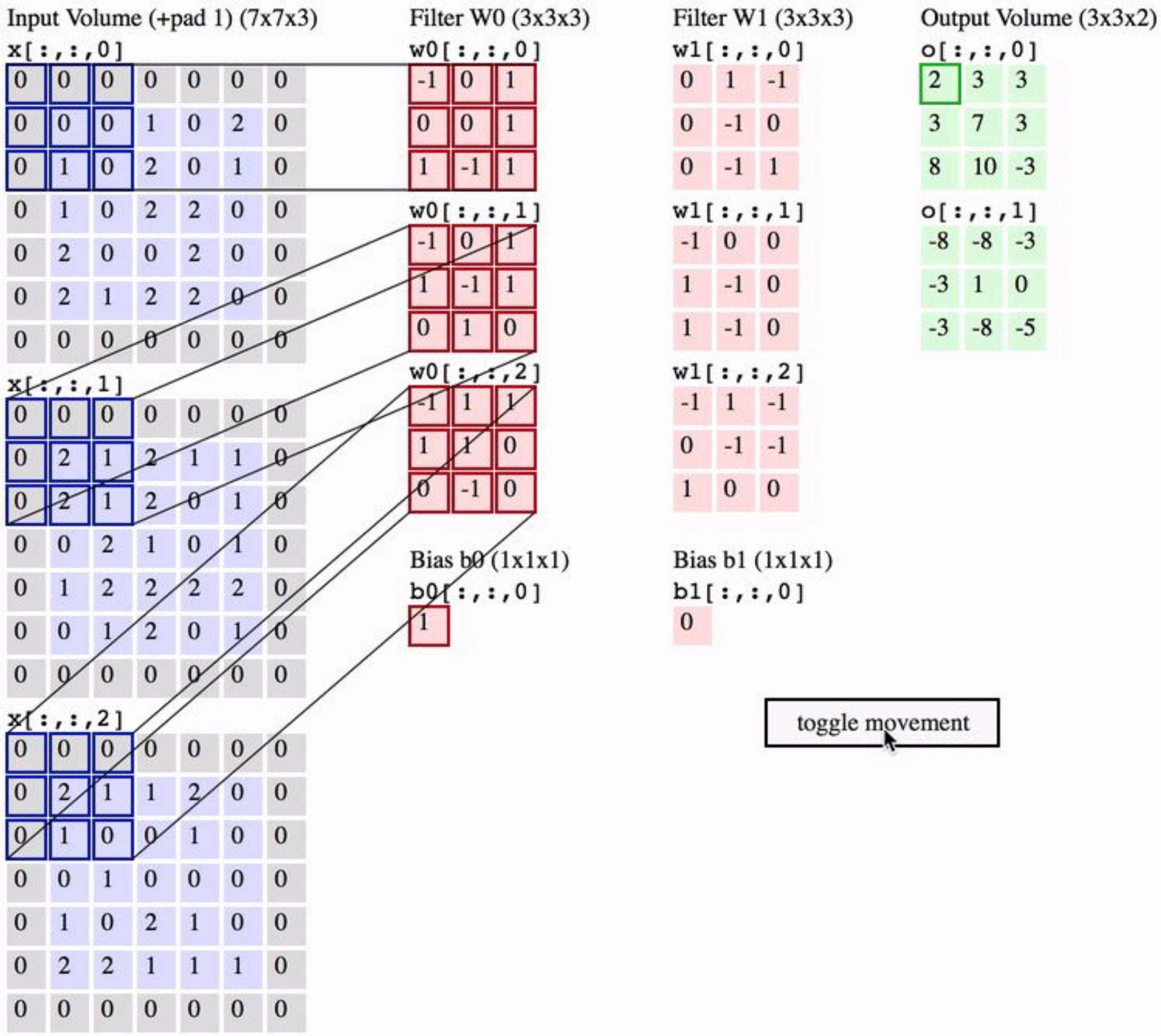
Blurred Image

Convolution operation is actually an inner dot product.

See more filters in Gimp:
https://docs.gimp.org/2.6/en/plug-in-convmatrix.html

Source: https://en.wikipedia.org/wiki/Kernel_(image_processing)

Convolution Operation



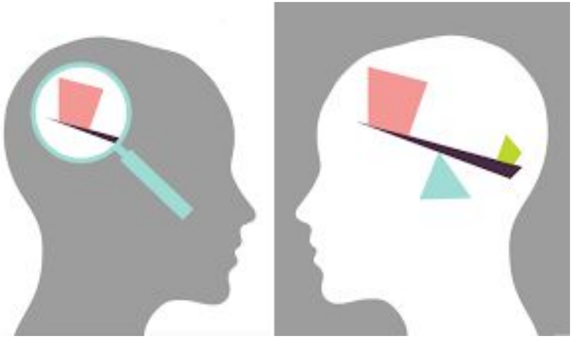
Source: http://cs231n.github.io/convolutional-networks/

Bias

$Y=\theta x+Thai$

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: (input_size+2*padding_size-filter_size)/stride + 1
y: (input_size+2*padding_size-filter_size)/stride + 1
```

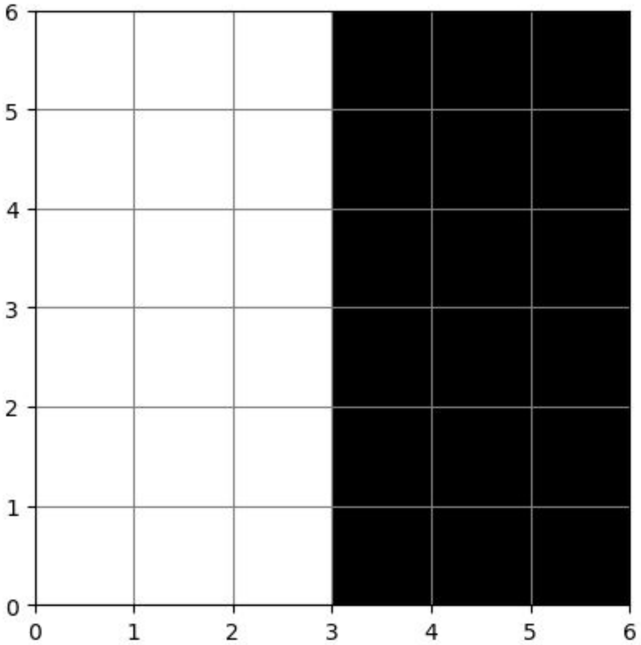
Calculate Feature Maps Dimension (2)

Given an input image size 6x6, kernel size 3x3, stride 2, padding 1.

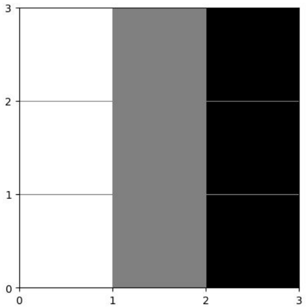
What is the dimension of an output?

Convolution as Edge Detection

Vertical Edge Detection



$\circledast$



Kernel

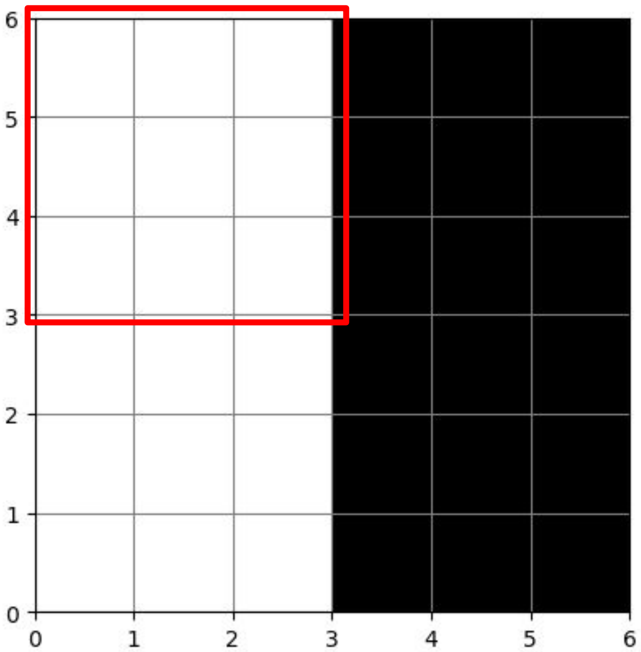
```
[[ 1  0 -1]
 [ 1  0 -1]
 [ 1  0 -1]]
```

= ?

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]]
```

