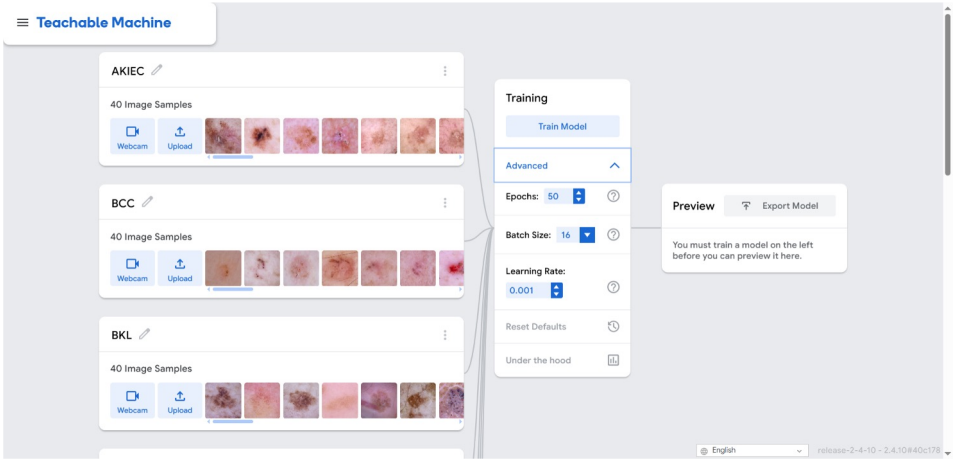
3099704: AI for Digital Health

Neural Networks & Deep Learning

Prof. Peerapon Vateekul, Ph.D.
Peerapon.v@chula.ac.th

Outlines

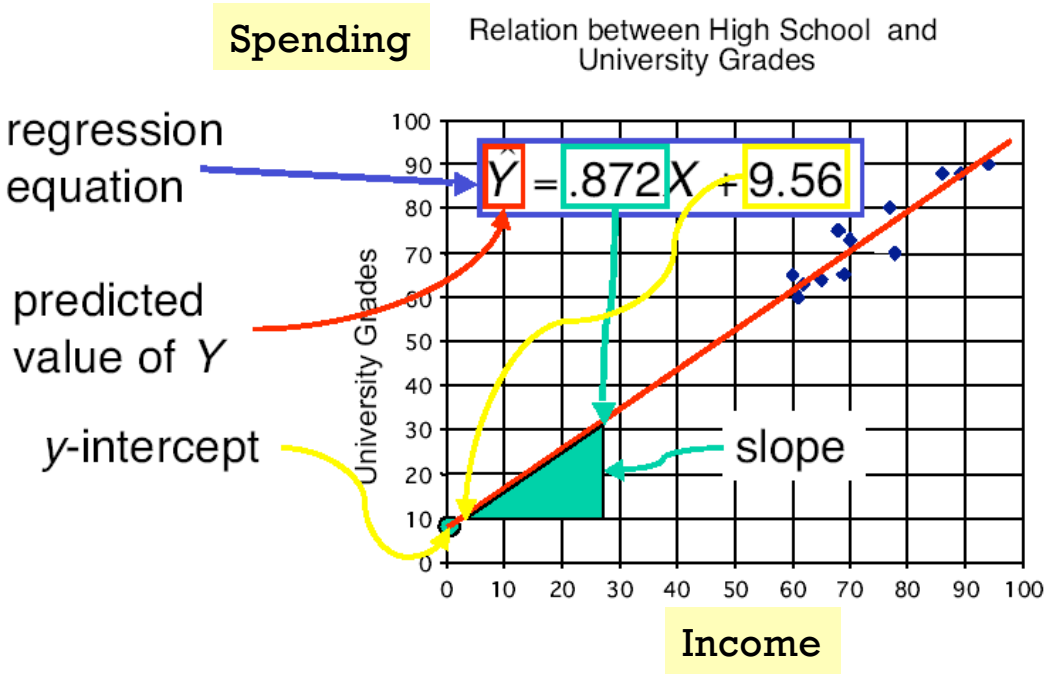
- Linear Regression (recap)
- Neural Networks
- Deep Learning
- Image Classification



- epoch: [5, 20, 50, 100]
- learning rate: [0.00001, 0.001, 0.1]

Linear Regression (recap)

Linear Regression



weight, coefficient

$$\underset{\text{target}}{\hat{y}} = \underset{\text{intercept}}{\hat{w}_0} + \hat{w}_1 x_1 + \hat{w}_2 x_2$$

input

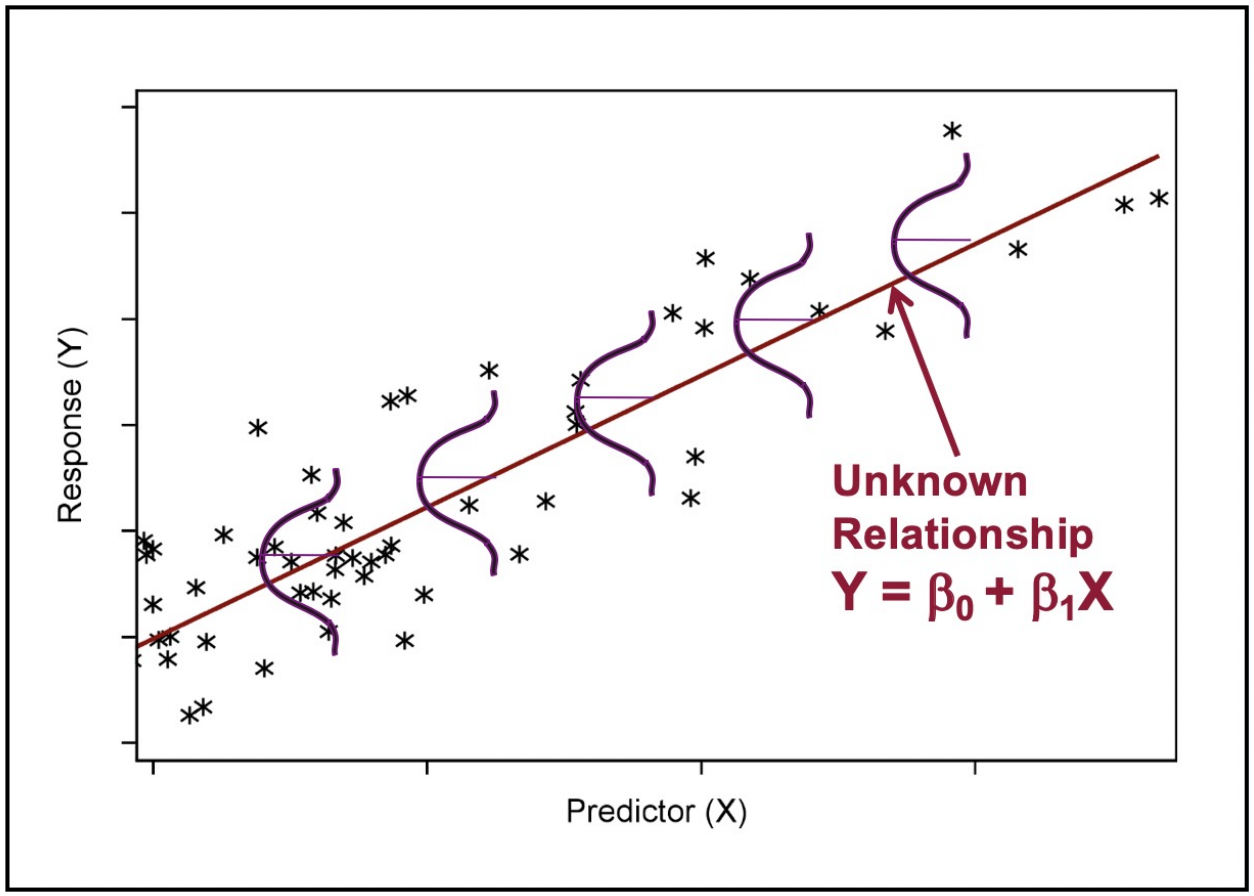
- **The least square method** aims to minimize the following term

$$\sum_{\substack{training \\ data}} (y_i - \hat{y}_i)^2$$

CHULA ENGINEERING COMPUTER
Foundation toward Innovation

Linear Regression Assumption

Spending = 500 + 10*Income10K + 2*Age

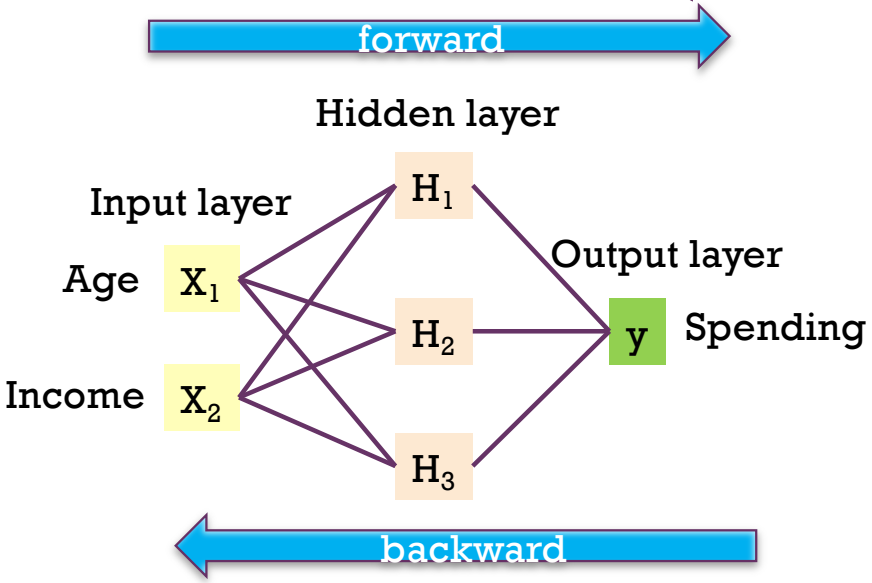


- Linear relationship between (y, x_i). [Pearson correlation]
- Error is normal distributed. [remove outliers, log-transformation]
- Error has equal variance (homoscedasticity) [remove outliers, log-transformation]
- Errors are independent from each other. [design new data correction]

+

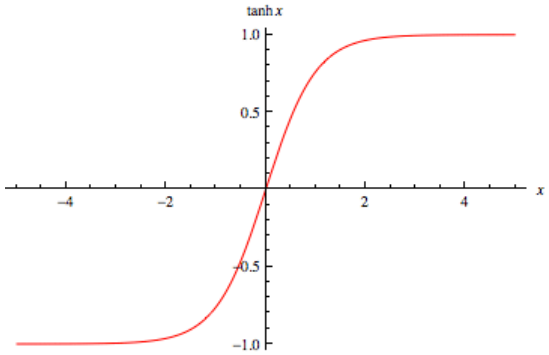
Neural Networks

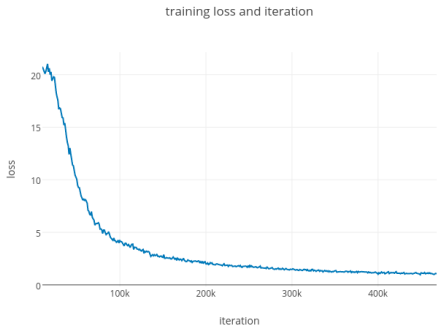
Neural Networks (universal approximator)



