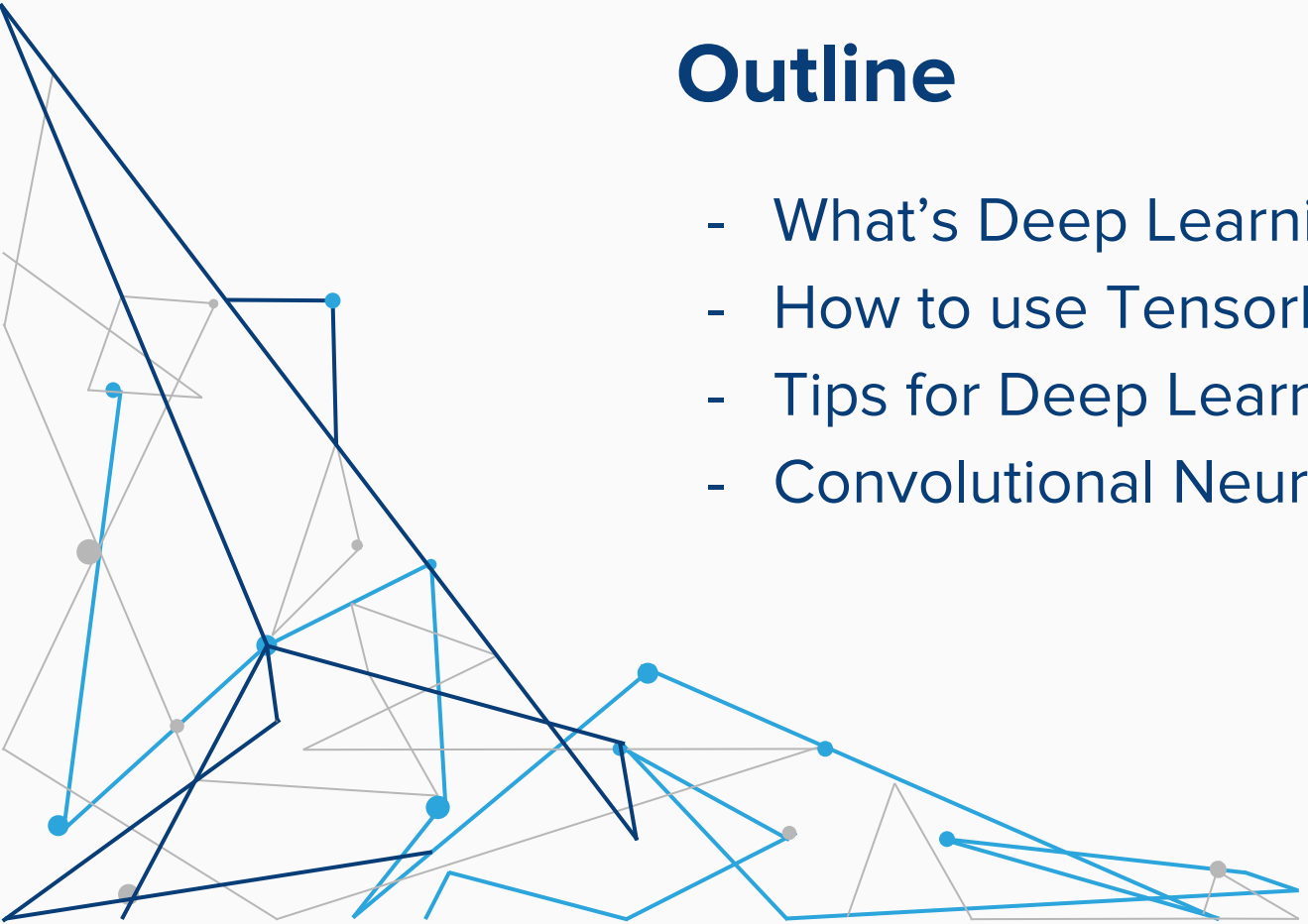
Introduction to Deep Learning Part I

Jeff Liu and Ching Lee
2018/10/16

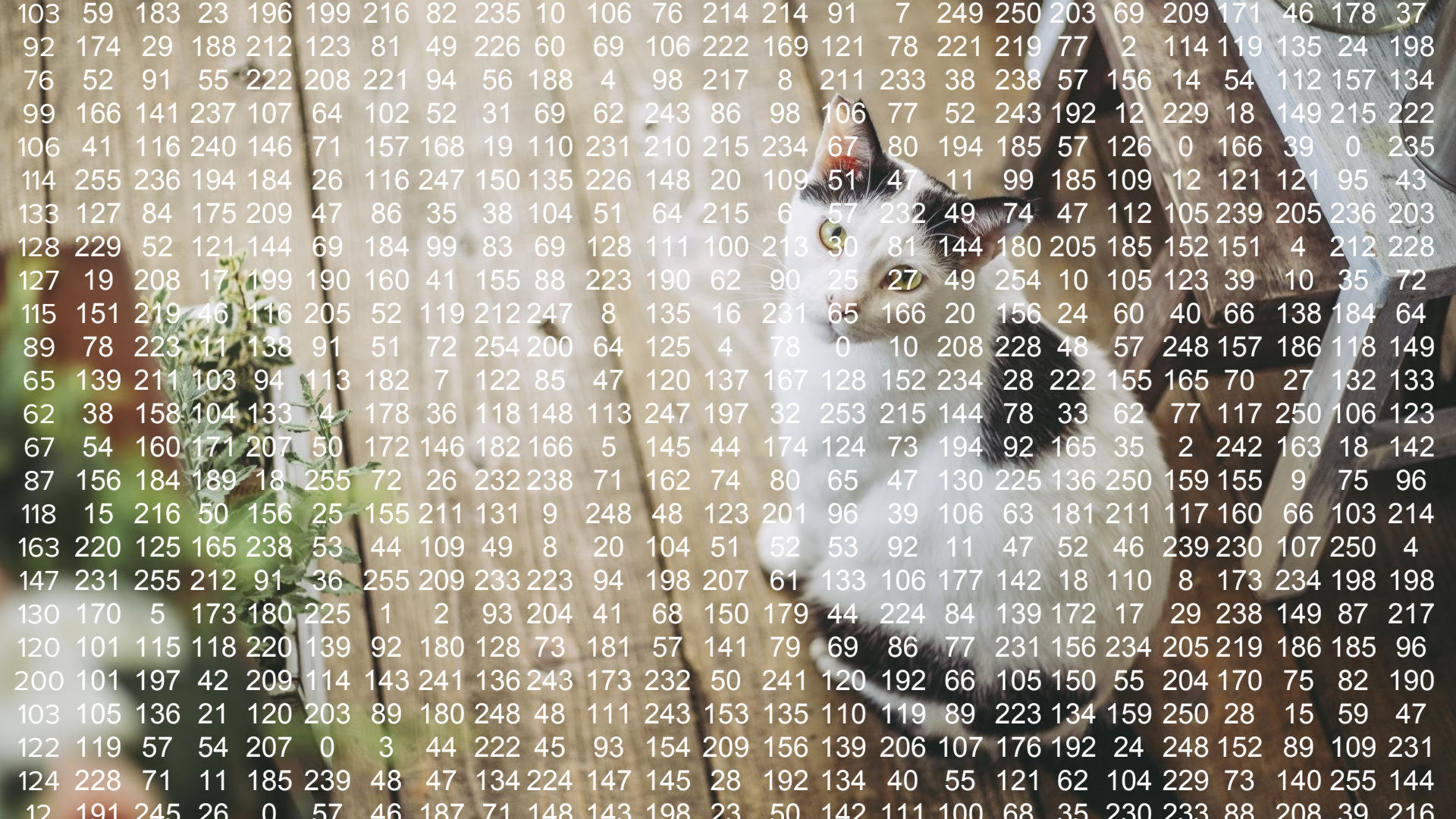
CloudMile

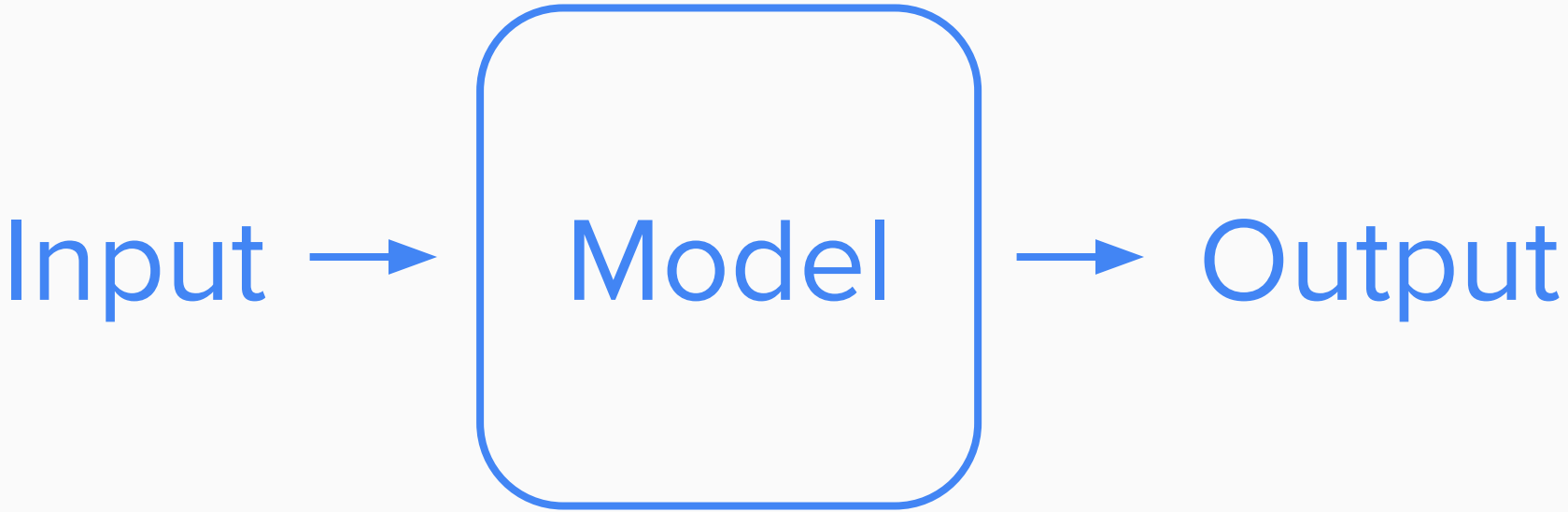


Outline

- What's Deep Learning?
- How to use TensorFlow and Keras
- Tips for Deep Learning
- Convolutional Neural Networks

CloudMile



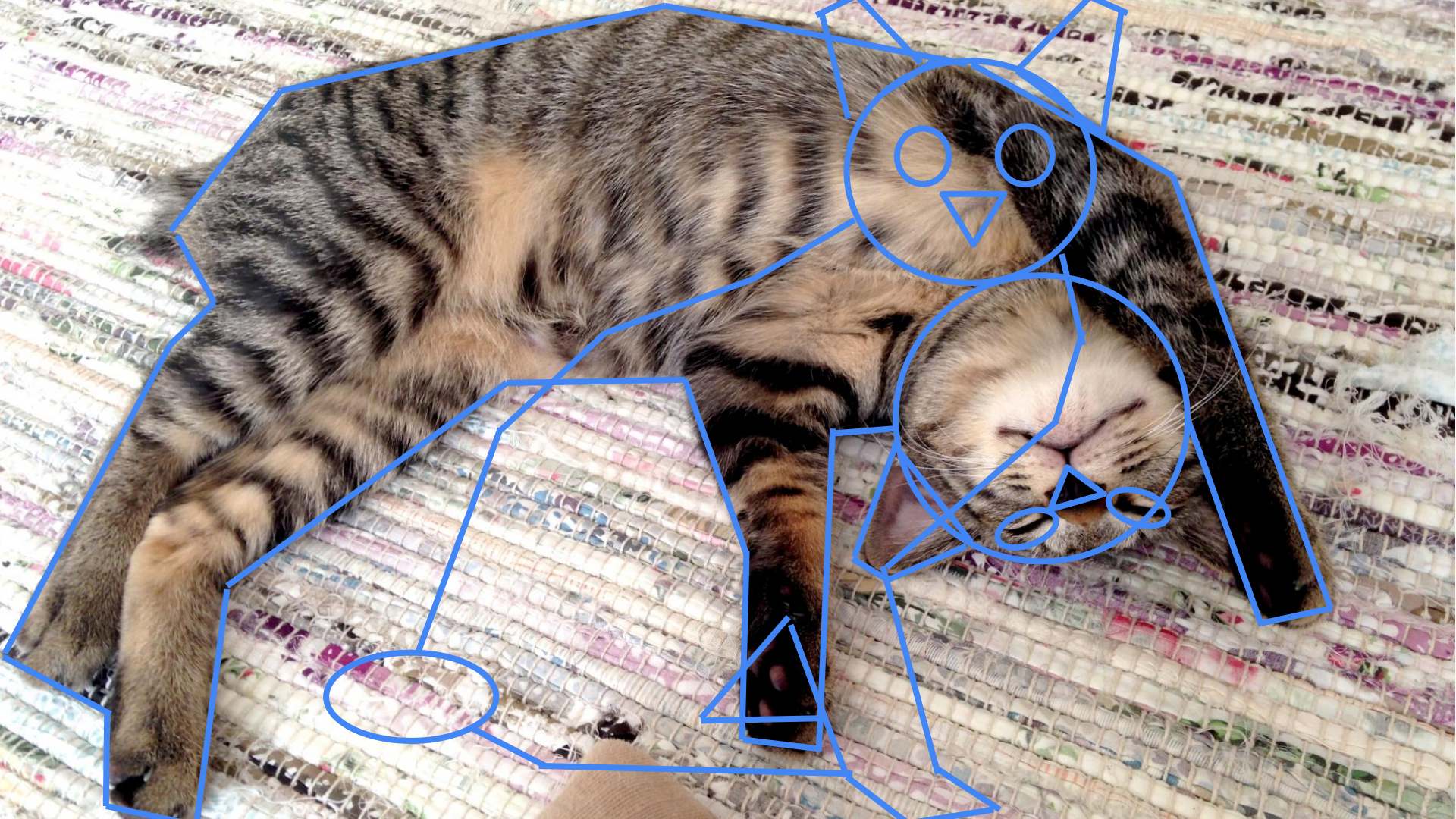


Model

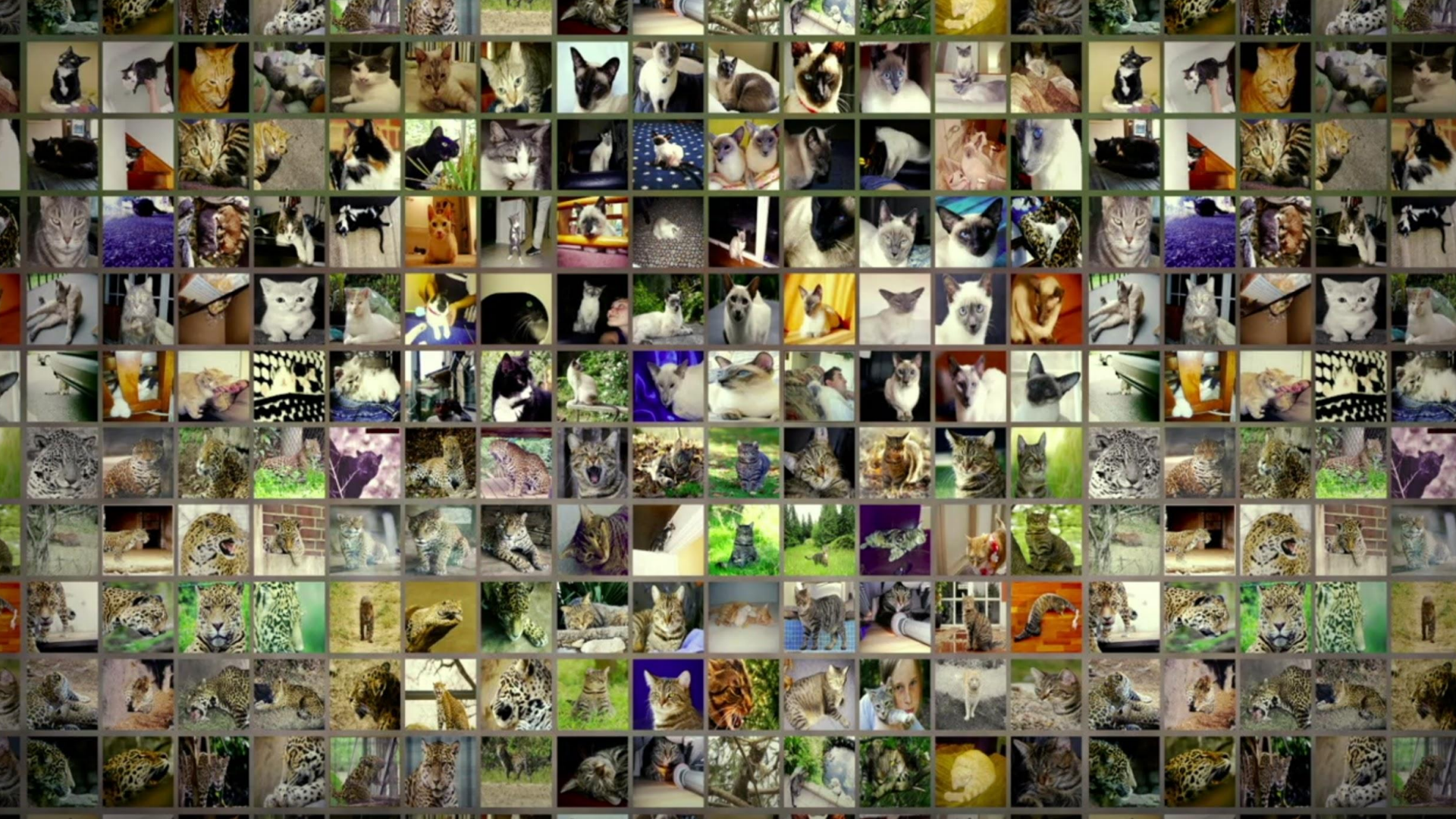
“Cat”

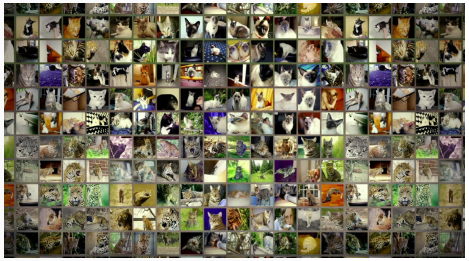
“Cat”

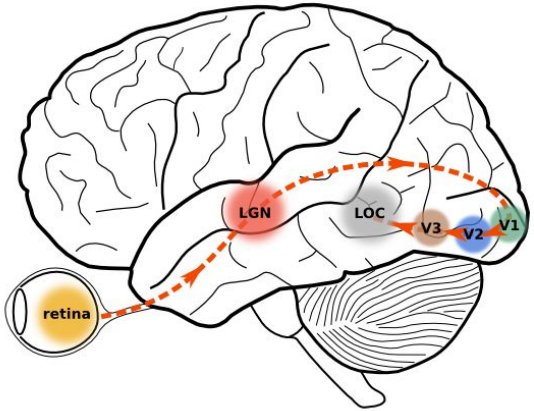
CloudMile

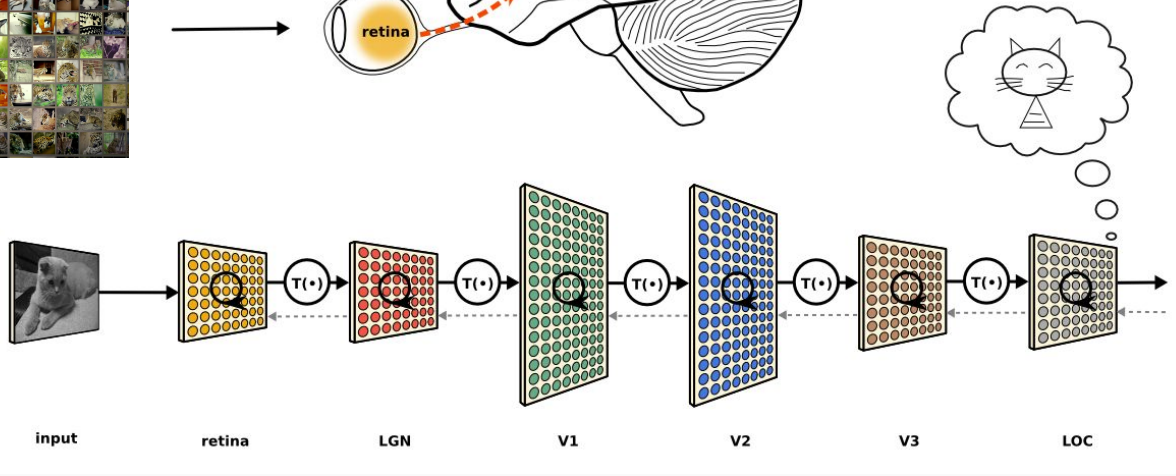



TAVUZ NAKLİYAT
EVDEN EVE
BÜRODAN BÜROYA
TAŞIMACILIK









(Deep) Neural Networks

Artificial Intelligence

目標

Machine Learning

手段

Deep Learning

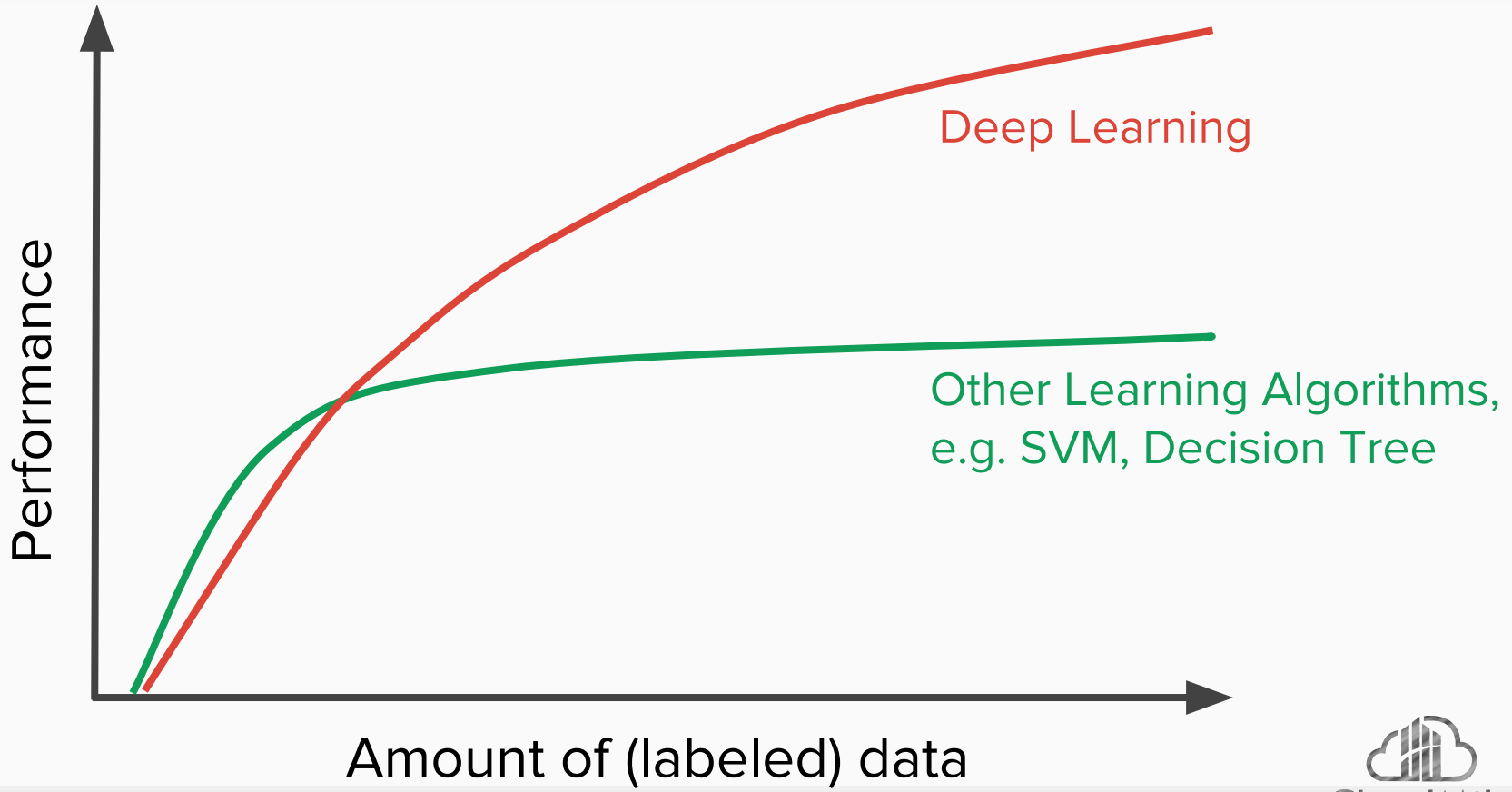
1950

1980

2010

CloudMile

Why Deep Learning?



CloudMile

The Basis of Artificial Intelligence

Data

Algorithm

Hardware

CloudMile

Structured Data

- Attributes (e.g. age, sex, income ...) as inputs
- Continuous (e.g. income) / Categorical (e.g. gender)

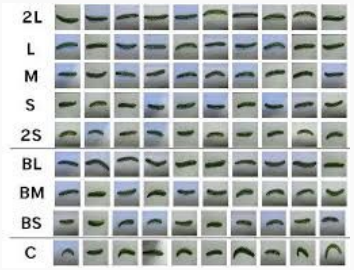
1 age		2 workclass		3 education		4 education–num		5 marital–status	
Min	17.	Private	22 696	HS-grad	10 501	Min	1.	Married-civ-spouse	14 976
1st Qu	28.	Self-emp-not-inc	2541	Some-college	7291	1st Qu	9.	Never-married	10 683
Median	37.	Local-gov	2093	Bachelors	5355	Median	10.	Divorced	4443
Mean	38.5816	?	1836	Masters	1723	Mean	10.0807	Separated	1025
3rd Qu	48.	State-gov	1298	Assoc-voc	1382	3rd Qu	12.	Widowed	993
Max	90.	Self-emp-inc	1116	11th	1175	Max	16.	Married-spouse-absent	418
		(Other)	981	(Other)	5134			Married-AF-spouse	23

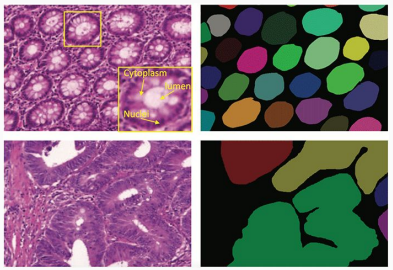
6 occupation		7 relationship		8 race		9 sex		10 capital–gain	
Prof-specialty	4140	Husband	13 193	White	27 816	Male	21 790	1st Qu	0.
Craft-repair	4099	Not-in-family	8305	Black	3124	Female	10 771	3rd Qu	0.
Exec-managerial	4066	Own-child	5068	Asian-Pac-Islander	1039			Median	0.
Adm-clerical	3770	Unmarried	3446	Amer-Indian-Eskimo	311			Min	0.
Sales	3650	Wife	1568	Other	271			Mean	1077.65
Other-service	3295	Other-relative	981					Max	99 999.
(Other)	9541								

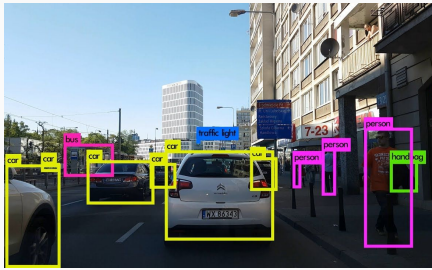
11 capital–loss		12 hours–per–week		13 native–country		14 income	
1st Qu	0.	Min	1.	United-States	29 170	<=50K	24 720
3rd Qu	0.	1st Qu	40.	Mexico	643	>50K	7841
Median	0.	Median	40.	?	583		
Min	0.	Mean	40.4375	Philippines	198		
Mean	87.3038	3rd Qu	45.	Germany	137		
Max	4356.	Max	99.	Canada	121		
				(Other)	1709		

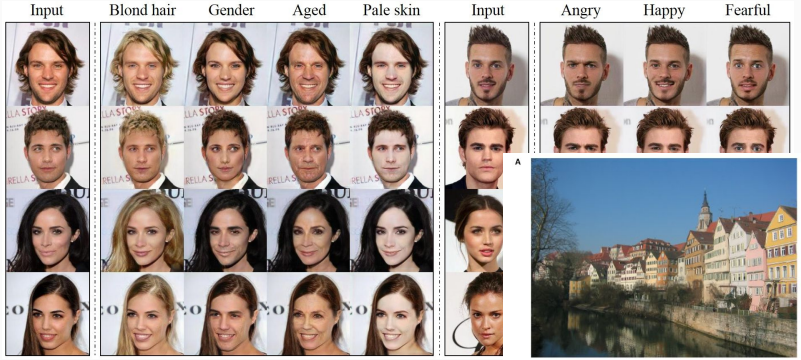
Images

- Classification
- Segmentation
- Object Detection
- Generative Models
- Art









Sequential Data

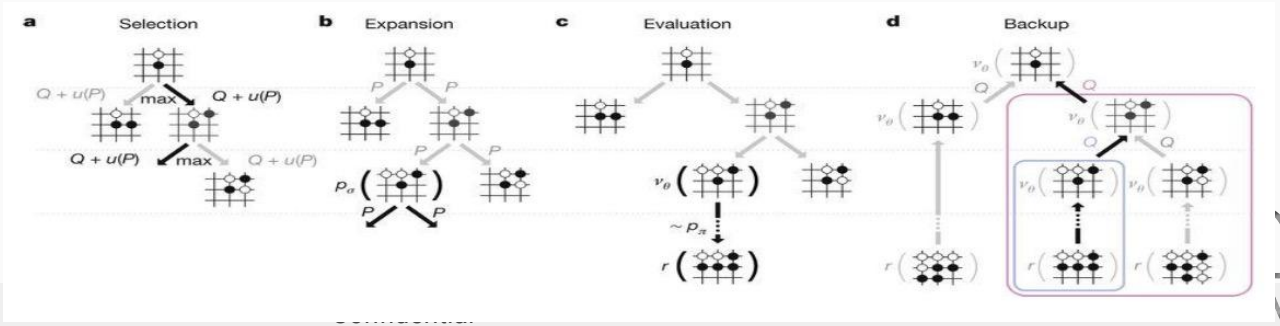
- Text

“CloudMile 成立於 2016 年，致力於 B2B 雲端與人工智慧應用。為客戶建立國際級雲端架構，並以機器學習及大數據分析技術為核心，協助企業進行商業預測與 產業升級。”

- Speech, Audio

- Videos

- Sequential Decision Making (RL)



Learning ≈ Looking for a Function

- Image Recognition

$f($  $) =$ “cat”

- Speech Recognition

$f($  $) =$ “你太skr了～”

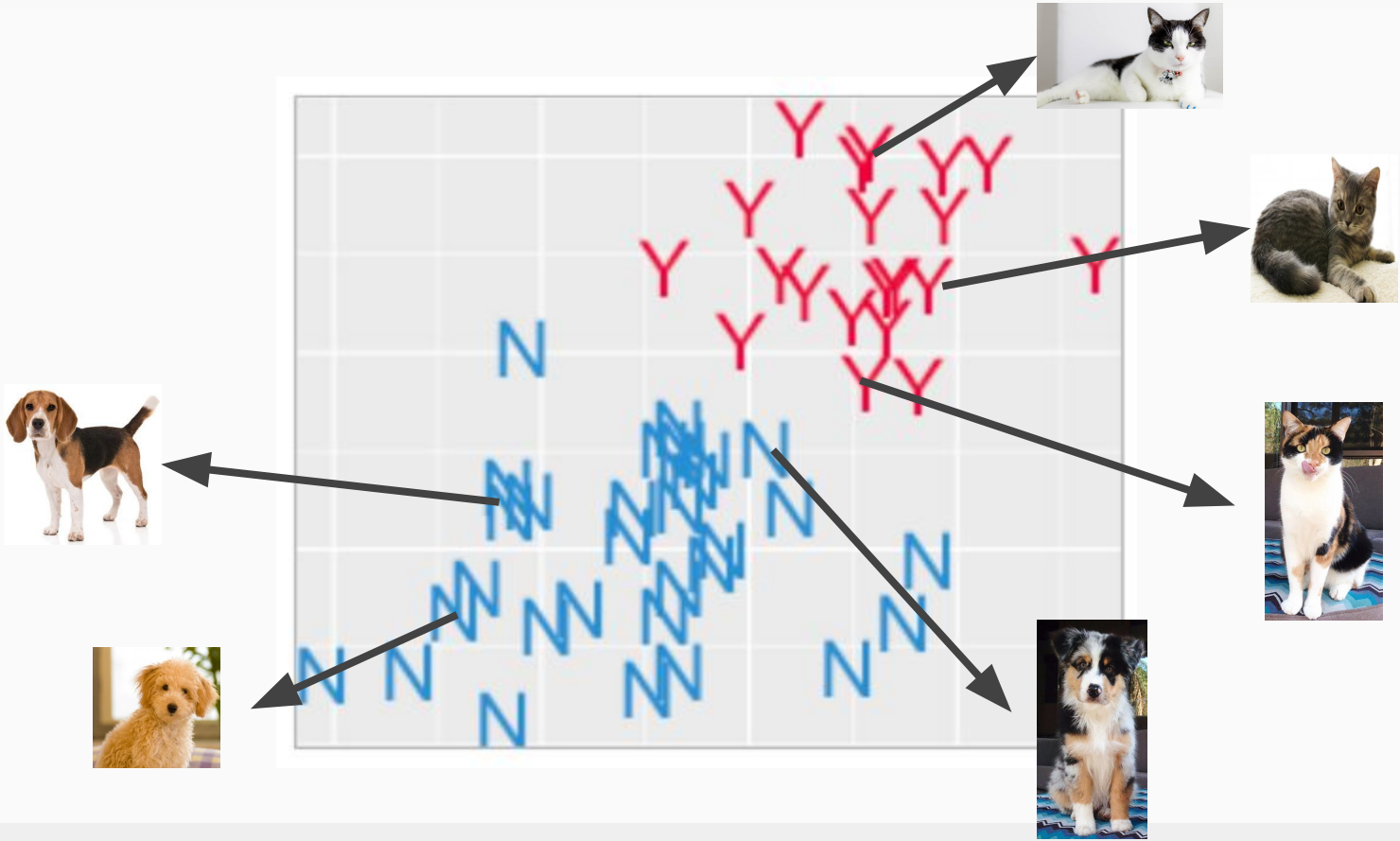
- Machine Translation

$f($ “母湯” $) =$ “不可以”

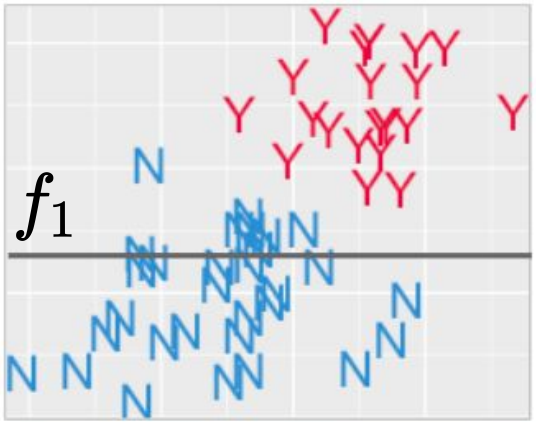
- Playing Go

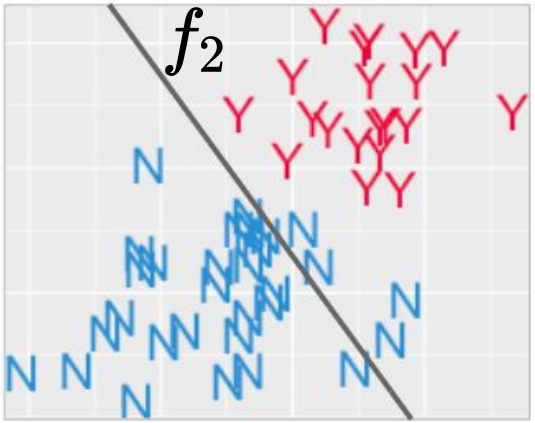
$f($  $) =$ “3-3” (next move)

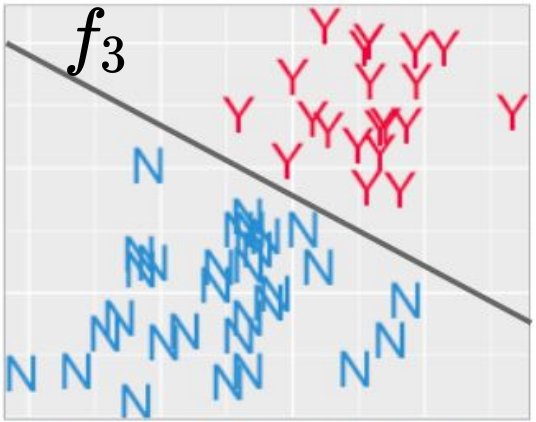
Cat/Dog Classification



Solutions may be something like ...