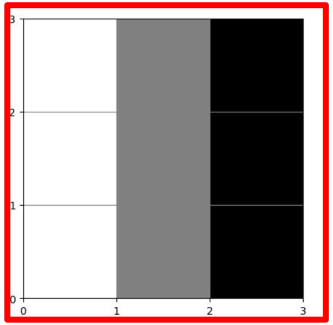
Vertical Edge Detection



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]]
```

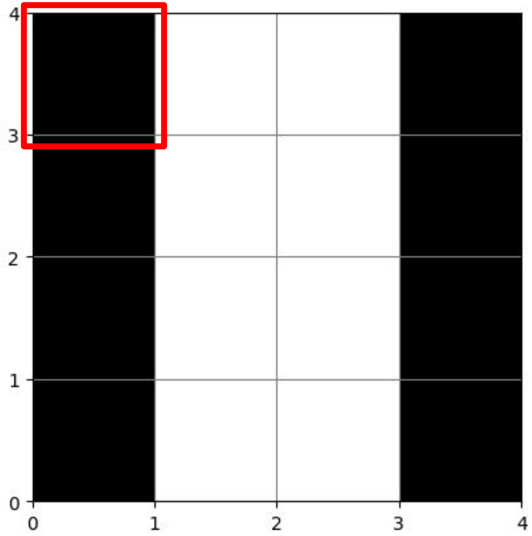
$\circledast$



Kernel

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

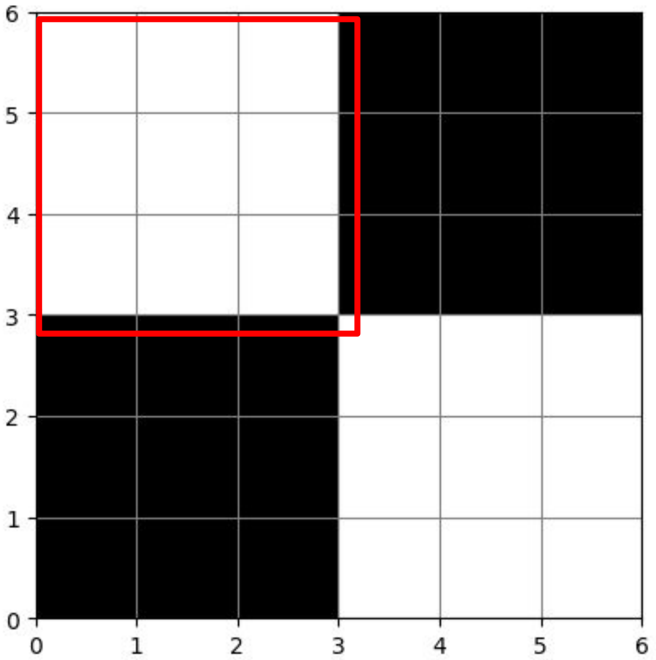
=



Output/Feature maps

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

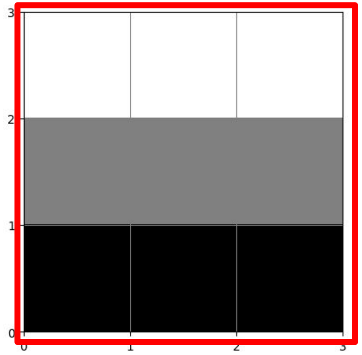
Horizontal Edge Detection



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]]
```