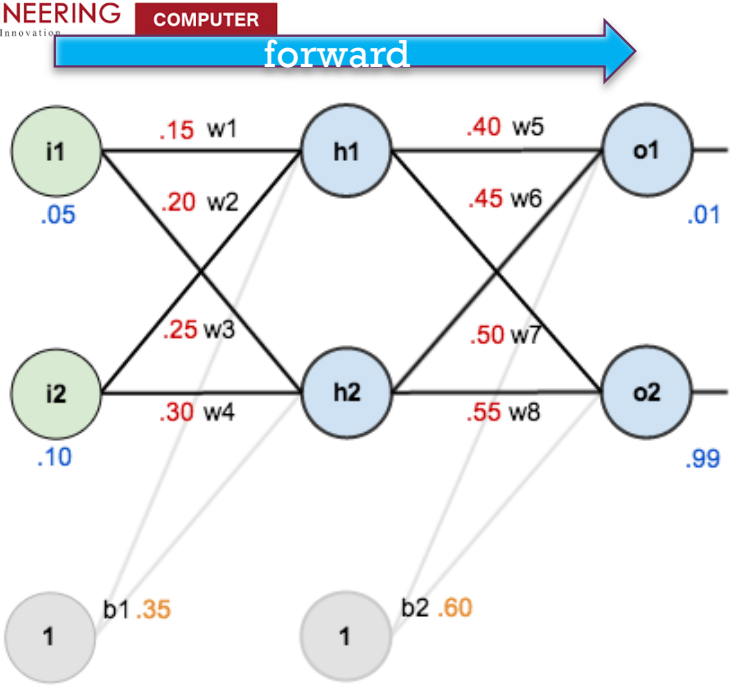
$$Spending = \hat{w}_0 + \hat{w}_1 H_1 + \hat{w}_2 H_2 + \hat{w}_3 H_3$$

$$H_1 = tanh(\hat{w}_{10} + \hat{w}_{11} x_1 + \hat{w}_{12} x_2)$$
$$H_2 = tanh(\hat{w}_{20} + \hat{w}_{21} x_1 + \hat{w}_{22} x_2)$$
$$H_3 = tanh(\hat{w}_{30} + \hat{w}_{31} x_1 + \hat{w}_{32} x_2)$$





How to update weight
https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/



$$E_{total} = \sum \frac{1}{2}(target - output)^2$$

$$E_{o1} = \frac{1}{2}(target_{o1} - out_{o1})^2 = \frac{1}{2}(0.01 - 0.75136507)^2 = 0.274811083$$

Repeating this process for o_2 (remembering that the target is 0.99) we get:

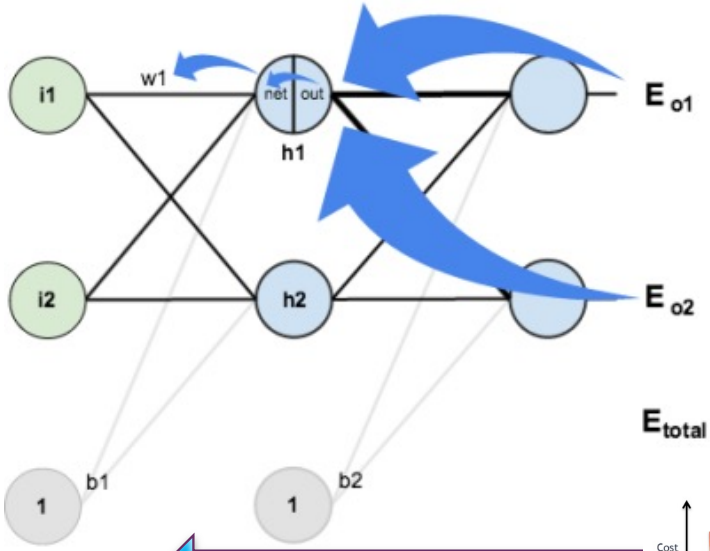
$$E_{o2} = 0.023560026$$

The total error for the neural network is the sum of these errors:

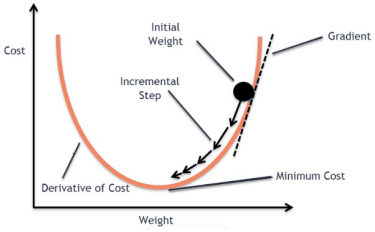
$$E_{total} = E_{o1} + E_{o2} = 0.274811083 + 0.023560026 = 0.298371109$$

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$



$$E_{total} = E_{o1} + E_{o2}$$



Putting it all together:

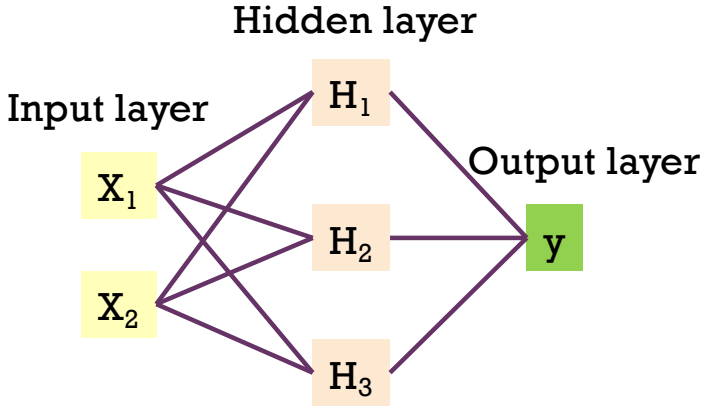
$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

$$\frac{\partial E_{total}}{\partial w_5} = 0.74136507 * 0.186815602 * 0.593269992 = 0.082167041$$

$$w_5^+ = w_5 - \eta * \frac{\partial E_{total}}{\partial w_5} = 0.4 - 0.5 * 0.082167041 = 0.35891648$$

Learning rate

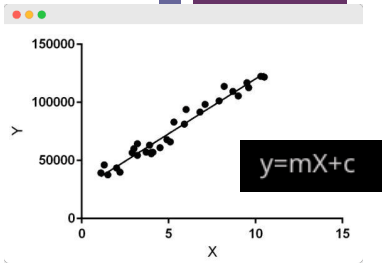
Gradient descent (solver to find weights)

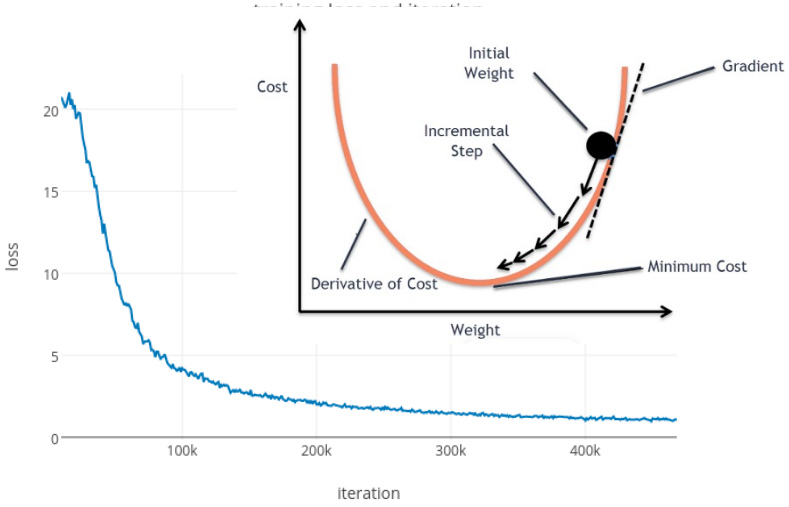


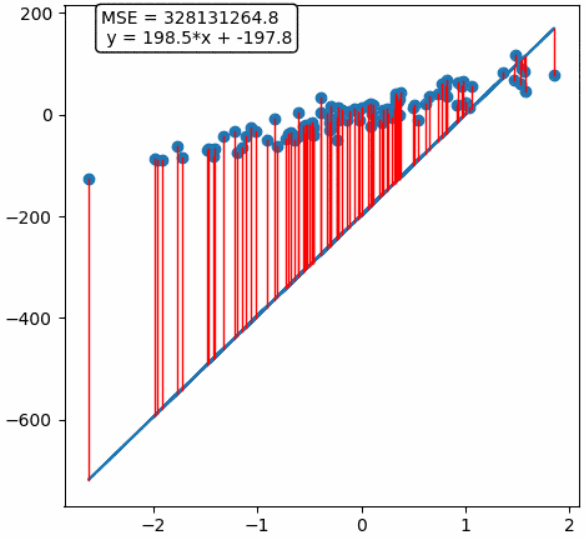
*New weight = weight – Learning rate * Gradient*

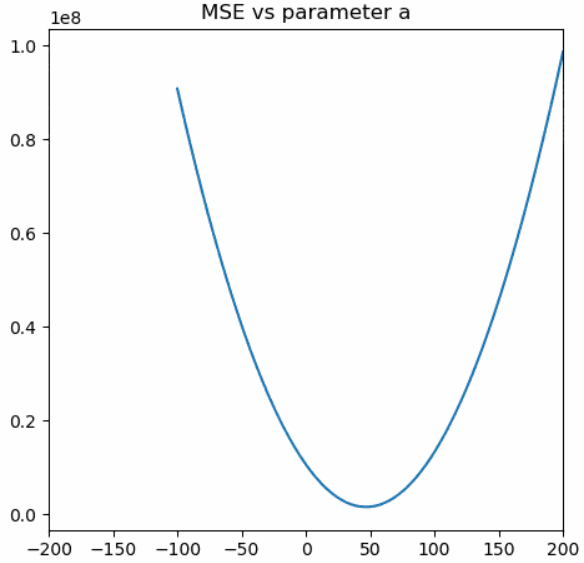
$$*W_x = W_x - a\left(\frac{\partial Error}{\partial W_x}\right)$$