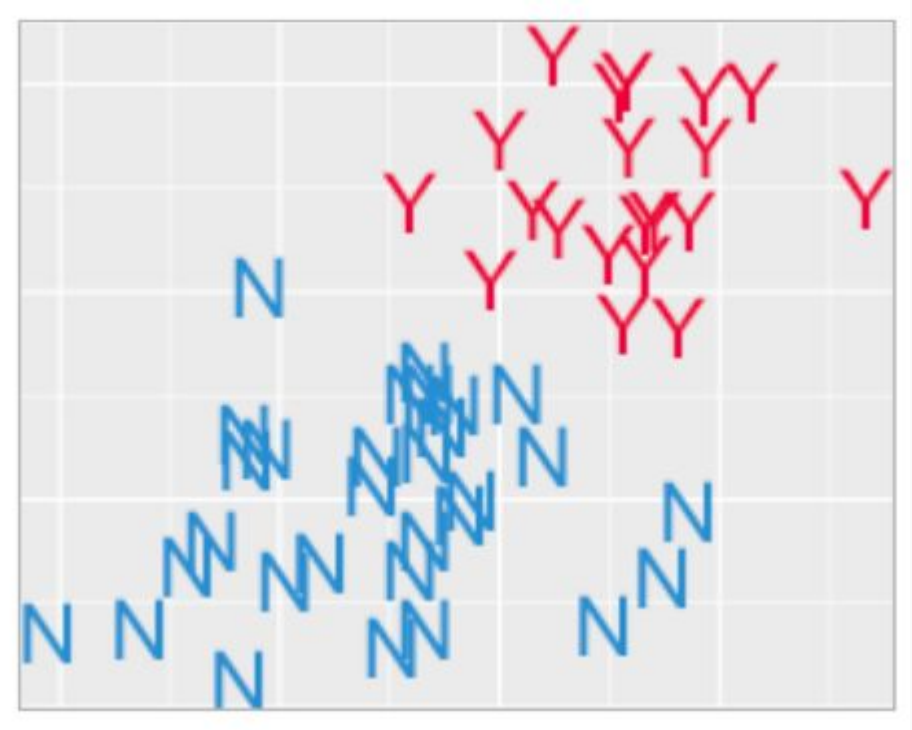






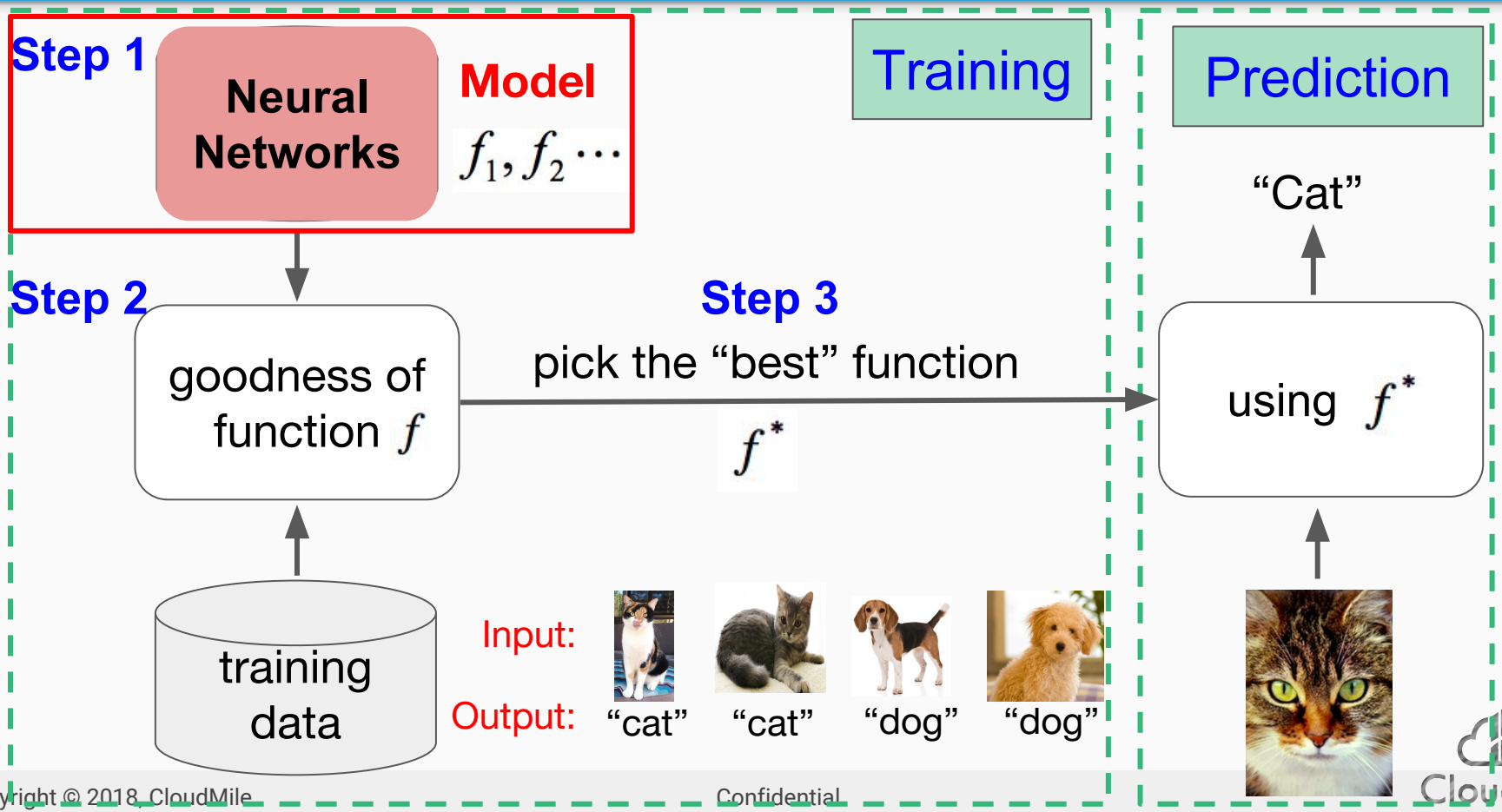
CloudMile

Select a Model

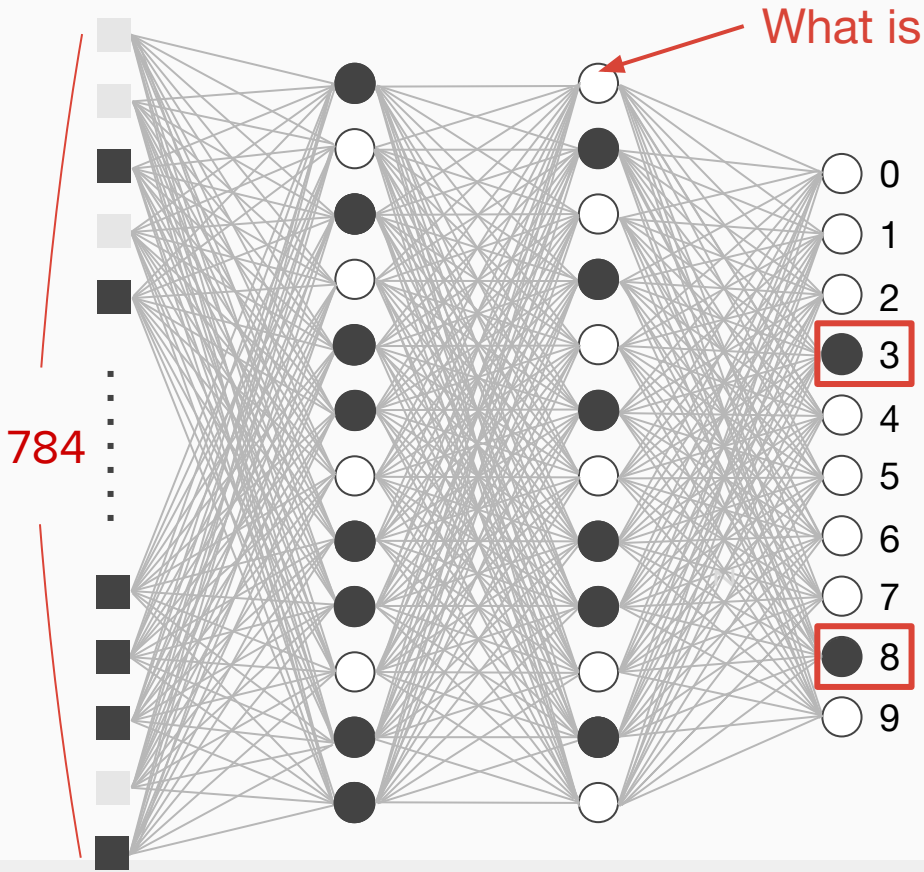


CloudMile

Machine Learning Framework

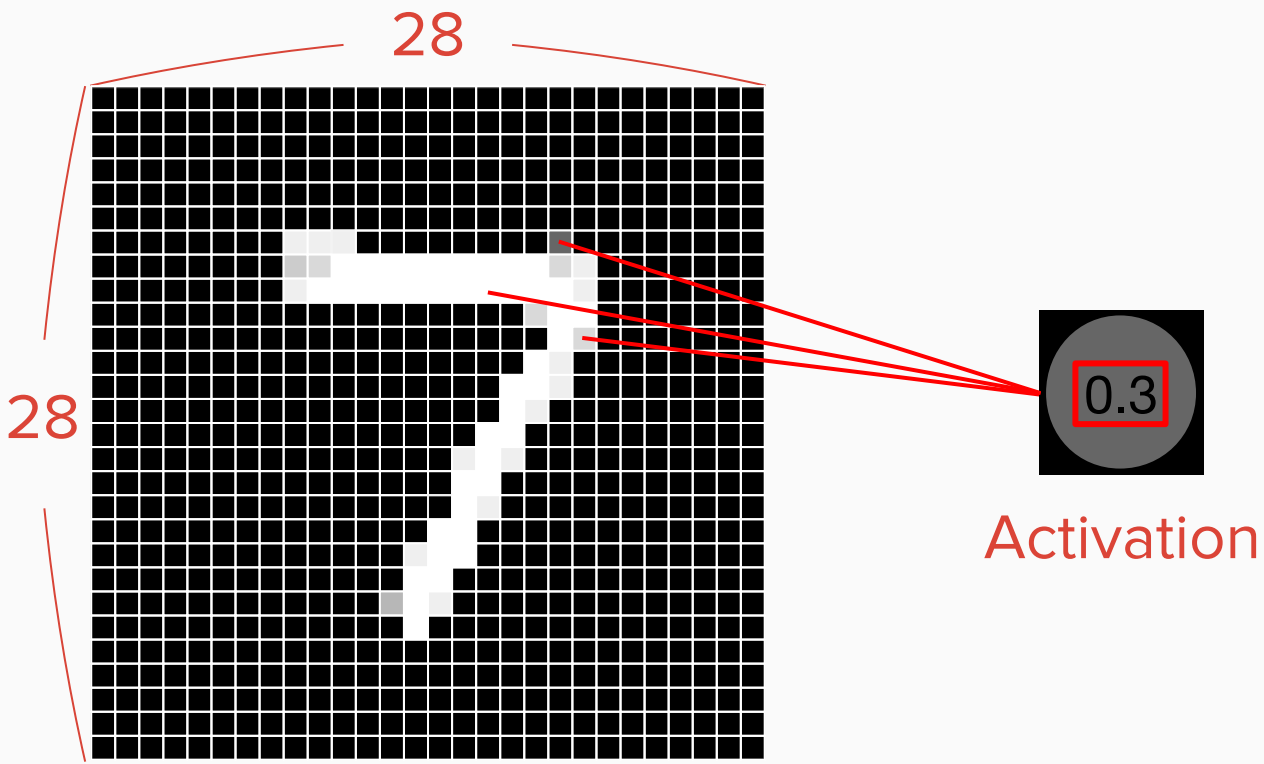


Neural Networks for MNIST

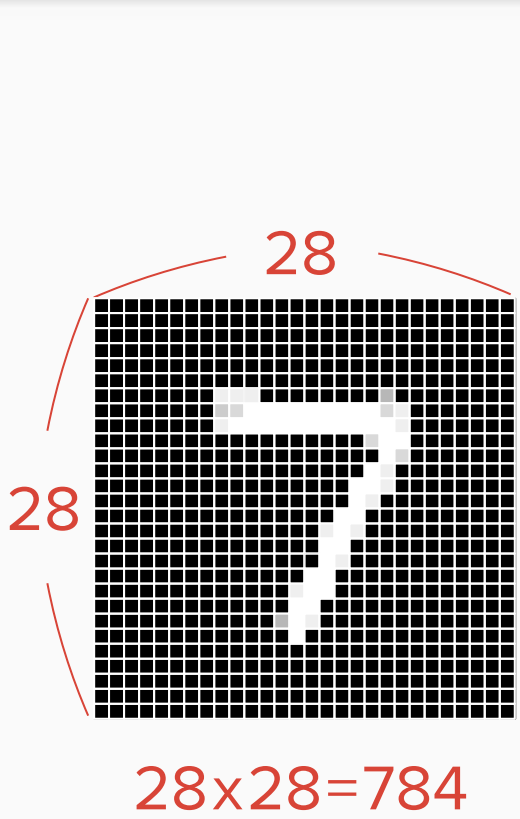


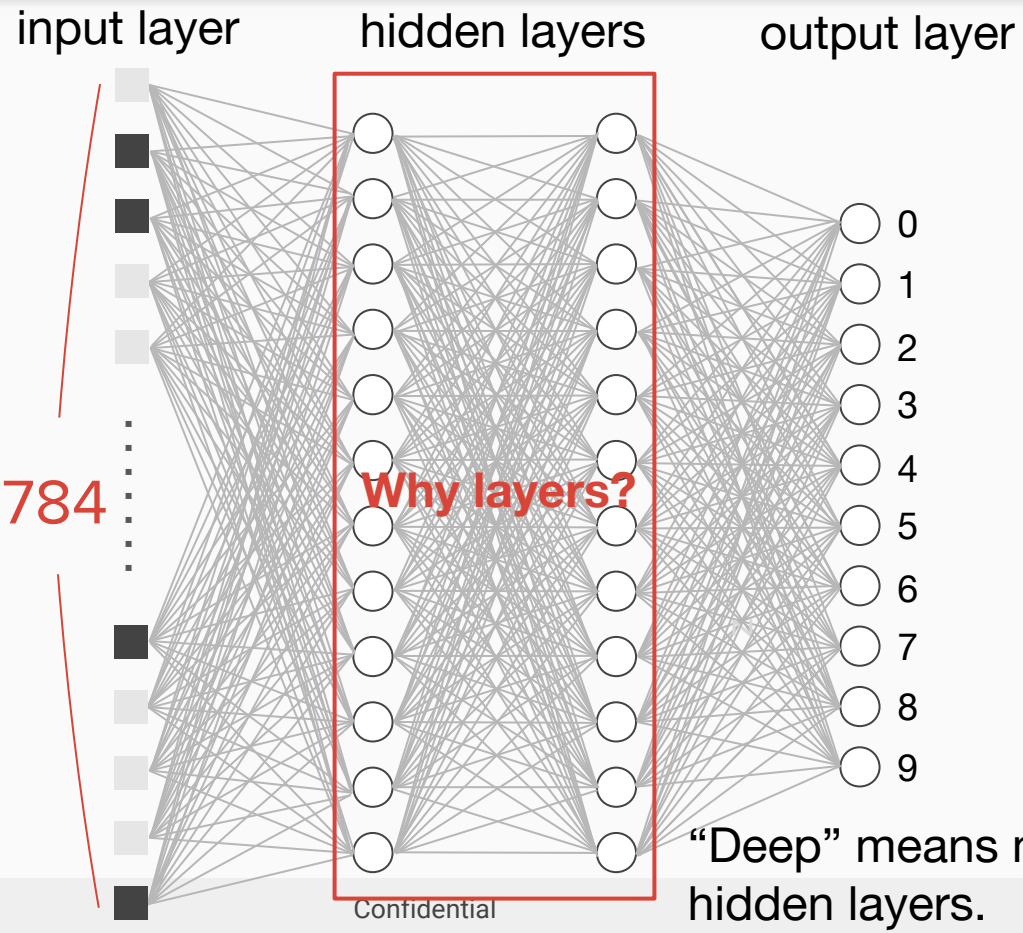
Neuron

Activation



Deep Neural Networks (Deep Learning)

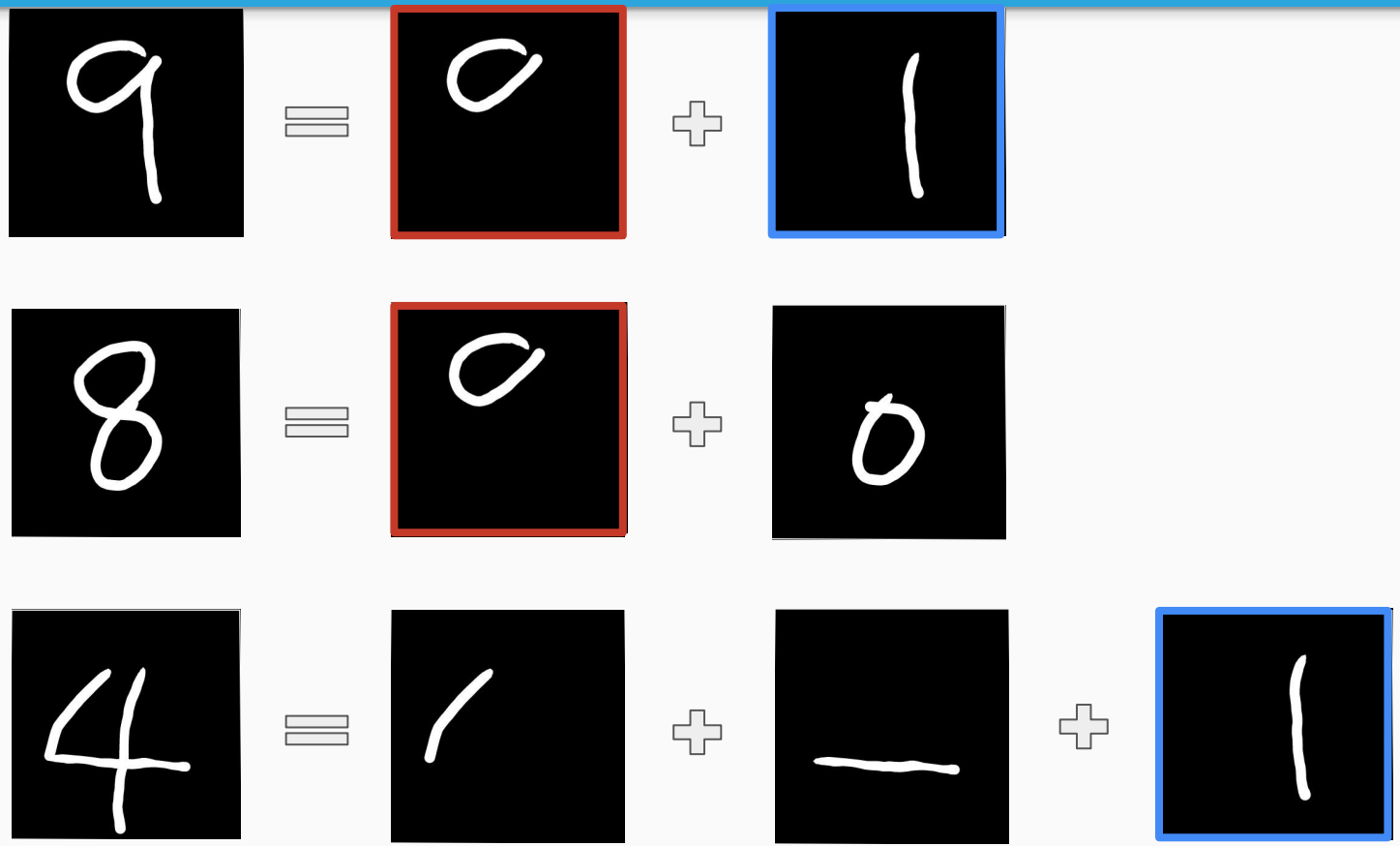




"Deep" means many hidden layers.

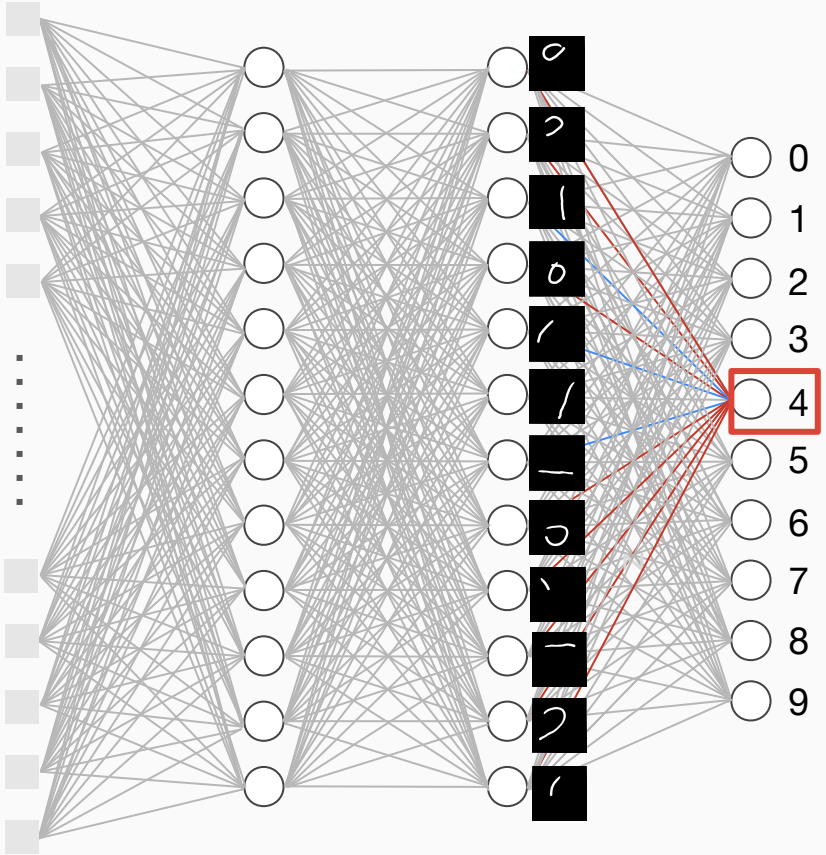
CloudMile

Meaningfulness of Deep Learning



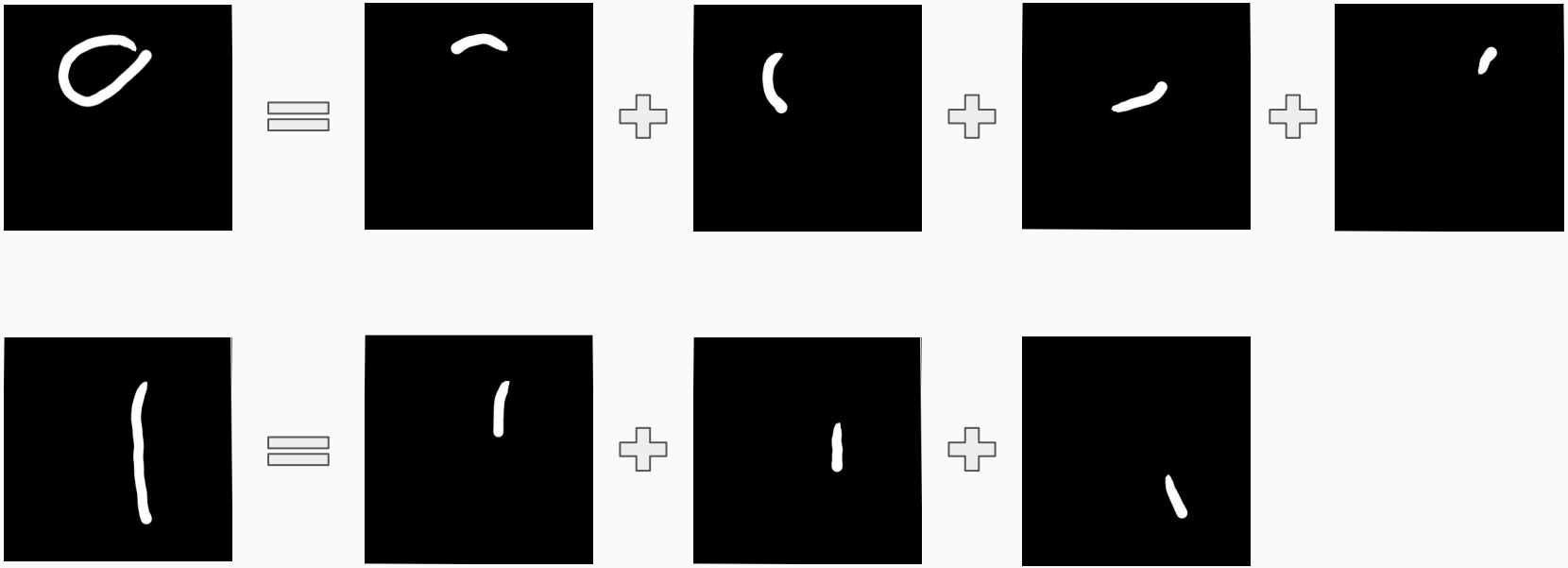
Meaningfulness of Deep Learning (Cont'd)

raw images



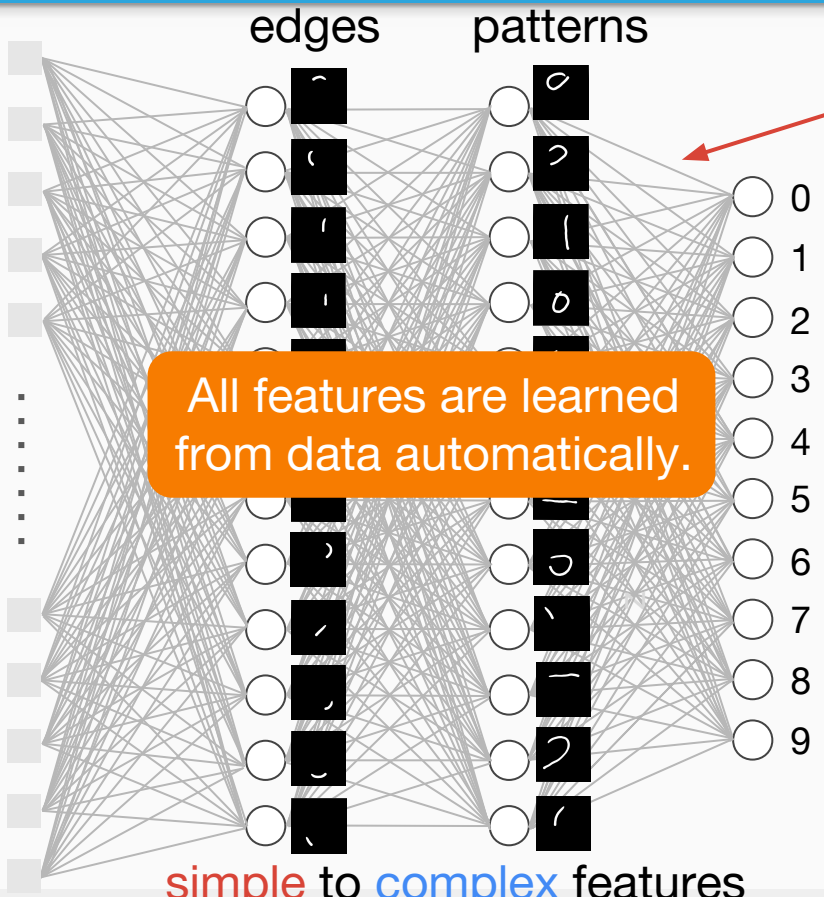
CloudMile

Meaningfulness of Deep Learning (Cont'd)



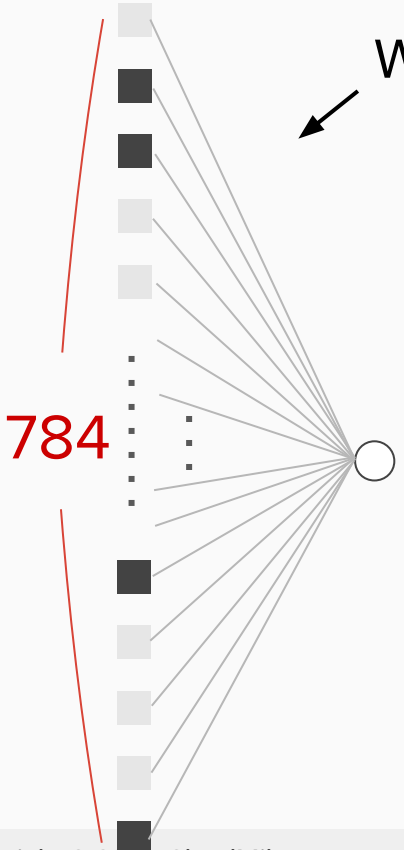
Meaningfulness of Deep Learning (Cont'd)

raw images

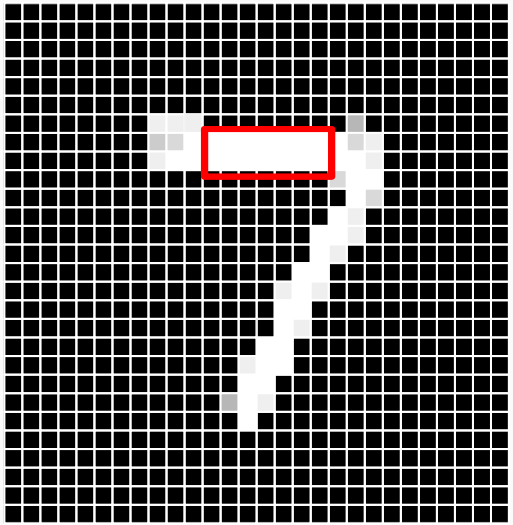


CloudMile

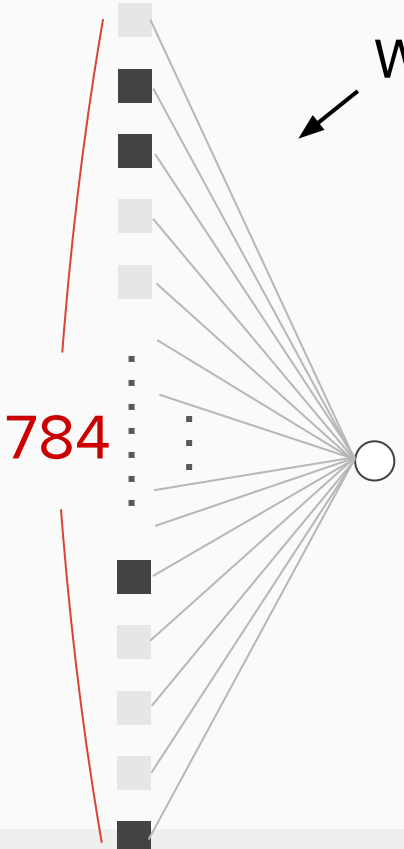
How to Extract an Edge?



$$w_1 : 1.91$$
$$w_2 : -1.68$$
$$w_3 : 0.78$$
$$\vdots$$
$$w_n : 0.23$$

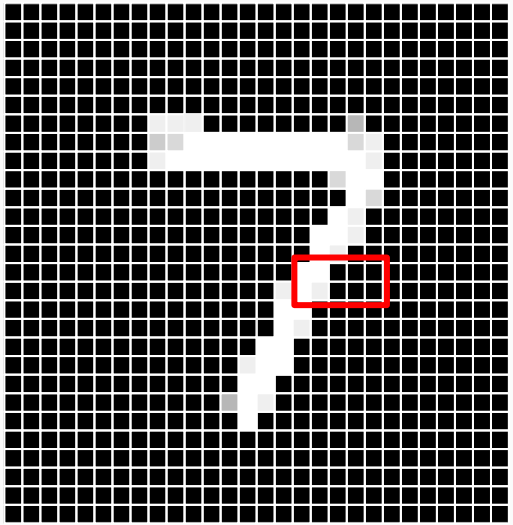


How to Extract an Edge?

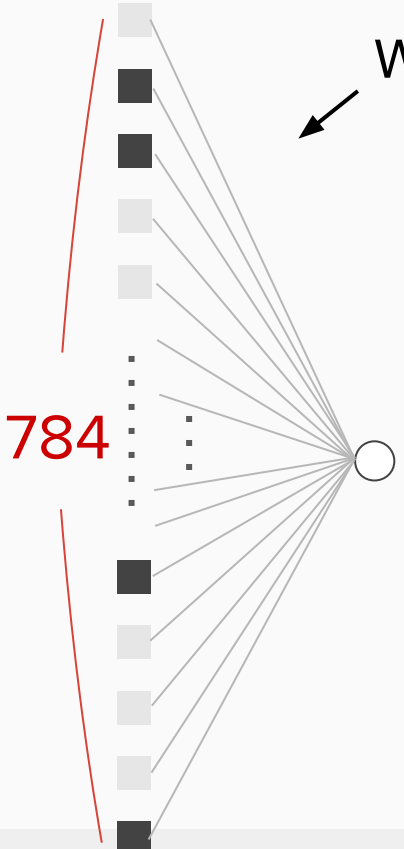


What parameters?

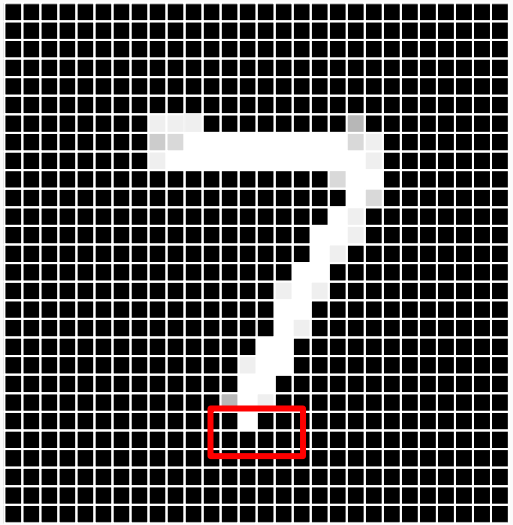
$$w_1 : 0.92$$
$$w_2 : -0.53$$
$$w_3 : -2.54$$
$$\vdots$$
$$w_n : -1.05$$



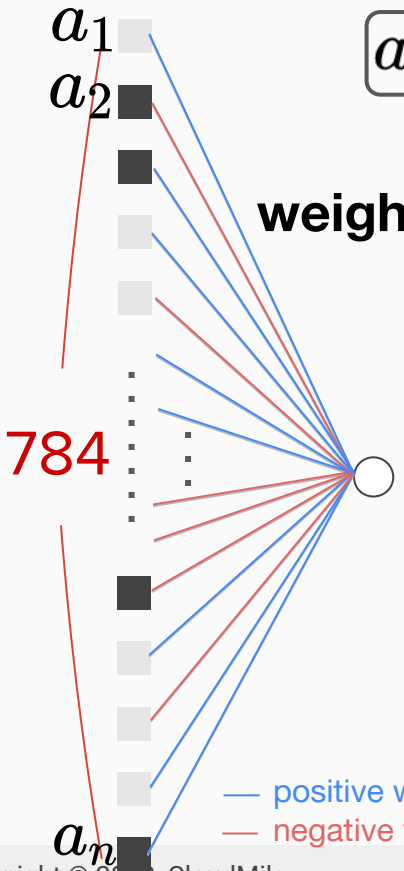
How to Extract an Edge?



$$w_1 : -0.98$$
$$w_2 : -0.87$$
$$w_3 : 1.23$$
$$\vdots$$
$$w_n : 3.14$$



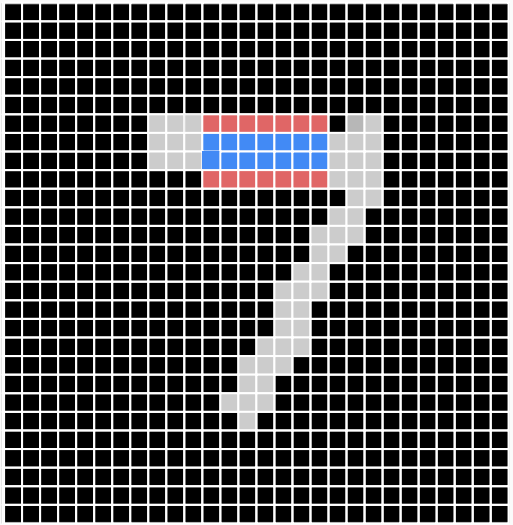
Weighted Sum



$$a_1 w_1 + a_2 w_2 + a_3 w_3 + \cdots + a_n w_n$$

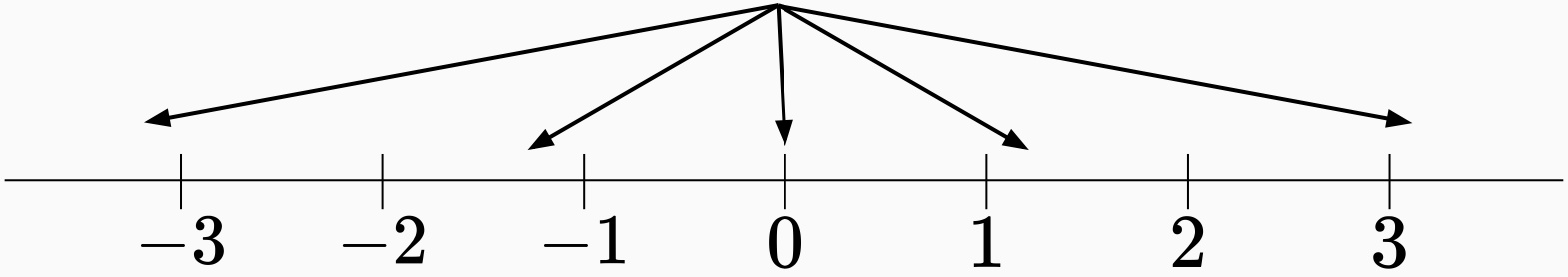
weights

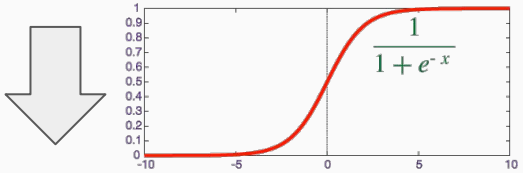
$$w_1 : 1.91$$
$$w_2 : -1.68$$
$$w_3 : 0.78$$
$$\vdots$$
$$w_n : 0.23$$



Activation Function

$$a_1 w_1 + a_2 w_2 + a_3 w_3 + \cdots + a_n w_n$$



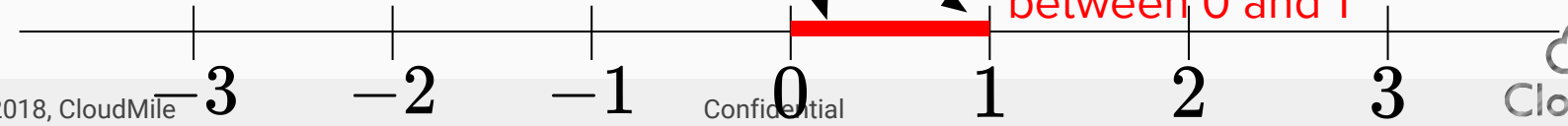


introduce non-linear complexities

sigmoid

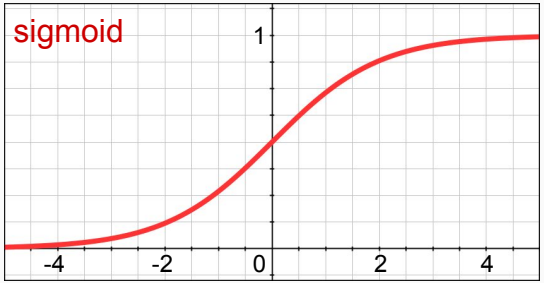
$$\sigma(a_1 w_1 + a_2 w_2 + a_3 w_3 + \cdots + a_n w_n)$$

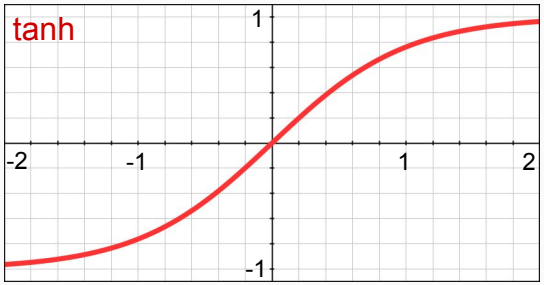
Activations should be between 0 and 1

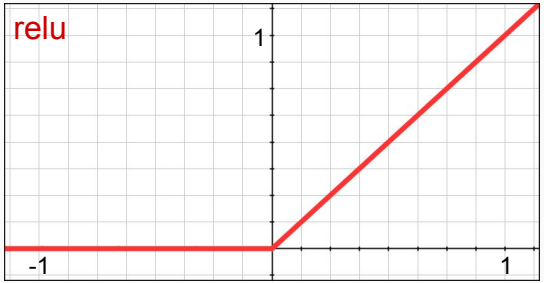


CloudMile

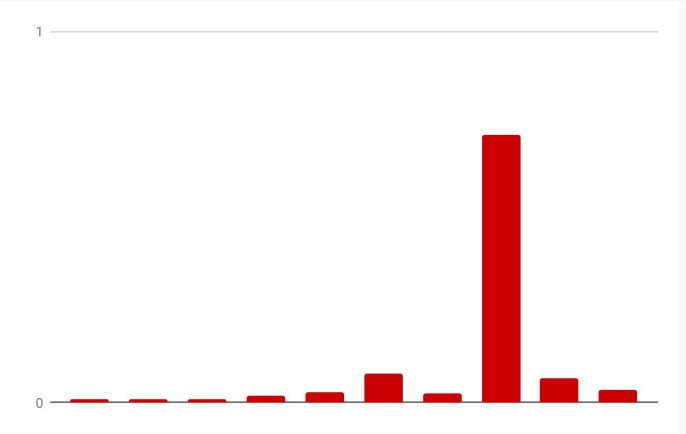
Other Activation Functions

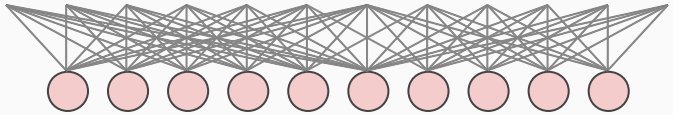






frequently used





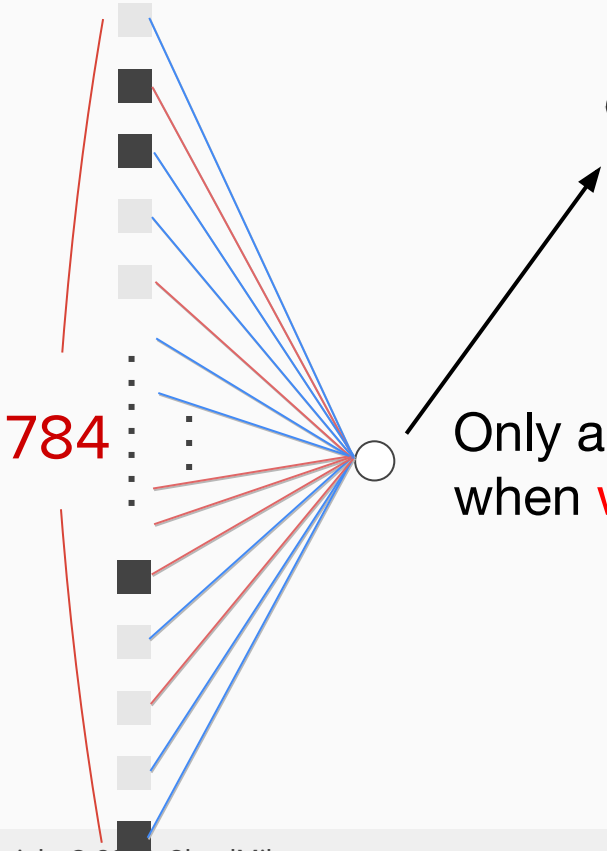
$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}$$

weighted sum+b

classification output

CloudMile

Bias for Inactivity

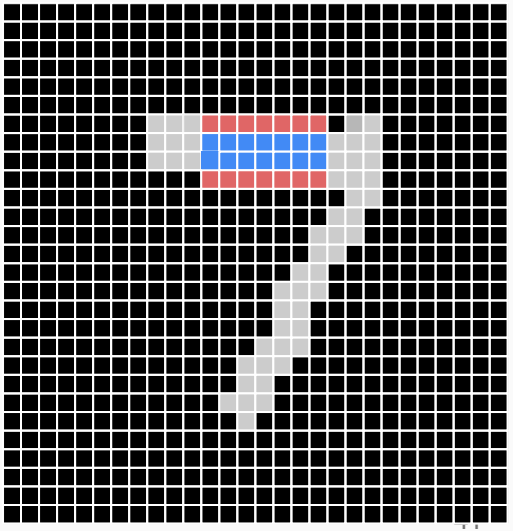


How positive is it?