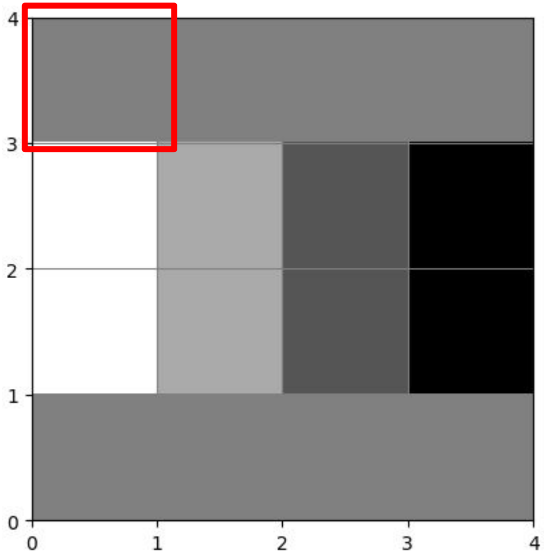
$\circledast$



Kernel

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

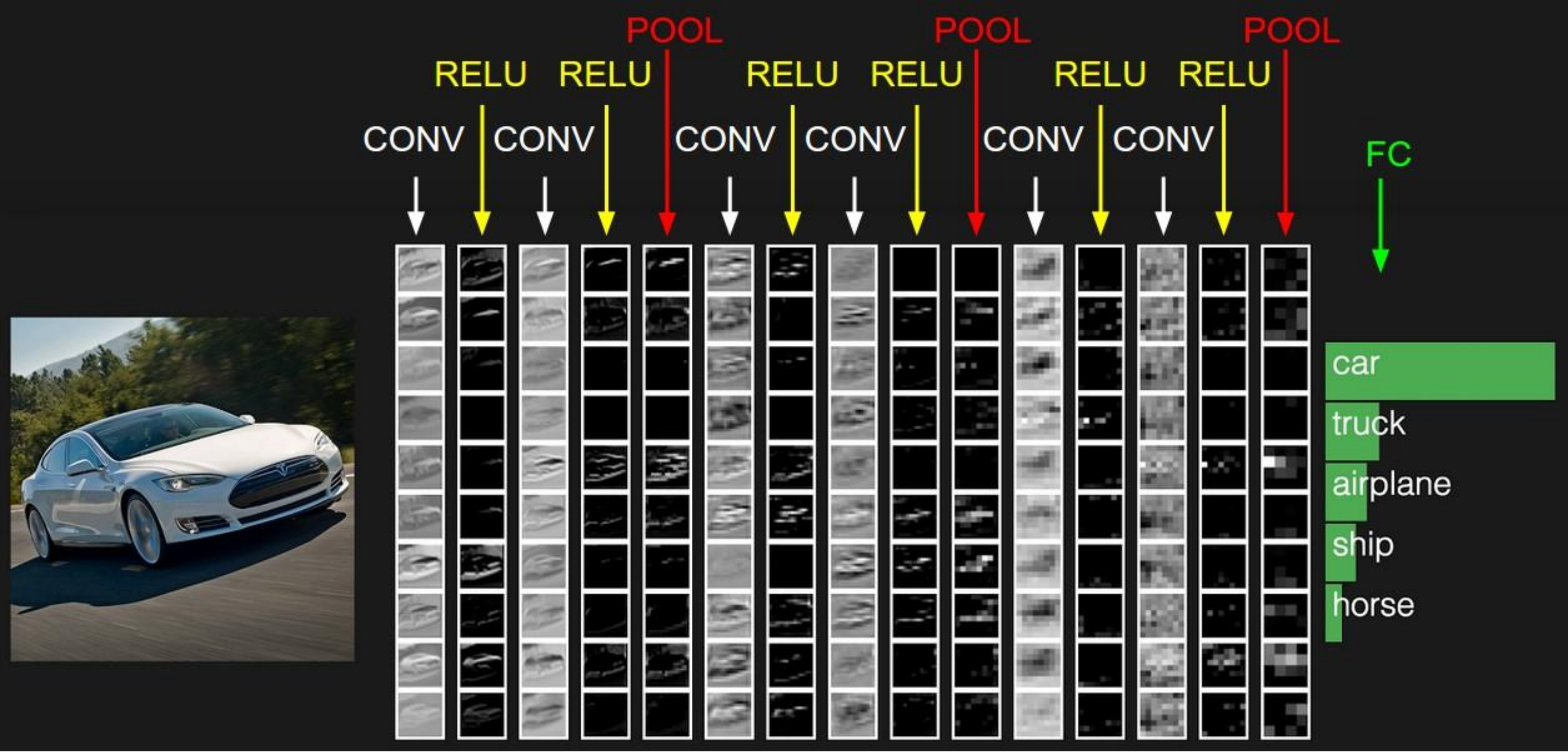
=



Output/Feature maps

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

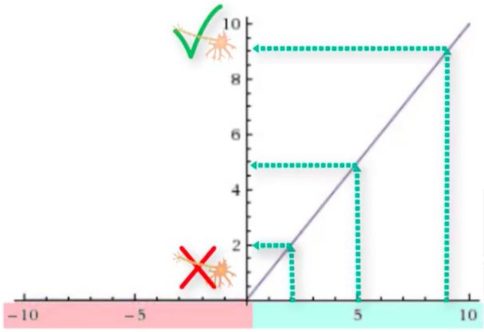
Convolutional Neural Network

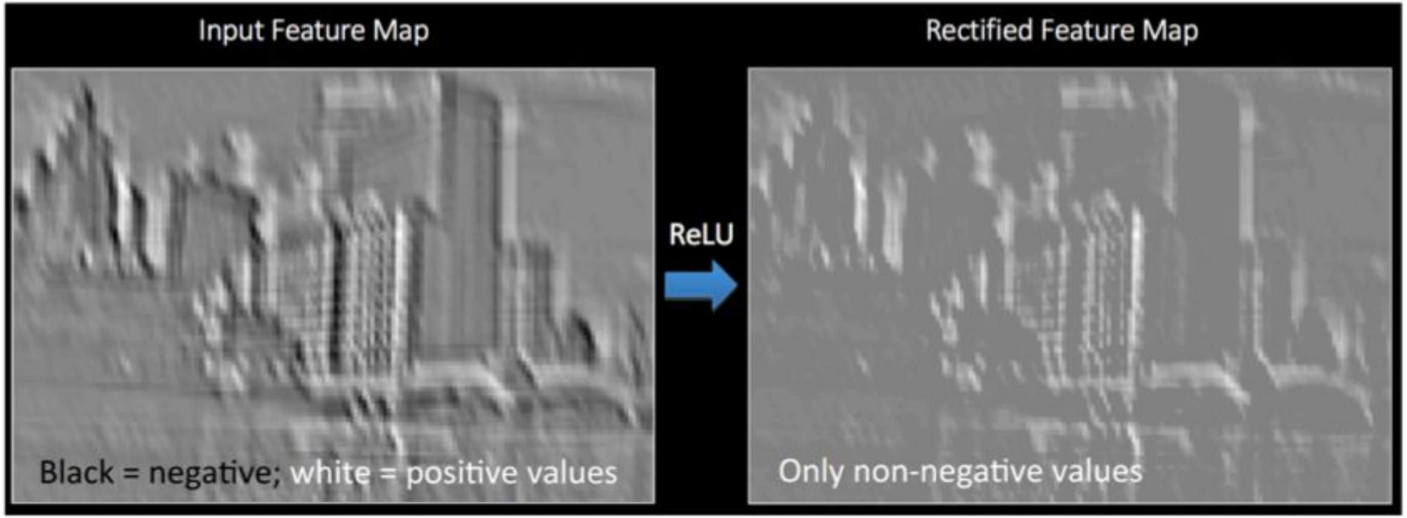


Source: http://cs231n.github.io/convolutional-networks/

ReLU (Rectified Linear Unit)