Old weight
Derivative of Error with respect to weight
New weight
Learning rate

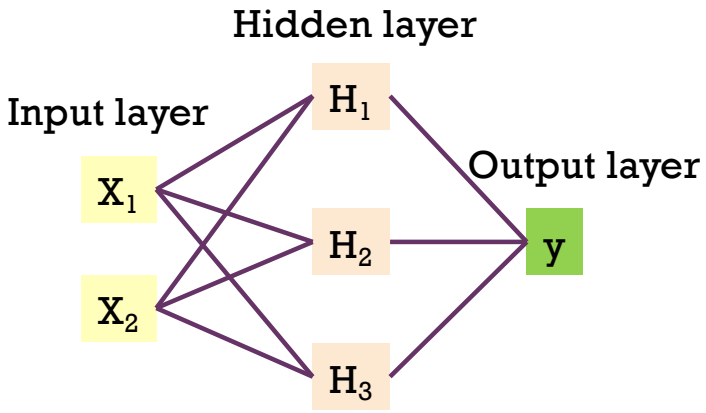








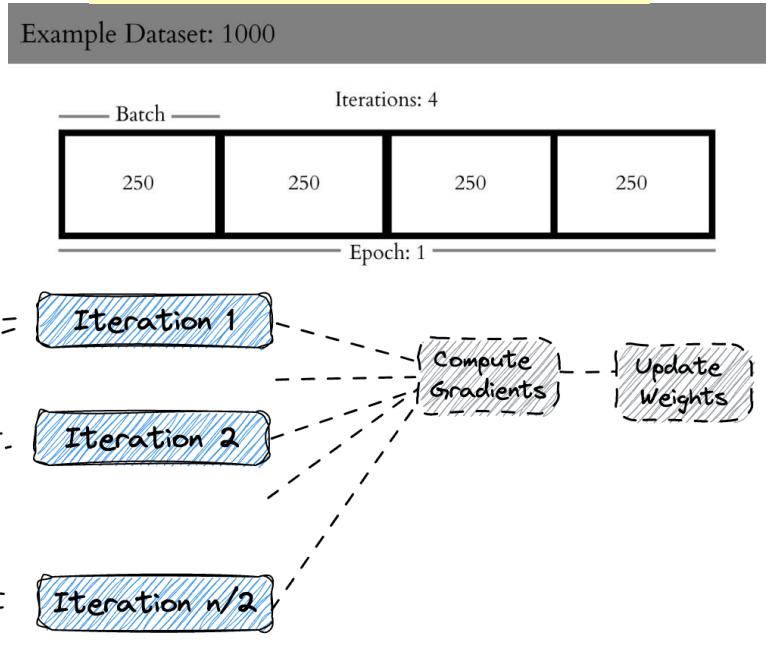
Batch, Iteration, Epoch



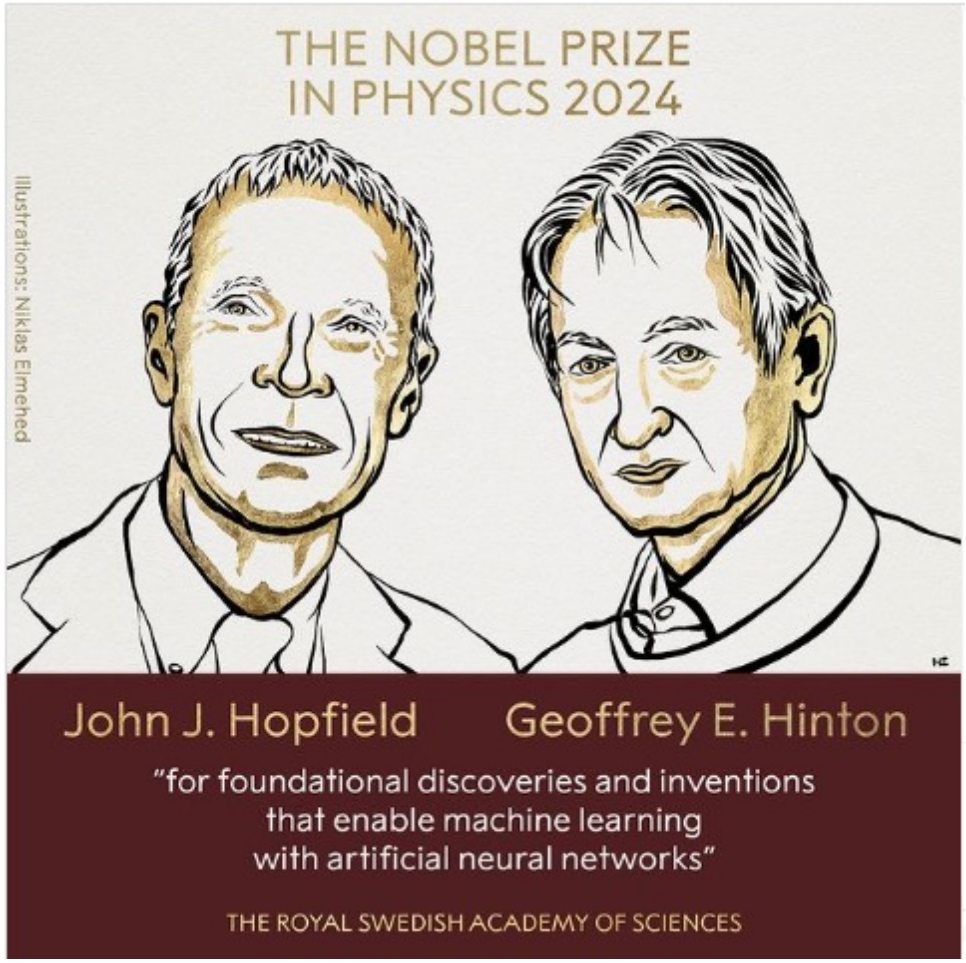
Stop when?
- Converge (no change in loss)
- Max epochs

Important Params:
- #hidden units, #hidden layers
- Learning rate, momentum, decay
- Seed number, etc.

Epoch vs Iterations vs Batch



Basic research (foundation research)

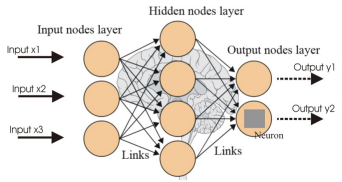


8 October 2024

The Royal Swedish Academy of Sciences has decided to award the Nobel Prize in Physics 2024 to

John J. Hopfield
Princeton University, NJ, USA

Geoffrey E. Hinton
University of Toronto, Canada

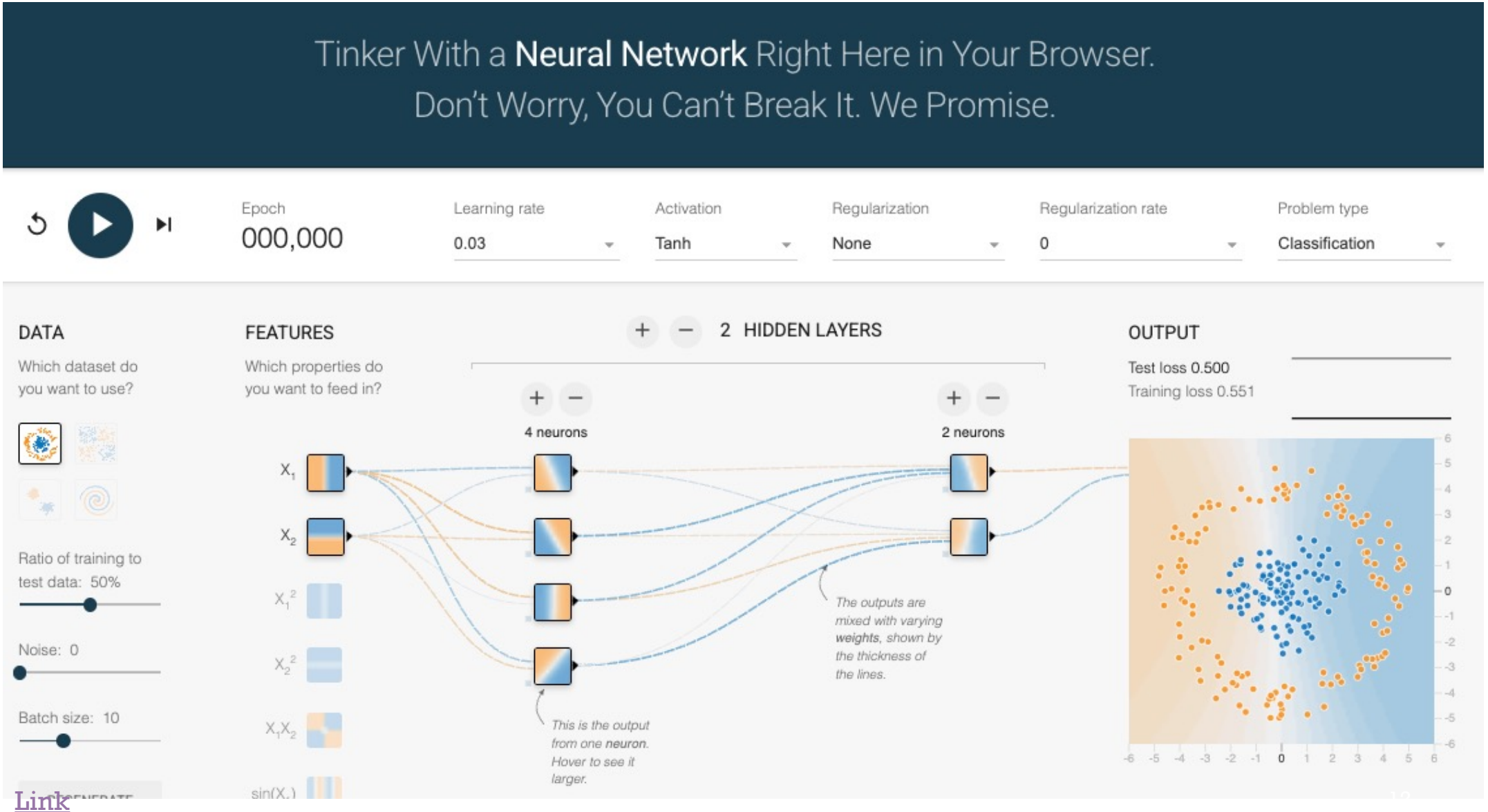


"for foundational discoveries and inventions that enable machine learning with artificial neural networks"

They trained artificial neural networks using physics

John Hopfield invented a network that uses a method for saving and recreating patterns. We can imagine the nodes as pixels. The *Hopfield network* utilises physics that describes a material's characteristics due to its atomic spin – a property that makes each atom a tiny magnet.

Geoffrey Hinton used the Hopfield network as the foundation for a new network that uses a different method: the *Boltzmann machine*. This can learn to recognise characteristic elements in a given type of data.


Tinker With a Neural Network Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.
Epoch
000,000
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
FEATURES
Which properties do you want to feed in?
2 HIDDEN LAYERS
4 neurons
2 neurons
The outputs are mixed with varying weights, shown by the thickness of the lines.
This is the output from one neuron. Hover to see it larger.
OUTPUT
Test loss 0.500
Training loss 0.551

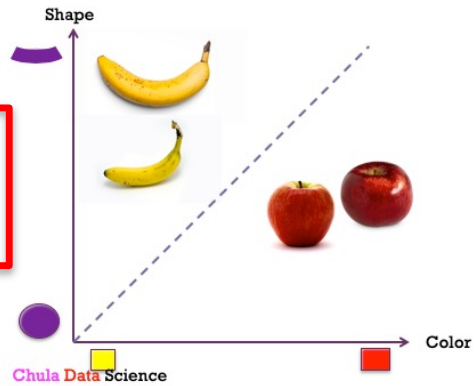
Link

+

Deep Learning

Handcrafted features

Age	Income	Gender	Province	Corona
25	25,000	Female	Bangkok	Yes
35	50,000	Female	Nontaburi	Yes
32	35,000	Male	Bangkok	No

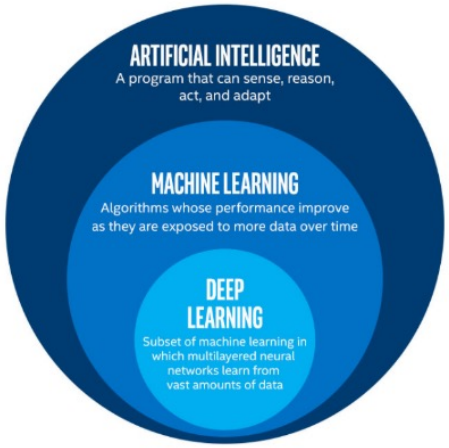


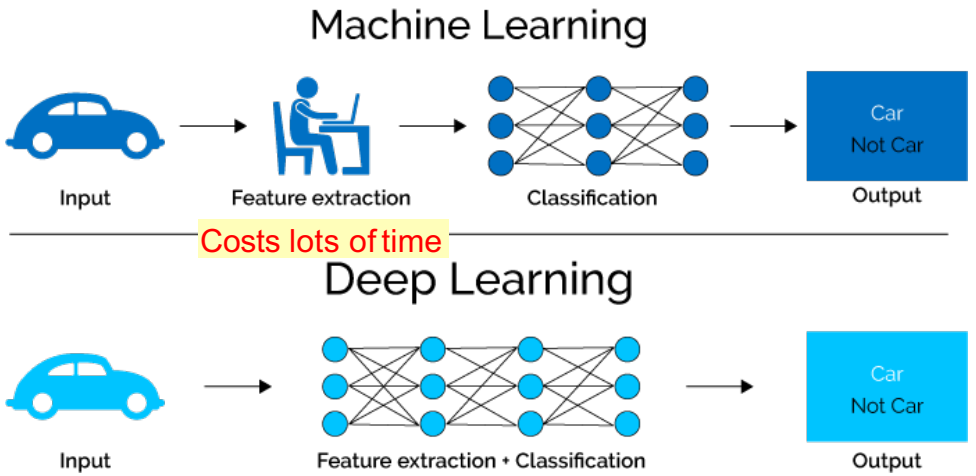
Can we still tell the features (columns)?

What is Deep Learning (DL)?

Part of the machine learning field of learning representations of data. Exceptional effective at learning patterns.

Utilizes learning algorithms that derive meaning out of data by using a hierarchy of multiple layers that mimic the neural networks of our brain.



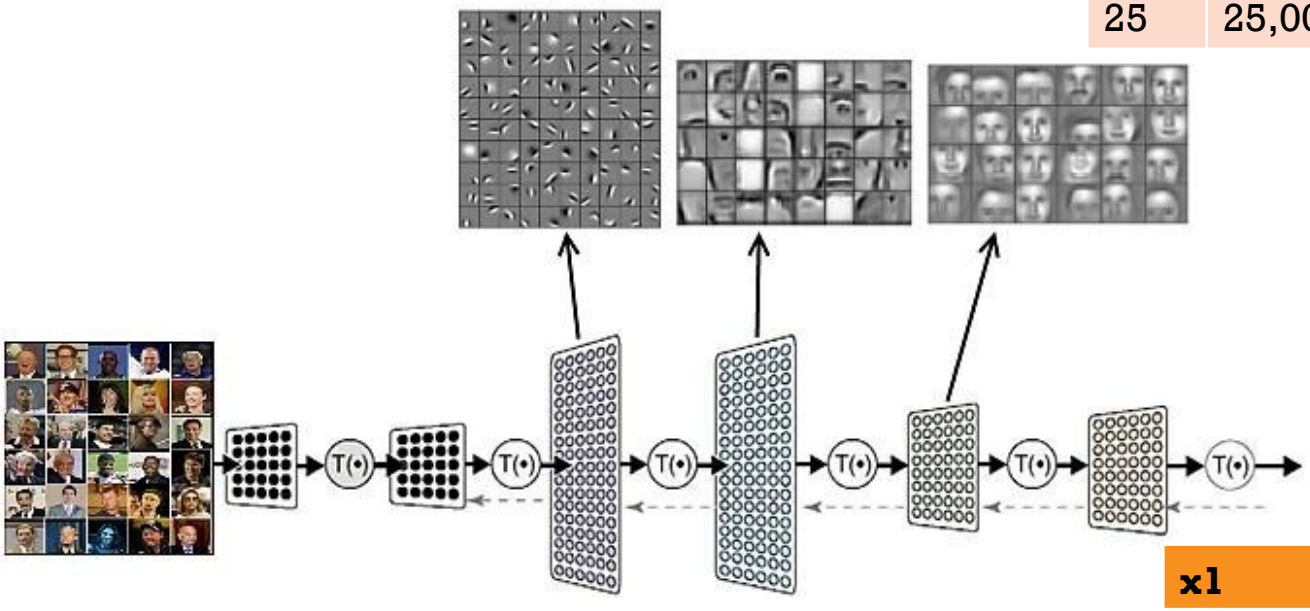


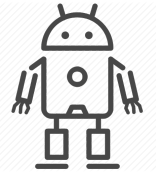
Deep Learning – Basics (cont.)