$$\sigma(a_1 w_1 + a_2 w_2 + a_3 w_3 + \cdots + a_n w_n - 8)$$

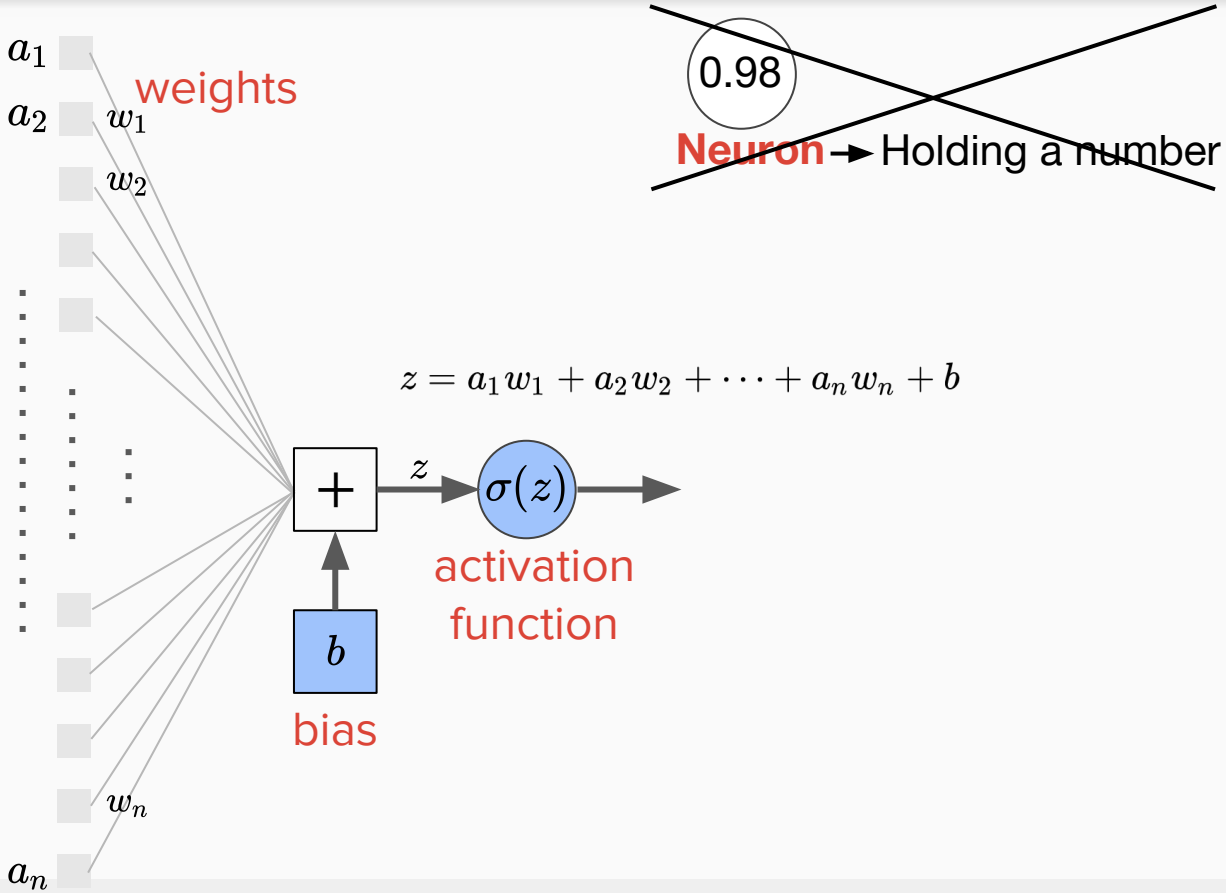
bias

Only activate meaningfully when weighted sum > 8

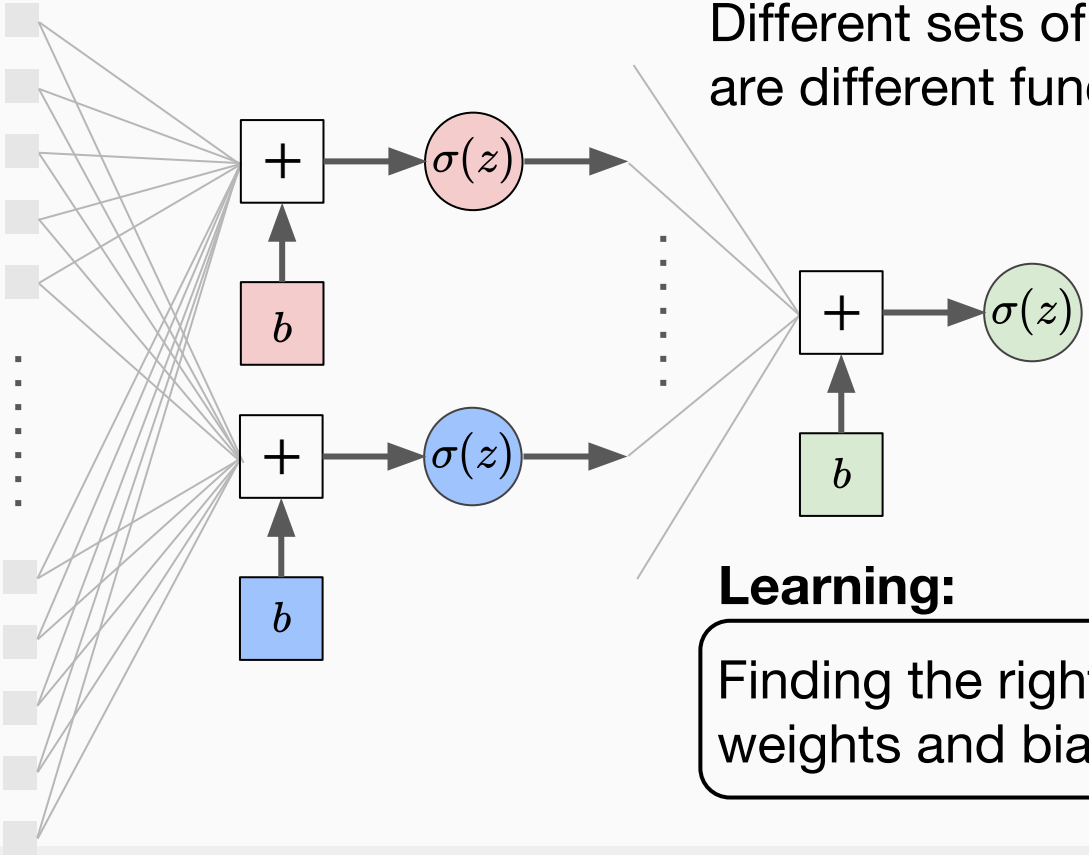


CloudMile

Neuron is a Simple Function



Neural Network is a Function



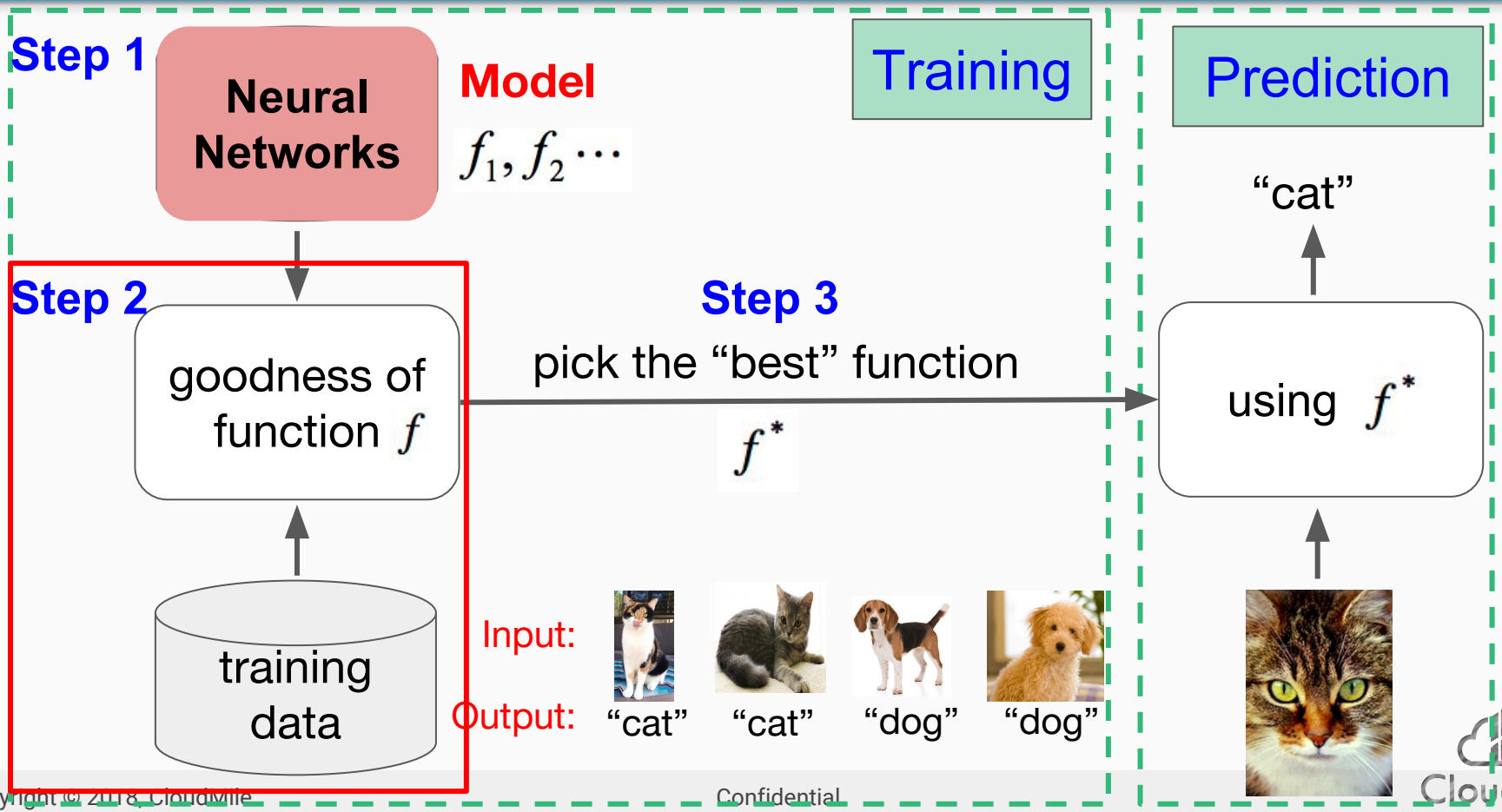
Different sets of parameters are different functions.

Learning:

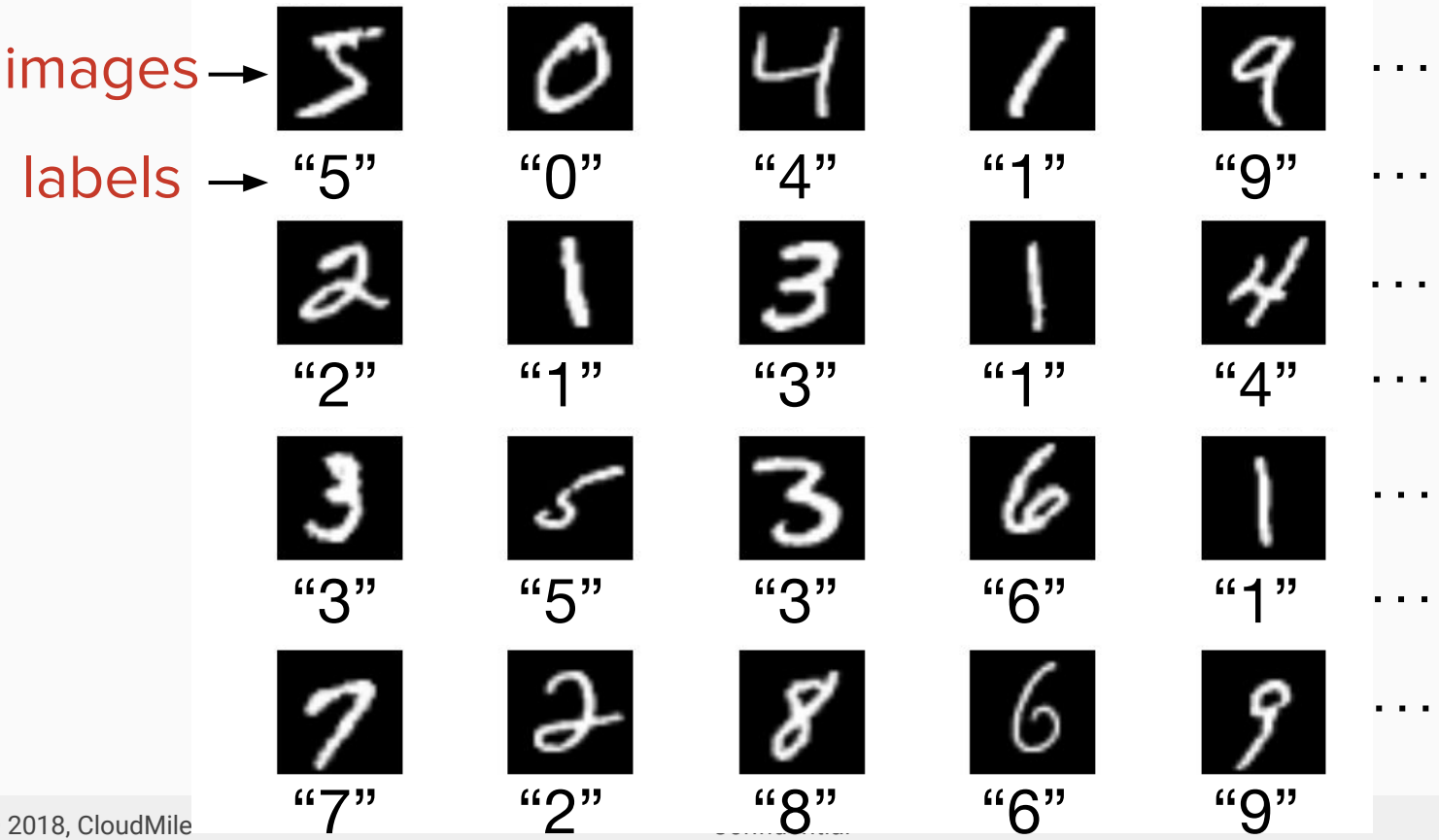
Finding the right weights and biases.

CloudMile

Machine Learning Framework

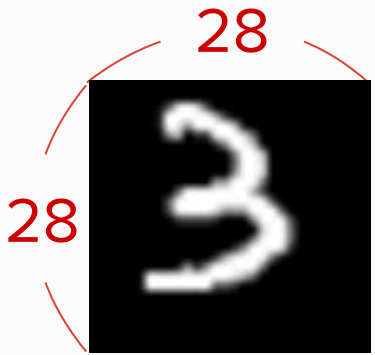


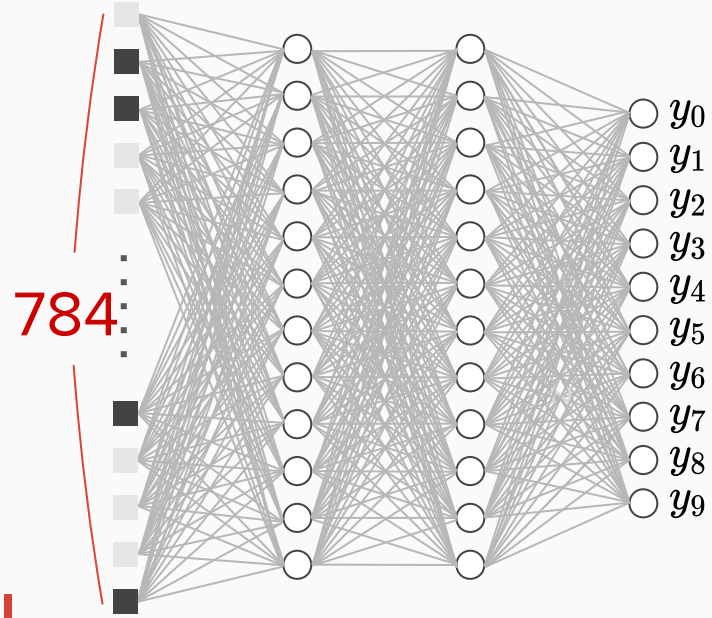
Training Data



CloudMile

Goal of Learning





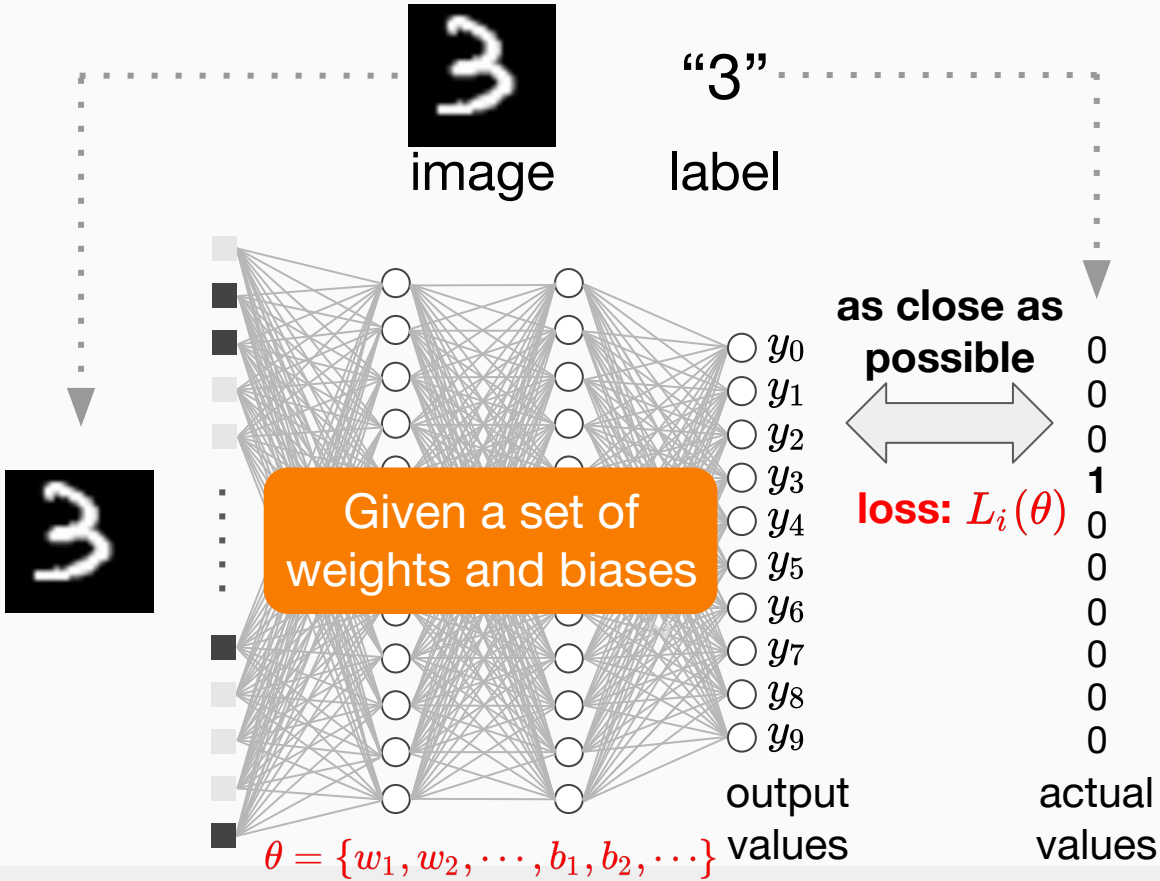
Goal

Input  : y_3 has the maximum value

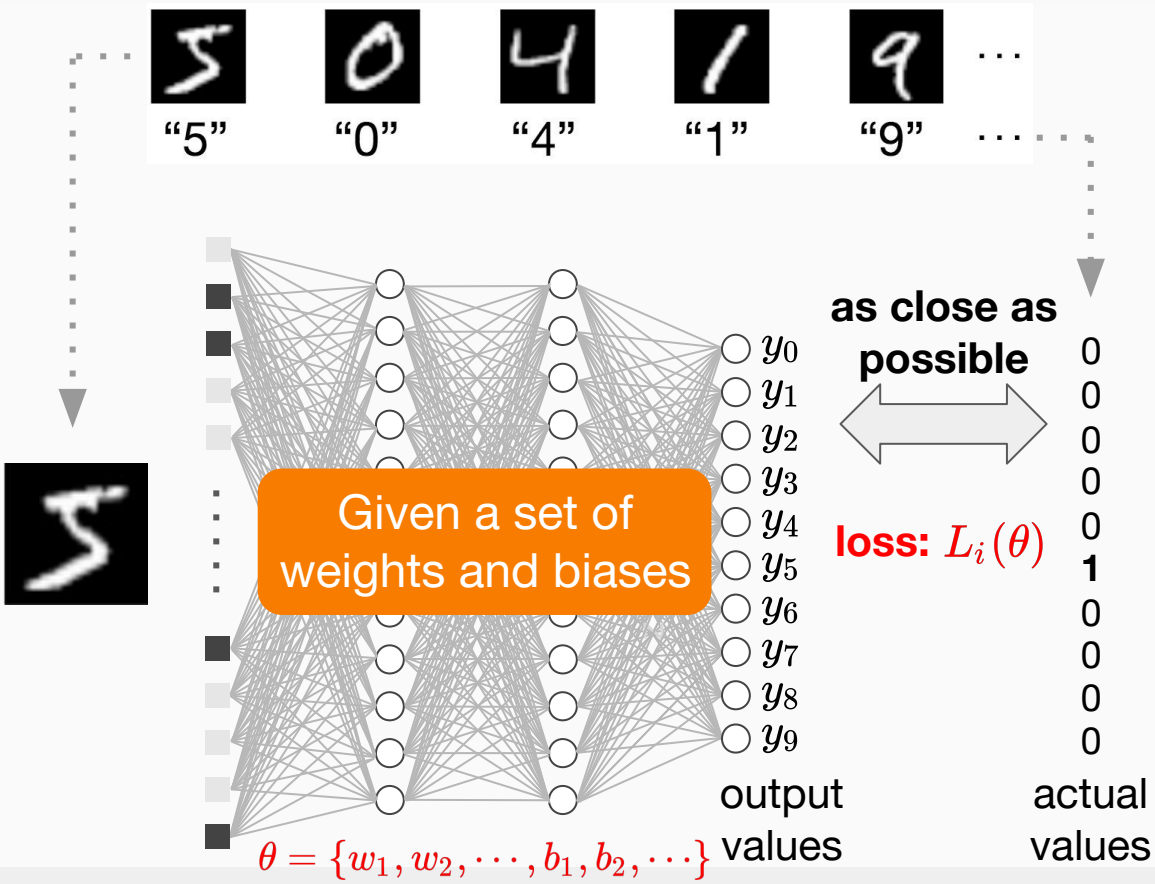
Input  : y_8 has the maximum value

CloudMile

Loss



Average Loss of all training data ...



as small as possible

average loss:

$$\frac{1}{N}\sum_{i=1}^{N} L_i(\theta)$$

Learning:

Finding the right weights and biases that minimize average loss

CloudMile

Some Loss Functions for (Binary) Classification

- Mean Squared Error (MSE): $-\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$

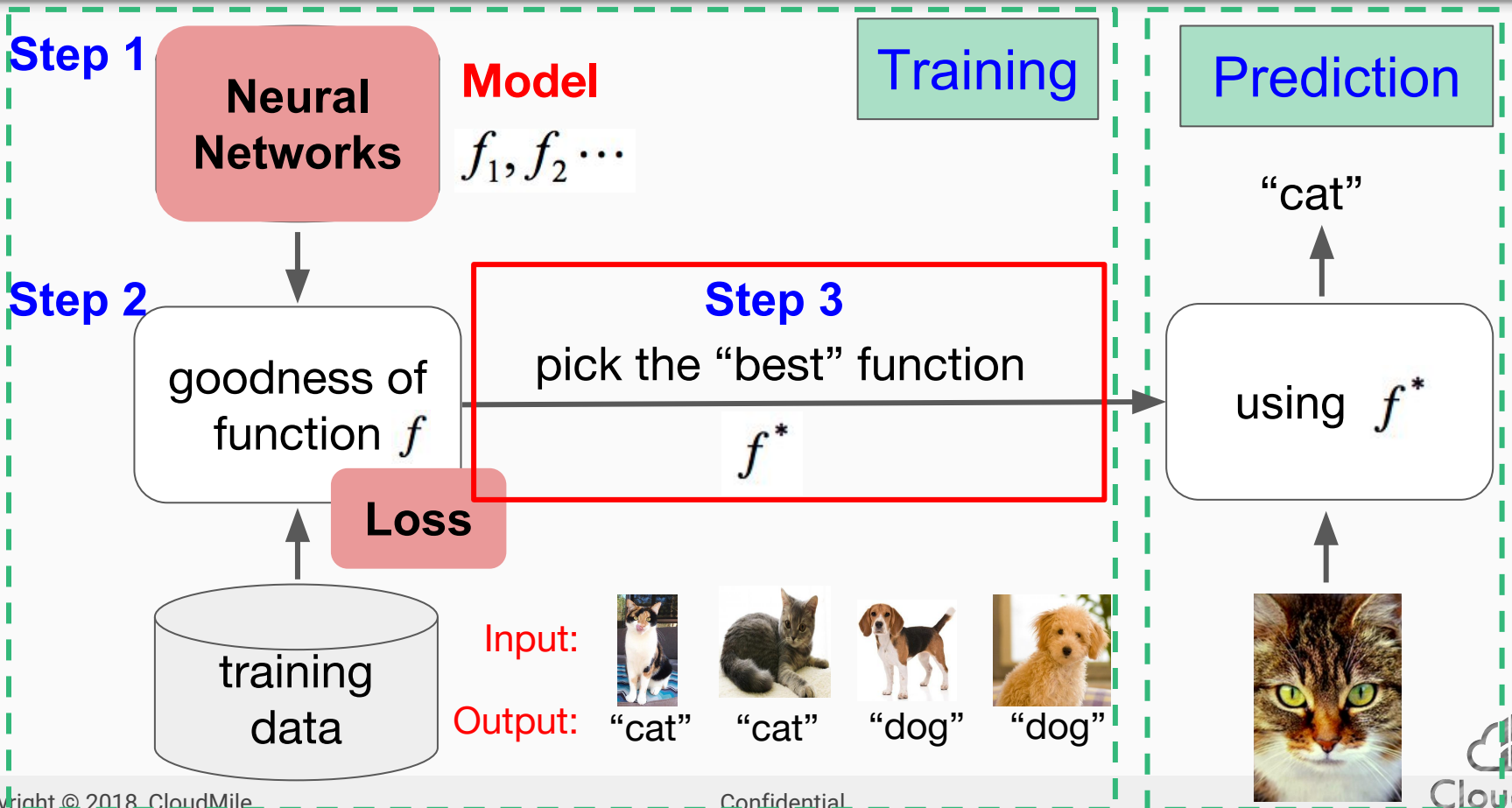
$$\frac{(0.7-1)^2+(0.1-0)^2+(0.2-0)^2}{3} = 0.0467$$

output value		actual value
(0.7	,	1)
(0.1	,	0)
(0.2	,	0)

- Cross Entropy: $-\frac{1}{N}\sum_{i=1}^{N}[\hat{y}_i \log(y_i) + (1-\hat{y}_i)\log(1-y_i)]$

$$-\frac{1\times\log(0.7)+(1-0)\times\log(1-0.1)+(1-0)\times\log(1-0.2)}{3} = 0.2284$$

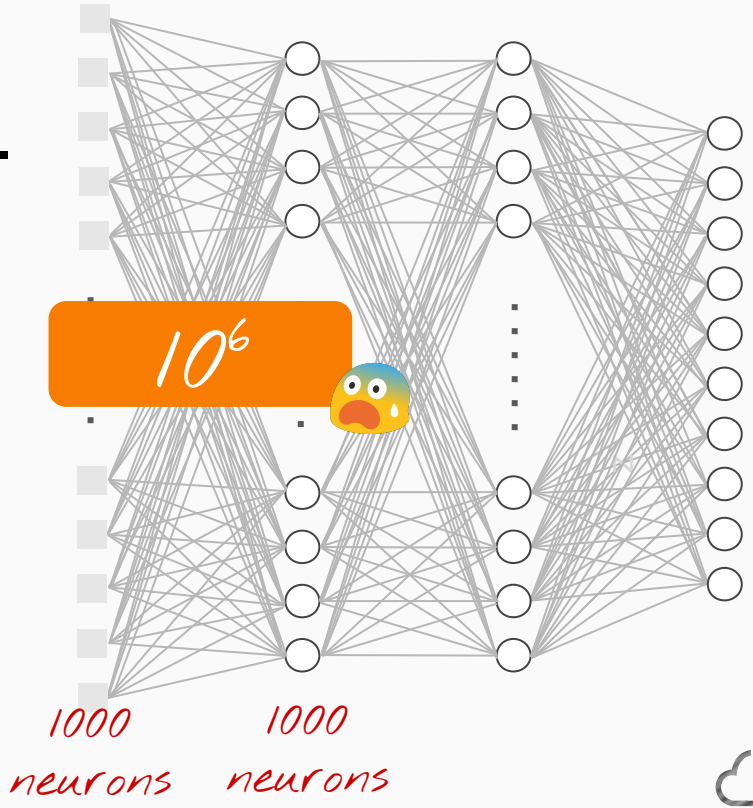
Machine Learning Framework



How to Find the Right Weights and Biases?

- ~~Enumerate all possible values.~~
- more efficient approach?

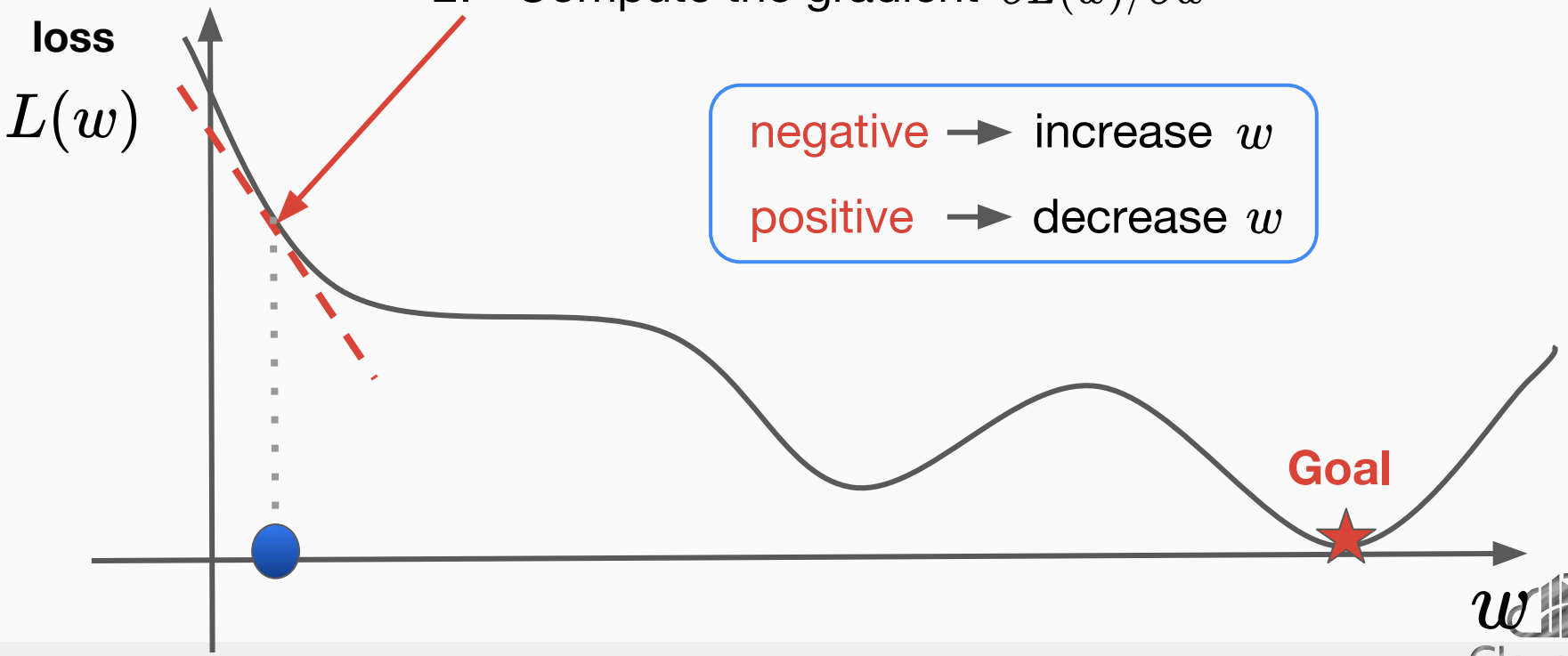
Gradient Descent



CloudMile

Gradient Descent

1. Randomly pick an initial value for w
2. Compute the gradient $\partial L(w)/\partial w$



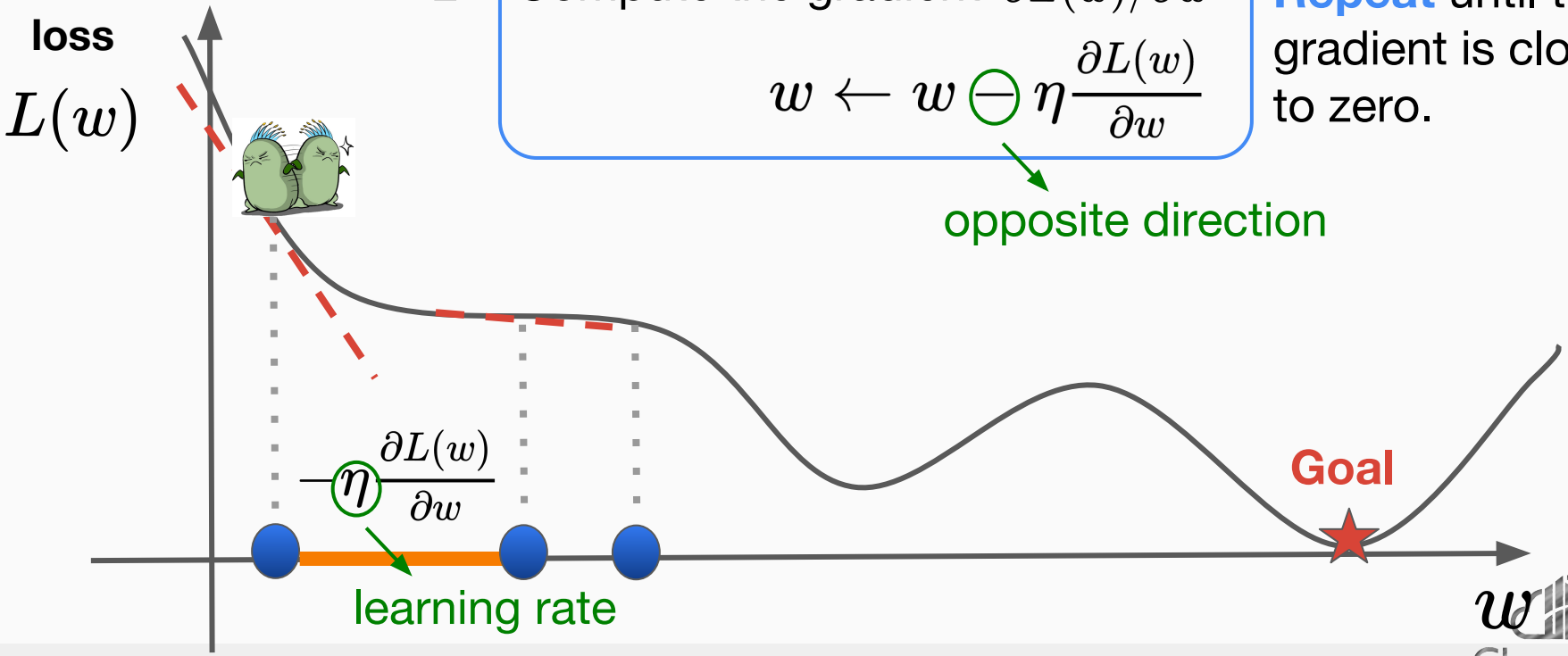
Gradient Descent (Cont'd)

1. Randomly pick an initial value for w
2. Compute the gradient $\partial L(w)/\partial w$

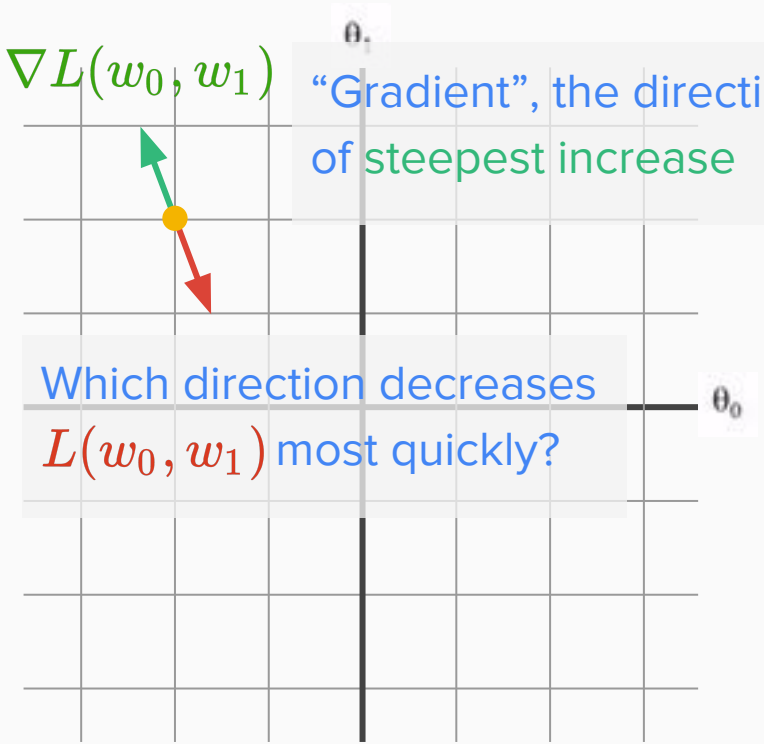
$$w \leftarrow w \ominus \eta \frac{\partial L(w)}{\partial w}$$

opposite direction

Repeat until the gradient is close to zero.