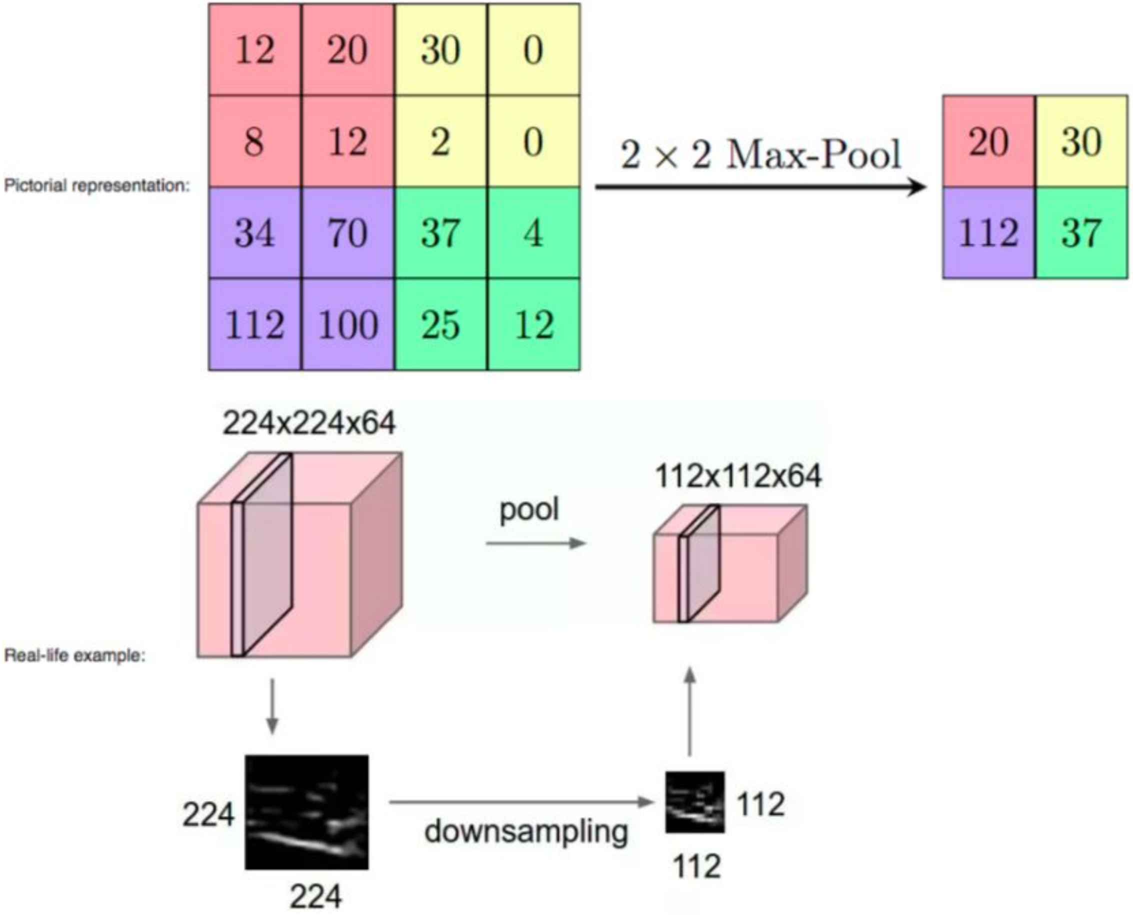
$f(x) = \max(0, x)$





Source: https://www.youtube.com/watch?v=m0pIlLfpXWE
https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