What did it learn?

A deep neural network consists of a hierarchy of layers, whereby each layer transforms the input data into more abstract representations (e.g., edge -> nose -> face). The output layer combines those features to make predictions.

Age	Income	Gender	Province	Corona
25	25,000	Female	Bangkok	Yes





x1	x2	x3	x4	Corona
0.7	0.2	-0.5	-0.1	Yes

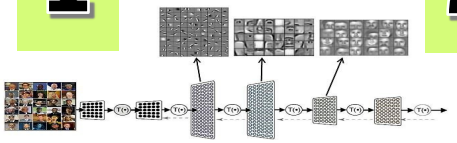
Deep Learning Application

Speech Recognition

Computer Vision

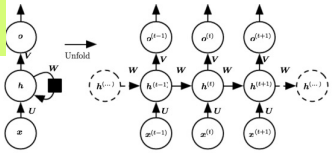
Natural Language Processing

1



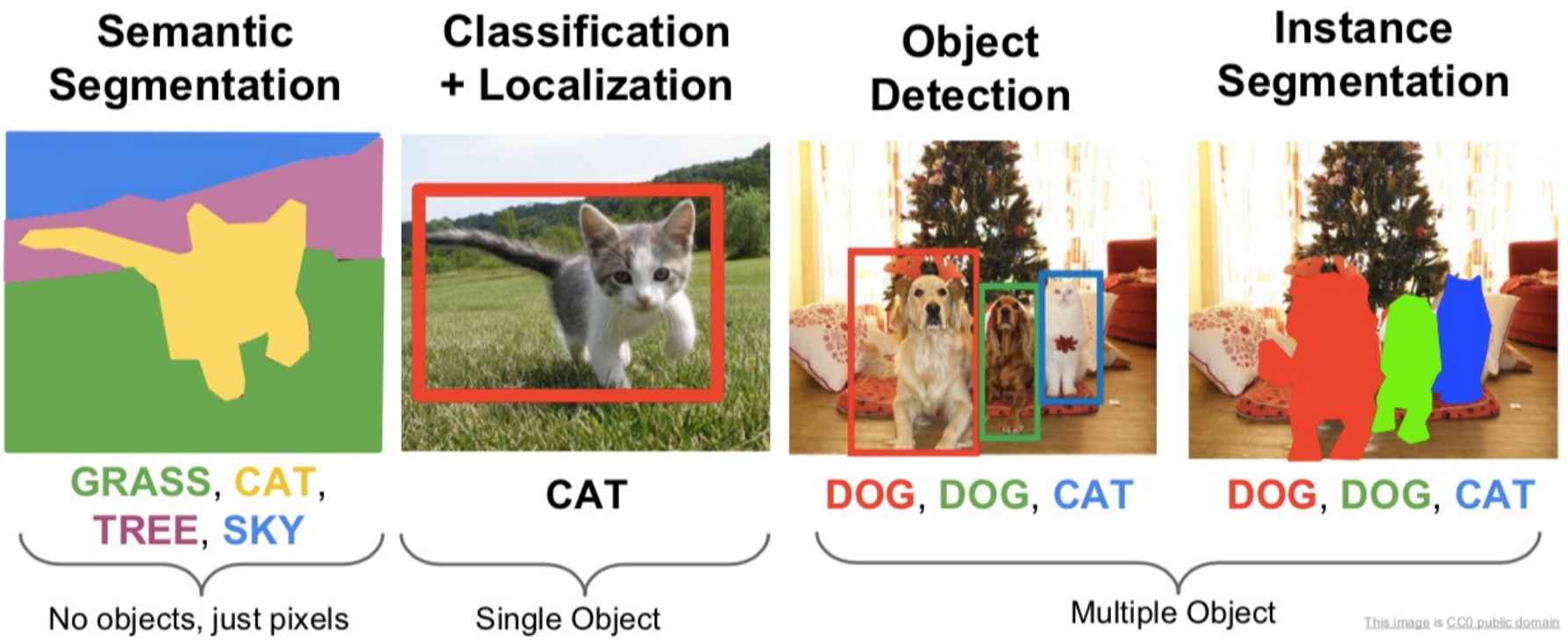
CNN

2



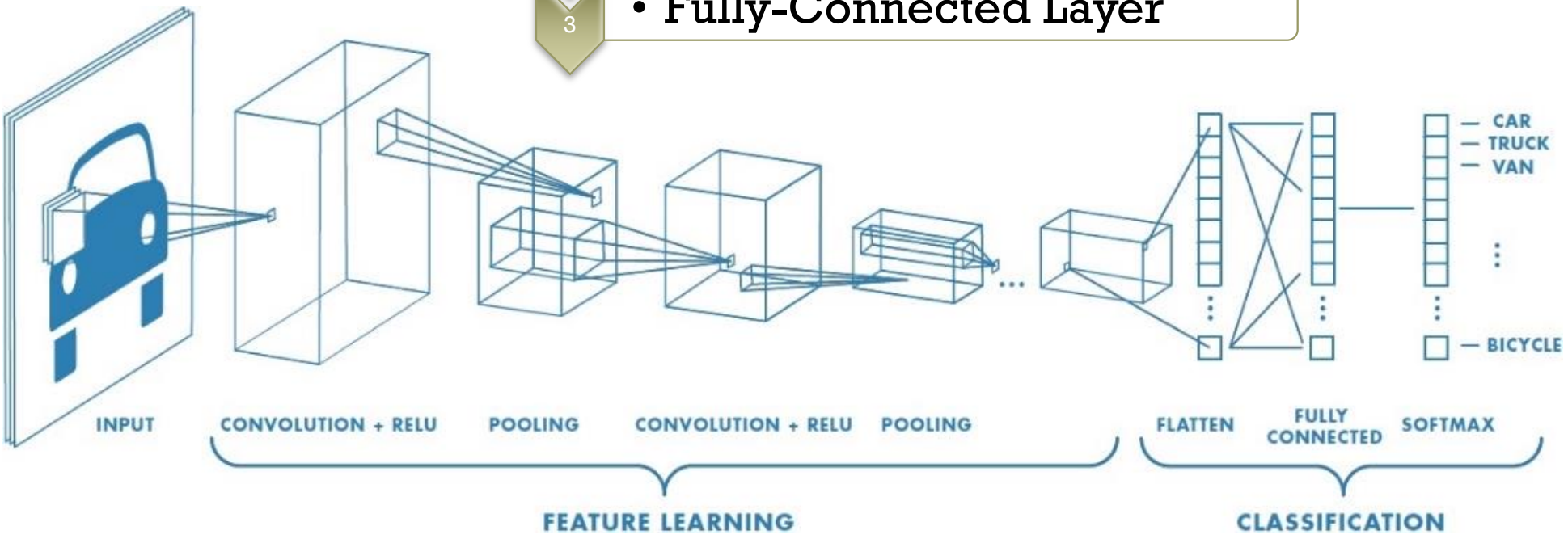
RNN

Type of image tasks

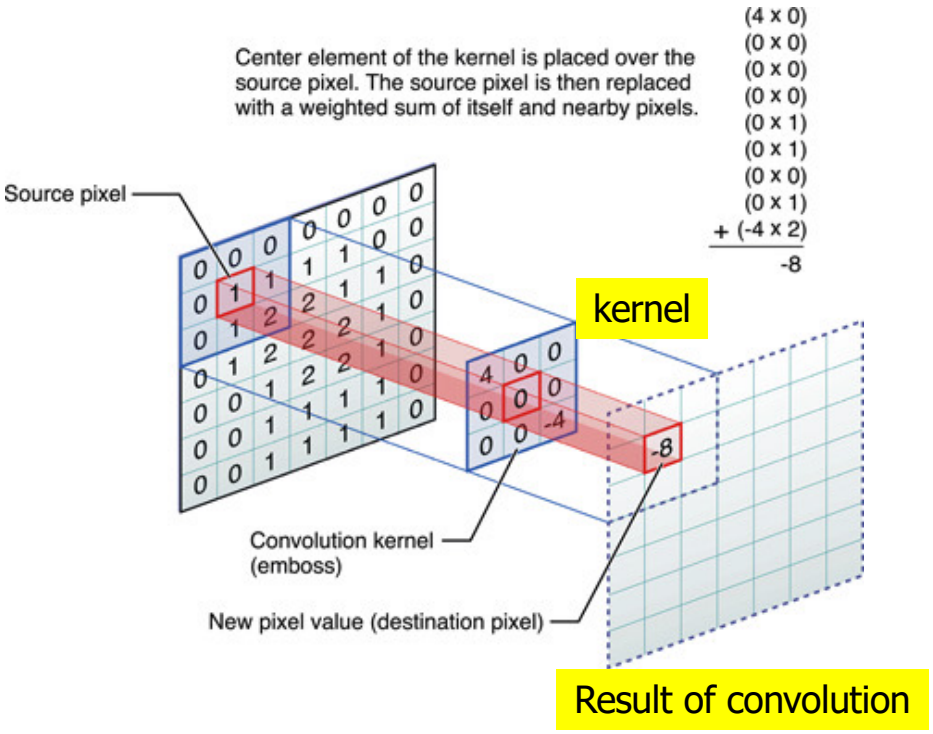


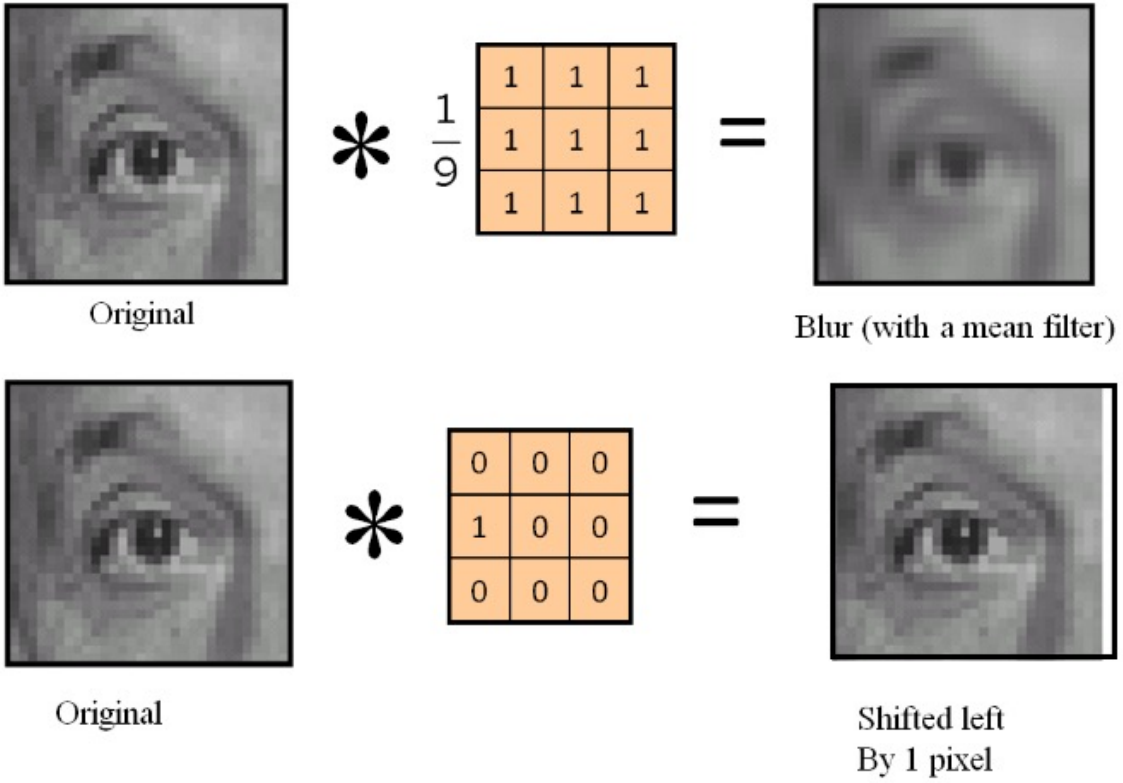
Convolutional Neural Networks (CNN)