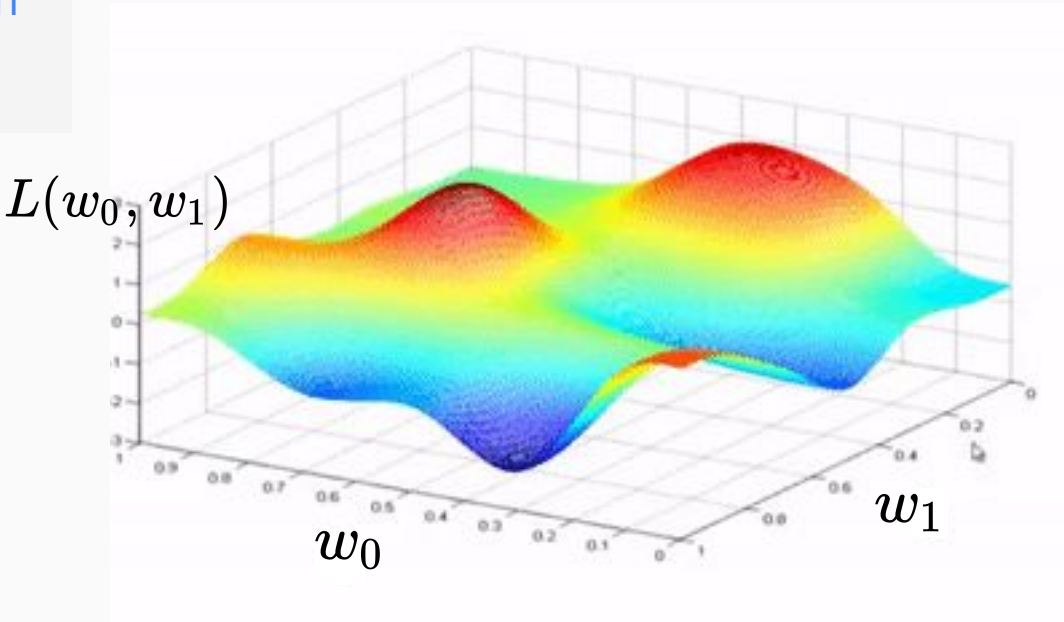


Gradient Descent for Two Parameters

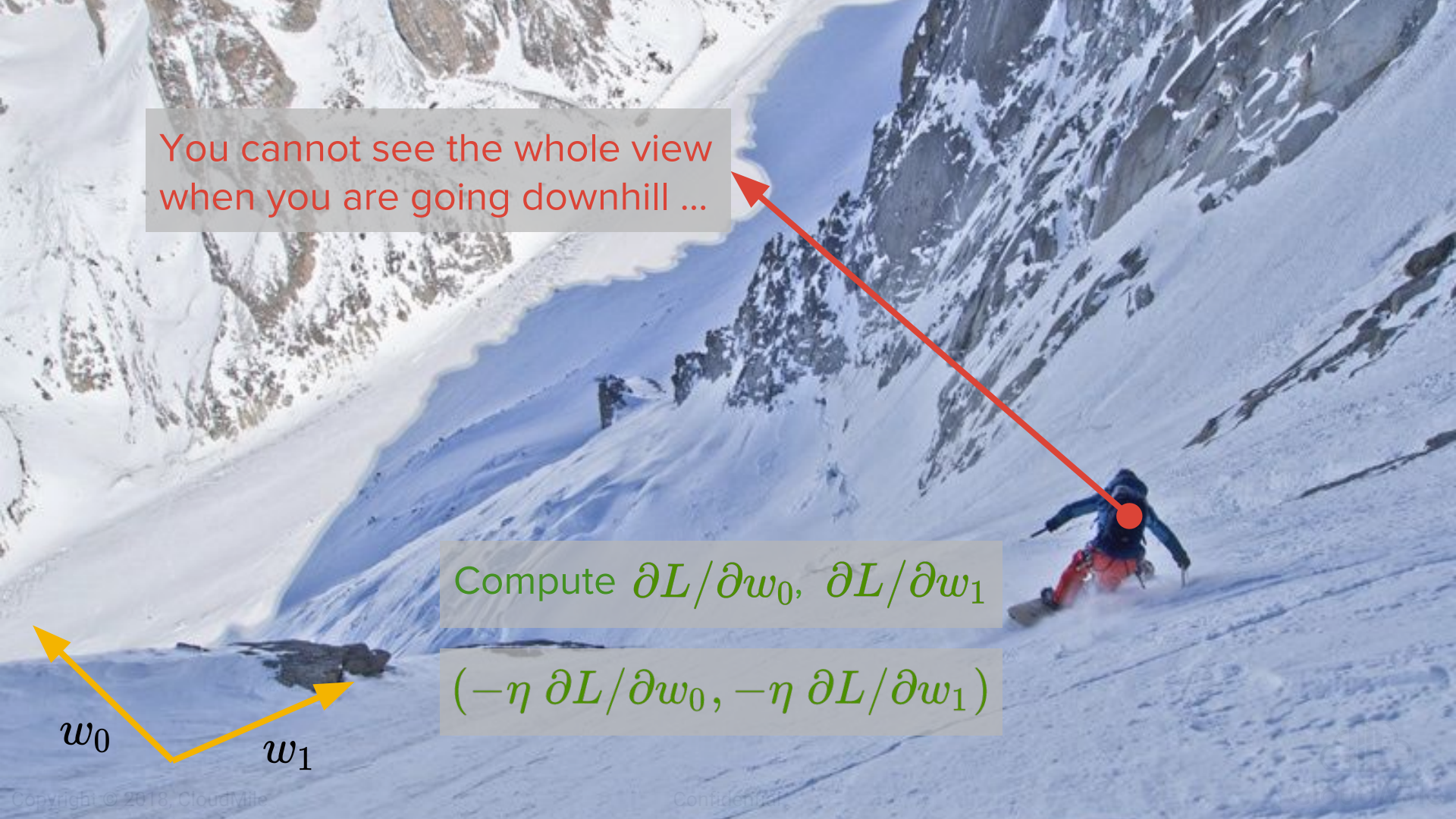


"Gradient", the direction of steepest increase

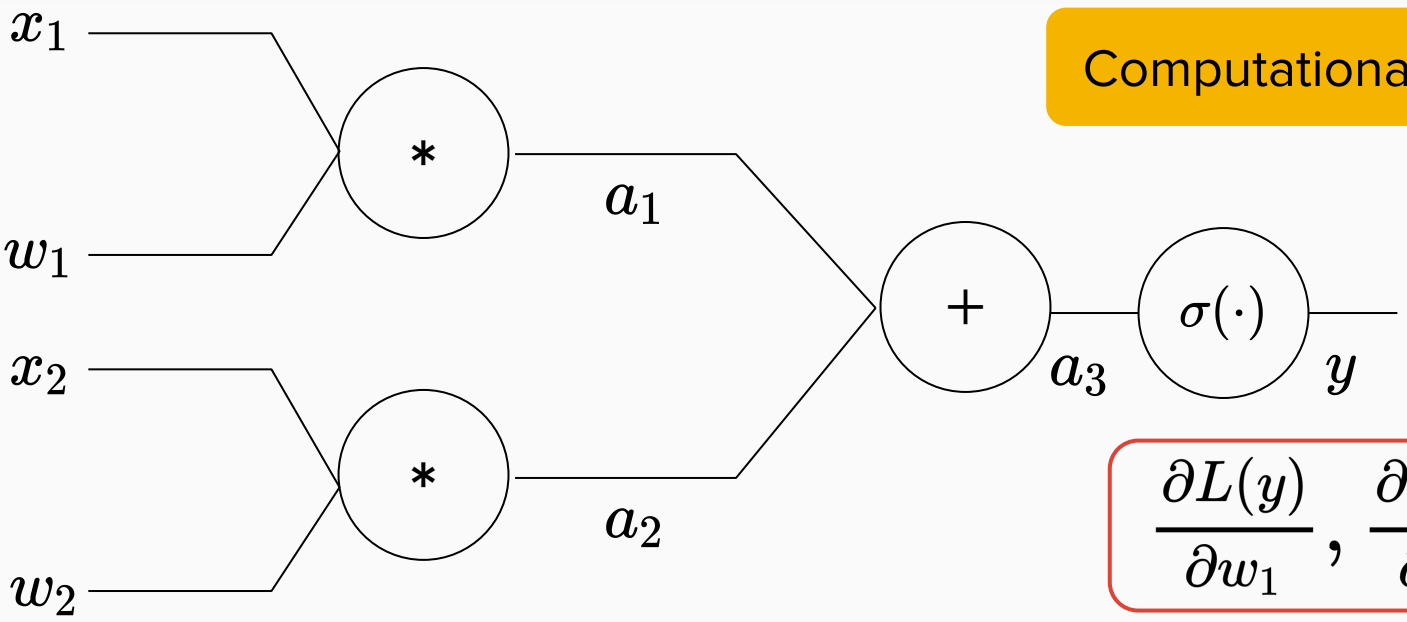
Which direction decreases $L(w_0, w_1)$ most quickly?



CloudMile



Let's see a simple case when there is only one neuron ...



Computational Graph

$$\frac{\partial L(y)}{\partial w_1}, \frac{\partial L(y)}{\partial w_2} ?$$

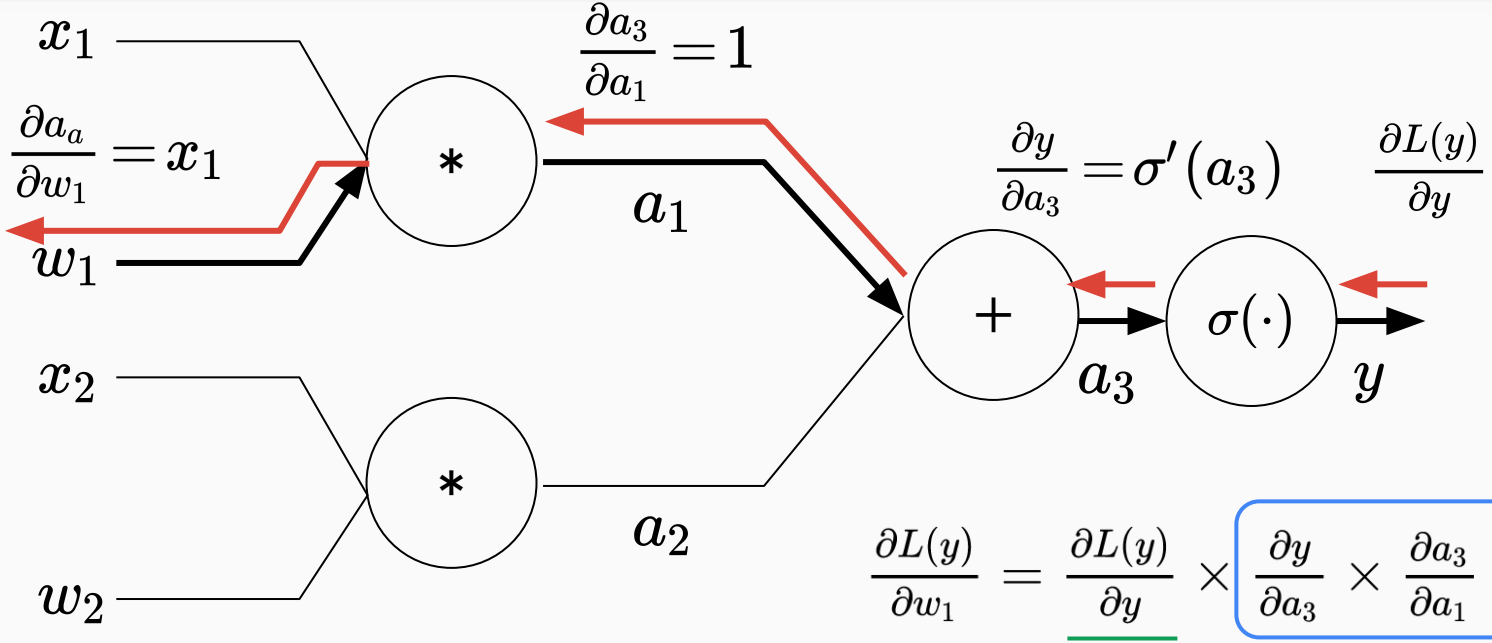
activation function

$$y = \sigma(w_1 x_1 + w_2 x_2)$$

weighted sum

$$a_1 = w_1 x_1$$
$$a_3 = a_1 + a_2$$
$$y = \sigma(a_3)$$

Back propagation: an efficient way to compute the gradient in NN



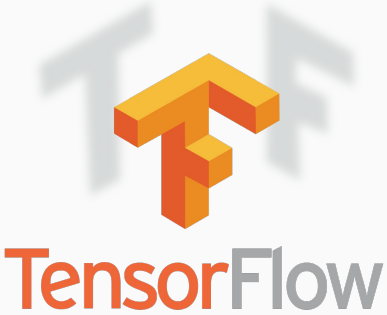
$$\frac{\partial L(y)}{\partial w_1} = \underline{\frac{\partial L(y)}{\partial y}} \times \boxed{\frac{\partial y}{\partial a_3} \times \frac{\partial a_3}{\partial a_1} \times \frac{\partial a_1}{\partial w_1}}$$

depend on loss function

$$\sigma'(a_3) \times 1 \times x_1$$

CloudMile

Don’t worry about the gradient since the toolkits will handle it :)



Ref: https://youtu.be/ibJpTrp5mcE

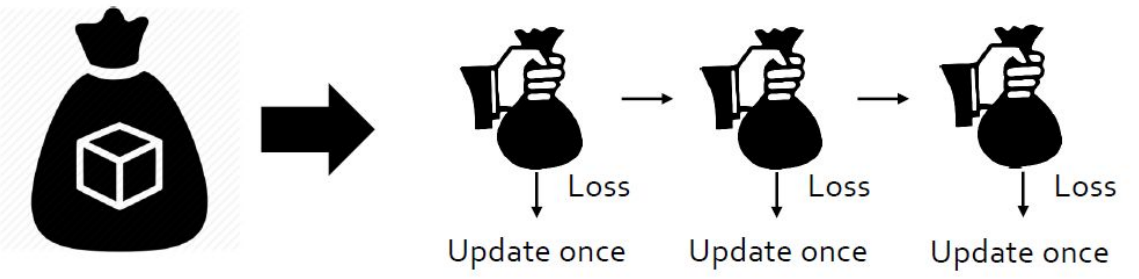
Stochastic/Mini-batch Gradient Descent

Learning:

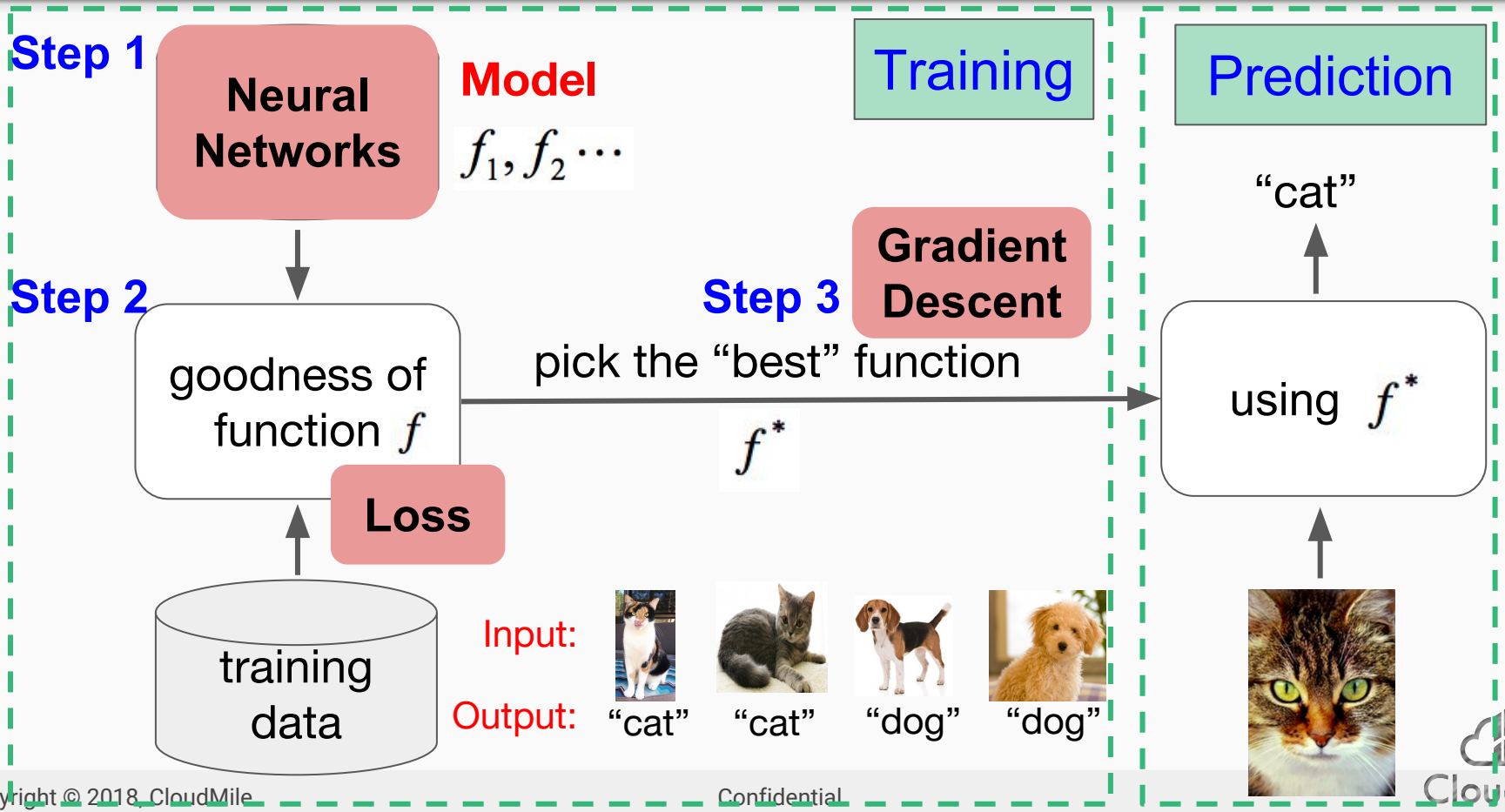
Finding the right weights and biases that minimize average loss

But, averaging the loss of all training data is very inefficient …

- Stochastic: pick a single training data, then perform a parameter update.
- Mini-batch: pick a batch training data, then perform a parameter update.
  - e.g., batch size: 16, 32, 64, …



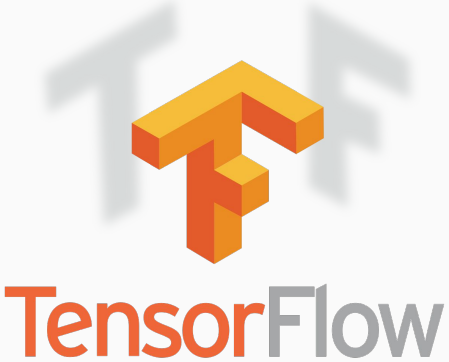
Machine Learning Framework



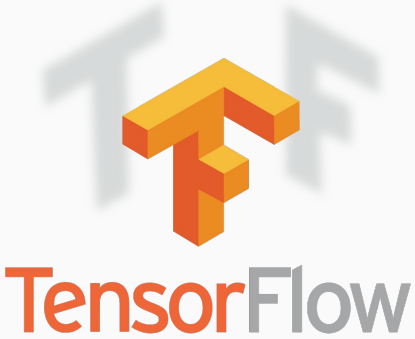
CloudMile

Build your first neural network!

Tools that we'll use today!

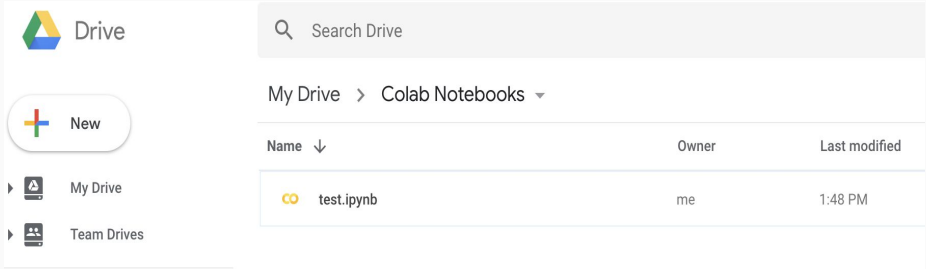


Colab



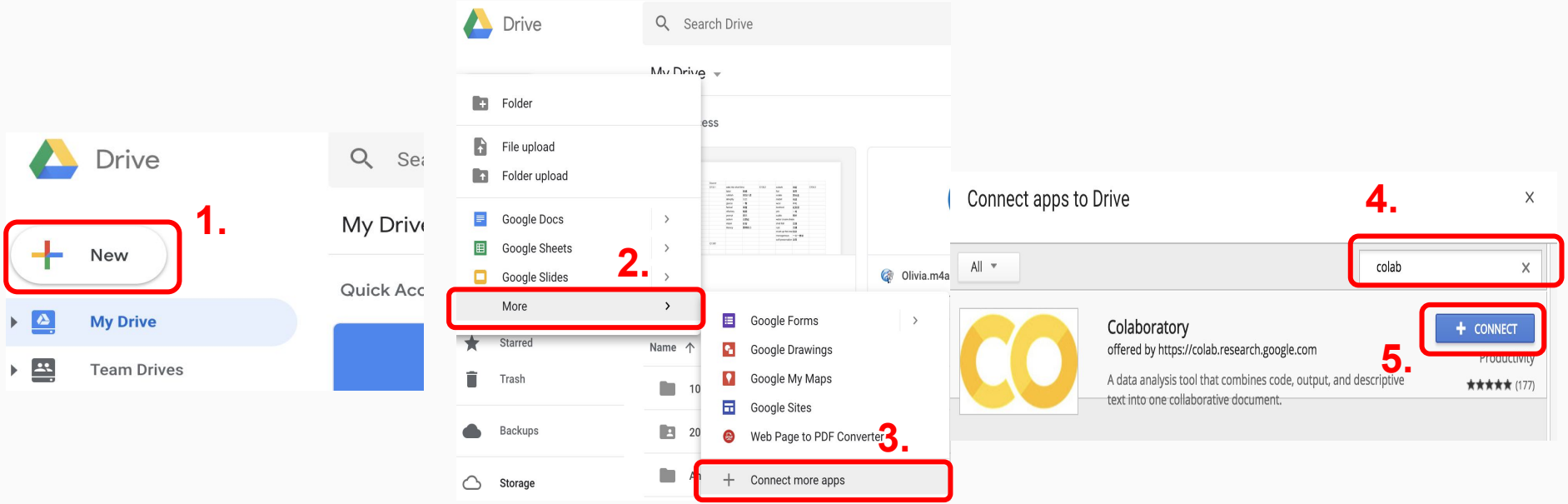
Colab

- 類似Jupyter Notebook，互動式介面，及時輸出結果
- 儲存在Drive 上，可以共同編輯
- 背後有VM 在運作
- 可連續使用免費的 Tesla K80 GPU 12小時(每次)
- 許多套件已事先安裝，例如：TensorFlow, scikit-learn, numpy, matplotlib



Colab

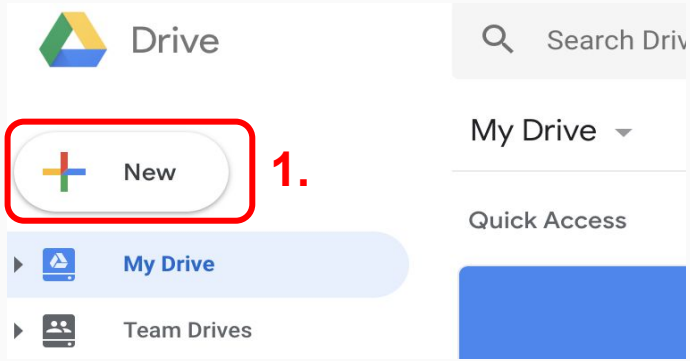
- 進入Google Drive 連接到Colaboratory

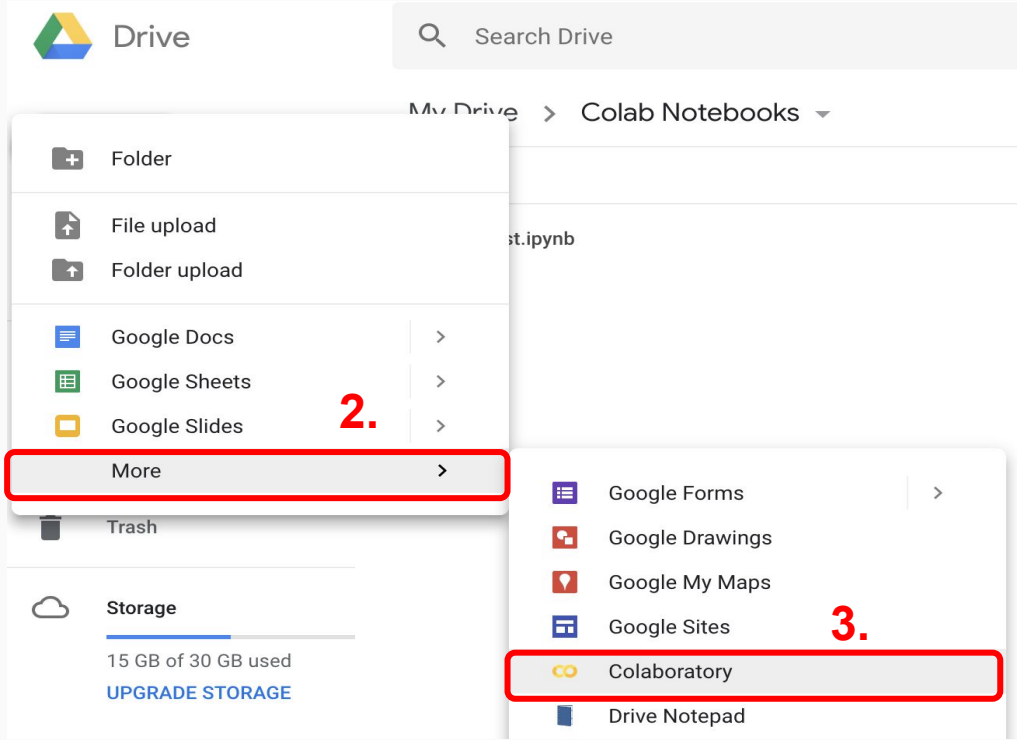


CloudMile

Colab

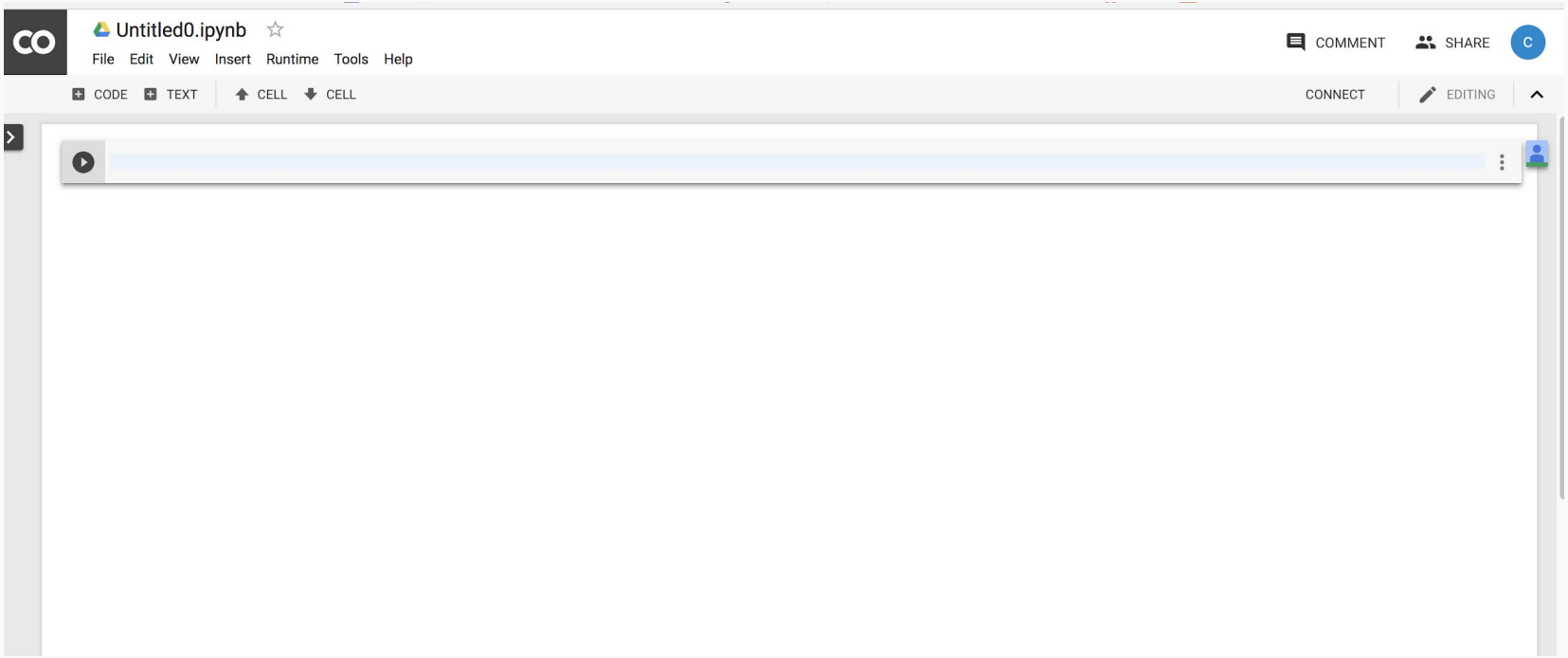
- 建立一個新的Colaboratory





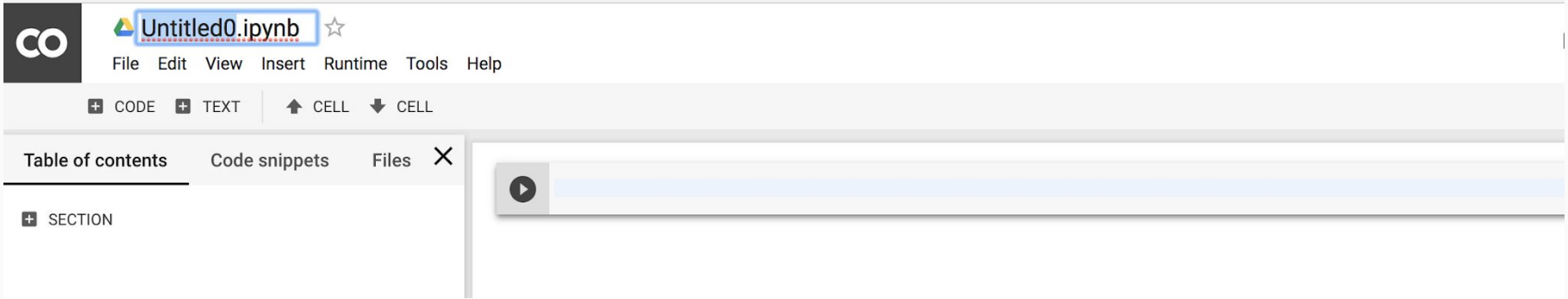
CloudMile

Colab



CloudMile

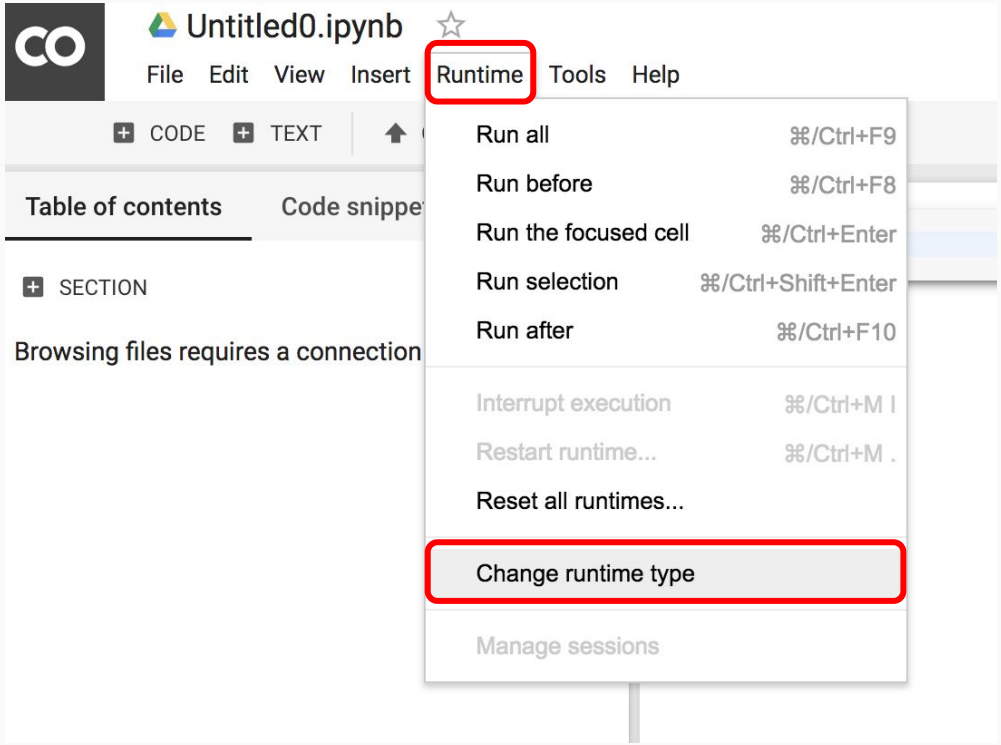
Colab



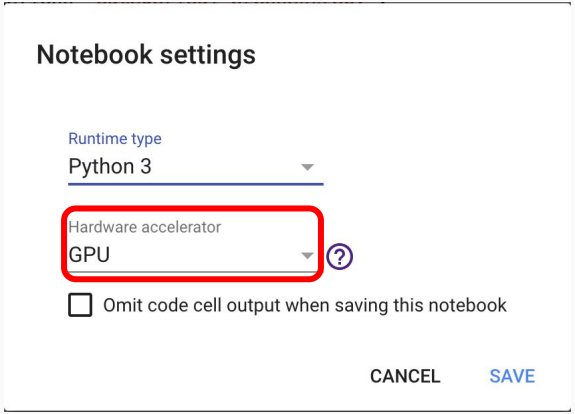
Colab

CloudMile

Colab



- 使用 Python 3

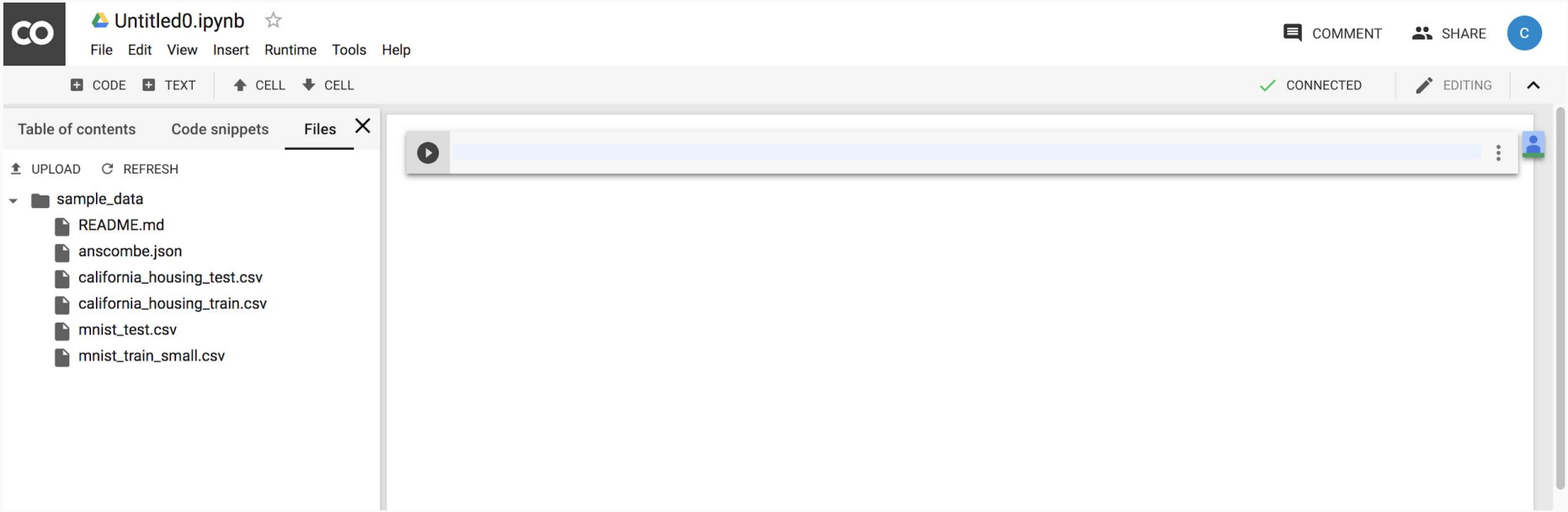


CloudMile

Colab

CloudMile

Colab



基本語法 - Python

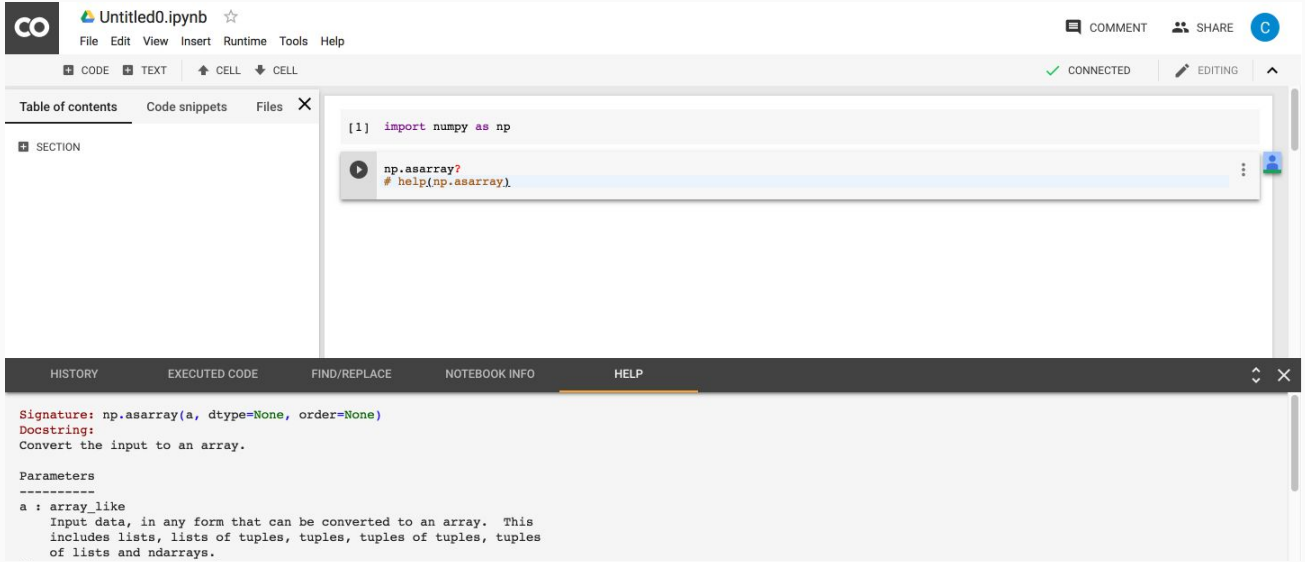
- 輸入 Python 程式碼，點擊  或是「Shift + Enter」執行程式碼區塊。