Pooling Layer



Source: https://computersciencewiki.org/index.php/Max-pooling_/_Pooling

A Simple Convolutional Neural Network

```python
simple_model = 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'))

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

Batch Size, Iteration, and Epoch

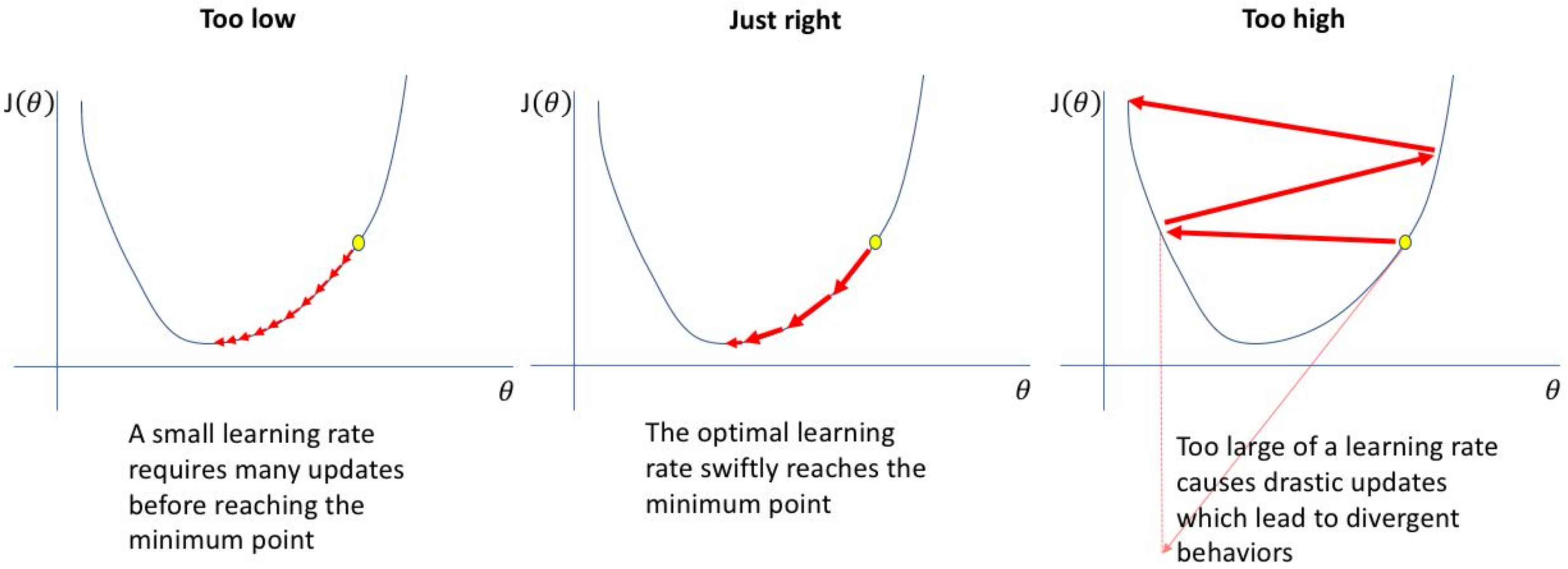
Given a training set with 48,000 samples.

- Batch Gradient Descent. Batch Size = Size of Training Set.
- Stochastic Gradient Descent. Batch Size = 1.
- Mini-Batch Gradient Descent. 1 < Batch Size < 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.

Source: https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/

Learning Rate

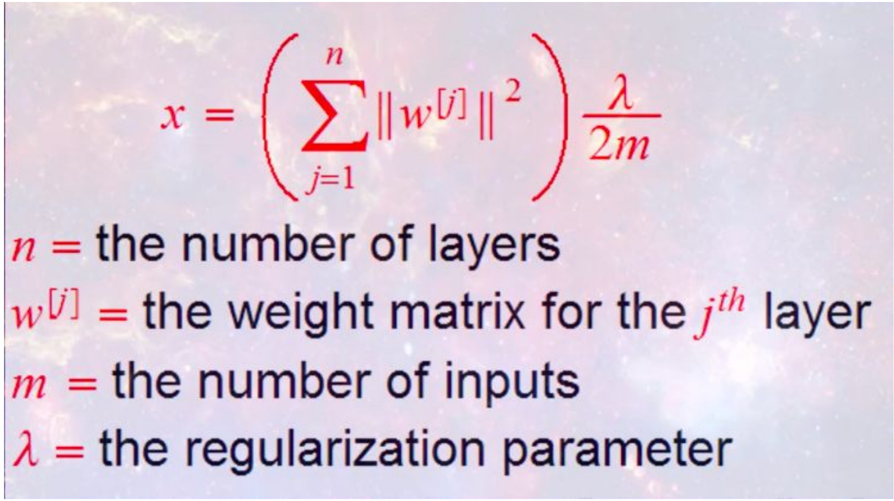


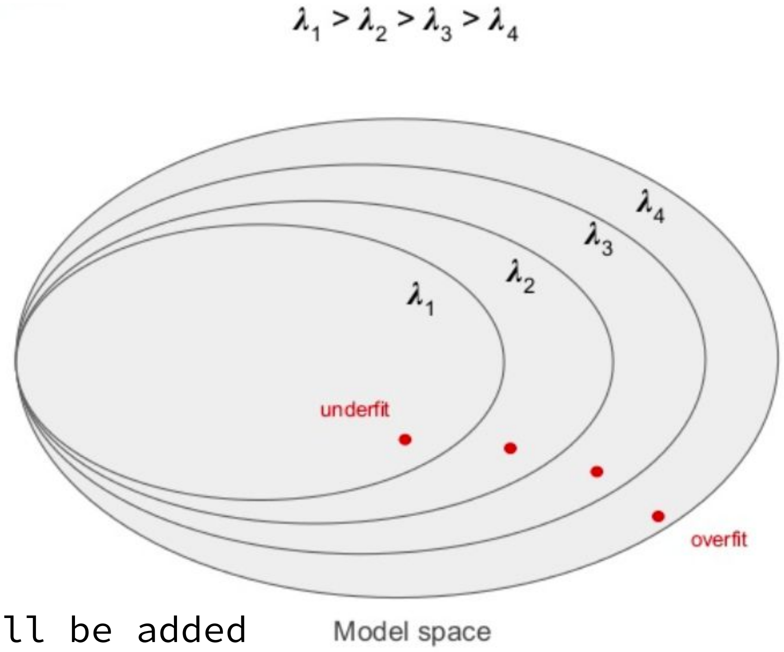
Source: https://www.jeremyjordan.me/nn-learning-rate/

Momentum

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)



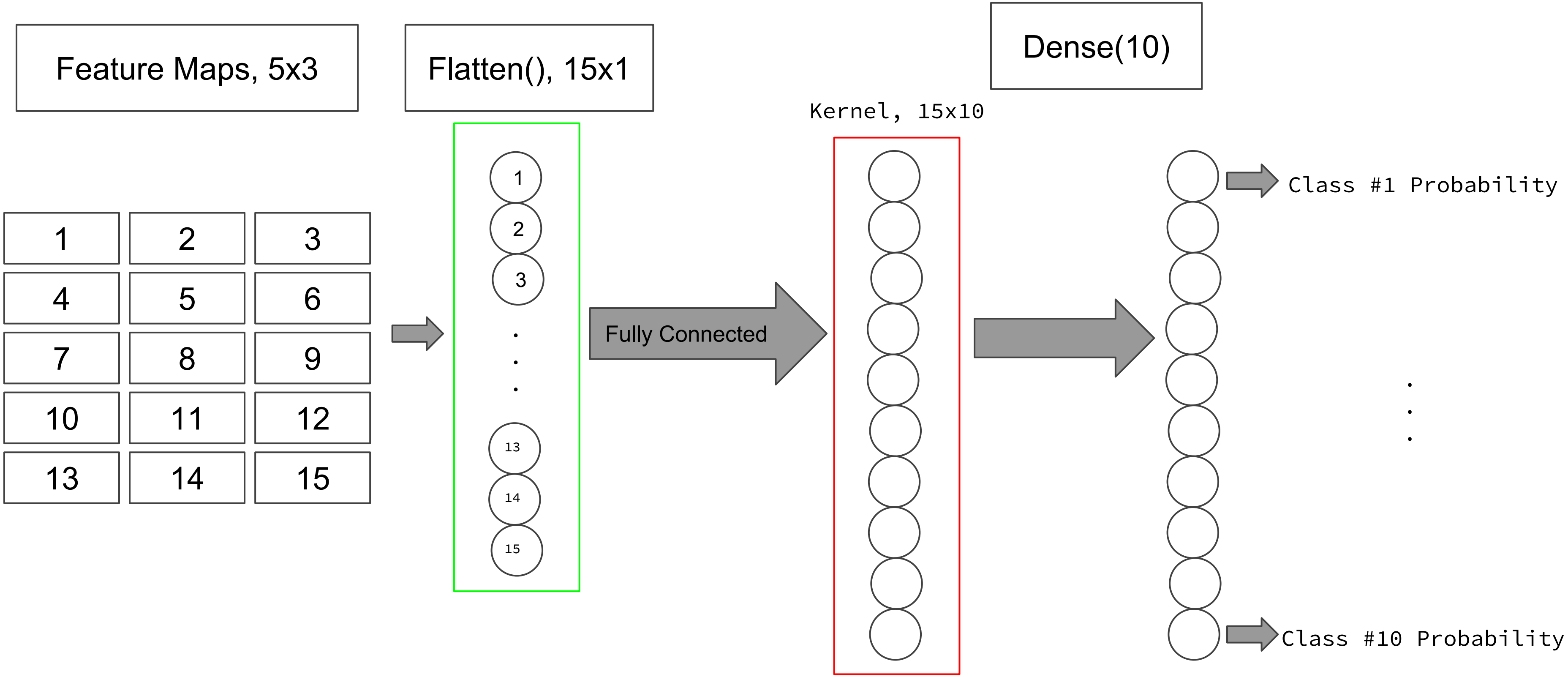


Weight decay penalize the large weights. x will be added to loss function. If some weights are large, the value of loss function will be increased.
Lambda is a weight decay hyperparameter.
The default value in Keras is 0. AlexNet set it to 0.0005.

Source: https://www.slideshare.net/xavigiro/training-deep-networks-d1l5-2017-upc-deep-learning-for-computer-vision
https://www.youtube.com/watch?v=iuJgyiS7BKM

Classification Layers

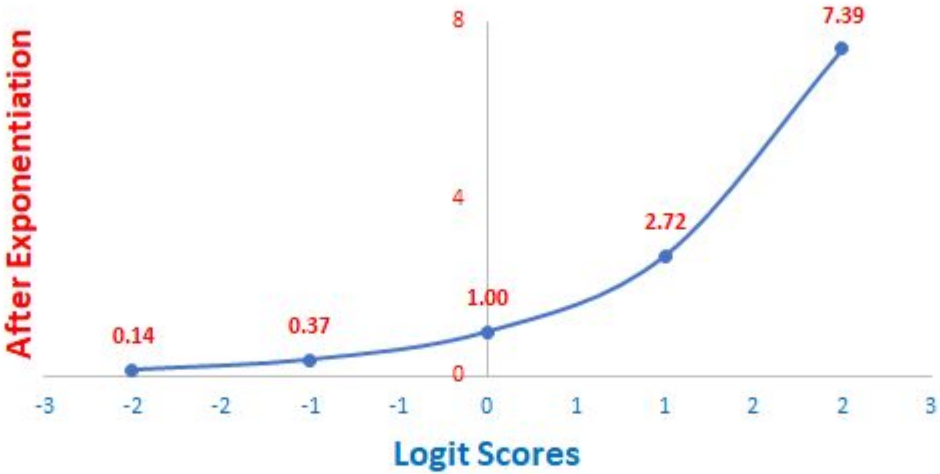
Flatten Layer and Dense Layer



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

Softmax (1)

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



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