1. Convolutional Layer
2. Pooling Layer
3. Fully-Connected Layer



Convolution



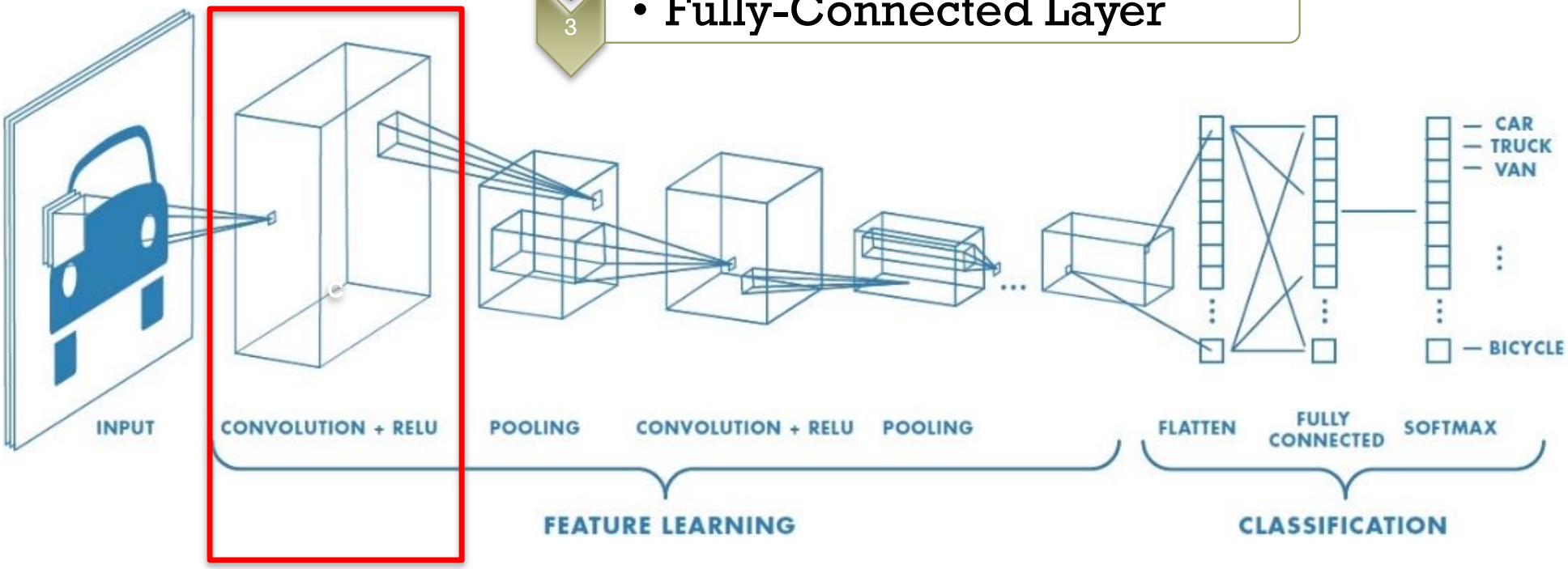


Changing the kernel (filter) results in different image processing outcomes

Convolutional Neural Networks (CNN)

1) Convolutional Layer

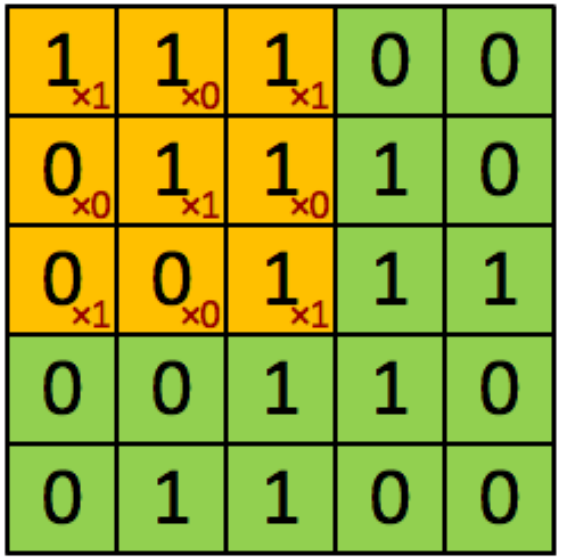
1. Convolutional Layer
2. Pooling Layer
3. Fully-Connected Layer



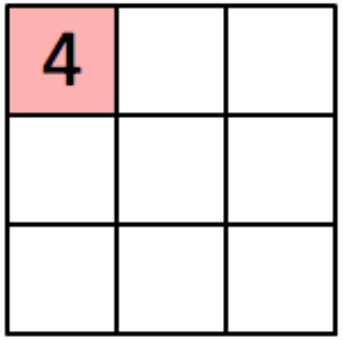
Layer 1 : Convolutional Layer

(3*3 filter)

1	0	1
0	1	0
1	0	1



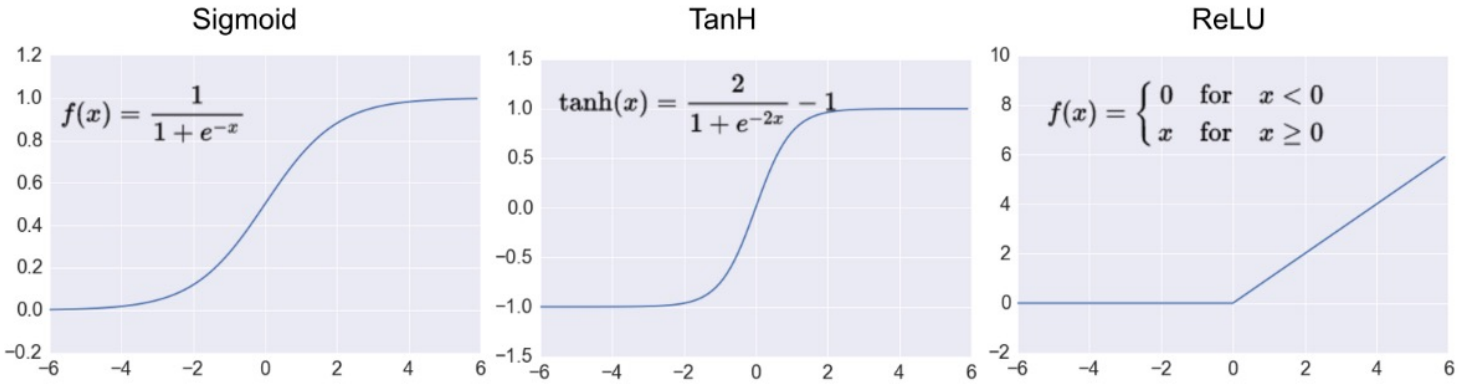
Image

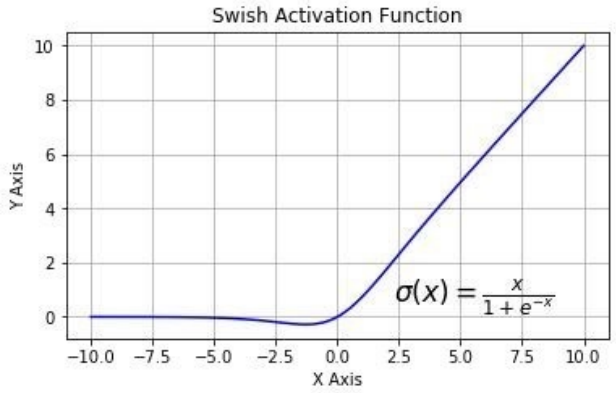


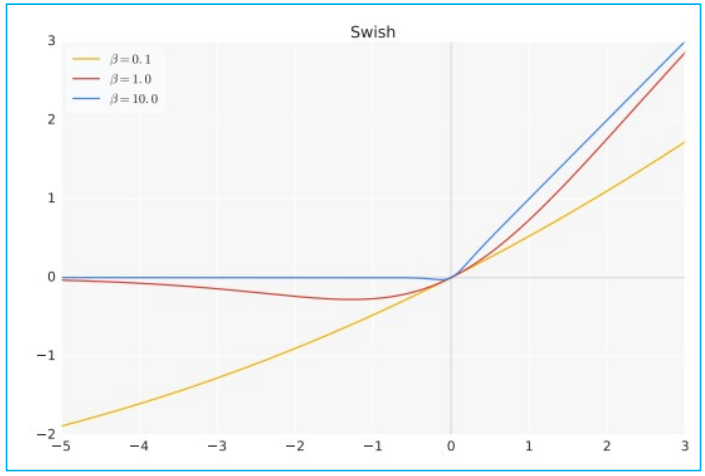
Convolved Feature

CNNs: 1) Convolutional Layer (cont.)

Non-Linear Activation Function



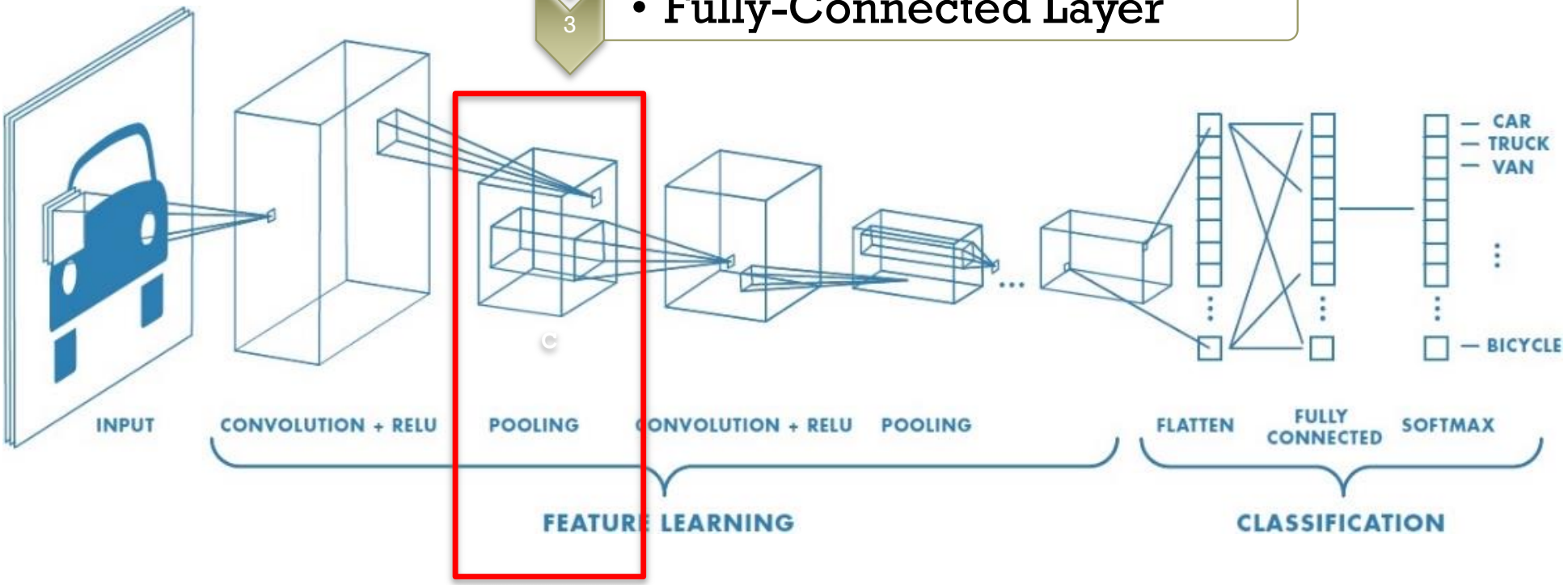




Convolutional Neural Networks (CNN)