基本語法 - Shell

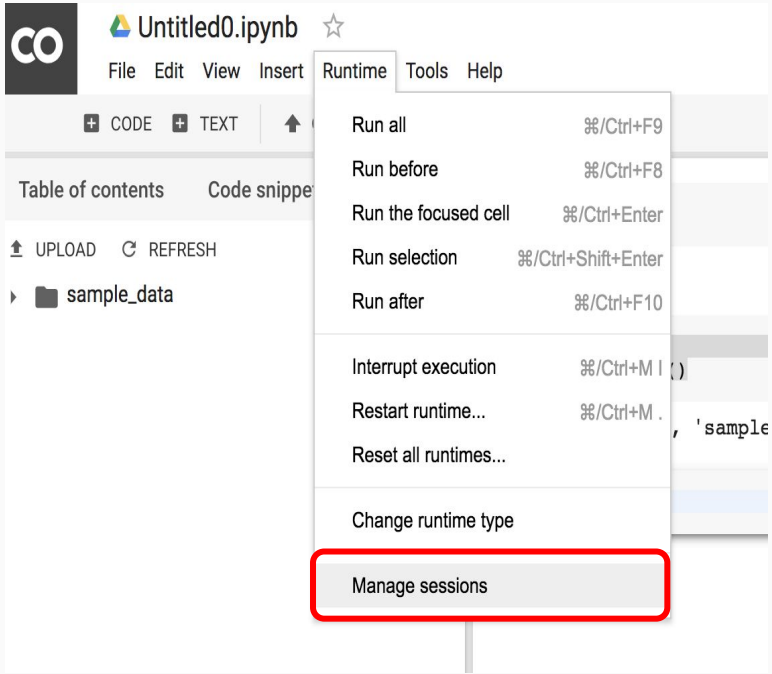
- 前面加上！

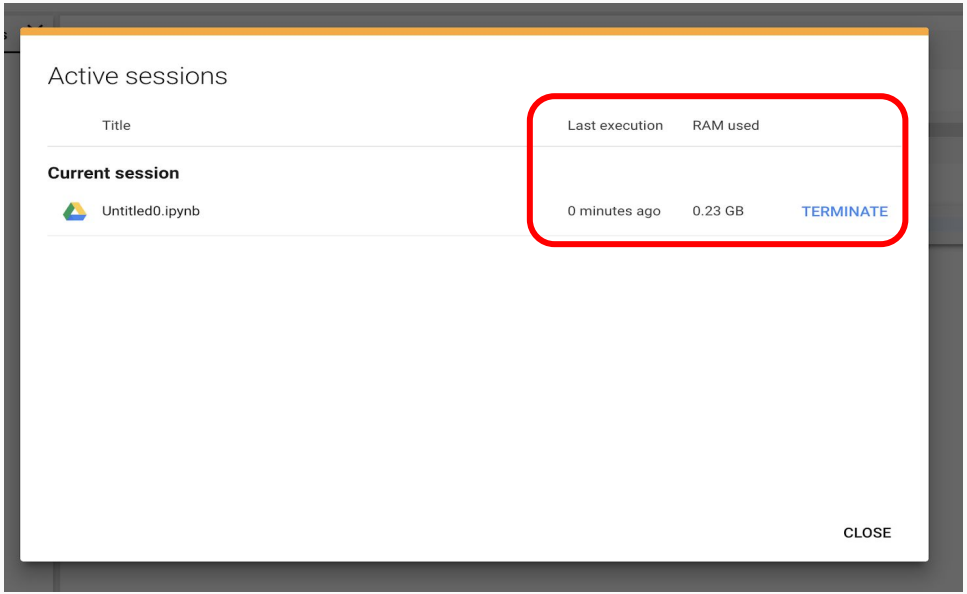
特殊語法 - 查看

- 在function 後面加上? 或是用help( )可以查看documentation

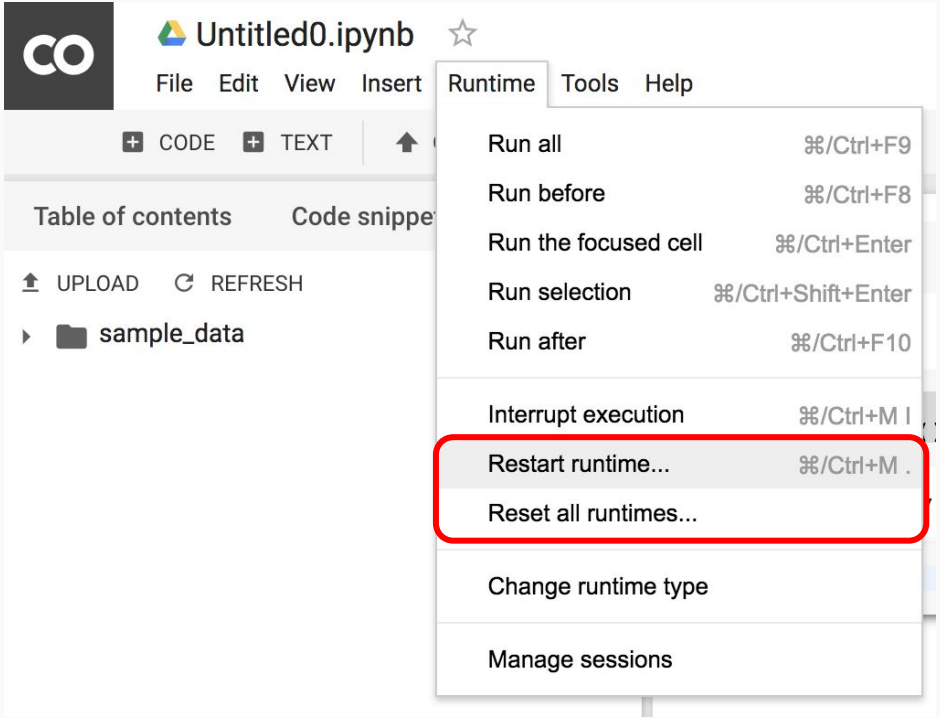


Manage Session



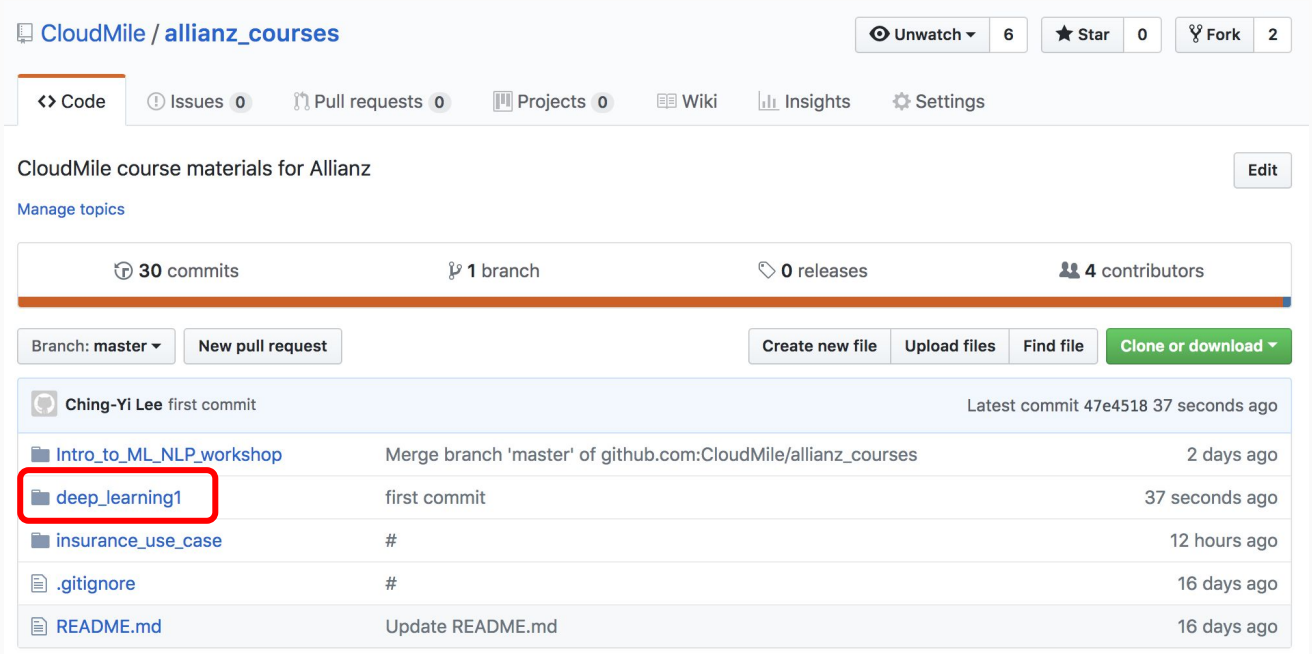


Restart runtime/ Reset runtime



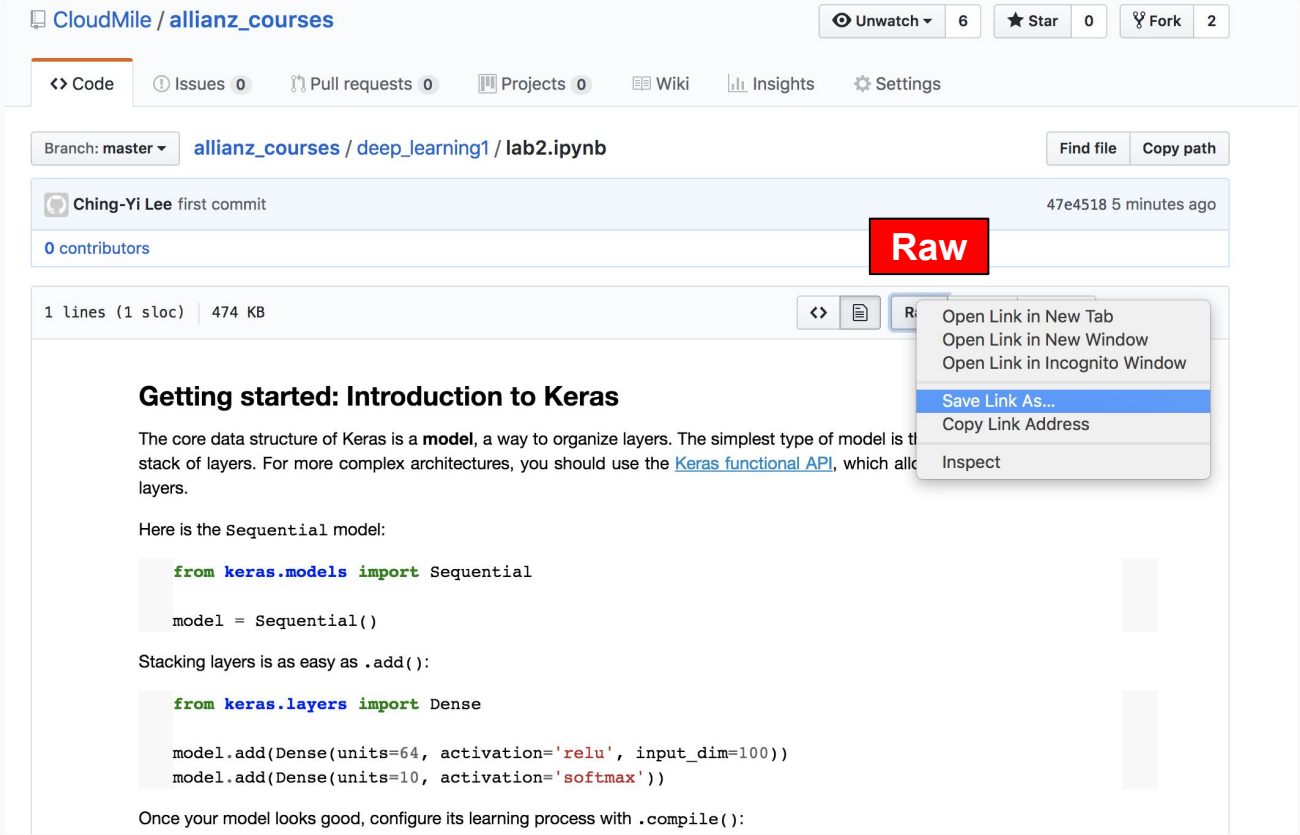
Colab

從網站下載 code 到本機 -> 網址 : https://github.com/CloudMile/allianz_courses



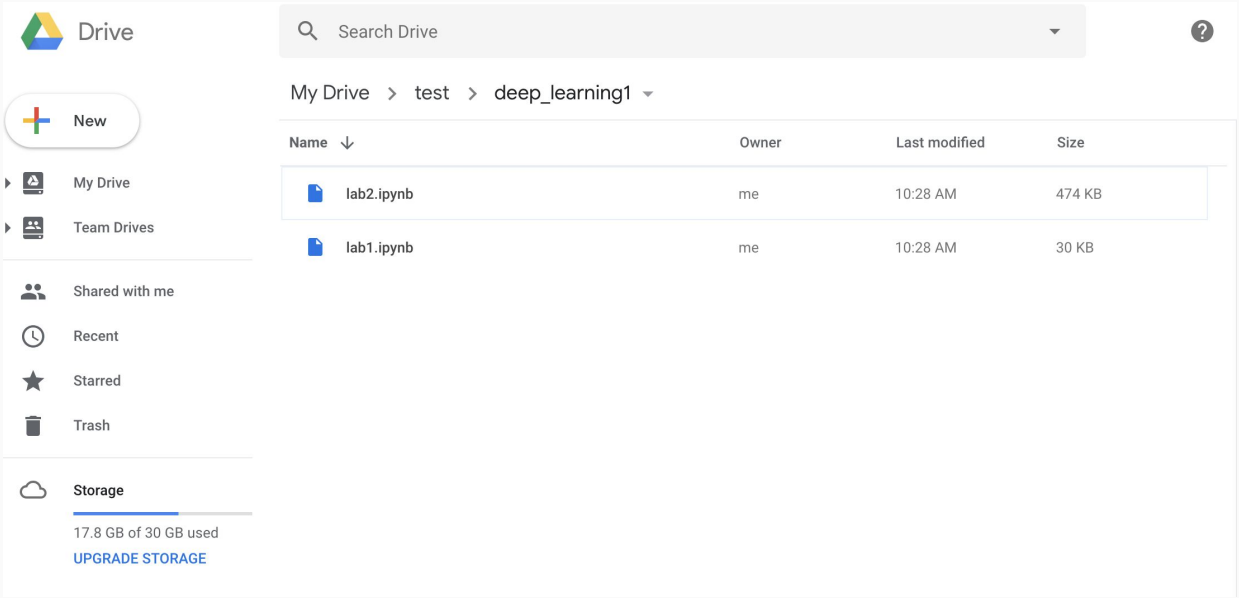
CloudMile

Colab

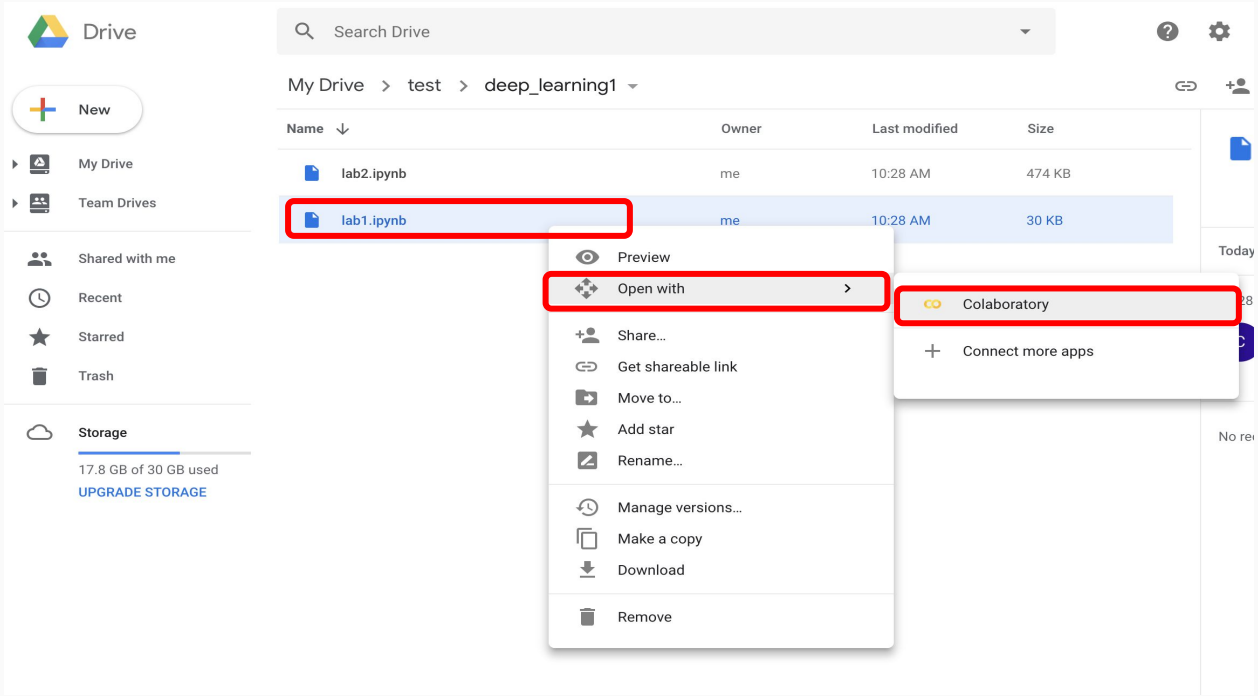


Open jupyter notebook from Colab

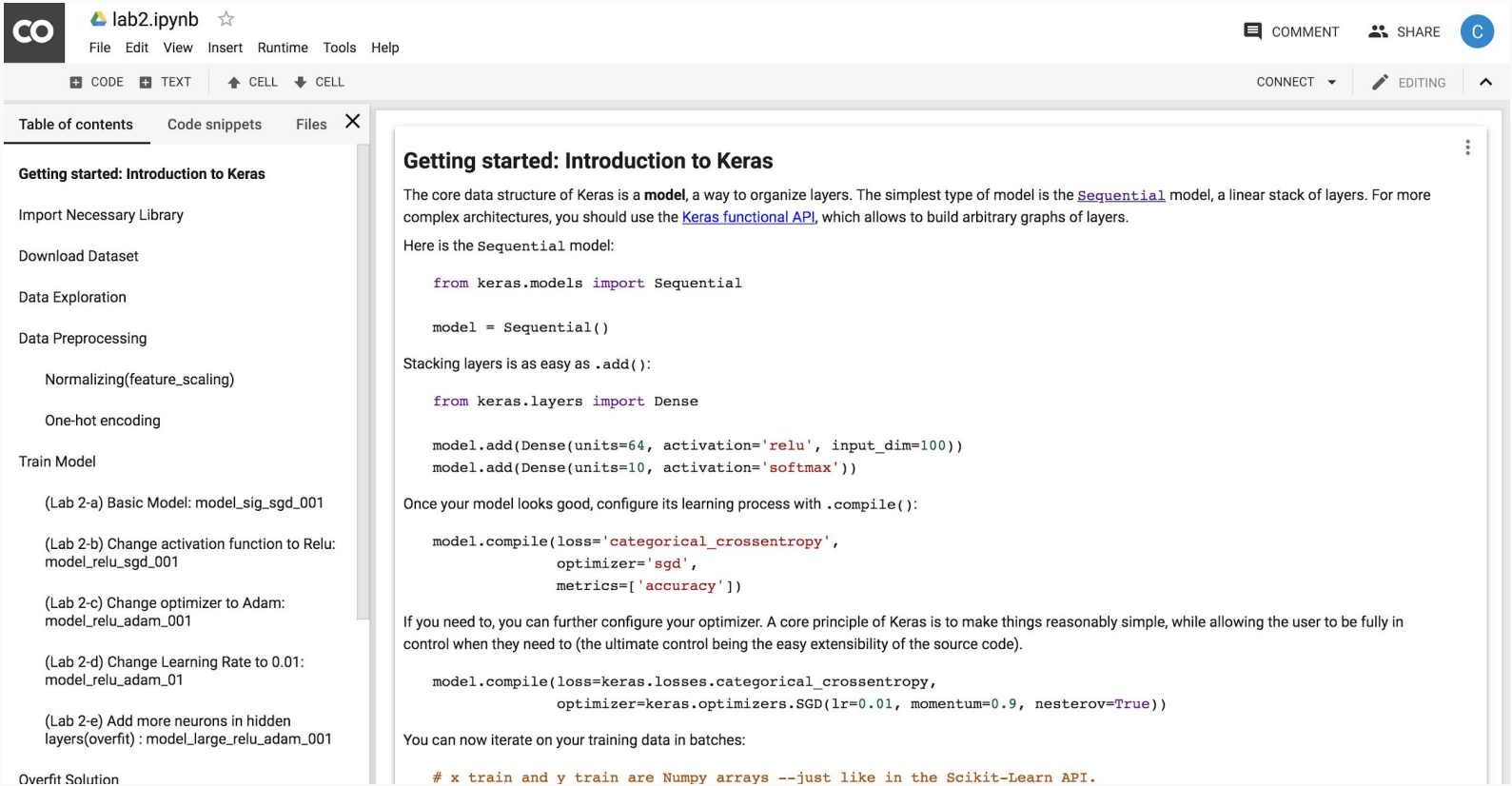
- Upload files to drive



Open jupyter notebook from Colab



Colab



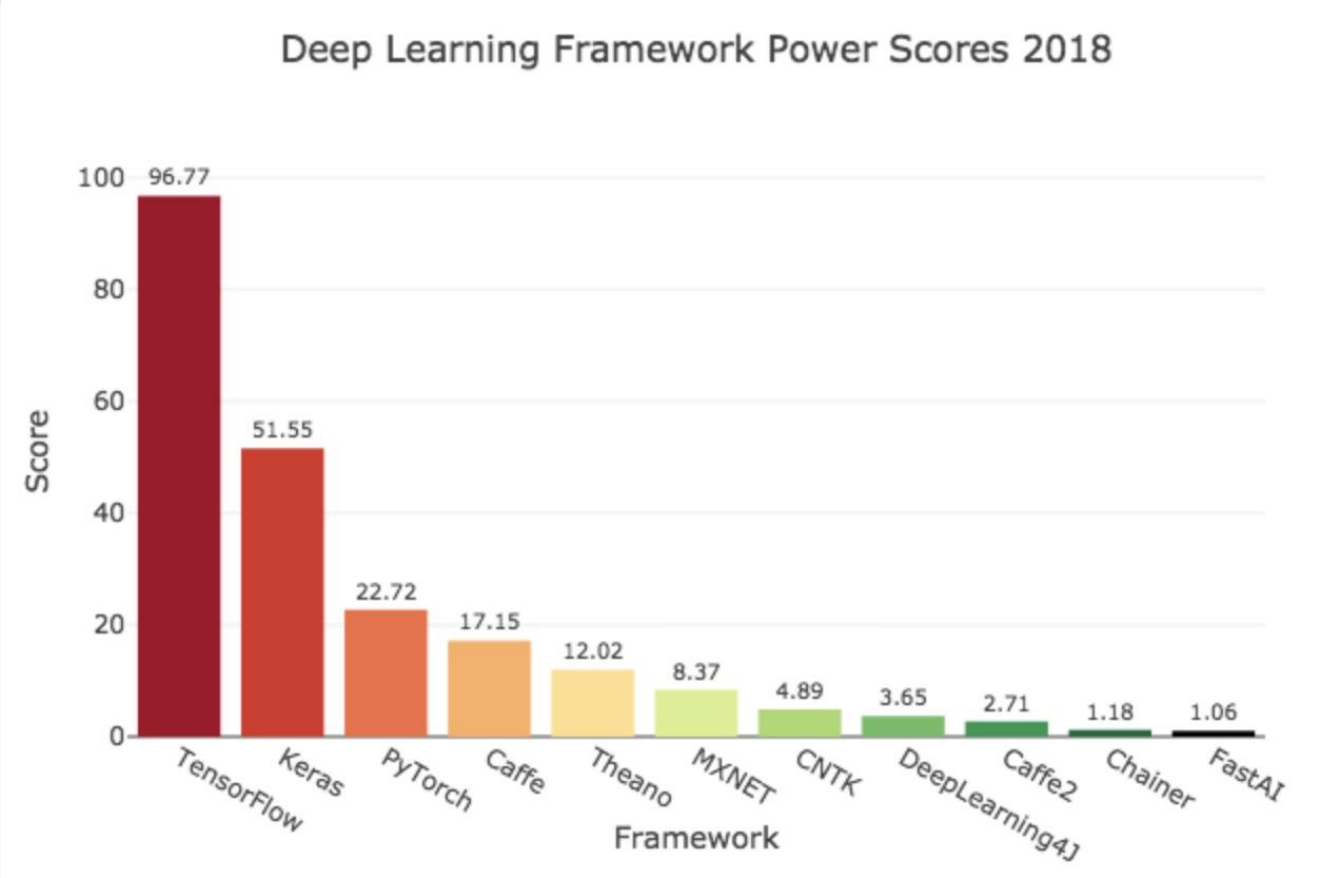
Tensorflow

TensorFlow

colab

Keras

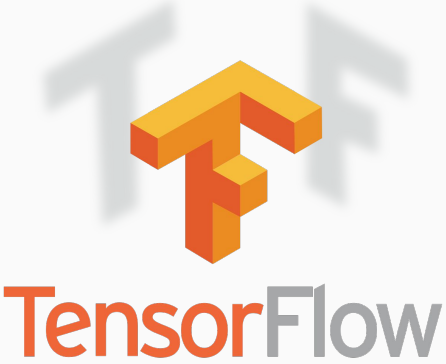
Different Deep Learning Frameworks



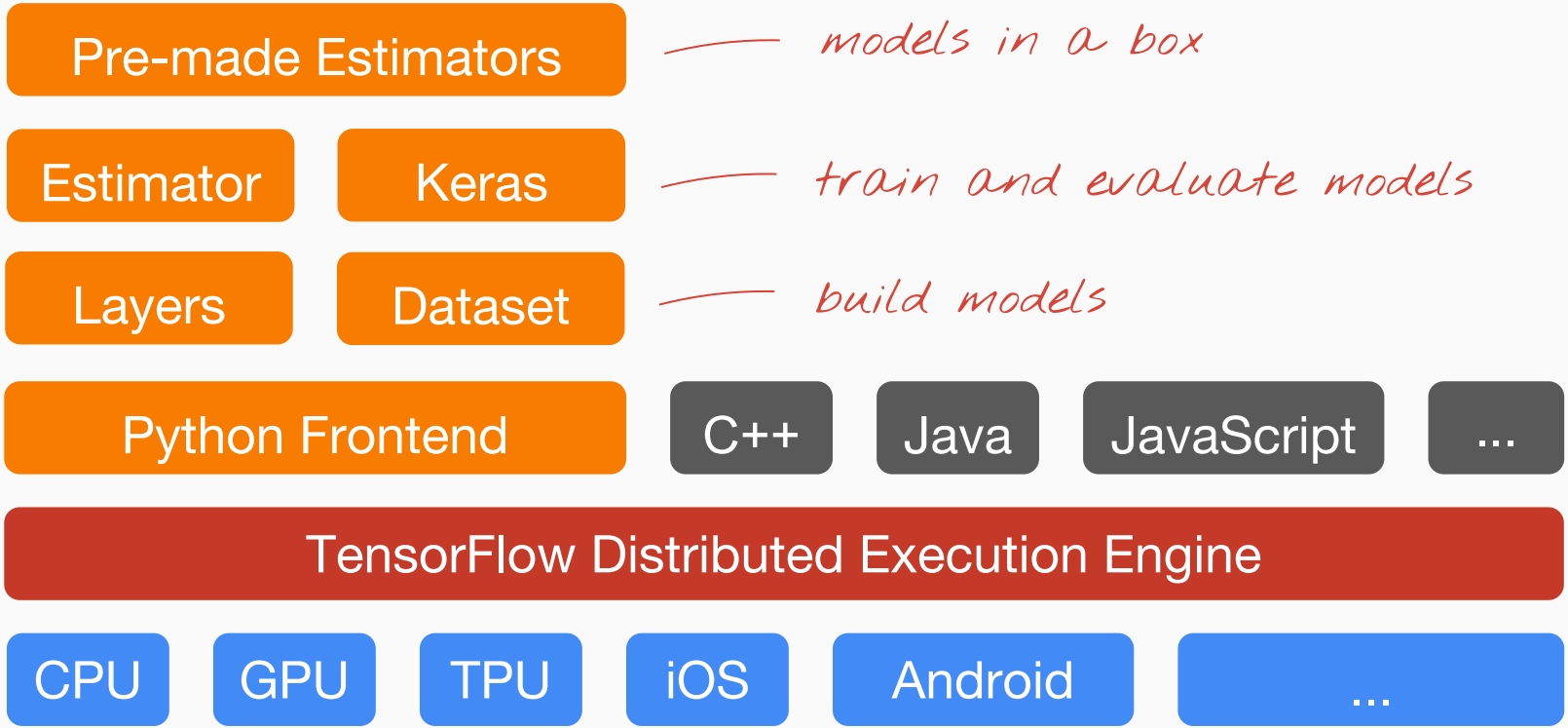
CloudMile

TensorFlow

- Open source software library created by Google
- Google 開發的開源工具
- 透過 Computational Graph 運算



TensorFlow



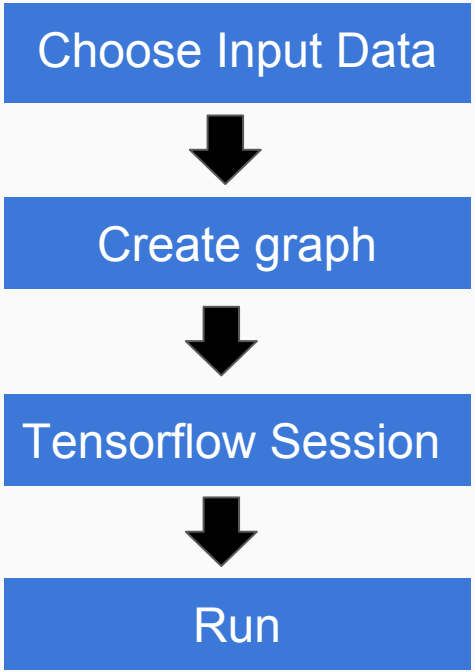
TensorFlow Fundamental

```python
import numpy as np
import tensorflow as tf

x_data = np.random.rand(100).astype(np.float32)
y_data = x_data * 0.1 + 0.3

W = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
b = tf.Variable(tf.zeros([1]))
y = W * x_data + b

loss = tf.reduce_mean(tf.square(y - y_data))
optimizer = tf.train.GradientDescentOptimizer(learning_rate=0.5)
train = optimizer.minimize(loss)

init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init)

for step in range(201):
    sess.run(train)
    if step % 20 == 0:
        print(step, sess.run(W), sess.run(b))

sess.close()
```

Input data

Graph

Session

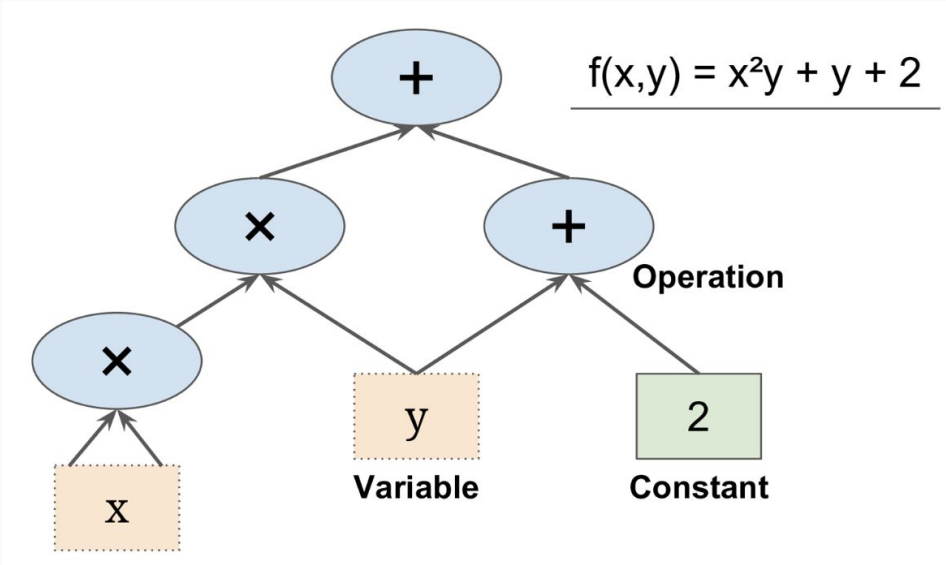


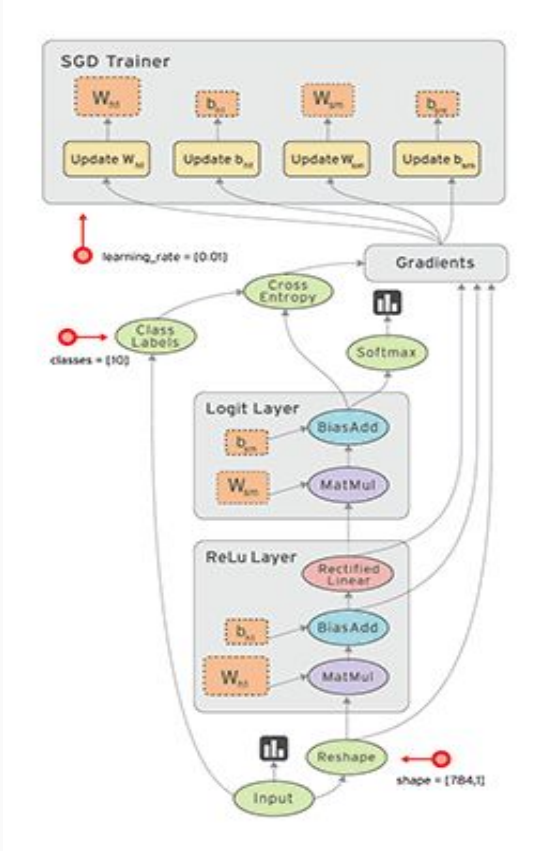
CloudMile

TensorFlow Fundamental

- TensorFlow 是由 **Tensor** 和 **Flow** 組成

Computational Graph





CloudMile

TensorFlow Fundamental

- **Build Graph**

```
import tensorflow as tf

# 1x2 matrix
matrix1 = tf.constant([[3, 3]])

# 2x1 matrix
matrix2 = tf.constant([[2],
                       [2]])

# matmul() is multiplication of matrixs
product = tf.matmul(matrix1, matrix2)
```

```
product
```

Must run session to see the result.

Output:

```
<tf.Tensor 'MatMul_5:0' shape=(1, 1) dtype=int32>
```


CloudMile

TensorFlow Fundamental

- **Run session**

```
import tensorflow as tf

# 1x2 matrix
matrix1 = tf.constant([[3, 3]])

# 2x1 matrix
matrix2 = tf.constant([[2],
                       [2]])

# matmul() is multiplication of matrixs
product = tf.matmul(matrix1, matrix2)

# open session

with tf.Session() as sess:
    result = sess.run(product)
    print(result)
```

另一種寫法:

```
# open session
sess = tf.Session()
result = sess.run(product)
print(result)

# close session
sess.close()
```

Output: [[12]]

TensorFlow Fundamental

- **Input Data**

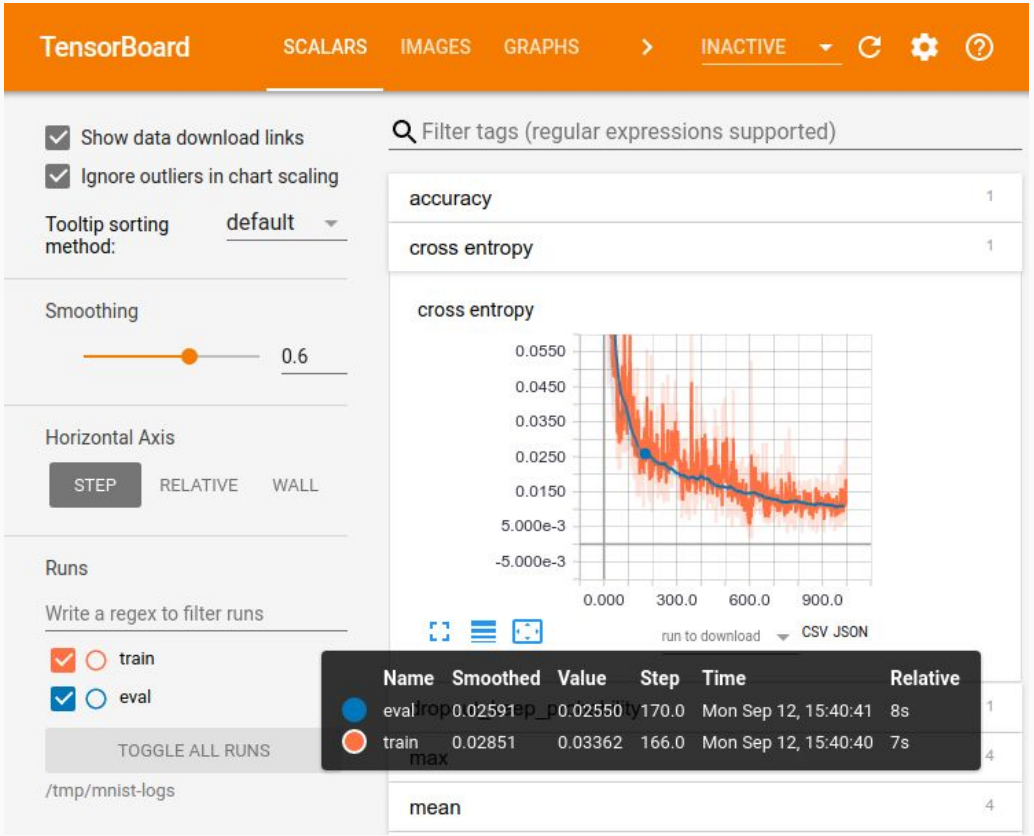
```
import tensorflow as tf

# Create placeholder for input data

input1 = tf.placeholder(tf.float32)
input2 = tf.placeholder(tf.float32)
output = tf.add(input1, input2)


# Use 8.0 for input1 and 3.0 for input2

with tf.Session() as sess:
    result = sess.run([output], feed_dict = {input1: [8.0] , input2: [3.0]})
    print(result)
```

Output : `[array([11.], dtype=float32)]`

Tensorboard

Visualize Learning

- Scalars
- Images
- Graphs
- Audio
- Histograms



CloudMile

Lab 1 : TensorFlow Basics

Topic : TensorFlow Linear Regression Example

Start coding !

Filename	lab1.ipynb
x	10000 of random generated value between 0 and 1
Target function	y = wx + b
Loss function	mean square error
Learning rate	0.5
optimizer	tf.train.GradientDescentOptimizer

Keras

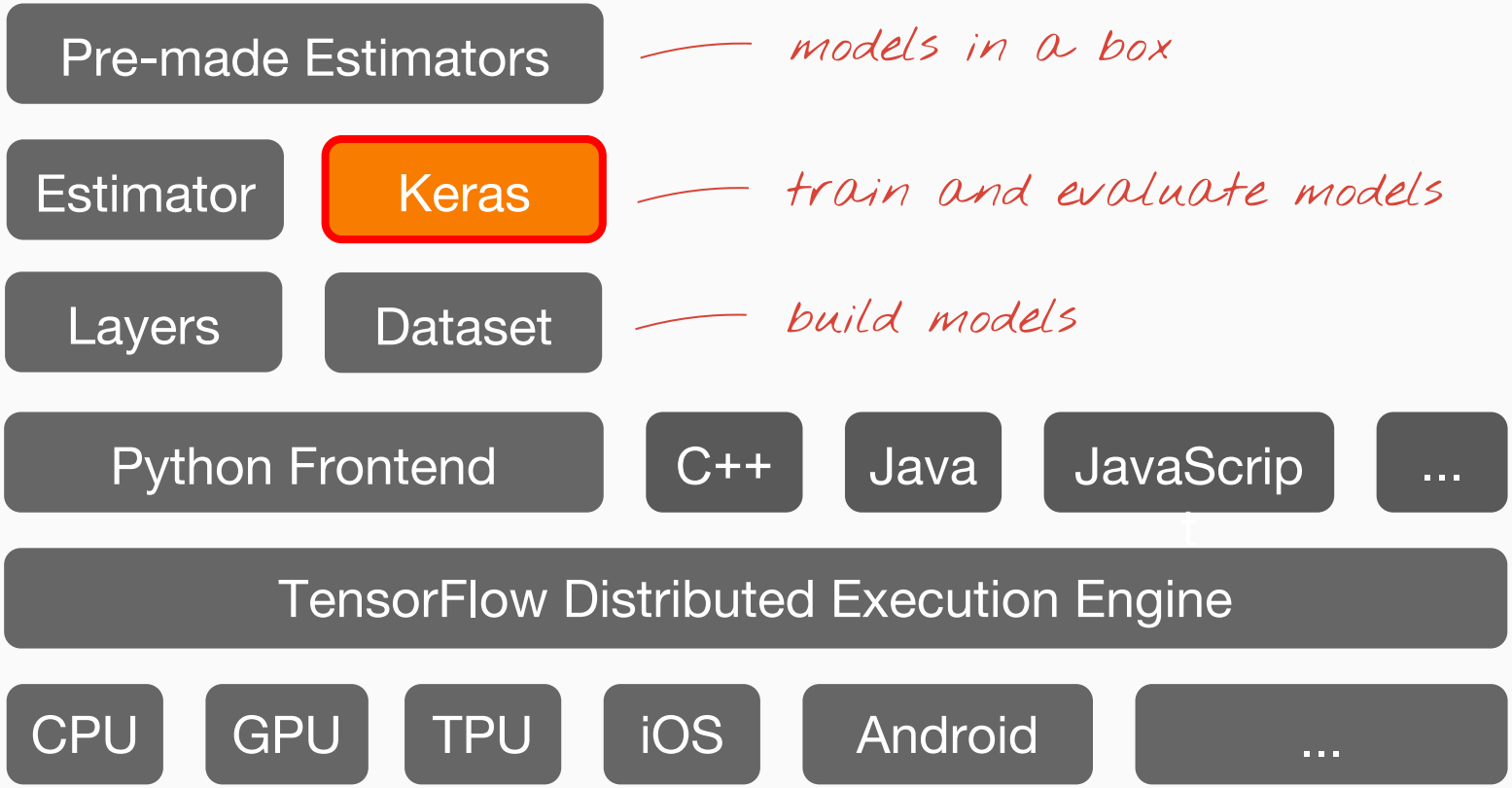
Keras

colab

TensorFlow

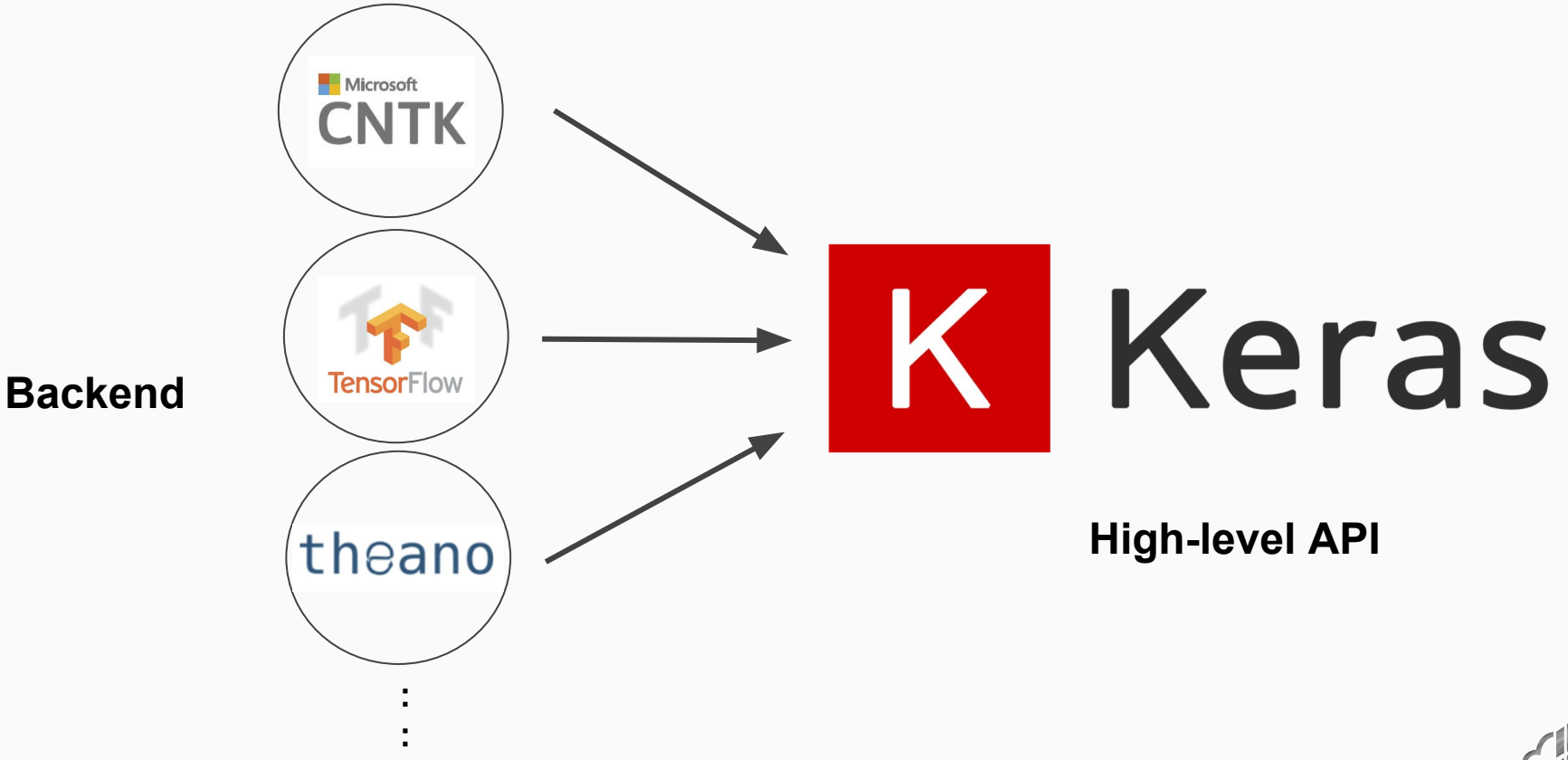
CloudMile

Keras



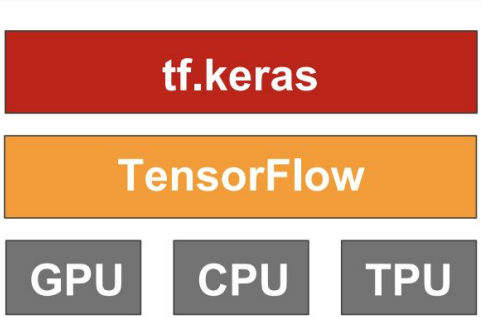
CloudMile

Keras



Keras

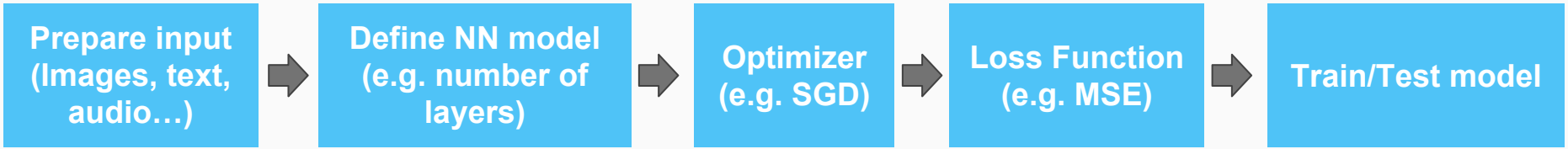
- Keras is an API designed for human beings, not machines.
- Part of core Tensorflow (tf.keras)
- Strong multi-GPU support and distributed training support