Source: https://towardsdatascience.com/cutting-edge-face-recognition-is-complicated-these-spreadsheets-make-it-easier-e7864dbf0e1a

Softmax (2)

Actual Image 

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

	Raw Probability	$\xrightarrow{e^{\text{logit}}}$	Unnormalized Probability	$\xrightarrow{\text{normalize}}$	Normalized Probability S(y)
Cat	14.2		1,468,864		?
Dog	10.2		26,903		0.02
Tiger	9.4		12,088		0.01
		Sum	1,507,856	Sum	1.0

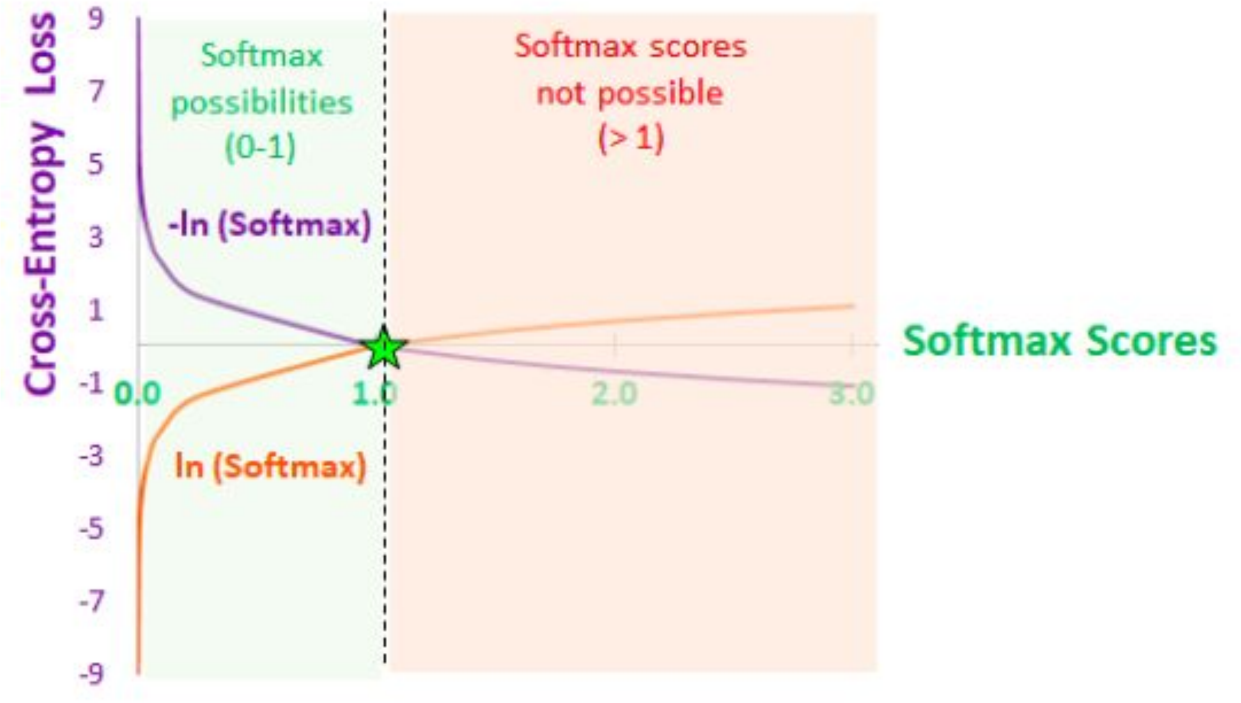
Cross-Entropy Loss

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

Normalized Probability S(y)		Cross-entropy (D)		Label (L)
0.97		0.03		1.00
0.02	→	0.00	←	0.00
0.01		0.00		0.00
	Loss	0.03		

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

Softmax Cross-Entropy Loss



Source: https://towardsdatascience.com/cutting-edge-face-recognition-is-complicated-these-spreadsheets-make-it-easier-e7864dbf0e1a

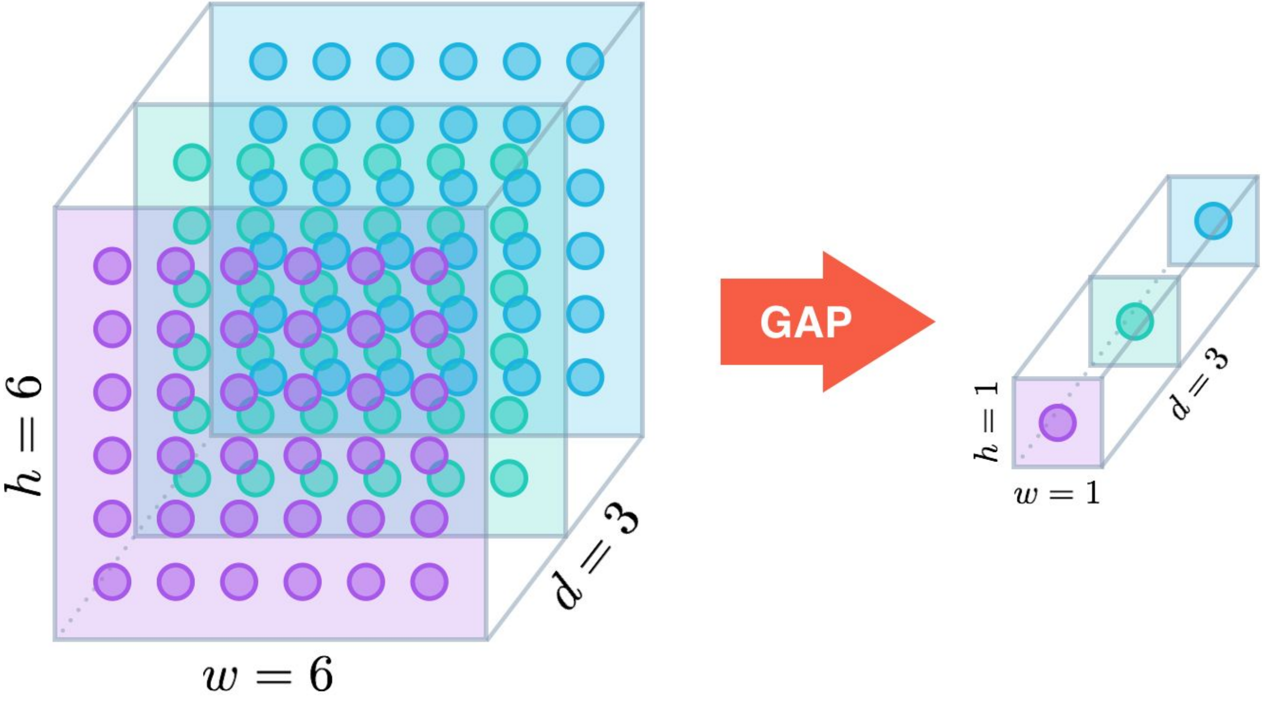
MobileNets

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	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
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	$1 \times 1 \times 1000$

Source: https://arxiv.org/abs/1704.04861

Global Average Pooling (GAP)

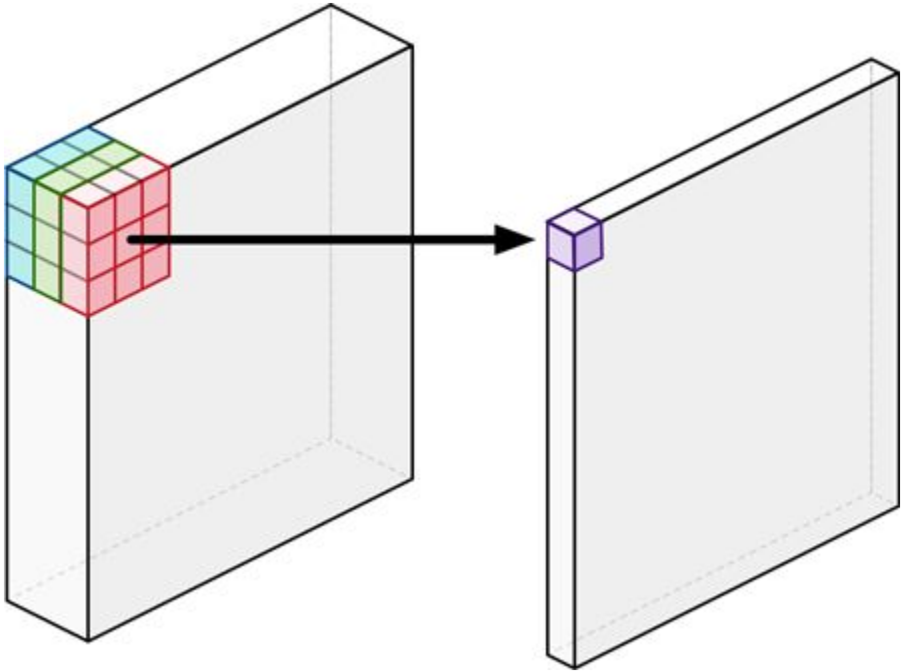


Global average pooling layer is similar to max pooling layer but it reduced dimensions more than max pooling layer.

Source: https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/

Depthwise Separable Convolution

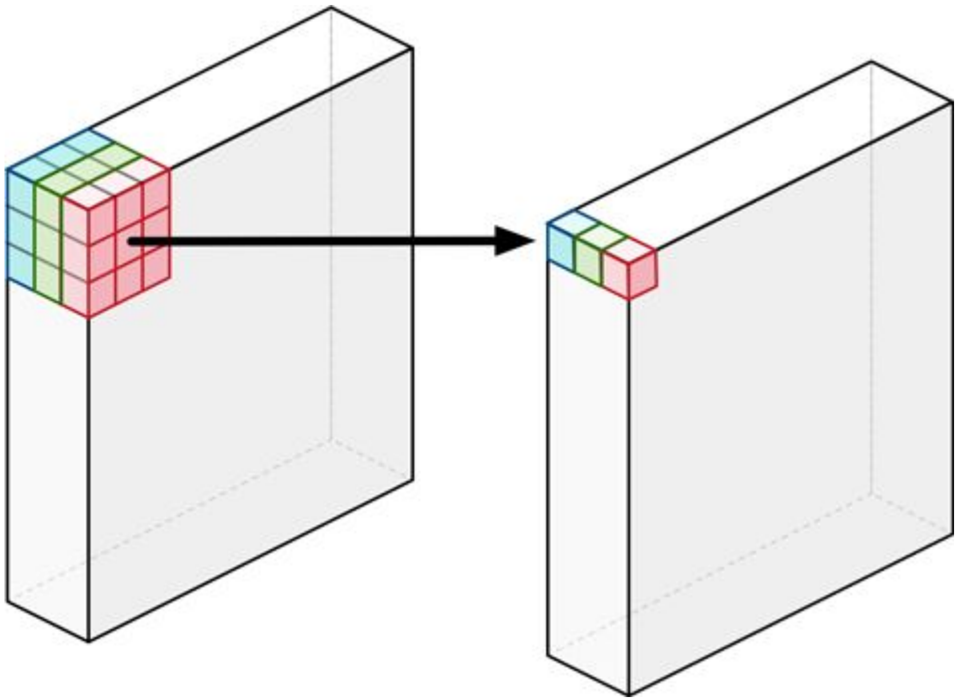
Regular Convolutional Layer



For example, each channel of 3x3x3 kernel slided through each channel of an input image, and sum all values into the feature maps.