2) Pooling Layer

1. Convolutional Layer
2. Pooling Layer
3. Fully-Connected Layer

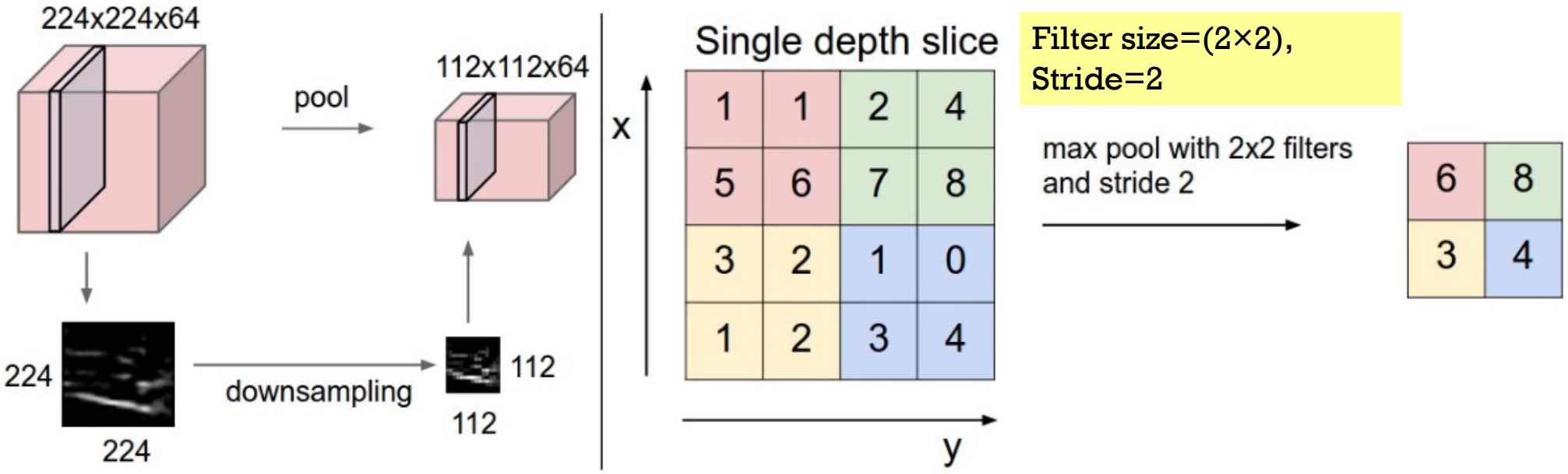


CNNs: 2) Pooling Layer

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

Summary: **2 parameters** for pooling layer
- Filter size, Stride

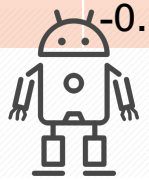
- Feature selection (reduction); downsampling



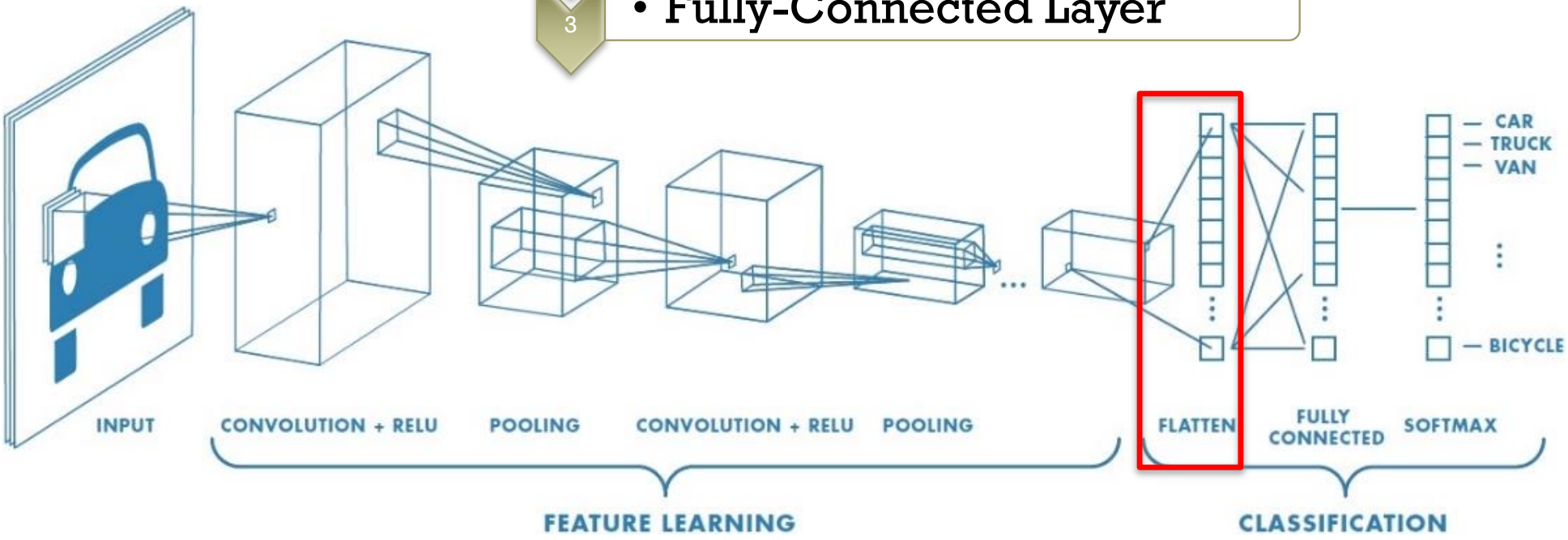
Convolutional Neural Networks

Embedding Vector

x1	x2	x3	x4	Corona
0.7	0.2	-0.5	-0.1	Yes



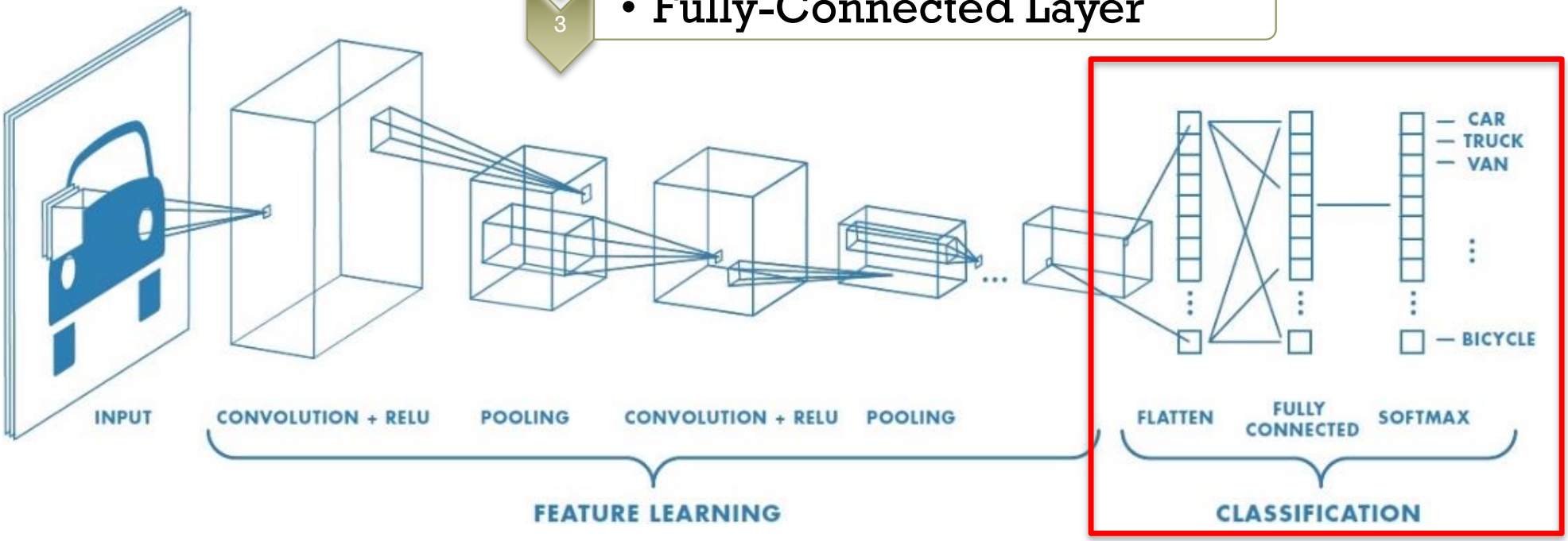
1. • Convolutional Layer
2. • Pooling Layer
3. • Fully-Connected Layer



Convolutional Neural Networks (CNN)

3) Fully-Connected Layer (FC) = Neural Networks

1. Convolutional Layer
2. Pooling Layer
3. Fully-Connected Layer



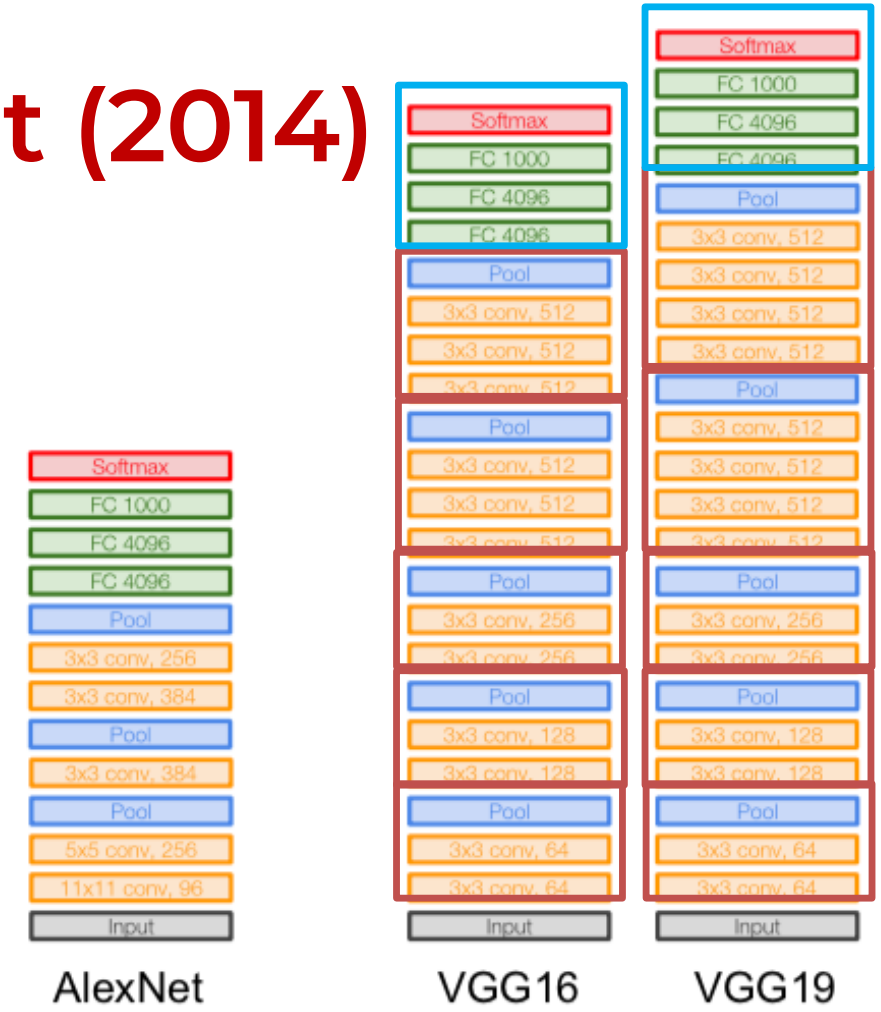
CNN Example: VGGNet (2014)

[Simonyan and Zisserman, 2014]