	Keras	TensorFlow
Difficulty	Easy	More Difficult
Flexibility	Medium	Flexible
Rapid Prototyping	Fast	Medium
Suitable User	Beginner	Advanced User

Implementing a NN in Keras

- **Steps to build a model**
  - Prepare the input
  - Define the model architecture
  - Specify the optimizer and learning rate
  - Specify the loss function
  - Train and test the model on the dataset



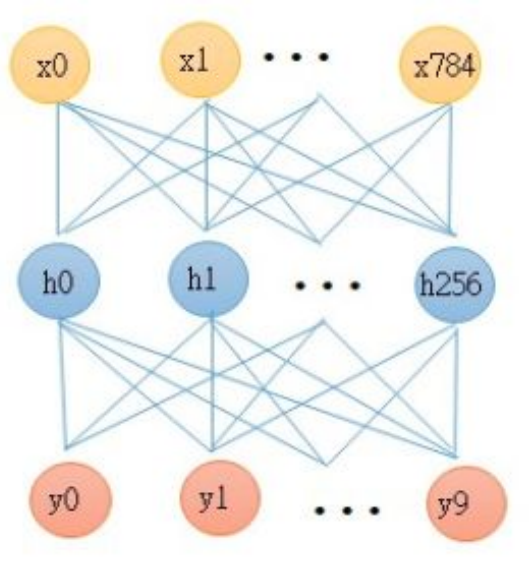
Keras Fundamental

- Building neural network is like making a cake

Input layer (x) **784**

Hidden layer(h1) **256**

Output layer(y) **10**



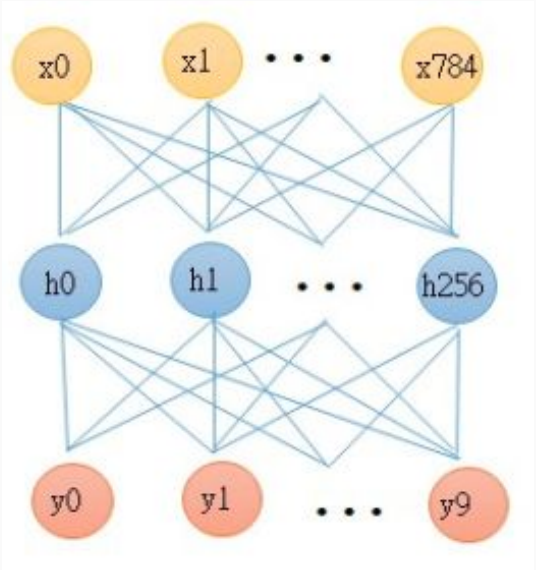
Keras Fundamental

- **Import *Sequential* class from *keras.model***

```
from tensorflow.keras import Sequential

model = Sequential()
```

- **Stacking layers using *.add()* method**

```
from tensorflow.keras.layers import Dense

model.add(Dense(256,
          input_dim = 784,
          activation = 'sigmoid'))

model.add(Dense(10,
          activation = 'sigmoid'))
```



Keras Fundamental

- **Configure learning process using *.compile()* method**

```
model.compile(loss='mean_squared_error',
        optimizer='sgd')
```

- **Train the model on train dataset using *.fit()* method**

```
model.fit(x_train, y_train, epochs=5, batch_size=32)
```

- **Evaluate the model on test dataset using .evaluate() method**

```
loss_and_metrics = model.evaluate(x_test, y_test, batch_size=128)
```

- **Predict the data on test dataset using .predict() method**

```
model.predict(x_test)
```

Discussion (~5 mins)

Term	Meaning
Label	
Input	
Example	
Model	
Training	
Prediction	

Discussion (~5 mins)

Term	Meaning
Label	True Answer
Input	Predictor variables, what you can use to predict the label
Example	Input and corresponding label
Model	Math function that takes variables and predict label
Training	Adjust model to minimize error
Prediction	Using model on unlabeled data

Lab 2-a

Topic : Fashion MNIST classification

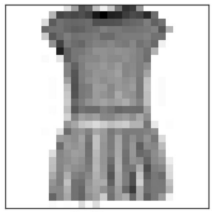
Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Import dataset ➔ Data exploration ➔ Data preprocessing ➔ Train a classification model

Lab 2a - Data Exploration


9: Ankle boot


0: T-shirt/top


0: T-shirt/top


3: Dress

```
Train image shape:(60000, 28, 28)
Test image shape:(10000, 28, 28)
Train class: [0 1 2 3 4 5 6 7 8 9]
Test class: [0 1 2 3 4 5 6 7 8 9]
```

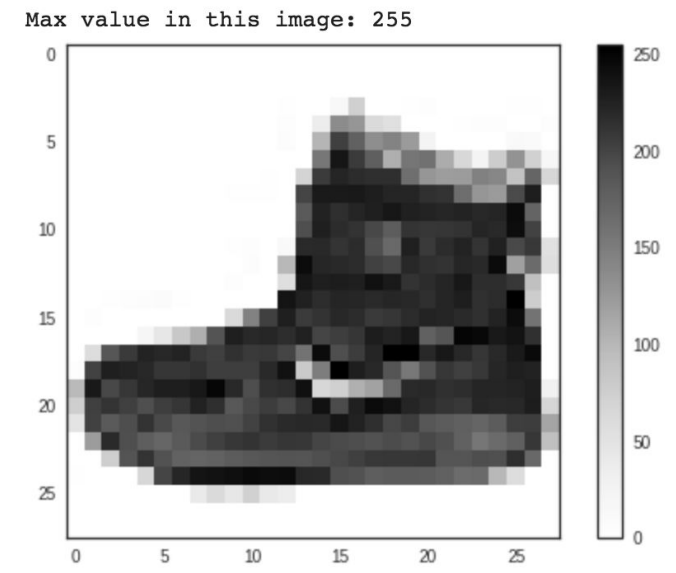
Label	Class
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

- There are 60000 images for training, and 10000 images for testing.
- The image size is 28 pixels x 28 pixels
- 10 classes in total

Lab 2a - Data Preprocessing

Normalization

- Be able to learn the real structures instead of dealing with the scale differences
- Treat all features in the same scale ⇨ treat all images in the same way



$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

```
# Normalize Data
train_images = train_images / 255.0
test_images = test_images / 255.0
```

Before Normalization:

Image pixel range: **0~255**

After Normalization:

Image pixel range: **0~1**

CloudMile

Lab 2a - Data Preprocessing

One hot encoding

- Label = 9 doesn't mean it is larger and Label = 0 doesn't mean it is smaller
- Don't mathematically imply any ordinal relationship between the classes.

Label	Class	One-hot Encoding
0	T-shirt/top	[1,0,0,0,0,0,0,0,0,0]
1	Trouser	[0,1,0,0,0,0,0,0,0,0]
2	Pullover	[0,0,1,0,0,0,0,0,0,0]
3	Dress	[0,0,0,1,0,0,0,0,0,0]
4	Coat	[0,0,0,0,1,0,0,0,0,0]
5	Sandal	[0,0,0,0,0,1,0,0,0,0]
6	Shirt	[0,0,0,0,0,0,1,0,0,0]
7	Sneaker	[0,0,0,0,0,0,0,1,0,0]
8	Bag	[0,0,0,0,0,0,0,0,1,0]
9	Ankle boot	[0,0,0,0,0,0,0,0,0,1]

```
train_labels = keras.utils.to_categorical(train_labels, num_classes=10)
```

```
array([[0., 0., 0., ..., 0., 0., 1.],
       [1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]], dtype=float32)
```

Lab 2a - training

```
# Clean session fisrt
K.clear_session()

# Start building the model
model_sig_sgd_001 = Sequential()
model_sig_sgd_001.add(Flatten(input_shape=(28, 28)))
model_sig_sgd_001.add(Dense(128, activation = 'sigmoid'))

############# START CODING HERE #############

# add one dense layer with 64 neurons and with 'sigmoid' activation function (~ 1 line)
model_sig_sgd_001.add(Dense(64, activation = 'sigmoid'))

############# END CODING HERE ###############

model_sig_sgd_001.add(Dense(10, activation='softmax'))
opt = tf.train.GradientDescentOptimizer(learning_rate = 0.001)
model_sig_sgd_001.compile(loss='categorical_crossentropy',
                          optimizer = opt,
                          metrics = ['accuracy'])

# Use .summary() to see model details
model_sig_sgd_001.summary()
```

Clean graph

Flatten input image from (28, 28) to (784)

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Create an optimizer

Lab 2a - training


```
train_sig_sgd_001 = model_sig_sgd_001.fit(train_images,
                                          train_labels,
                                          epochs = 20,
                                          batch_size = 128,
                                          validation_split = 0.2,
                                          shuffle = False)
```

Lab 2-a

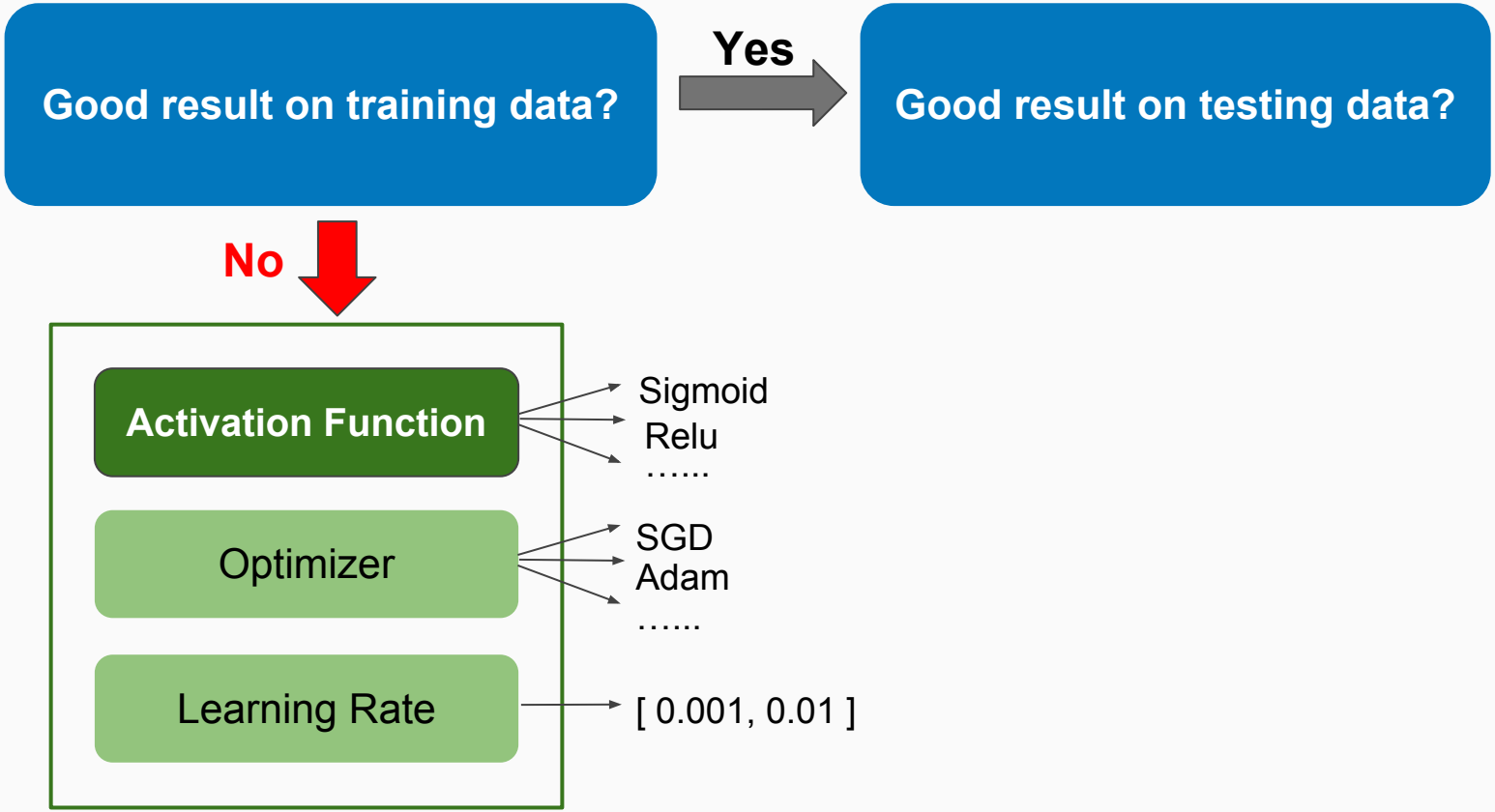
Topic : Fashion MNIST classification

Start coding !

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Import dataset ➔ Data exploration ➔ Data preprocessing ➔ Train a classification model
Duration	~ 15 mins

Tips for Deep Learning

Tips for Deep Learning

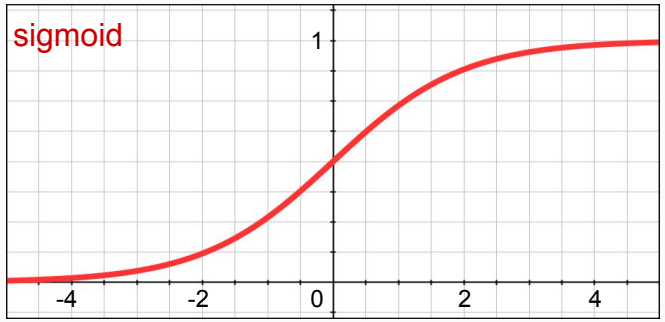


CloudMile

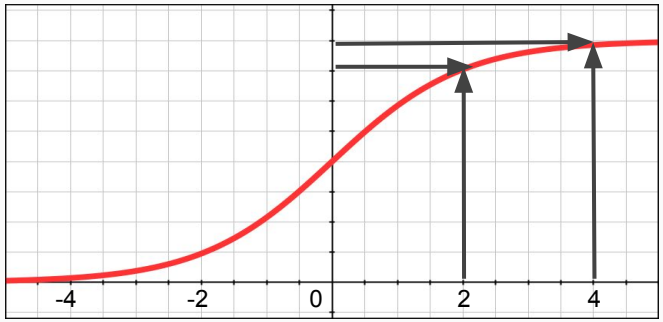
Activation Function

Sigmoid

WX + B



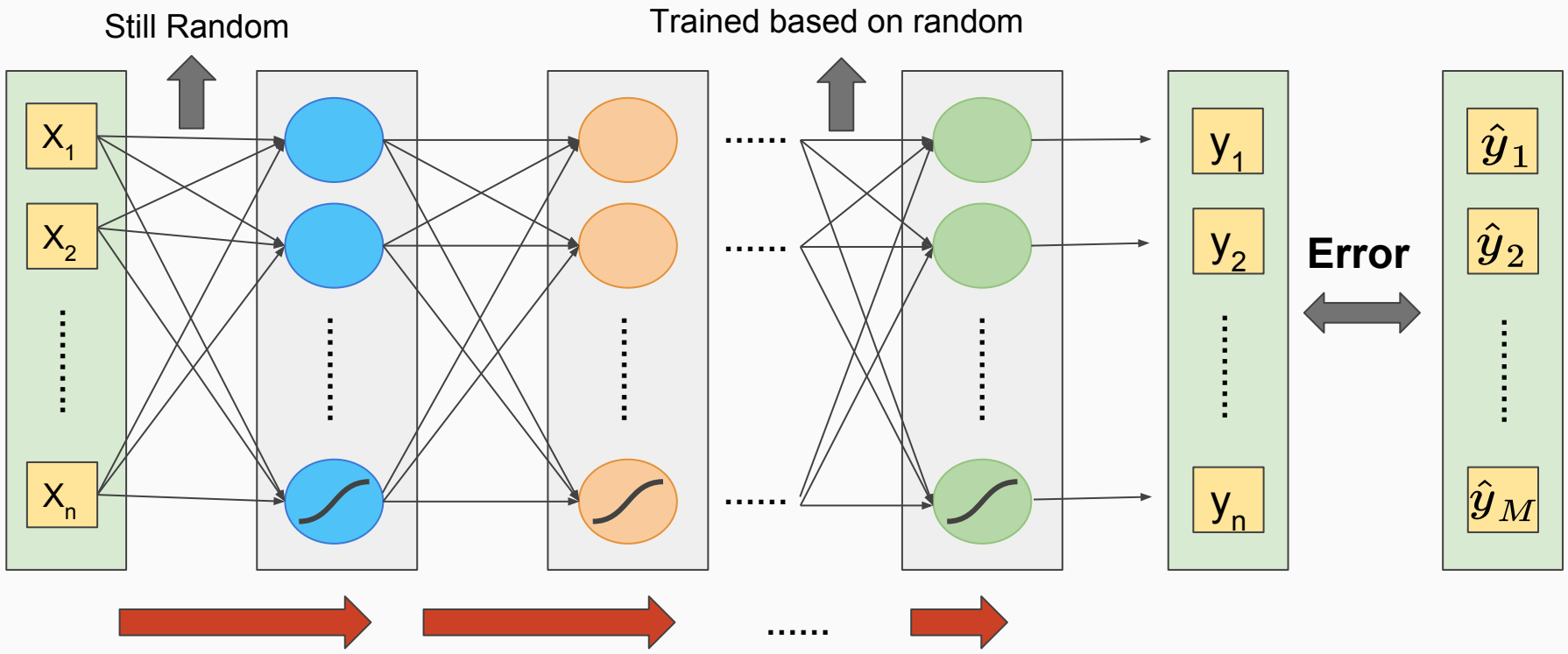
Small output difference



Large input difference

CloudMile

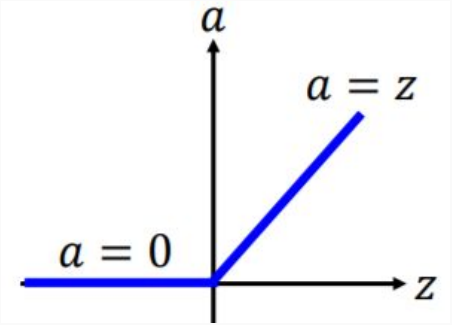
Activation Function



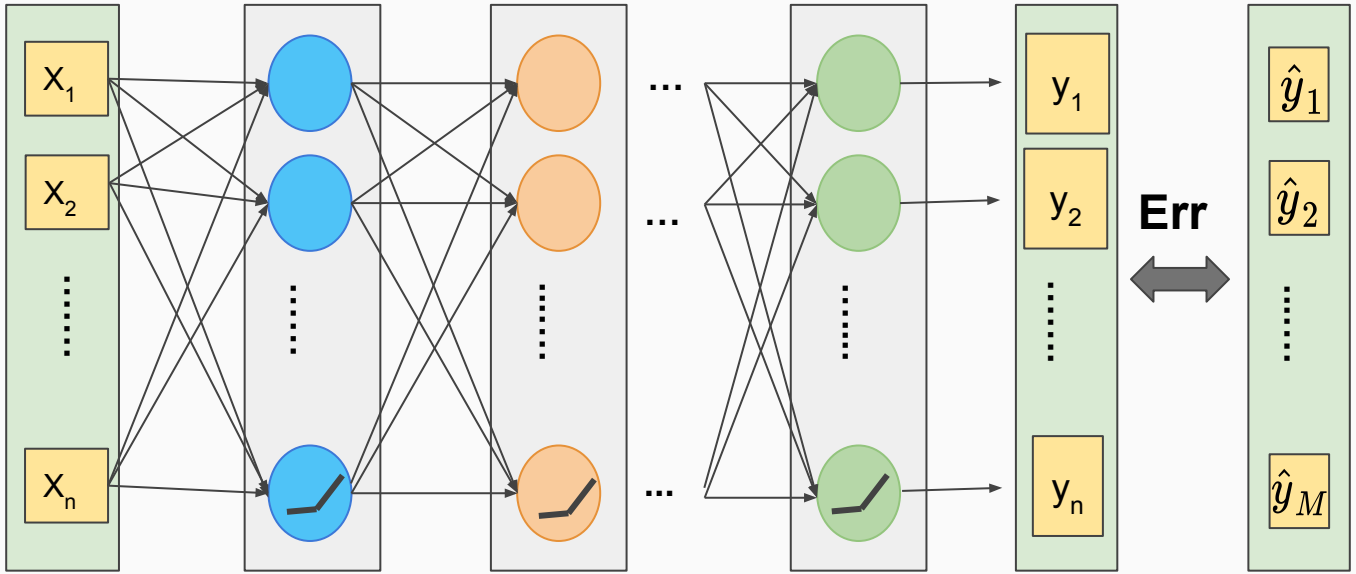
CloudMile

ReLU

ReLU: Rectified Linear Unit



$$RELU(x) = \begin{cases} 0 \ if \ x < 0 \\ x \ if \ x >= 0 \end{cases}$$



CloudMile

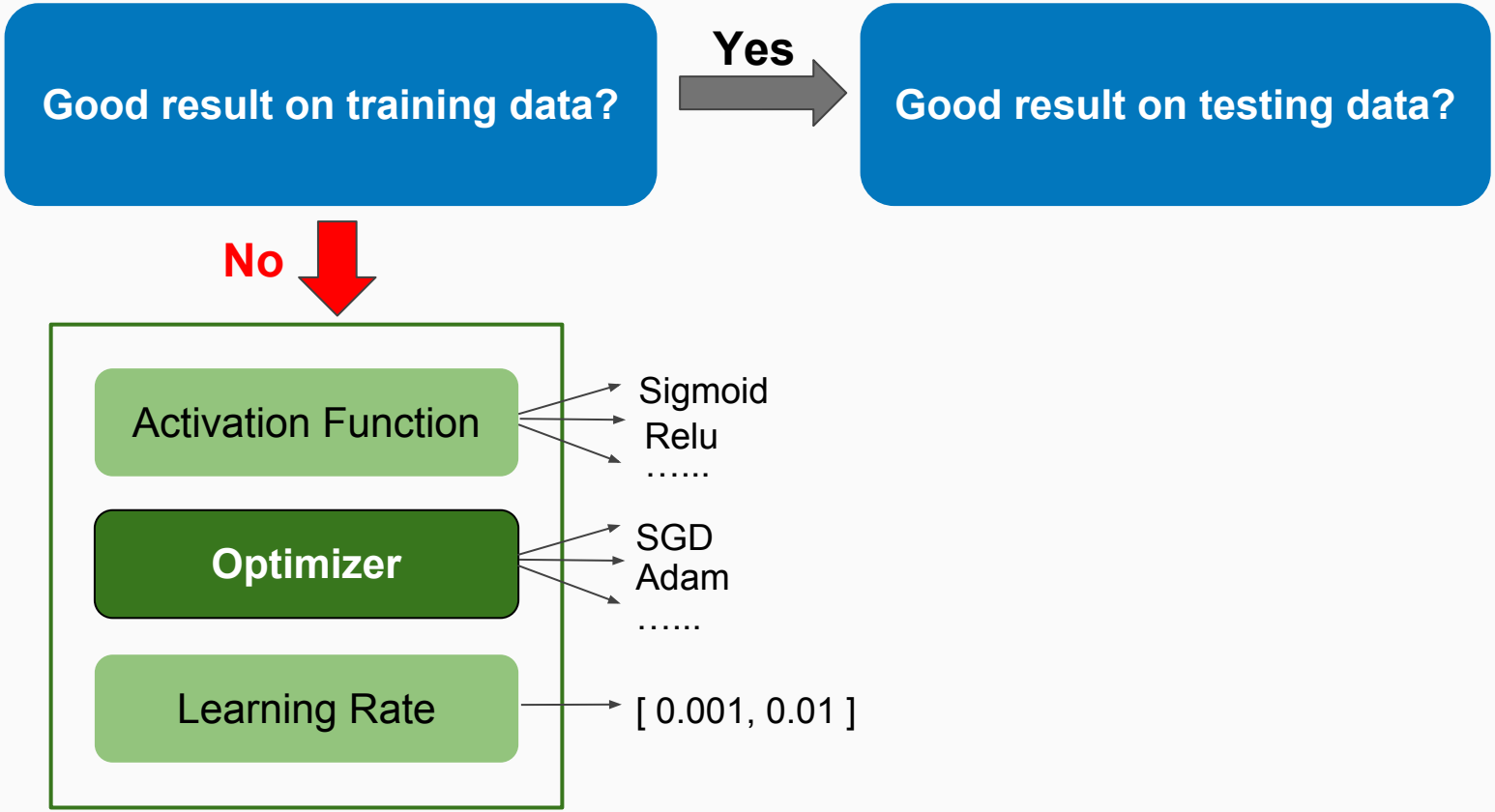
Lab 2-b

Topic : Fashion MNIST classification with RELU

Start coding !

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Change activation function to RELU ➔ Train a classification model
Duration	~ 10 mins

Tips for Deep Learning



CloudMile

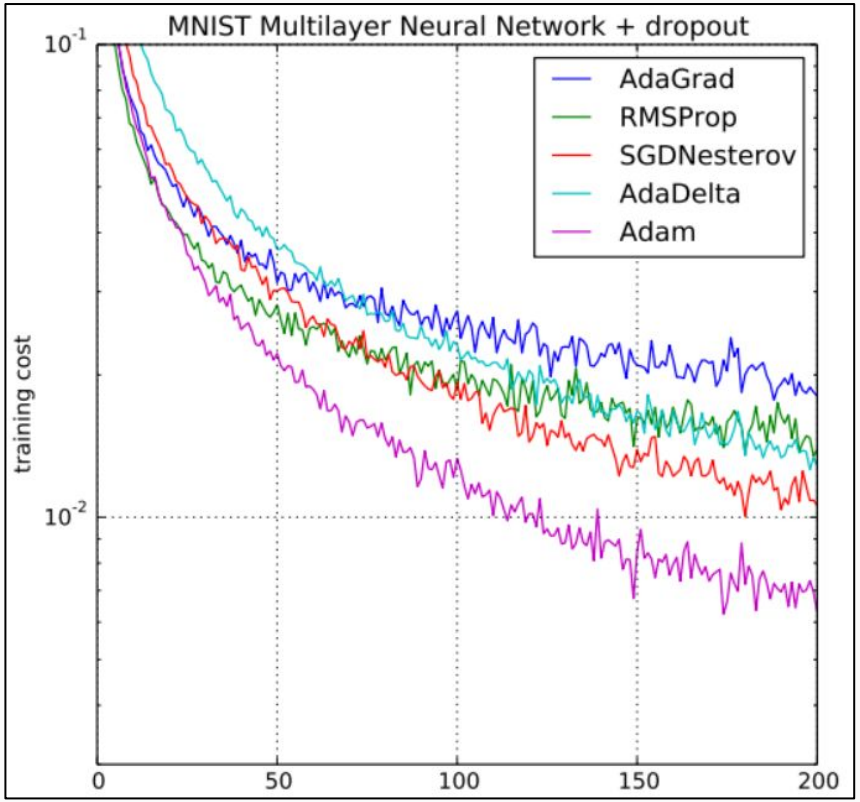
Optimizer

Techniques

- Adaptive learning rate
- Momentum

Optimizer

- Vanilla Gradient Descent
- Adagrad
- RMSprop
- Adam



CloudMile

Vanilla Gradient Descent

- Reduce the learning rate by some factor every few epochs.
- At the beginning, we are far from the destination, so we use larger learning rate.
- After several epochs, we are close to the destination, so we reduce the learning rate.

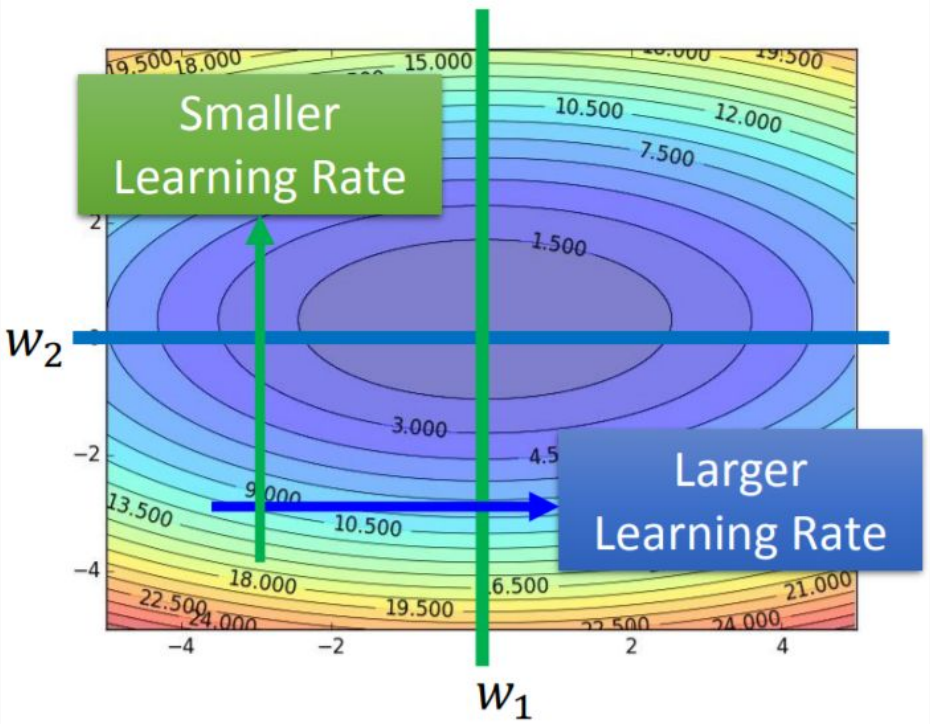
E.g. 1/t decay: $\eta^t = \eta/\sqrt{t+1}$

$$w^{t+1} \leftarrow w^t - \eta^t g^t$$

Adagrad

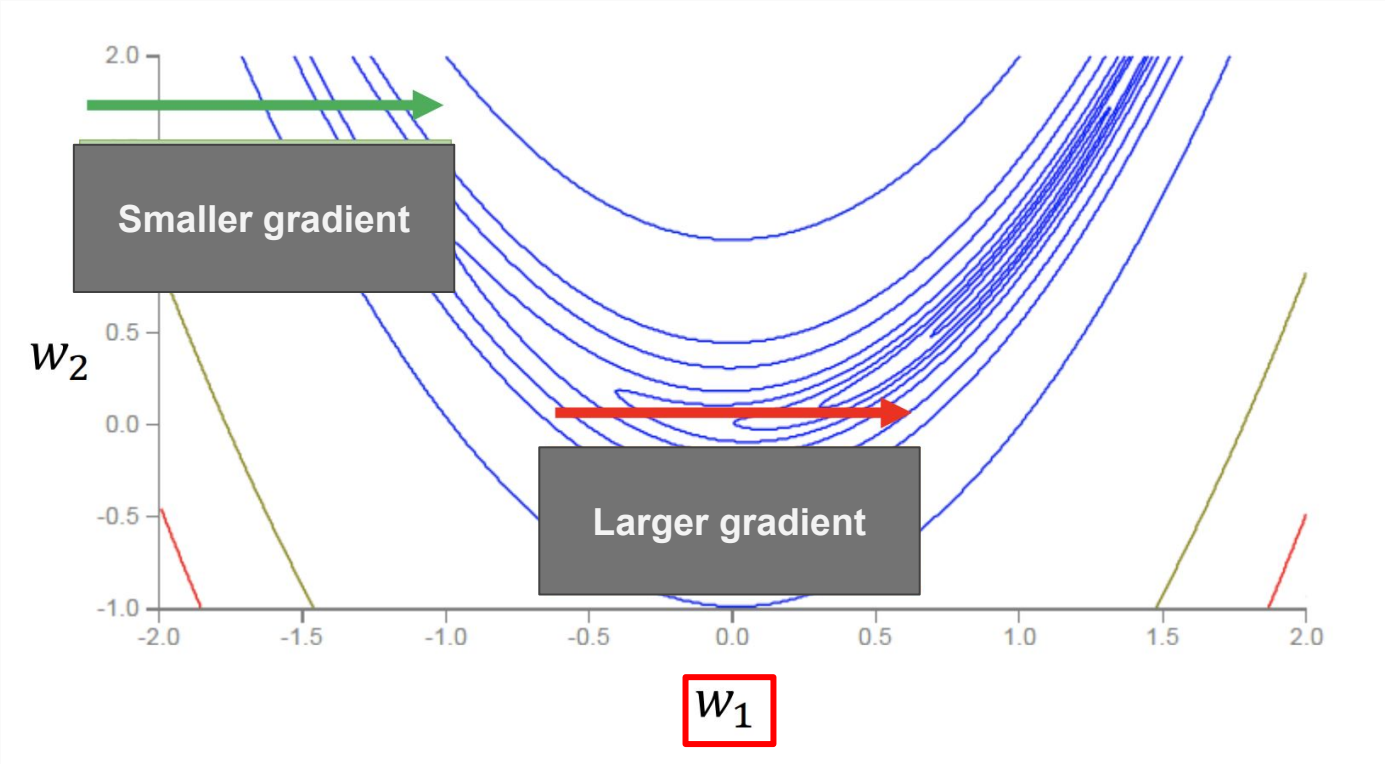
$$\eta^t = \frac{\eta}{\sqrt{t+1}} \qquad \sigma^t = \sqrt{\frac{1}{t+1}\sum_{i=0}^{t}(g^i)^2}$$

$$w^{t+1} \leftarrow w^t - \frac{\eta^t}{\sigma^t} g^t$$



Adaptive Learning Rate

RMSProp



Adaptive Learning Rate

RMSProp

$$w^{t+1} = w^t - \frac{\eta}{\sigma^t} g^t$$

$$\sigma^t = \sqrt{\alpha(\sigma^{t-1})^2 + (1-\alpha)(g^t)^2}$$

$$w^1 \leftarrow w^0 - \frac{\eta}{\sigma^0} g^0 \qquad \sigma^0 = g^0$$

$$w^2 \leftarrow w^1 - \frac{\eta}{\sigma^1} g^1 \qquad \sigma^1 = \sqrt{\alpha(\sigma^0)^2 + (1-\alpha)(g^1)^2}$$

$$w^3 \leftarrow w^2 - \frac{\eta}{\sigma^2} g^2 \qquad \sigma^2 = \sqrt{\alpha(\sigma^1)^2 + (1-\alpha)(g^2)^2}$$

Including g0, g1 Including g2

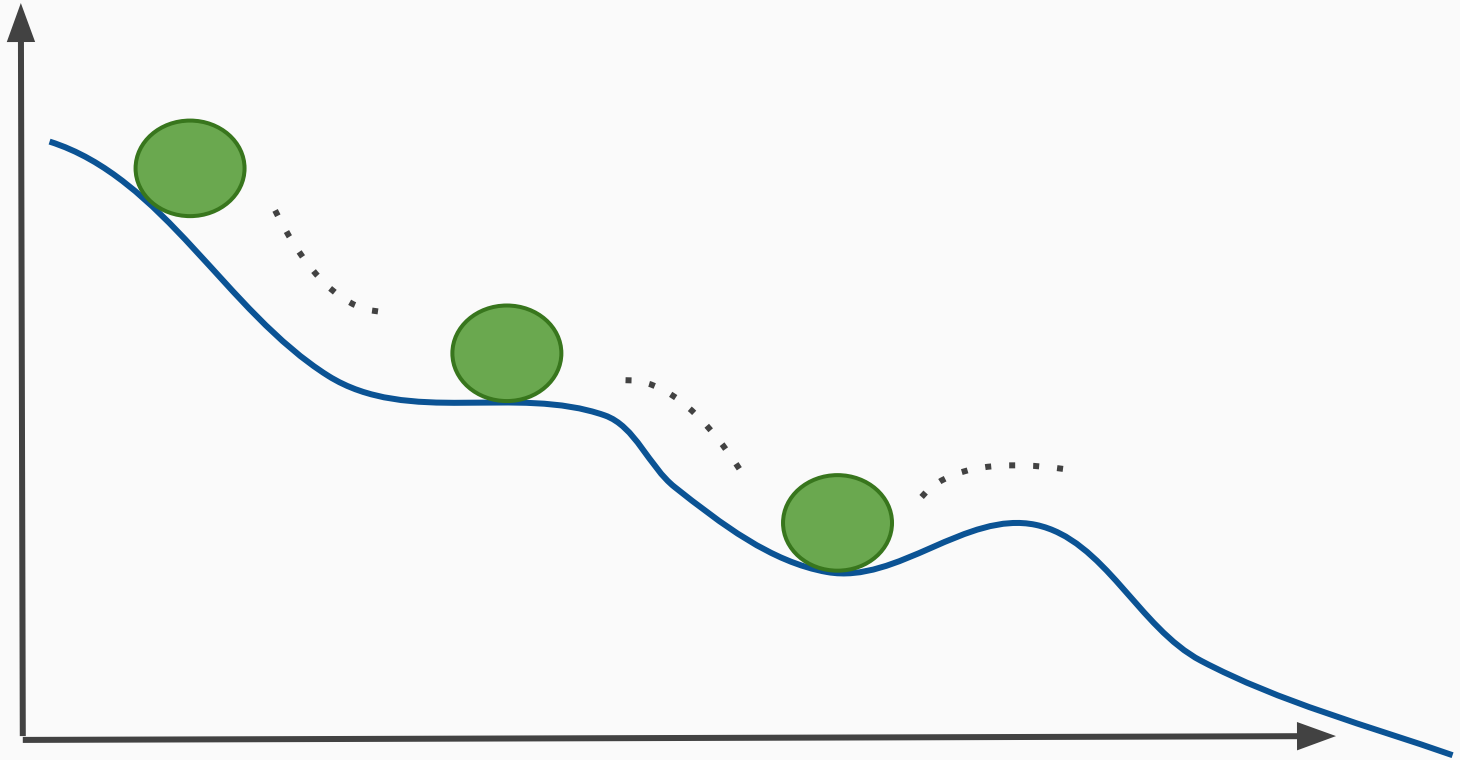
$$\vdots$$

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sigma^t} g^t \qquad \sigma^t = \sqrt{\alpha(\sigma^{t-1})^2 + (1-\alpha)(g^t)^2}$$

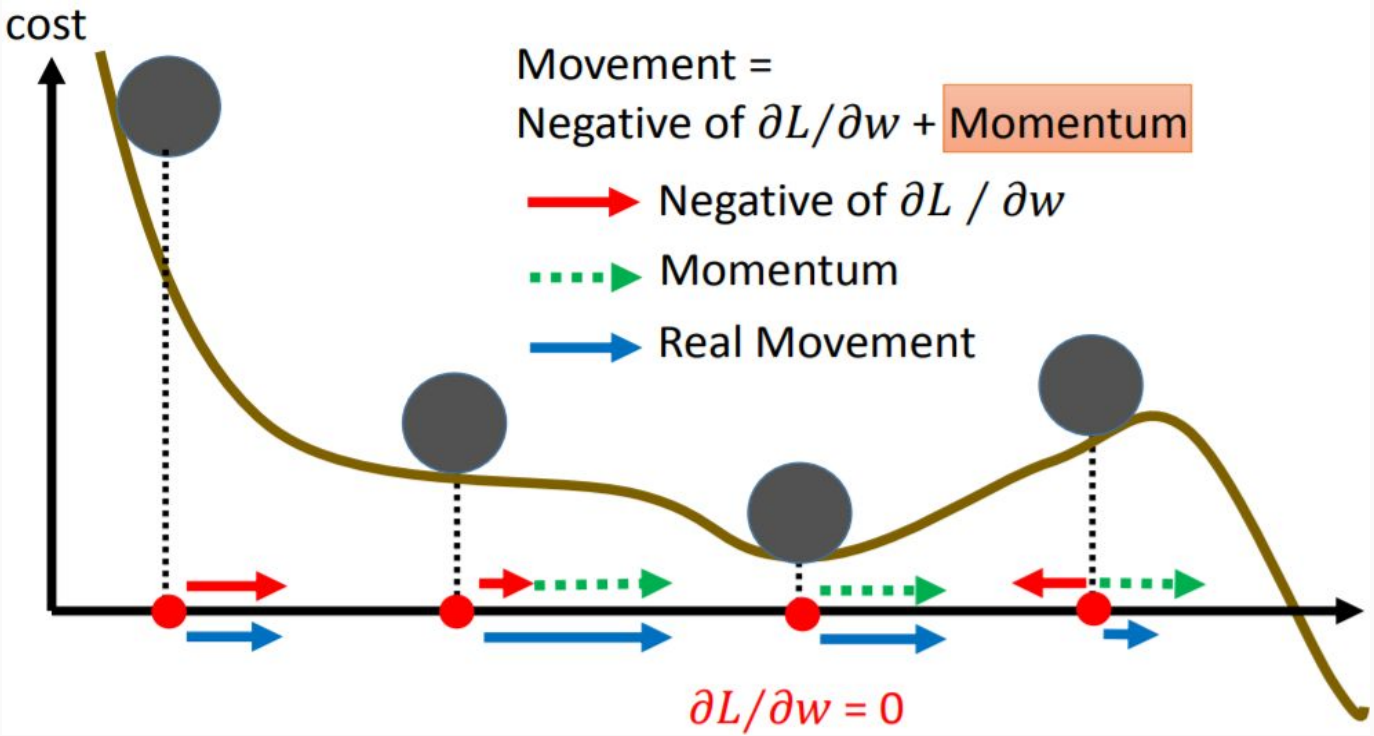
Choose your own α

- **Small α: Depend more on current gradient**
- **Large α: Depend more on previous gradients**

Momentum



Momentum



Optimizers

RMSProp $w^{t+1} = w^t - \frac{\eta}{\sigma^t} g^t$ $\sigma^t = \sqrt{\alpha(\sigma^{t-1})^2 + (1-\alpha)(g^t)^2}$

Adam

- Include the advantages from RMSProp and Momentum
- Try with Adam first !