Source: https://machinethink.net/blog/googles-mobile-net-architecture-on-iphone/

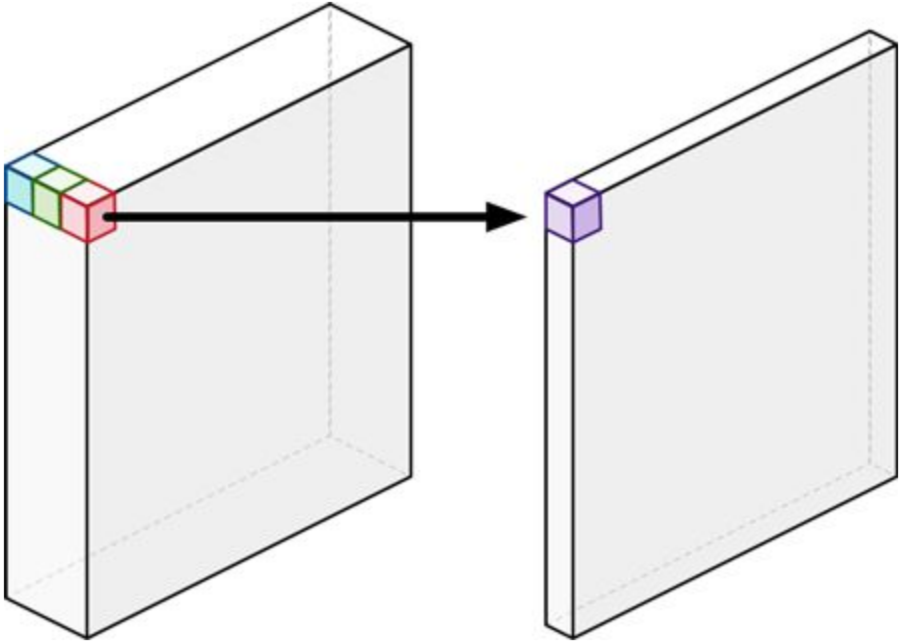
Depthwise Convolutional Layer



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.

Source: https://machinethink.net/blog/googles-mobile-net-architecture-on-iphone/

Pointwise Convolutional Layer



The depthwise convolution is followed by a pointwise convolution. Pointwise convolution is actually a regular convolution with 1x1 kernel size. So, pointwise convolution convert 3 channels input into one channel.

Source: https://machinethink.net/blog/googles-mobile-net-architecture-on-iphone/

Depthwise Separable Convolution is Fast!

Input Image: 224x224x3

Regular Convolution (64 filters)

$(224-3+1)^2$ x (3x3x3) x 64 = 85,162,752 multiplication operations

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

Depthwise Separable Convolution (64 filters)

Depthwise Convolution

$(224-3+1)^2$ x (3x3x1) x 3 = 1,330,668

\+ 3 kernels is number of channels from input image

Pointwise Convolution

$(224-3+1)^2$ x (1x1x3) x 64 = 9,462,528

Total: ? multiplication operations

Depthwise Separable Convolution is Fast!

Input Image: 224x224x3

Regular Convolution (64 filters)

$(224-3+1)^2$ x (3x3x3) x 64 = 85,162,752 multiplication operations

output size: input size - filter size | filter size | number of filters

Depthwise Separable Convolution (64 filters)

Depthwise Convolution

$(224-3+1)^2$ x (3x3x1) x 3 = 1,330,668

+ 3 kernels is number of channels from input image

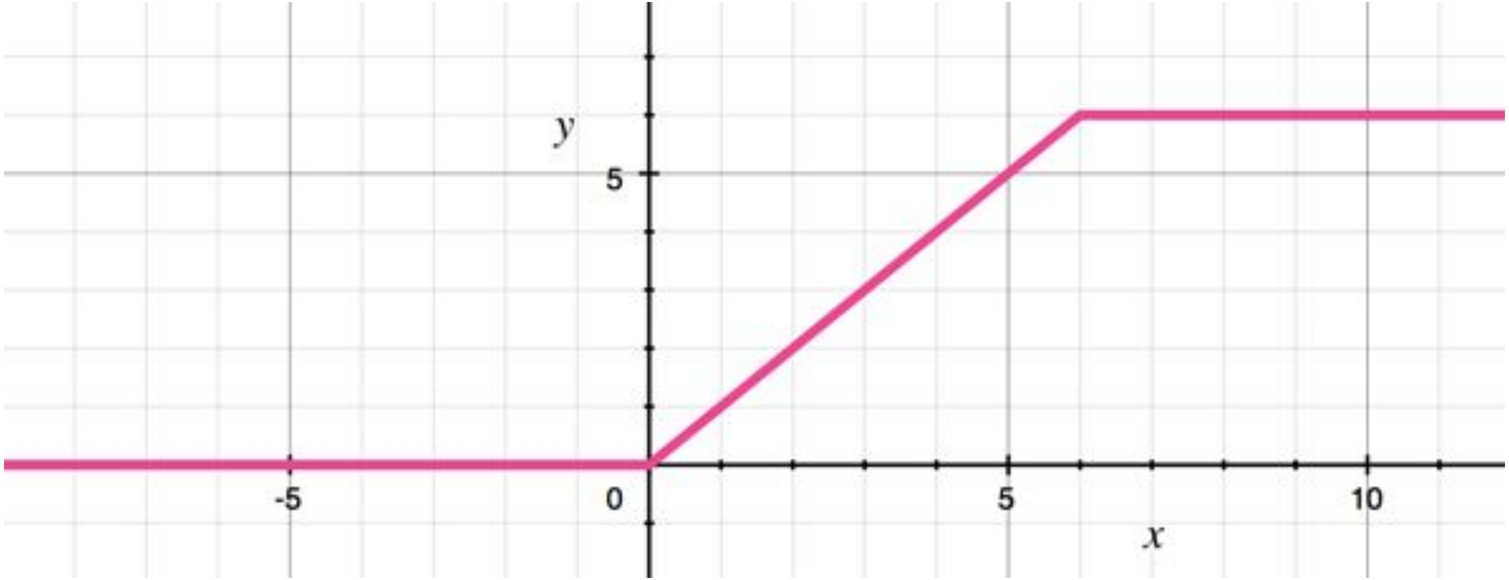
Pointwise Convolution

$(224-3+1)^2$ x (1x1x3) x 64 = 9,462,528