Small filters, Deeper networks

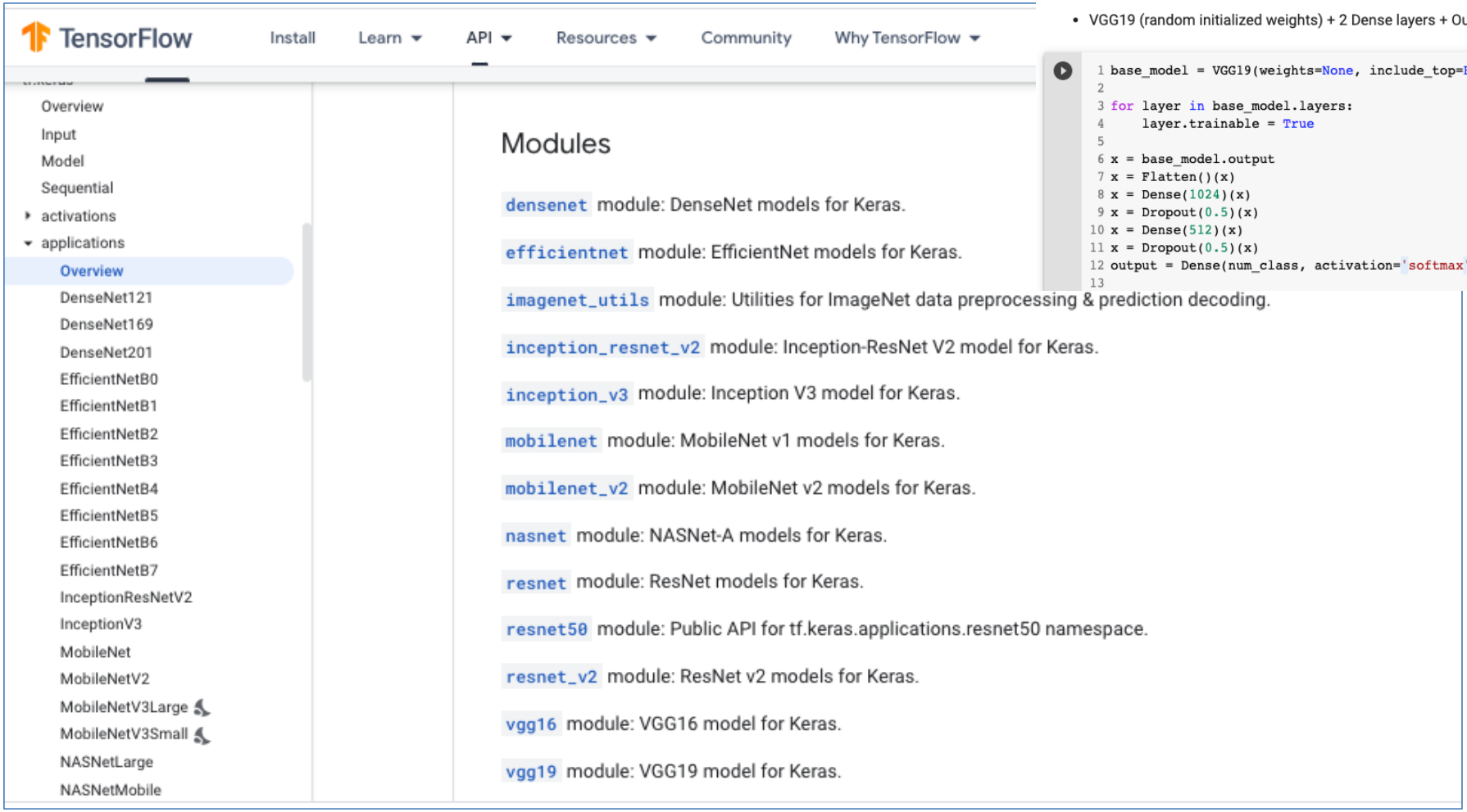
8 layers (AlexNet) → 16-19 layers (VGG16Net)

Only 3x3 CONV Stride 1, pad 1
And 2x2 MAX POOL stride 2



Network	A	B	C	D	E
Number of parameters (Millions)	133	133	134	138	144

28



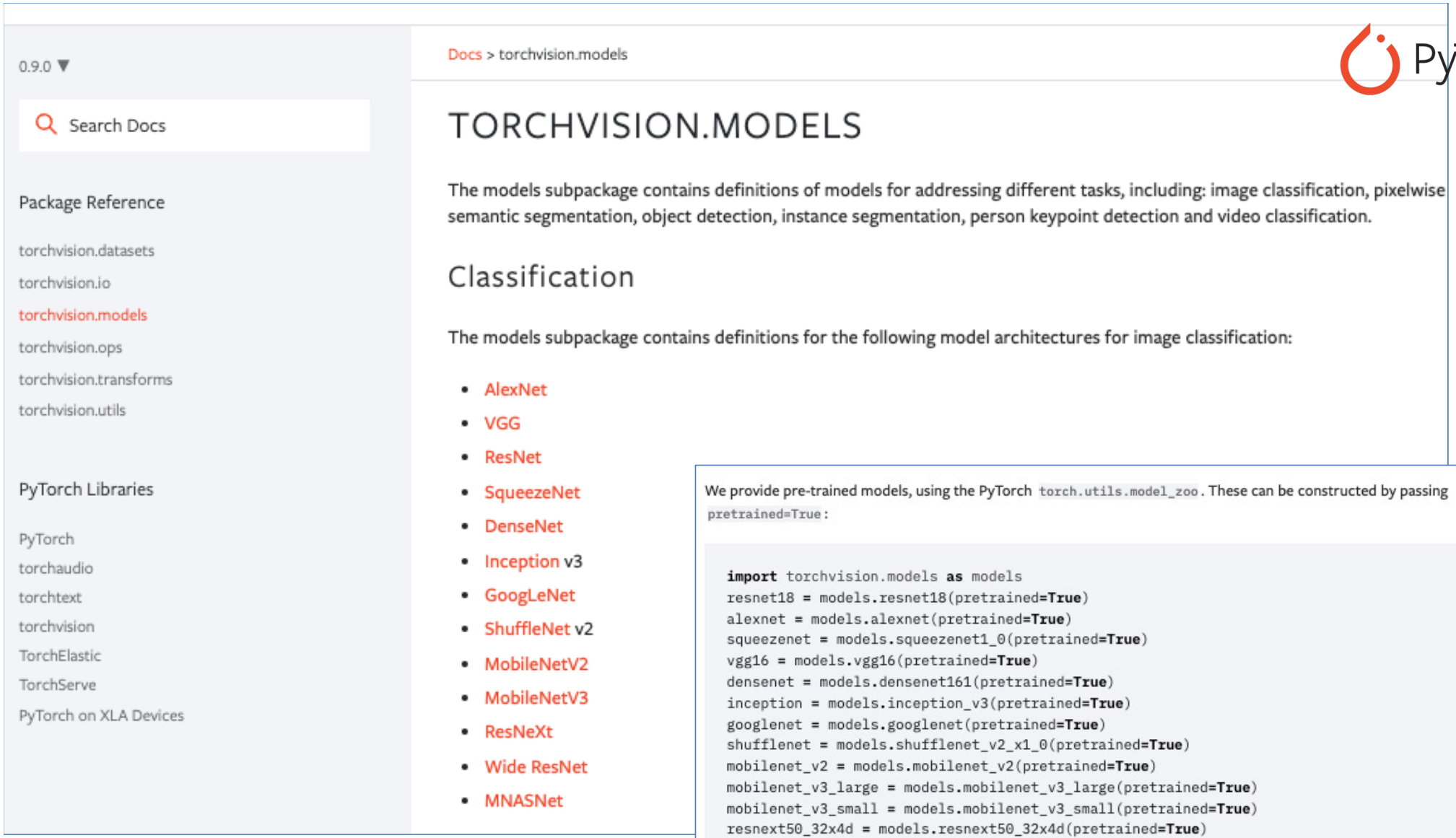
Model 1

- VGG19 (random initialized weights) + 2 Dense layers + Output layer

```
base_model = VGG19(weights=None, include_top=False, input_shape=(224, 224, 3))

for layer in base_model.layers:
    layer.trainable = True

x = base_model.output
x = Flatten()(x)
x = Dense(1024)(x)
x = Dropout(0.5)(x)
x = Dense(512)(x)
x = Dropout(0.5)(x)
output = Dense(num_class, activation='softmax')(x)

```

https://www.tensorflow.org/api_docs/python/tf/keras/applications

29

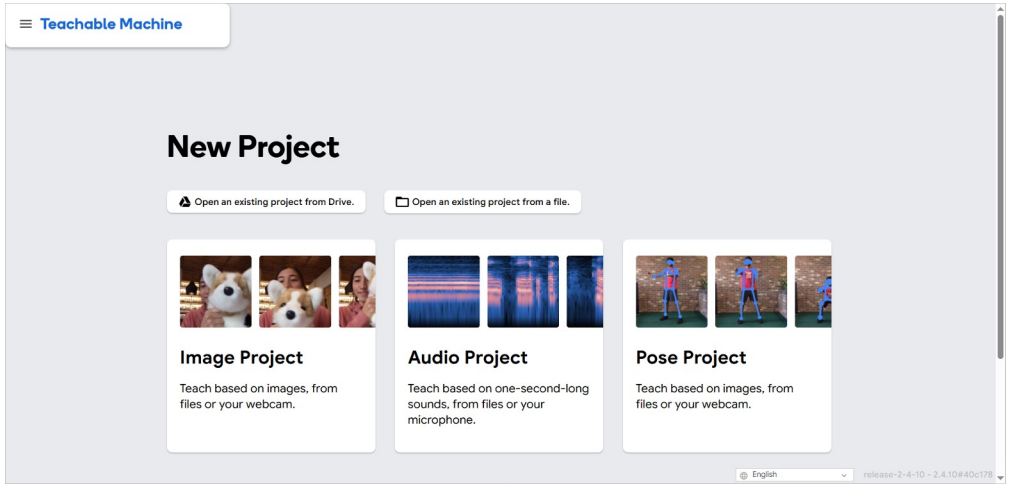


https://pytorch.org/vision/0.9/models.html

30

Google Teachable Machine

- **Teachable Machine** is a web-based tool from Google that allows you to easily create models without writing code or installing software. Simply access it through your web browser at https://teachablemachine.withgoogle.com/



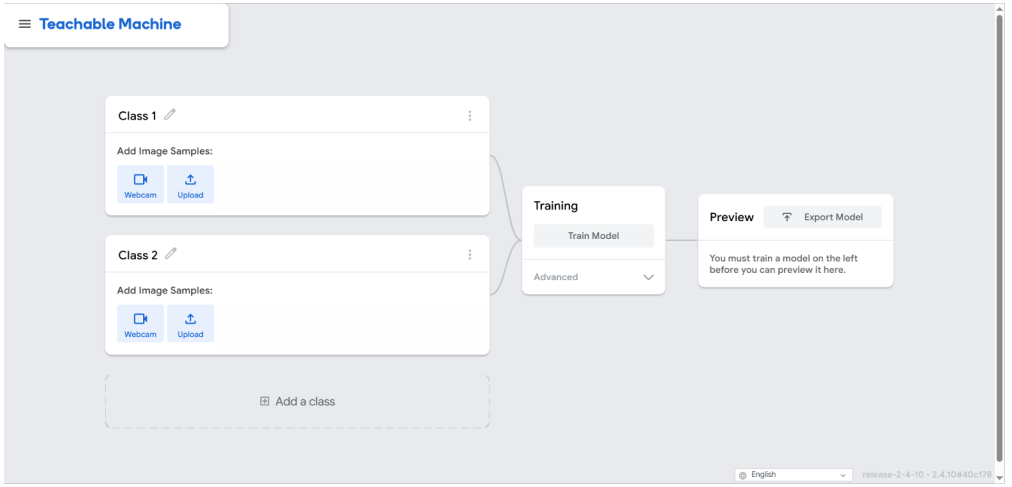
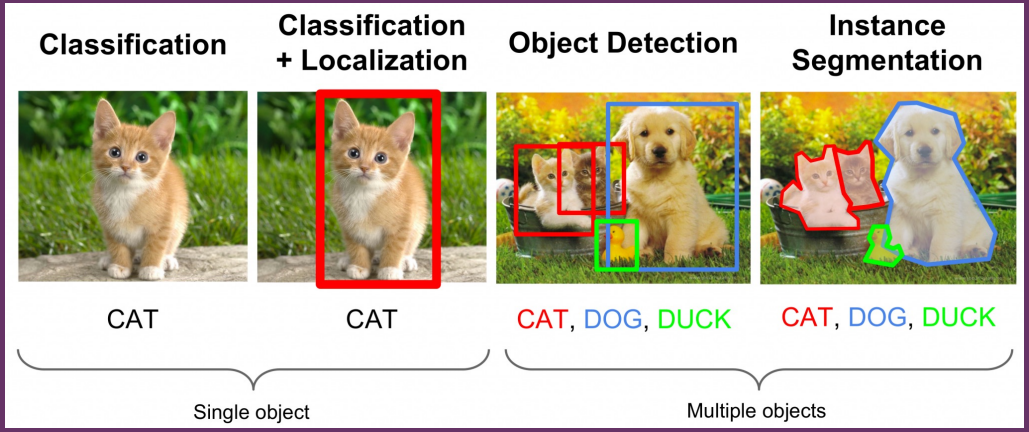
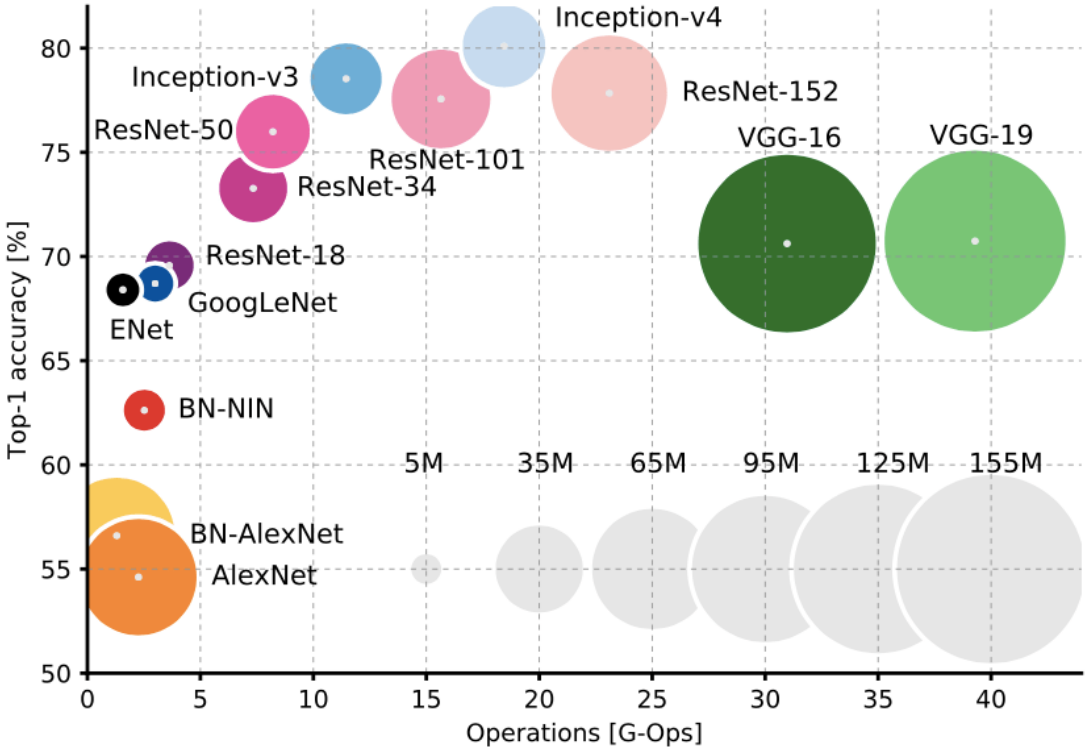