RMSProp + Momentum

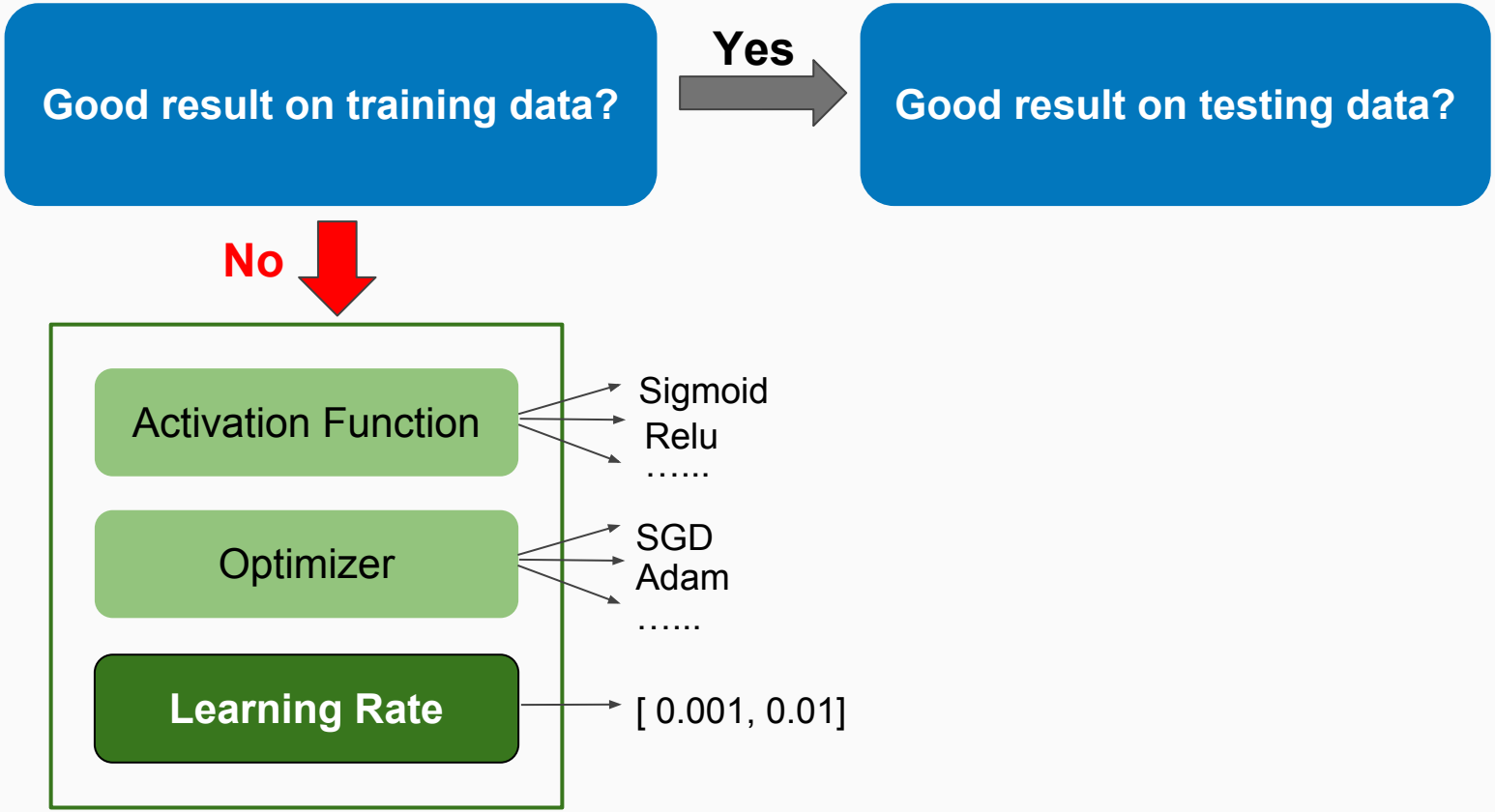
Lab 2-c

Start coding !

Topic : Fashion MNIST classification with ReLU + Adam

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Change optimizer to Adam ➔ Train a classification model
Duration	10 mins

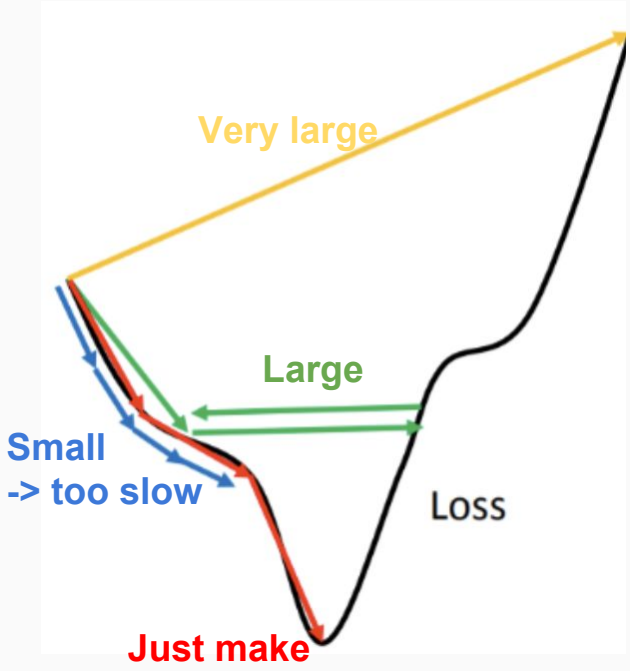
Tips for Deep Learning

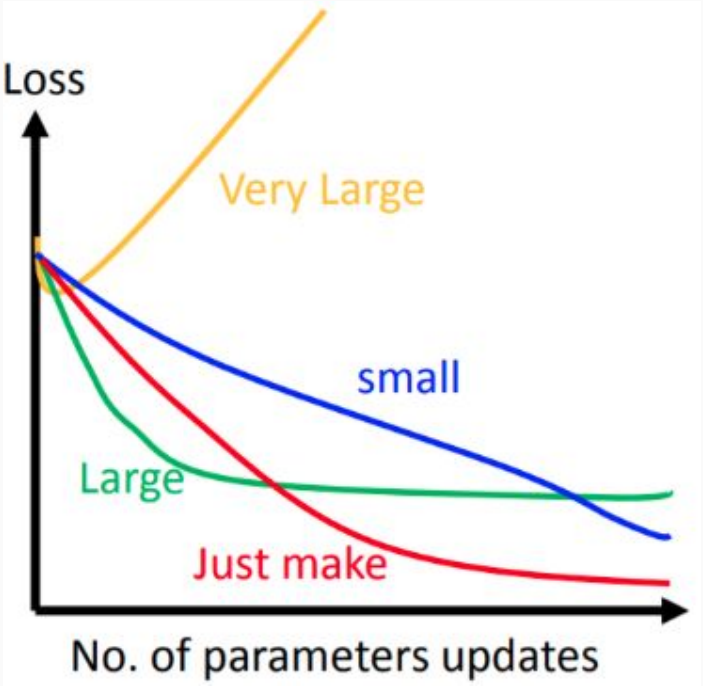


Learning rate

- Set learning rate carefully

$$\theta^i = \theta^{i-1} - \eta \nabla L\left(\theta^{i-1}\right)$$





CloudMile

Lab 2-d

Topic : Adjust learning rate in Fashion MNIST classification

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Change learning rate from 0.001 to 0.01 ➔ Train a classification model
Duration	~ 10 mins

Start coding !

Lab 2-e

Start coding !

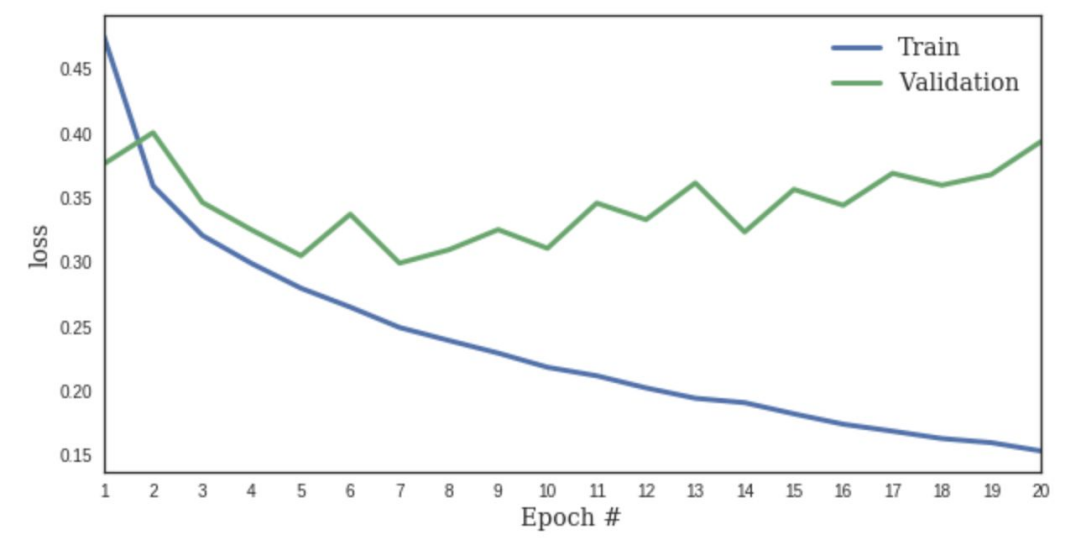
Topic : Change number of hidden layers neurons in Fashion MNIST classification

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Change hidden layers neurons to [2048 , 1024] ➔ Train a classification model
Duration	~ 10 mins

Lab 2-e result

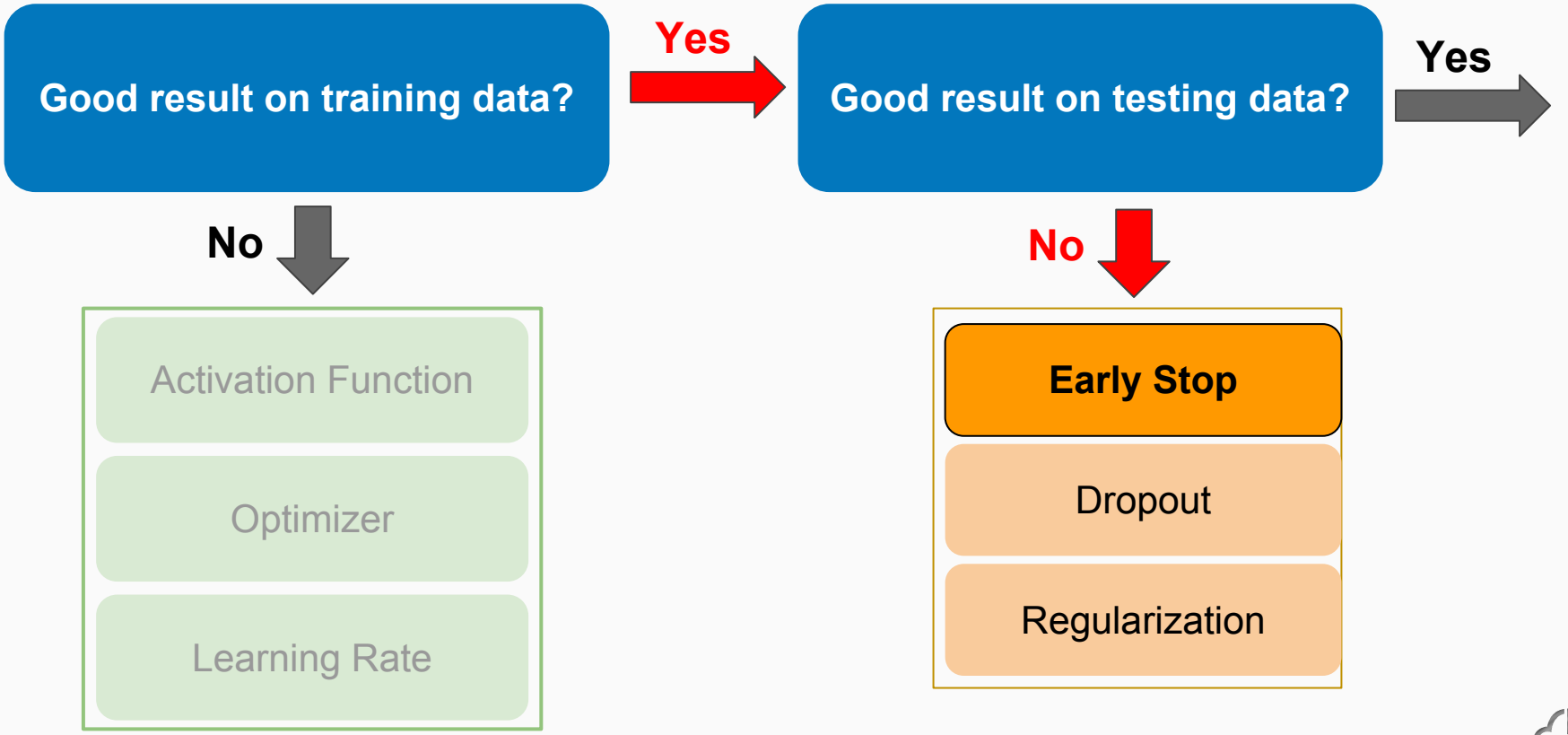
- Training loss is **decreasing**, however, validation loss is **increasing** after a few epochs.
- Train too well → Learn the details and noise in the training data

  → Negatively impacts the model performance on new data

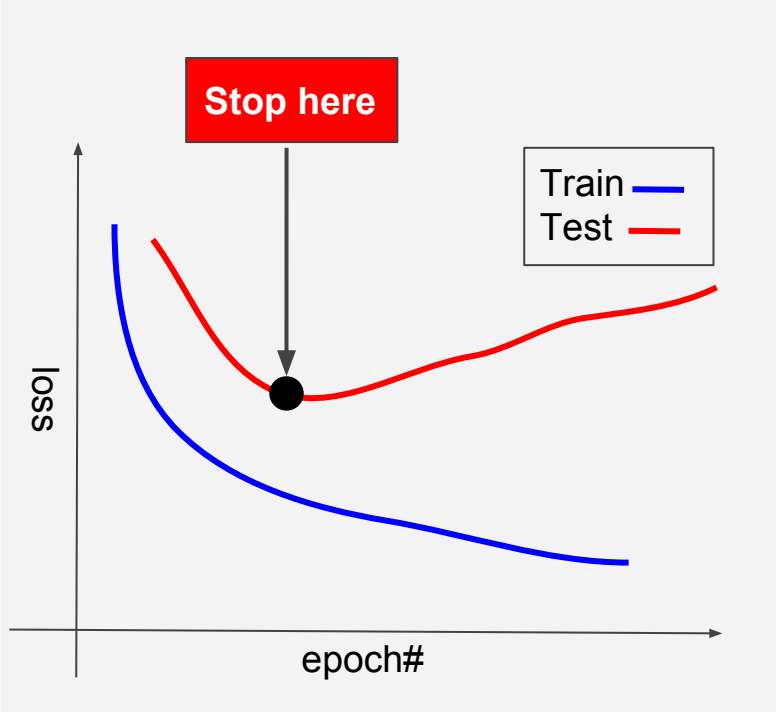


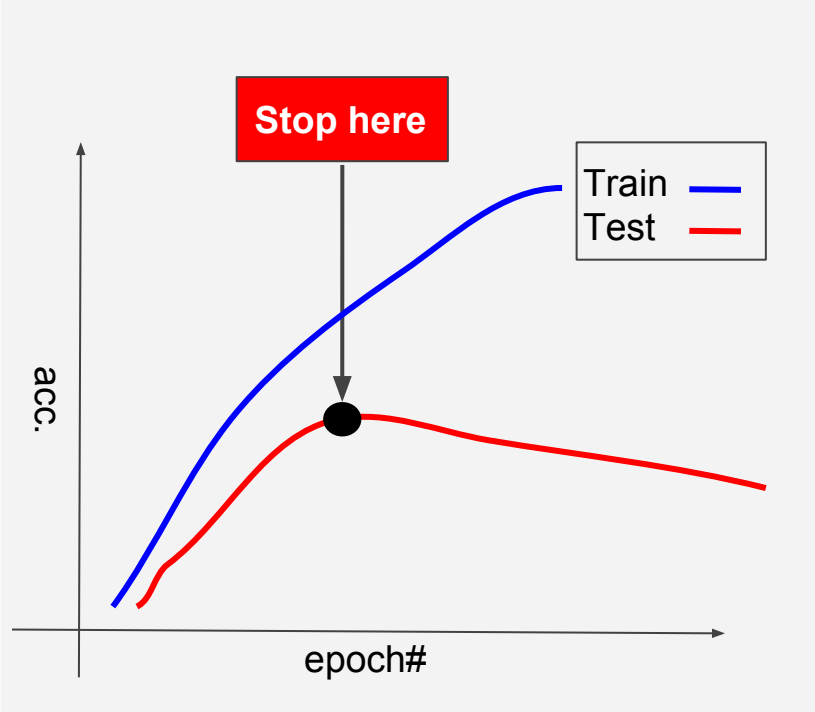
→ **May be overfitting**

Tips for Deep Learning



Early stopping





Keras - early stopping

```
keras.callbacks.EarlyStopping(monitor='val_loss', patience=3)
```

monitor: quantity to be monitored.

patience: number of epochs with no improvement after which training will be stopped.

Example Code

```
EarlyStopping = keras.callbacks.EarlyStopping(
                        monitor='val_loss',
                        patience=5)

callbacks_list = [EarlyStopping]

model_his = model.fit(train_features,
                      train_labels,
                      nb_epoch=100,
                      callbacks=callbacks_list)
```

Lab 2-f

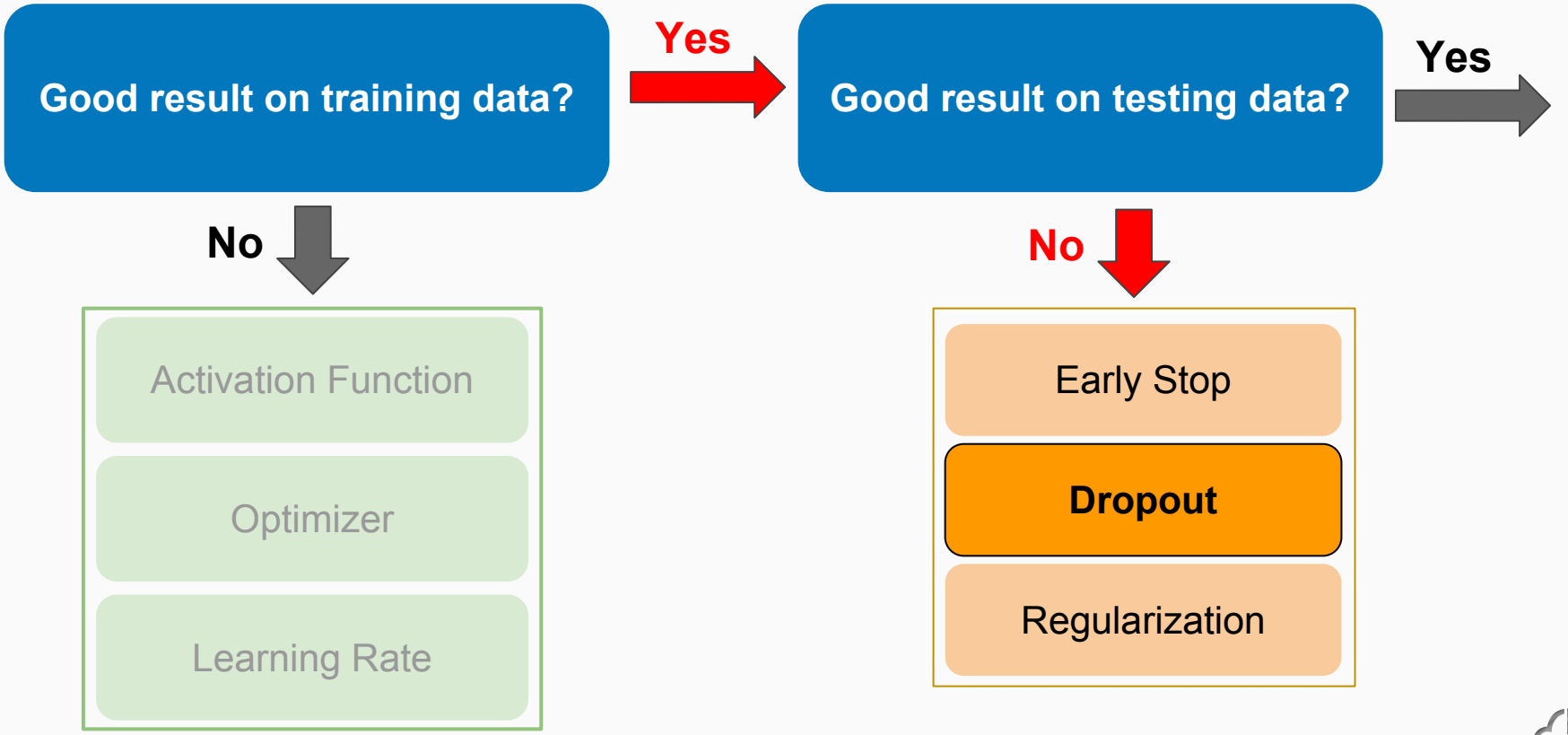
Topic : Early stop to avoid overfitting

Start coding !

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Create early stopping callback to avoid overfitting ➔ Train a classifier
Duration	~ 10 mins

Another solution: **save best model only**

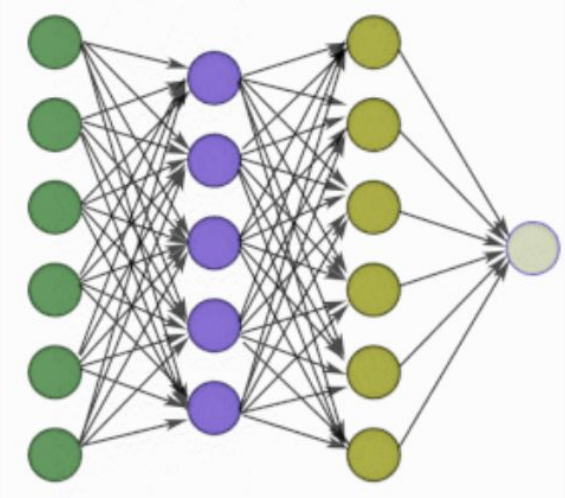
Tips for Deep Learning



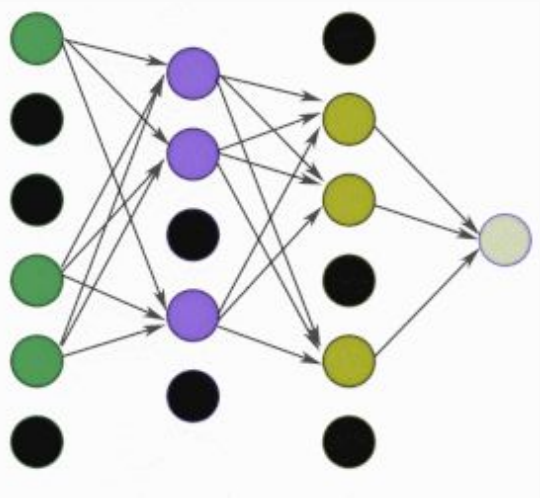
CloudMile

Dropout

Standard Neural Network

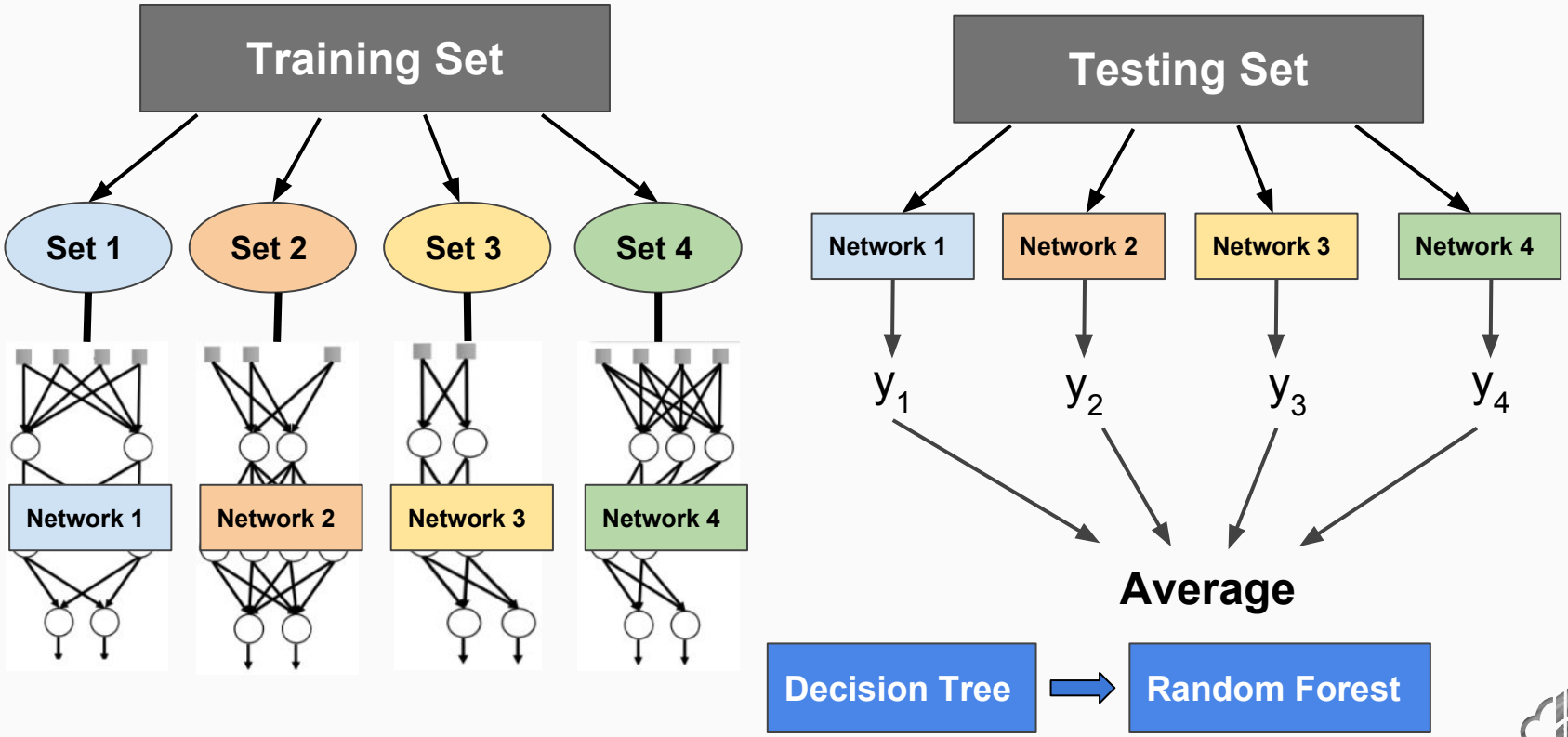


After applying dropout



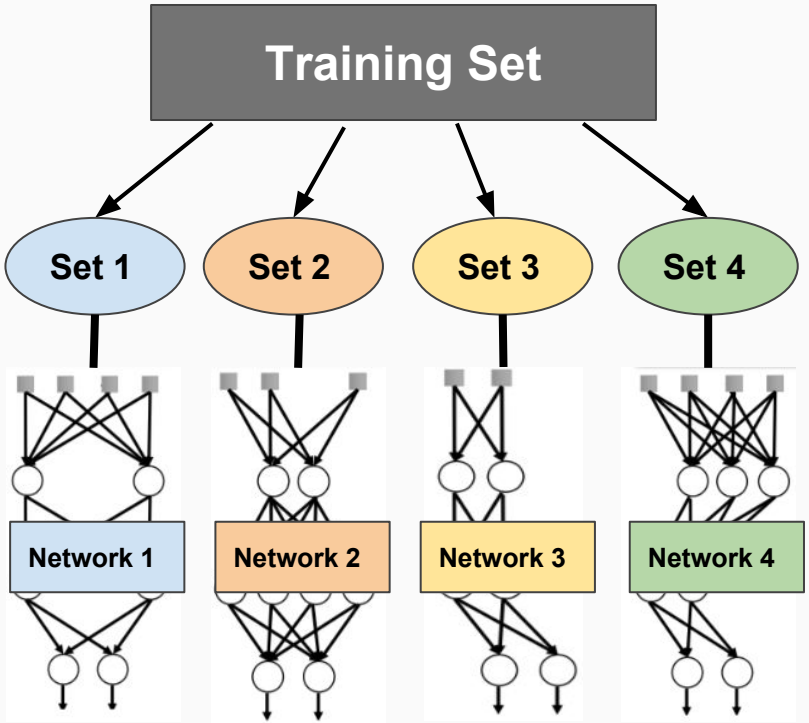
- When training, drop p% neurons in the batch.
- For each mini batch, we resample the dropout neurons.
- Dropout always work on training time, not on evaluate or prediction.

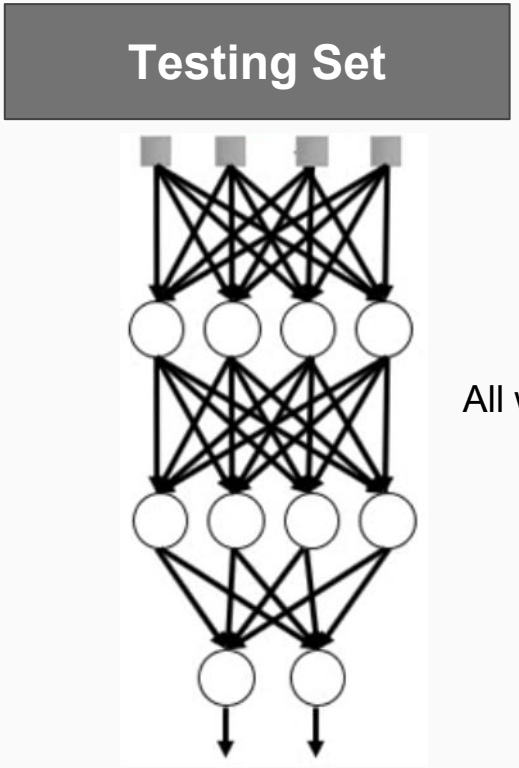
Dropout



CloudMile

Dropout





All weights multiply (1-p) %

Dropout - Keras

```
keras.layers.Dropout(rate)
```

rate: float between 0 and 1. Fraction of the input units to drop.

Example Code

```
model = Sequential()
model.add(Dense(60, input_dim=60, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(30, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(10, activation='softmax'))
```

rate

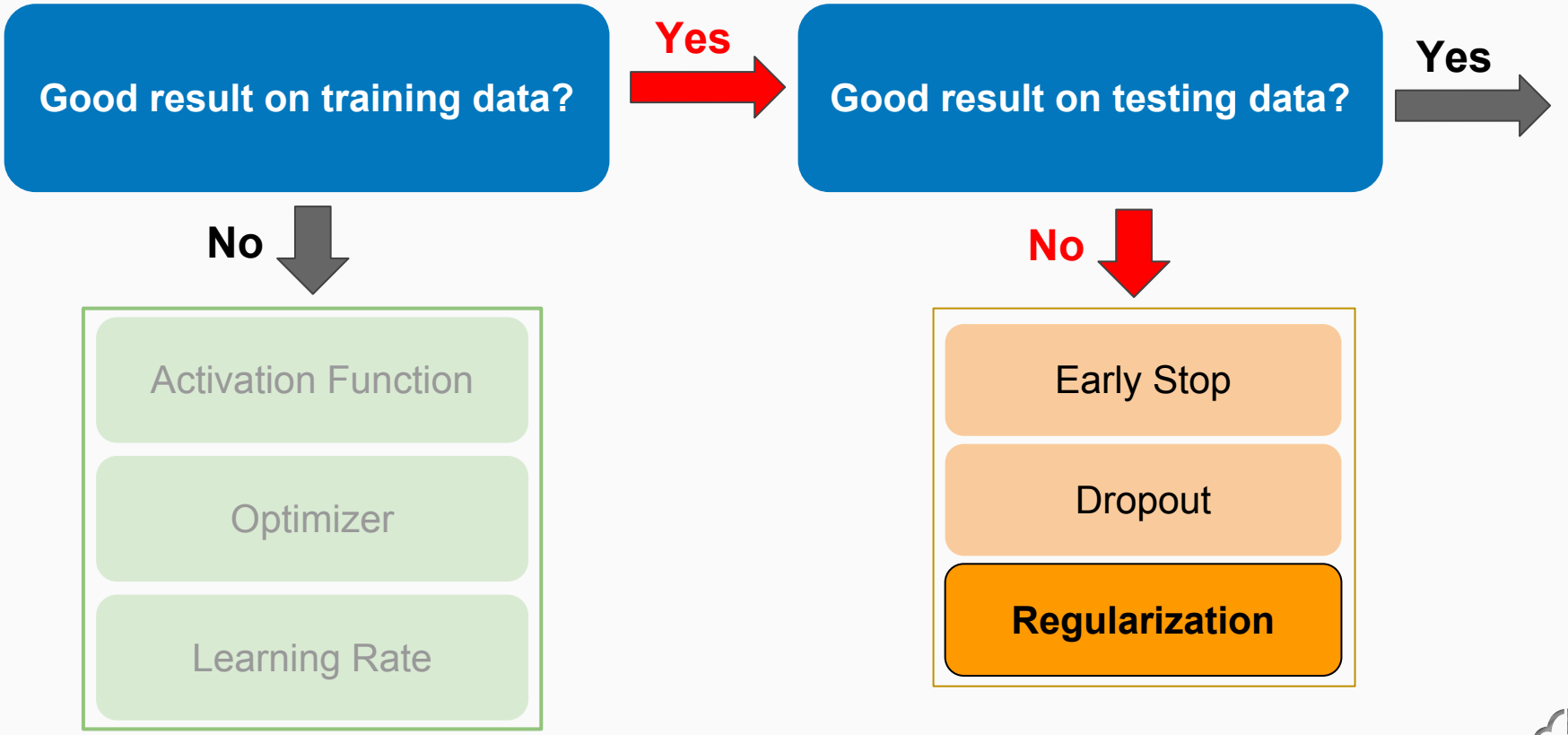
Lab 2-g

Start coding !

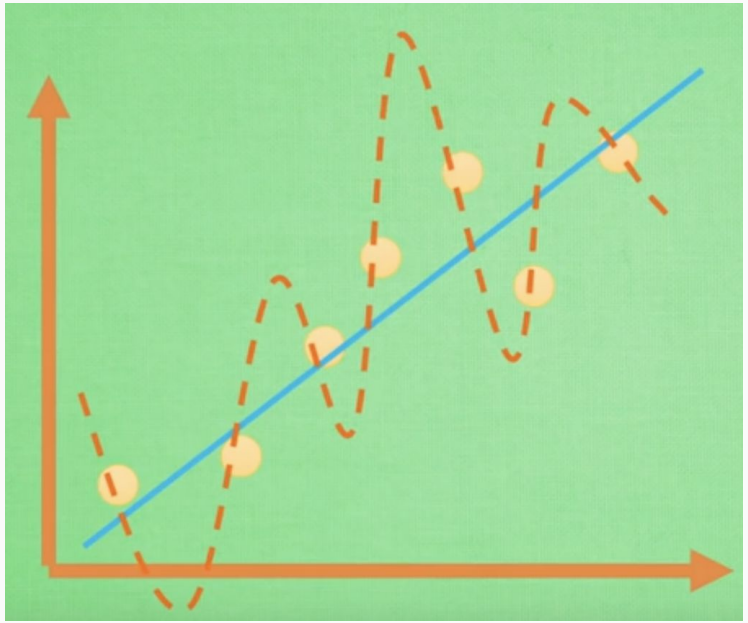
Topic : Use dropout to avoid overfitting

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Create dropout in hidden layers ➔ Train a classifier
Duration	~ 10 mins

Tips for Deep Learning



Regularization



$$y = a + bx + cx^2 + dx^3$$

$$y = a + bx$$

Regularization

- Add Reguarizer into loss

$$L'(\theta) = L(\theta) + \lambda(regularizer)$$

- λ is a parameter (usually below 0.01)
- L1 and L2

$$L_1 = \sum_i |w_i|$$

L1 norm: sum of absolute values

$$L_2 = \sum_i |w_i|^2$$

L2 norm: square sum of absolute values

Discussion

Application

1. Healthcare (Disease Identification , Alert and diagnostics from real-time patient data ….)
2. Manufacturing(Defection analysis, Demand forecasting, Condition monitoring ….)
3. Environment (Insects Identification, Earthquake prediction ….)
4. Financial (Risk Analysis, Customer segmentation ….)
5. Traveling (Aircraft scheduling, Traffic Patterns and congestion management ….)