Total: 10,793,196 multiplication operations

7.9 Times Faster!

ReLU6

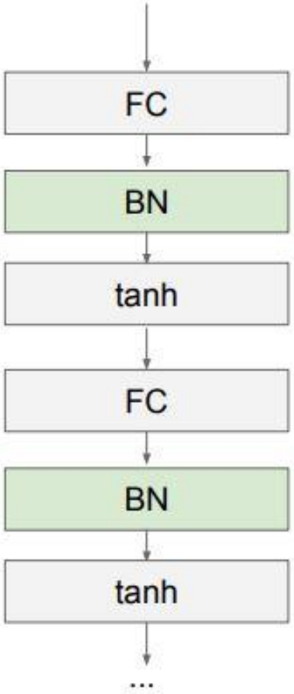


In MobileNet paper, they found that ReLU6 is more robust than regular ReLU when using low-precision computation such as mobile devices.

Source: https://machinethink.net/blog/mobilenet-v2/

Batch Normalization

[Ioffe and Szegedy, 2015]



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

$$\mu = \frac{1}{m} \sum_{i=1}^{m} z^{(i)}$$

$$\sigma^2 = \frac{1}{m} \sum_{i=1}^{m} (z^{(i)} - \mu)^2$$

$$z_{norm}^{(i)} = \frac{z^{(i)} - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

$$z_{bn}^{(i)} = \gamma z_{norm}^{(i)} + \beta$$

Source: https://www.youtube.com/watch?v=wEoyxE0GP2M

Transfer Learning

ImageNet dataset contains the 1000 categories and 1.2 million images.

Source: https://gluon-cv.mxnet.io/build/examples_datasets/imagenet.html

Case: Tiny Dataset
For example, less than 500 labels for each class.

Transferred weights from ImageNet. All these weights are frozen.

Train these classification layers on your dataset.

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	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
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: 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		$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$	Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
	Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
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	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
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	$1 \times 1 \times 1000$