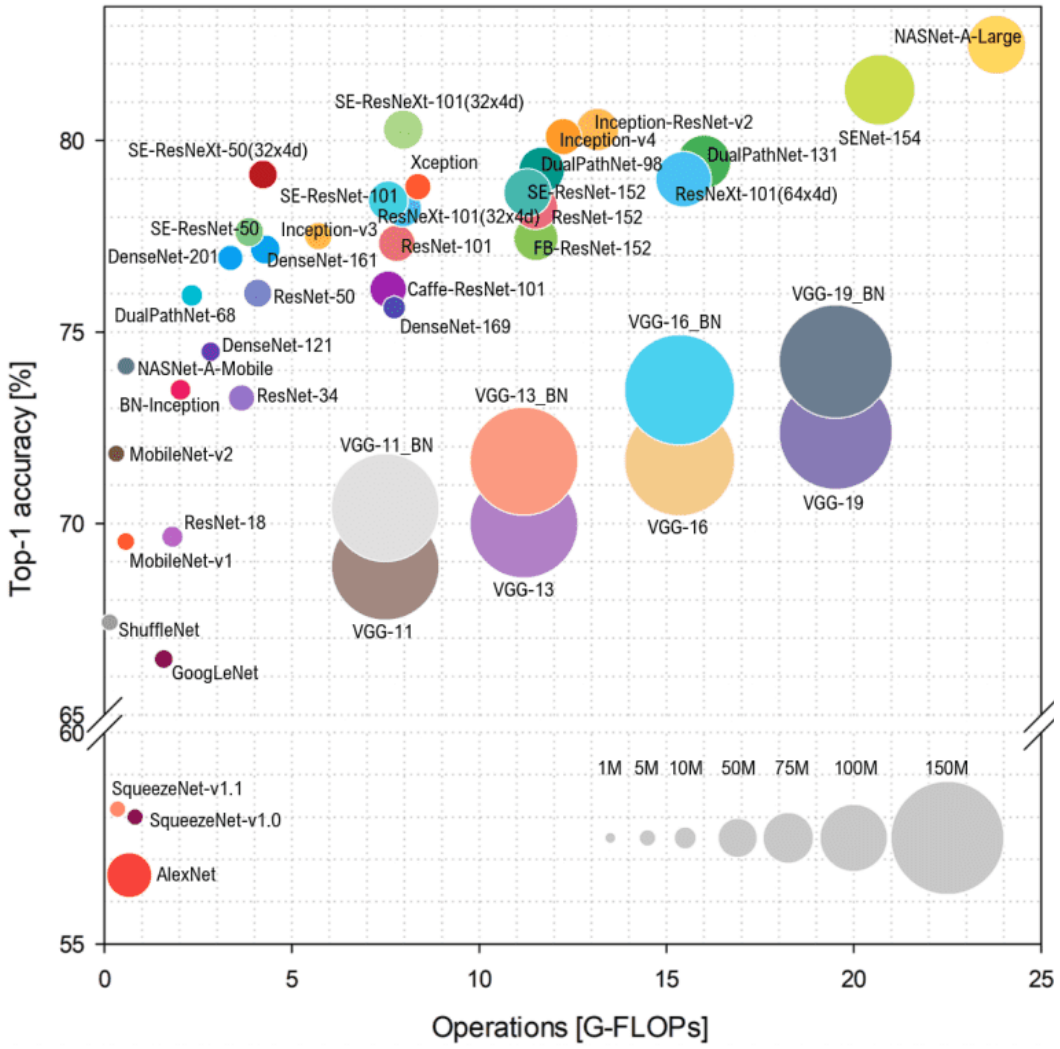
Image Classification



SOTA of Image Classification



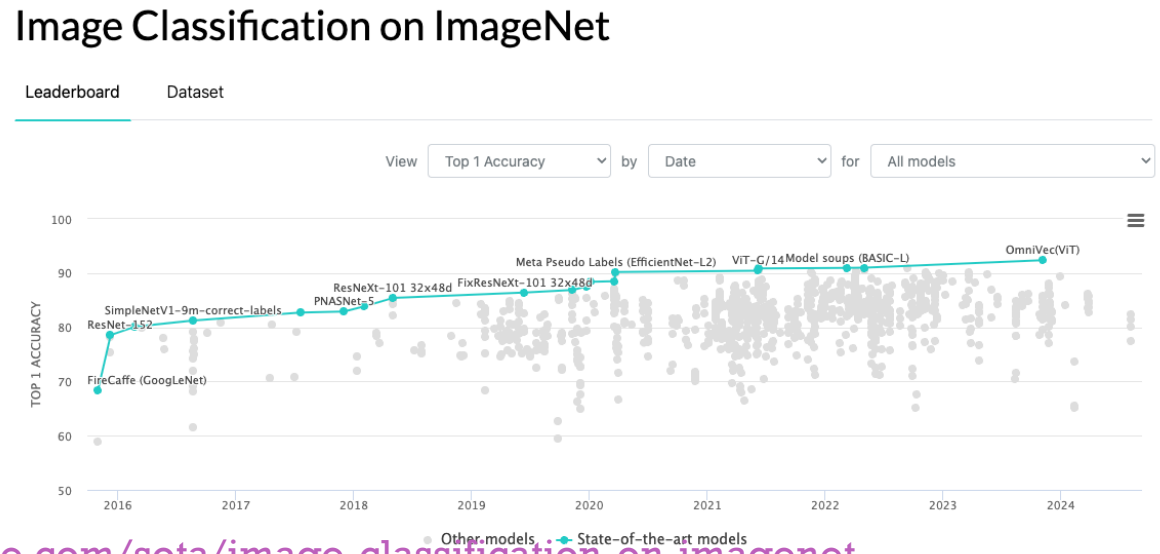
https://blog.csdn.net/qq_34216467/article/details/83061692



https://theaisummer.com/cnn-architectures/

Common Image Classification Models

Model	Year	Team / Organization
ResNet	2015	Microsoft Research
MobileNet	**2017**	**Google**
EfficientNet	**2019**	**Google**
Vision Transformer (ViT)	2020	Google Research
Swin Transformer	2021	Microsoft Research
ConvNeXt	2022	Meta AI



https://paperswithcode.com/sota/image-classification-on-imagenet

EfficientNet (May, 2019) → EffNetV2 (2021)

EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks

Mingxing Tan [1] Quoc V. Le [1]

Abstract

Convolutional Neural Networks (ConvNets) are commonly developed at a fixed resource budget, and then scaled up for better accuracy if more resources are available. In this paper, we systematically study model scaling and identify that carefully balancing network depth, width, and resolution can lead to better performance. Based on this observation, we propose a new scaling method that uniformly scales all dimensions of depth/width/resolution using a simple yet highly effective *compound coefficient*. We demonstrate the effectiveness of this method on scaling up MobileNets and ResNet.

To go even further, we use neural architecture

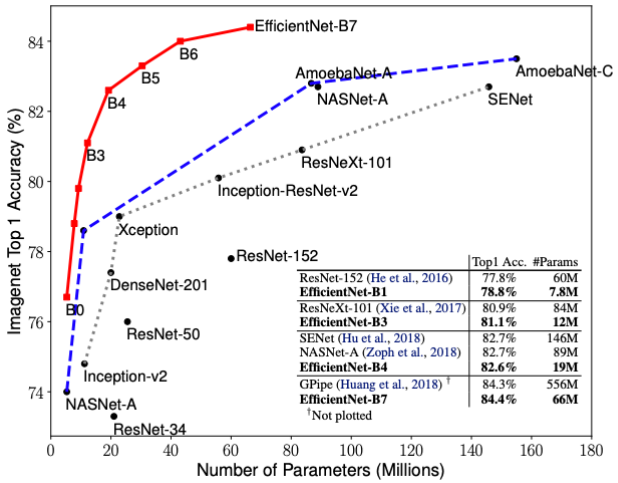
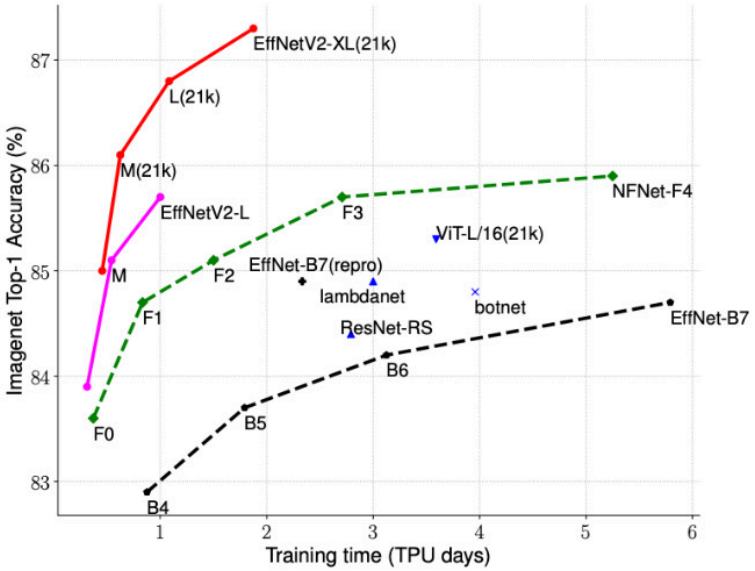


Figure 1. Model Size vs. ImageNet Accuracy. All numbers are



(a) Training efficiency.

	EfficientNet (2019)	ResNet-RS (2021)	DeiT/ViT (2021)	EfficientNetV2 (ours)
Top-1 Acc.	84.3%	84.0%	83.1%	83.9%
Parameters	43M	164M	86M	24M

(b) Parameter efficiency.

http://proceedings.mlr.press/v97/tan19a/tan19a.pdf

https://twitter.com/TensorFlow/status/1423359562724175873/photo/1

Thank you & any questions