Use Case	Label	Input(s)	Classification or regression

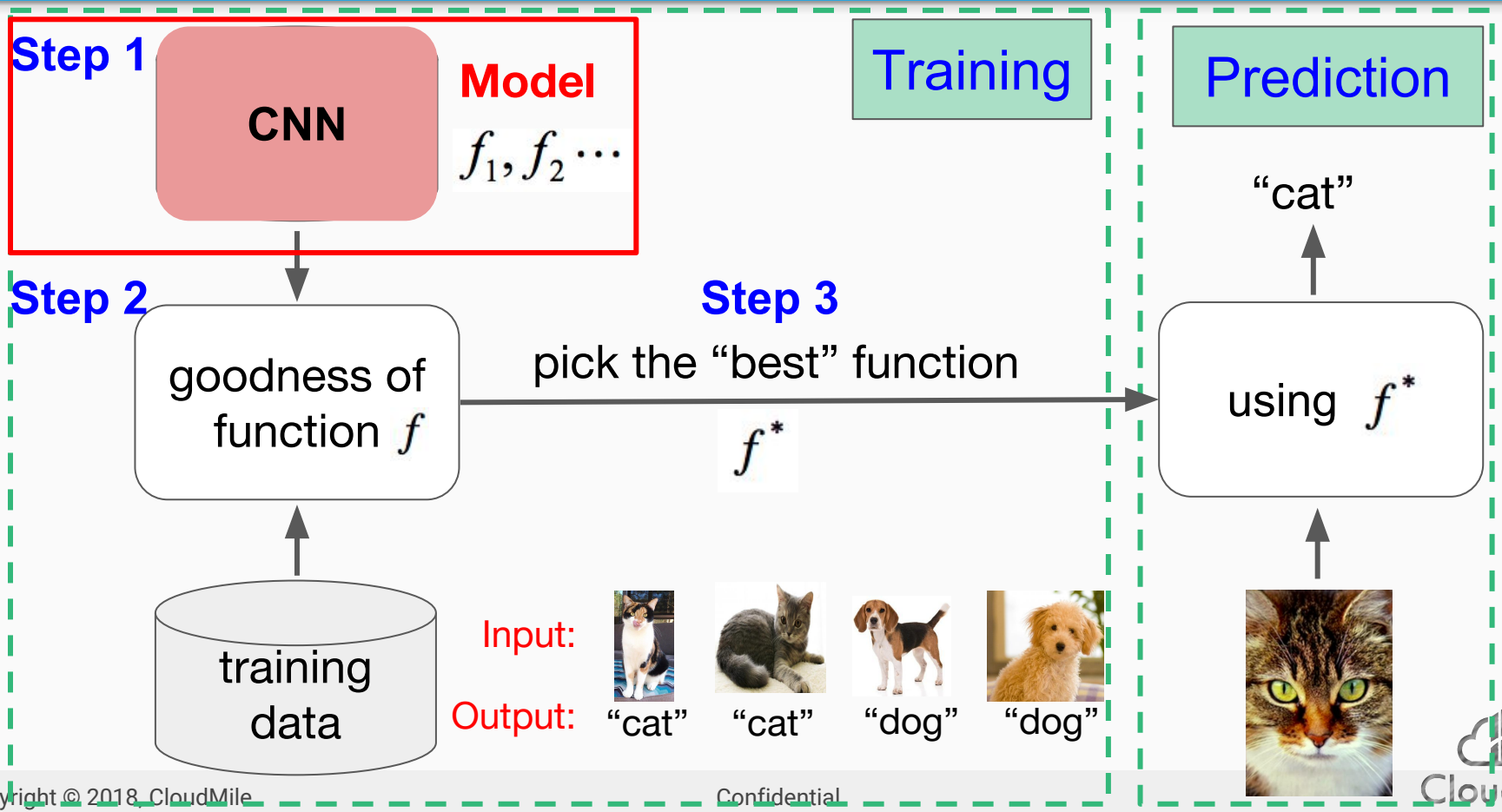
Variants of Neural Networks

Variants of Neural Networks

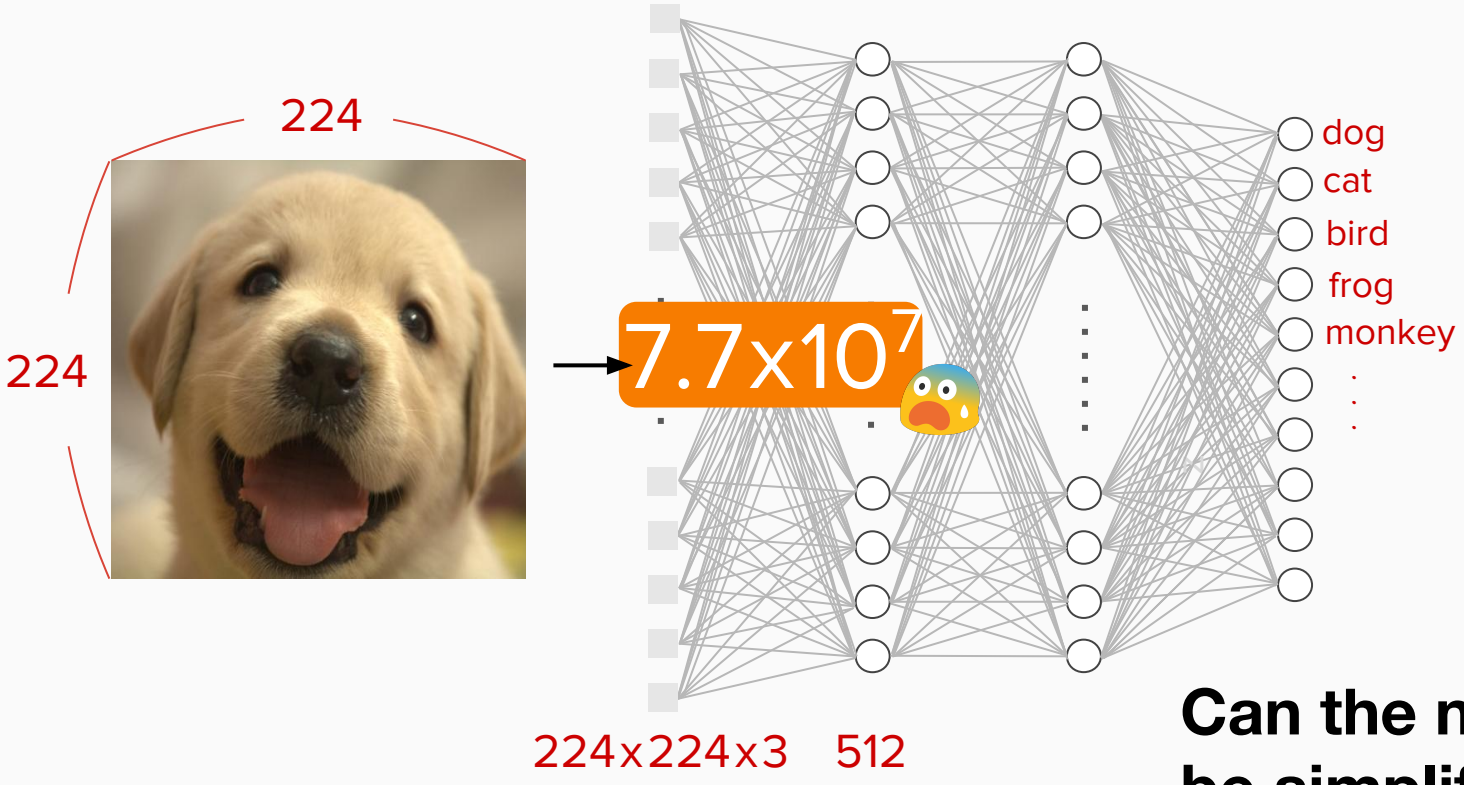
- Convolutional neural networks (CNN)
  → good for **images**

- Recurrent neural networks (RNN)
  → good for **sequential** data

Machine Learning Framework



Why CNN for Images?



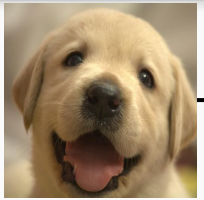
Can the network be simplified?

CloudMile

Properties of Image Recognition

- **Locality**: objects tend to have a local spatial support.
- **Translation invariance**: object appearance is independent of location.
- **Scale invariance**: subsampling the pixels will not change the object.

Whole CNN



Convolution

locality
translation invariance

Max Pooling

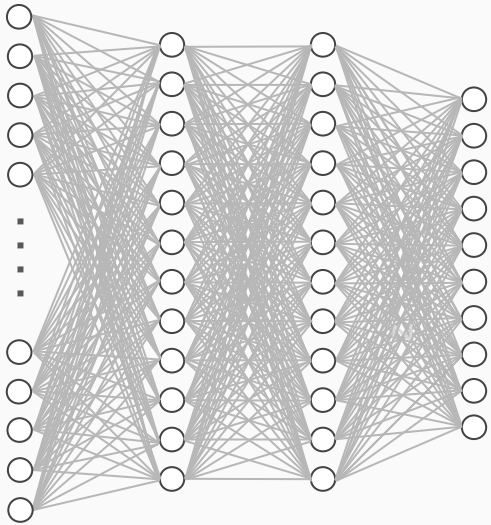
scale invariance

can repeat
many times

Convolution

Max Pooling

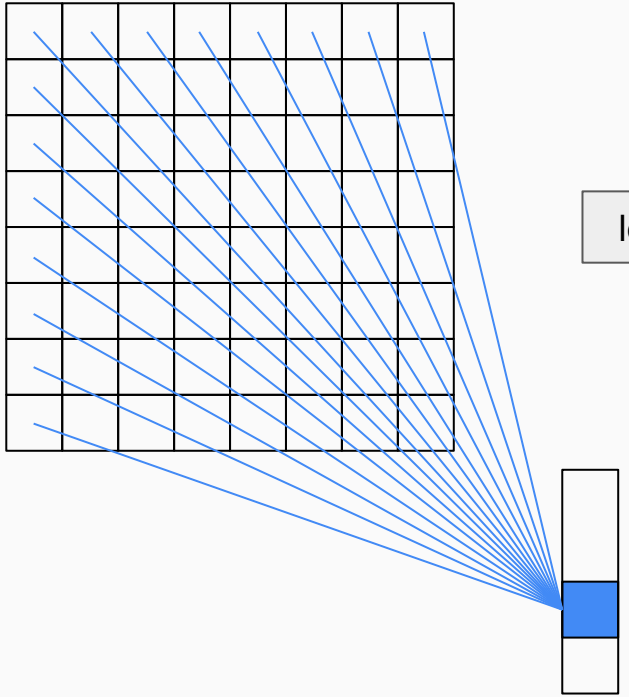
Flatten



CloudMile

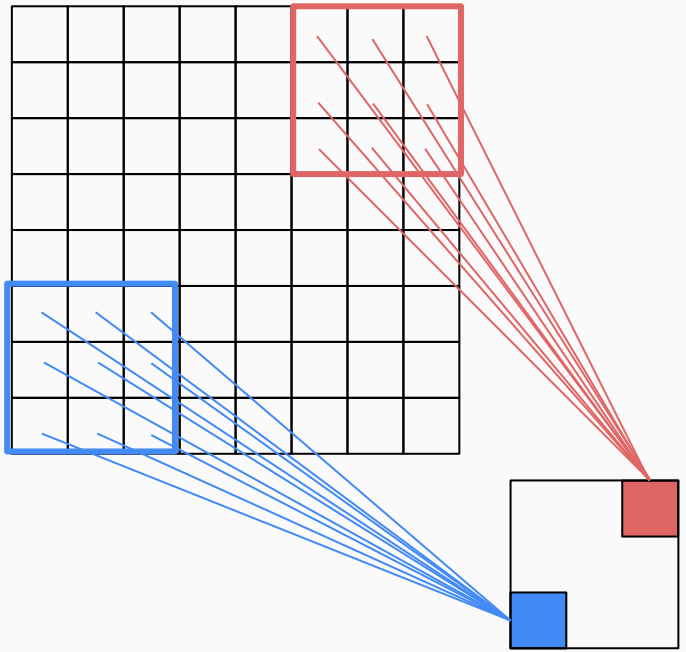
Incorporate Properties: Locality

fully-connected units



locality

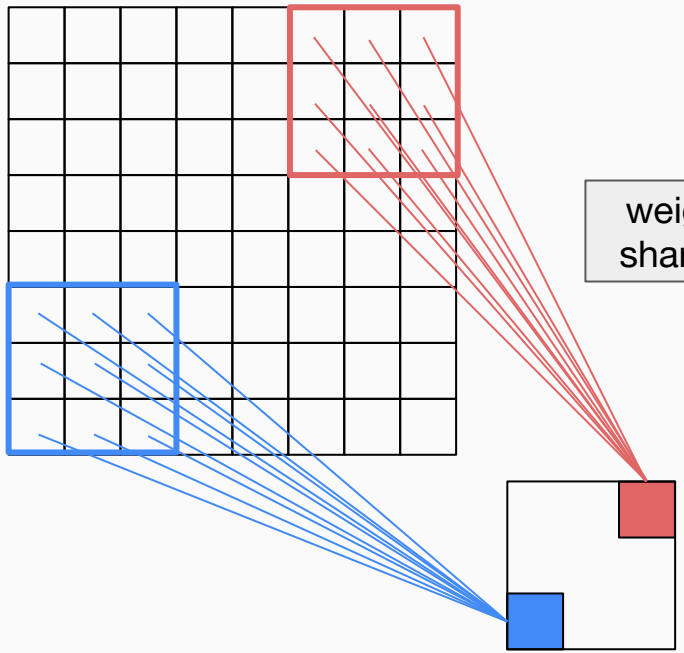
locally-connected units



less parameters!

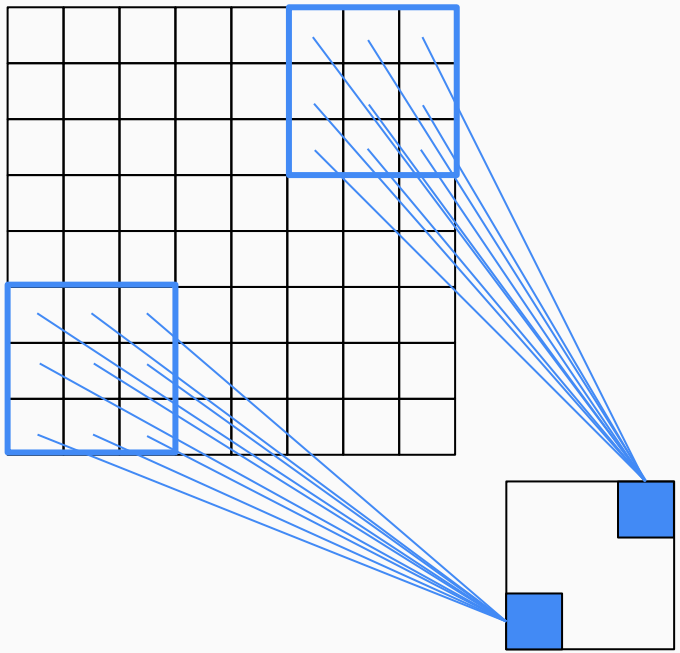
Incorporate Properties: Translation Invariance

locally-connected units



weight sharing

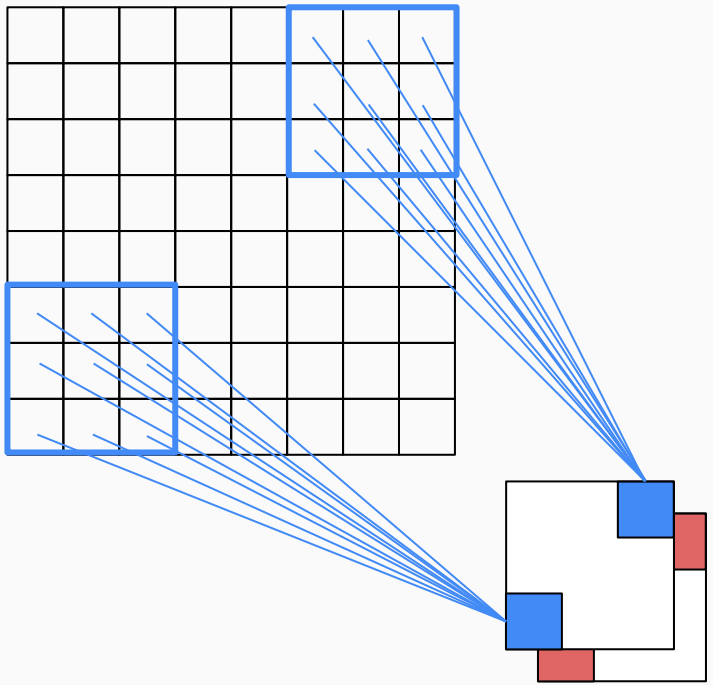
convolutional units



even less parameters!

Convolutions

weight matrix → filter (or kernel)



-1	1	-1
-1	1	-1
-1	1	-1

filter 1

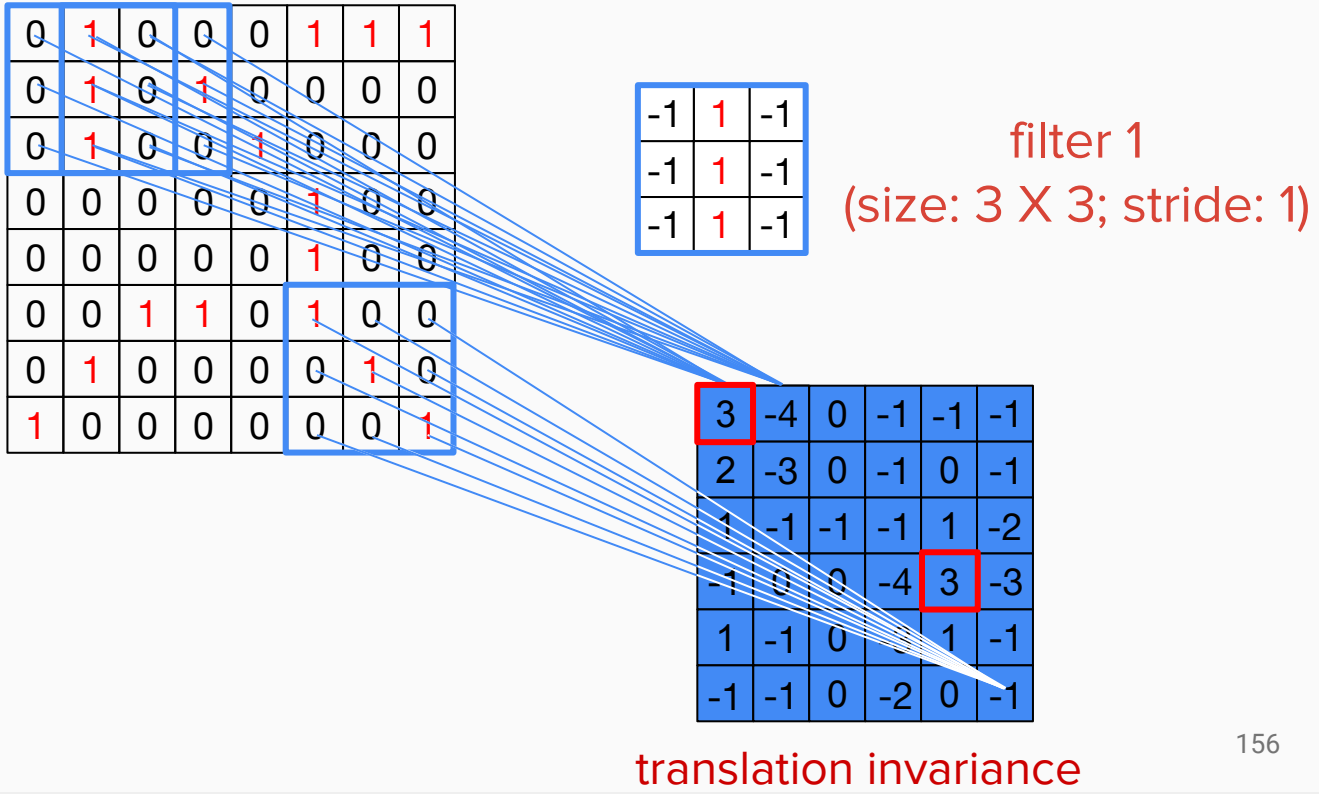
-1	-1	1
-1	1	-1
1	-1	-1

filter 2

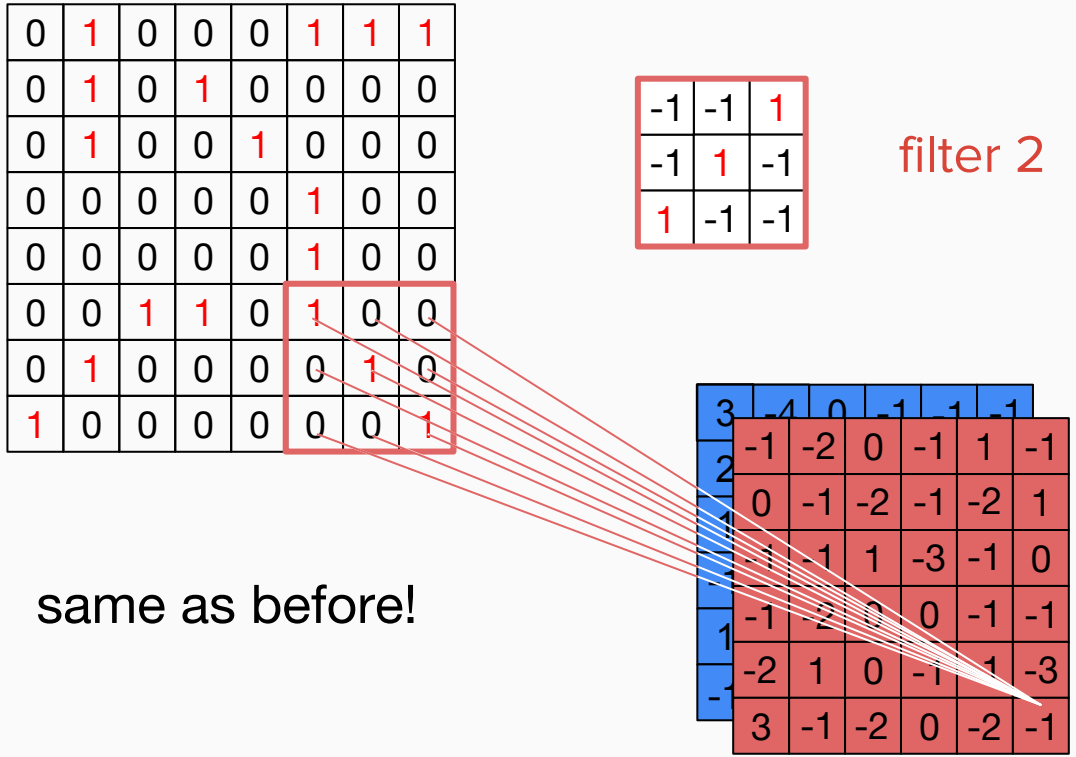
⋮

- Each filter extracts a small pattern.
- These are trainable parameters.

How to Compute the Output Feature?

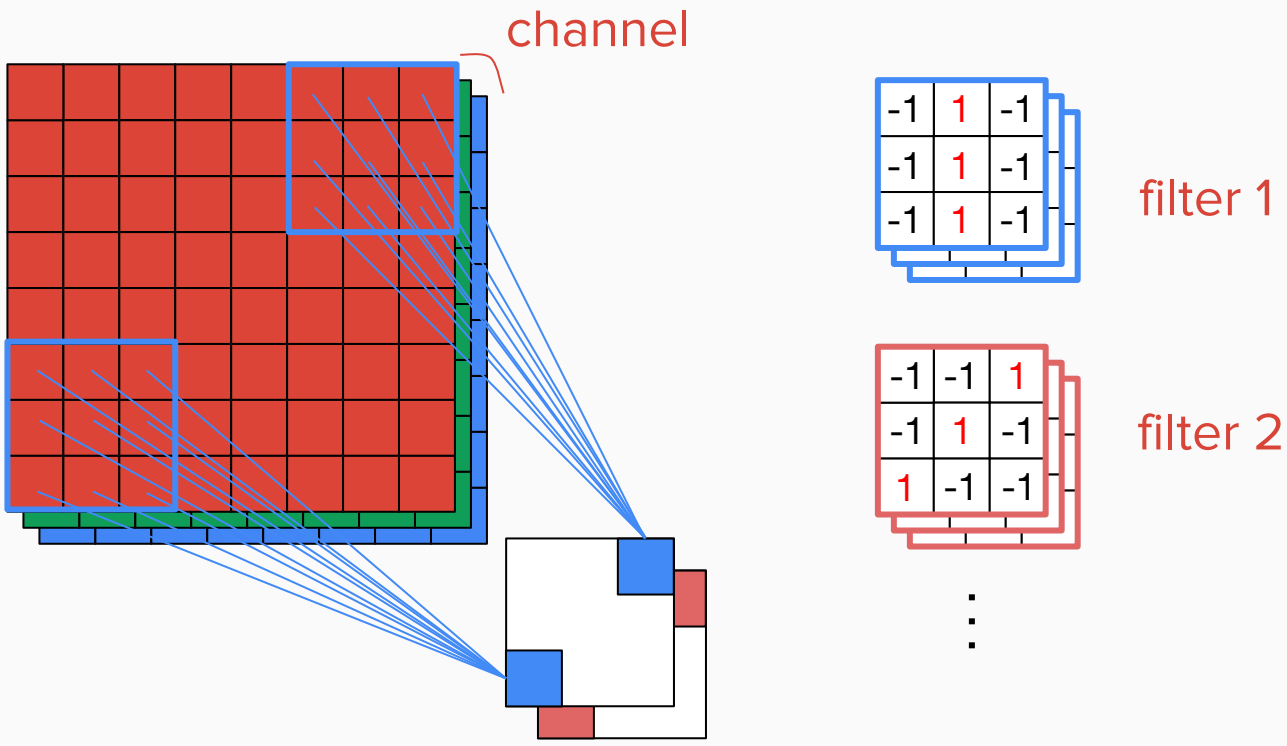


How to Compute the Output Features? (Cont'd)

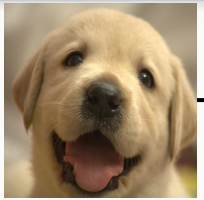


same as before!

Convolution for Colorful Images



Whole CNN



Convolution

locality
translation invariance

Max Pooling

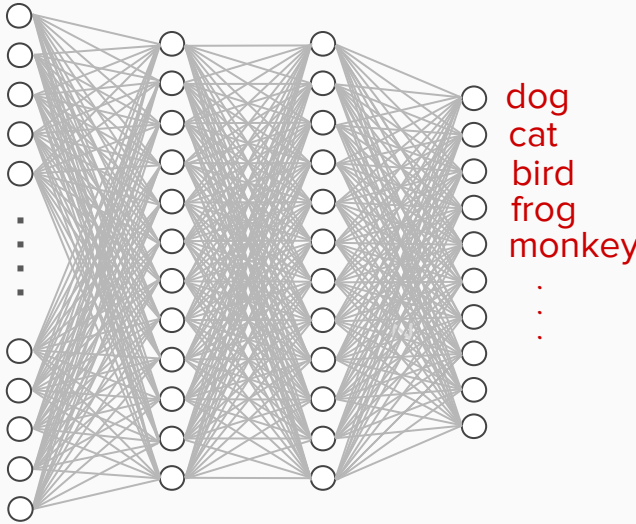
scale invariance

can repeat
many times

Convolution

Max Pooling

Flatten



CloudMile

Max Pooling

-1	1	-1
-1	1	-1
-1	1	-1

filter 1

-1	-1	1
-1	1	-1
1	-1	-1

filter 2

3	-4	0	-1	-1	-1
2	-3	0	-1	0	-1
1	-1	-1	-1	1	-2
-1	0	0	-4	3	-3
1	-1	0	-3	1	-1
-1	-1	0	-2	0	-1

Max Pooling

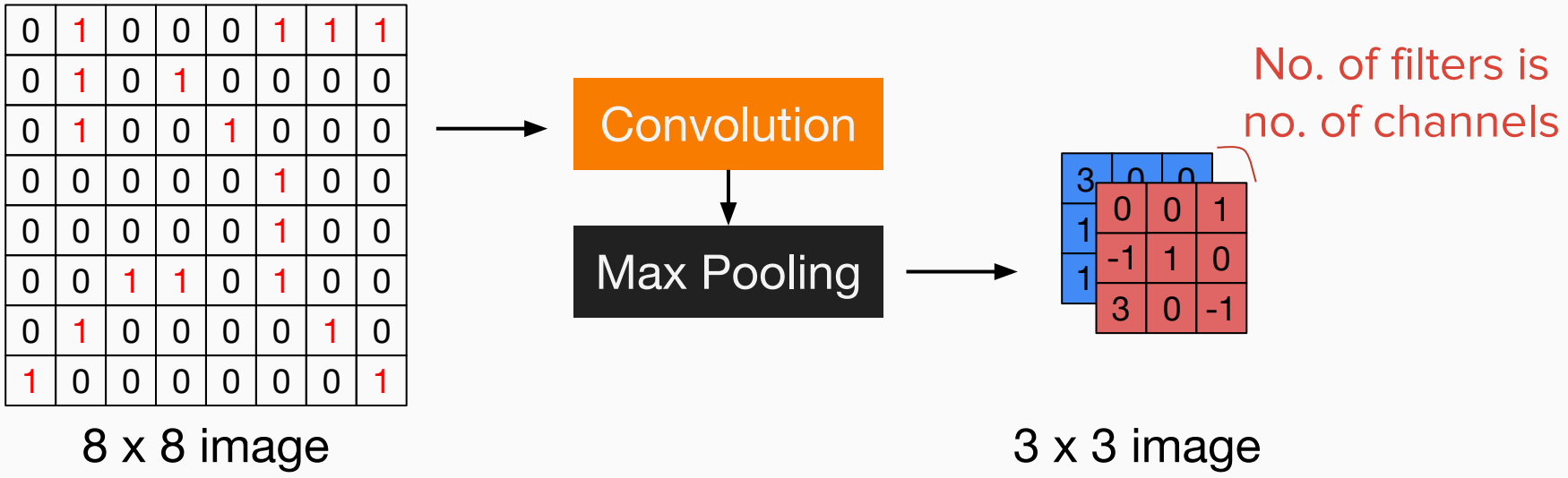
3	0	0
1	0	3
1	0	1

-1	-2	0	-1	1	-1
0	-1	-2	-1	-2	1
-1	-1	1	-3	-1	0
-1	-2	0	0	-1	-1
-2	1	0	-1	-1	-3
3	-1	-2	0	-2	-1

Max Pooling

0	0	1
-1	1	0
3	0	-1

Max Pooling (Cont'd)



8 x 8 image

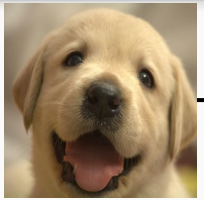
3 x 3 image

new image but smaller!

less parameters!

CloudMile

Whole CNN



Convolution

Max Pooling

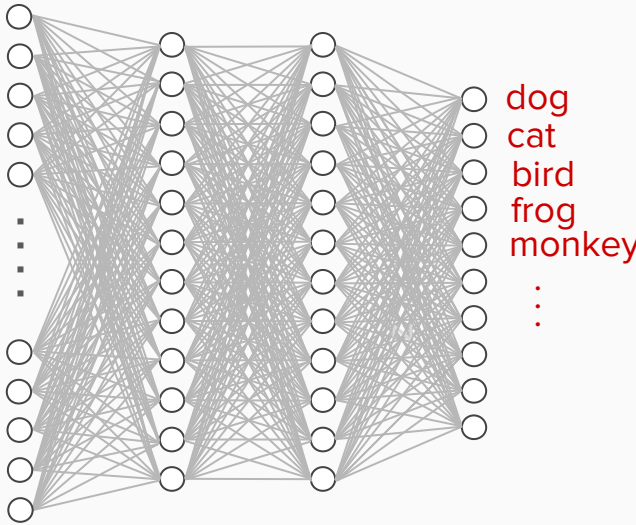
can repeat many times

new image

Convolution

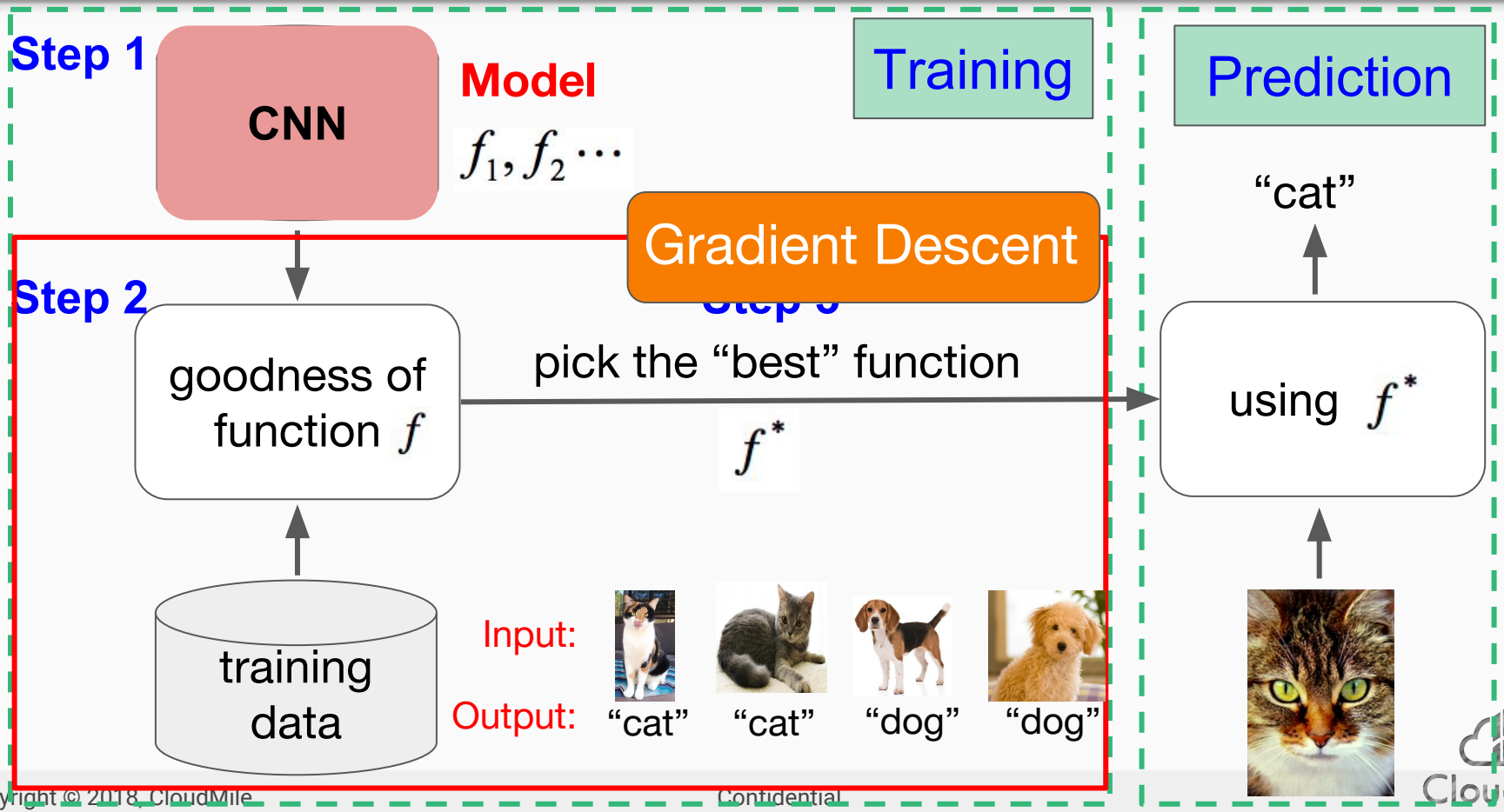
Max Pooling

Flatten



CloudMile

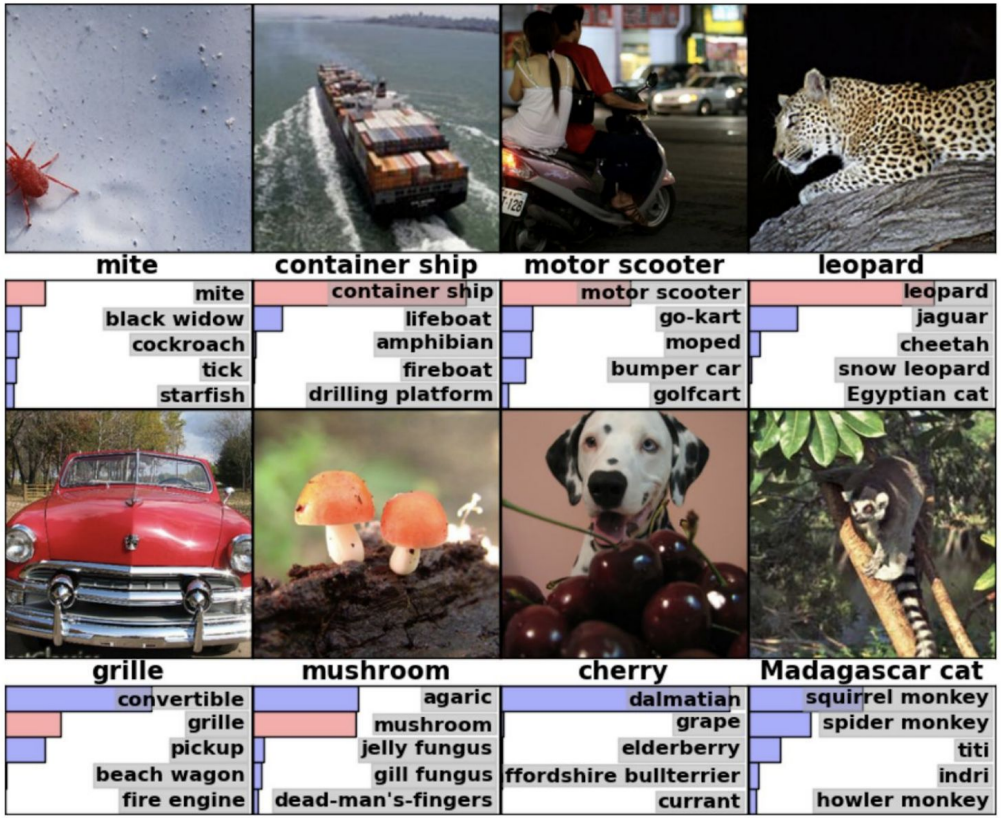
Machine Learning Framework



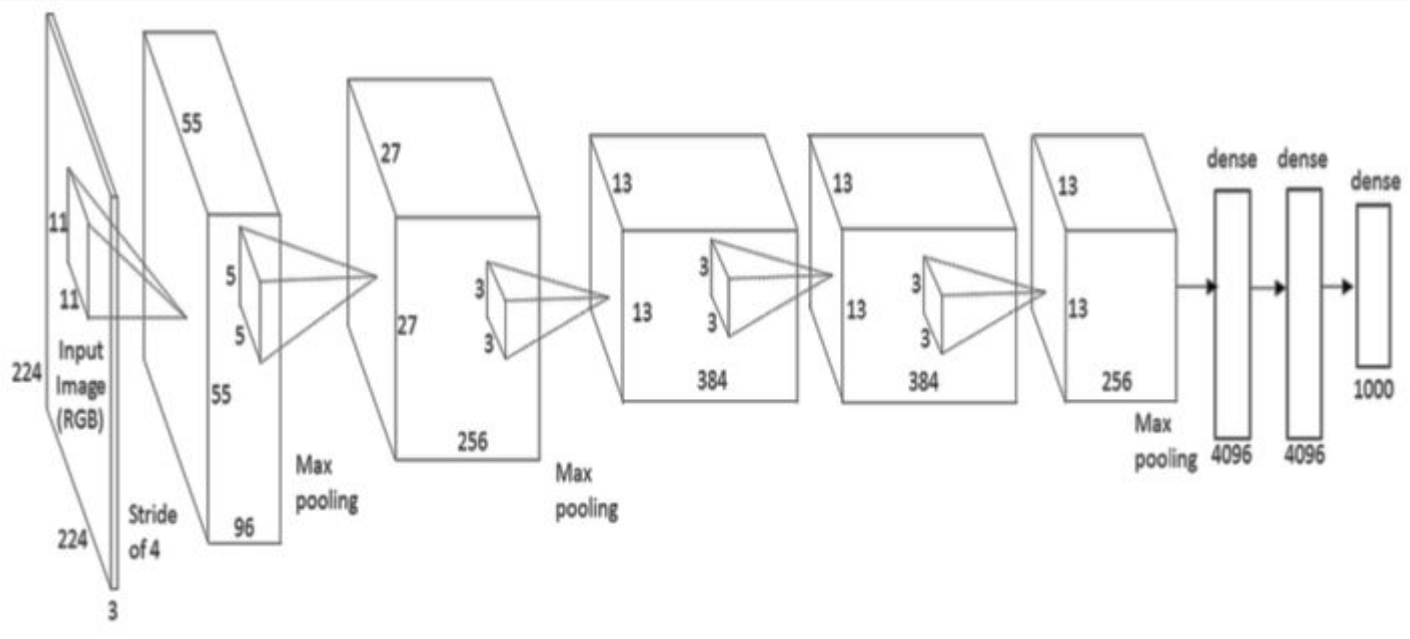
ImageNet Challenge

IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.



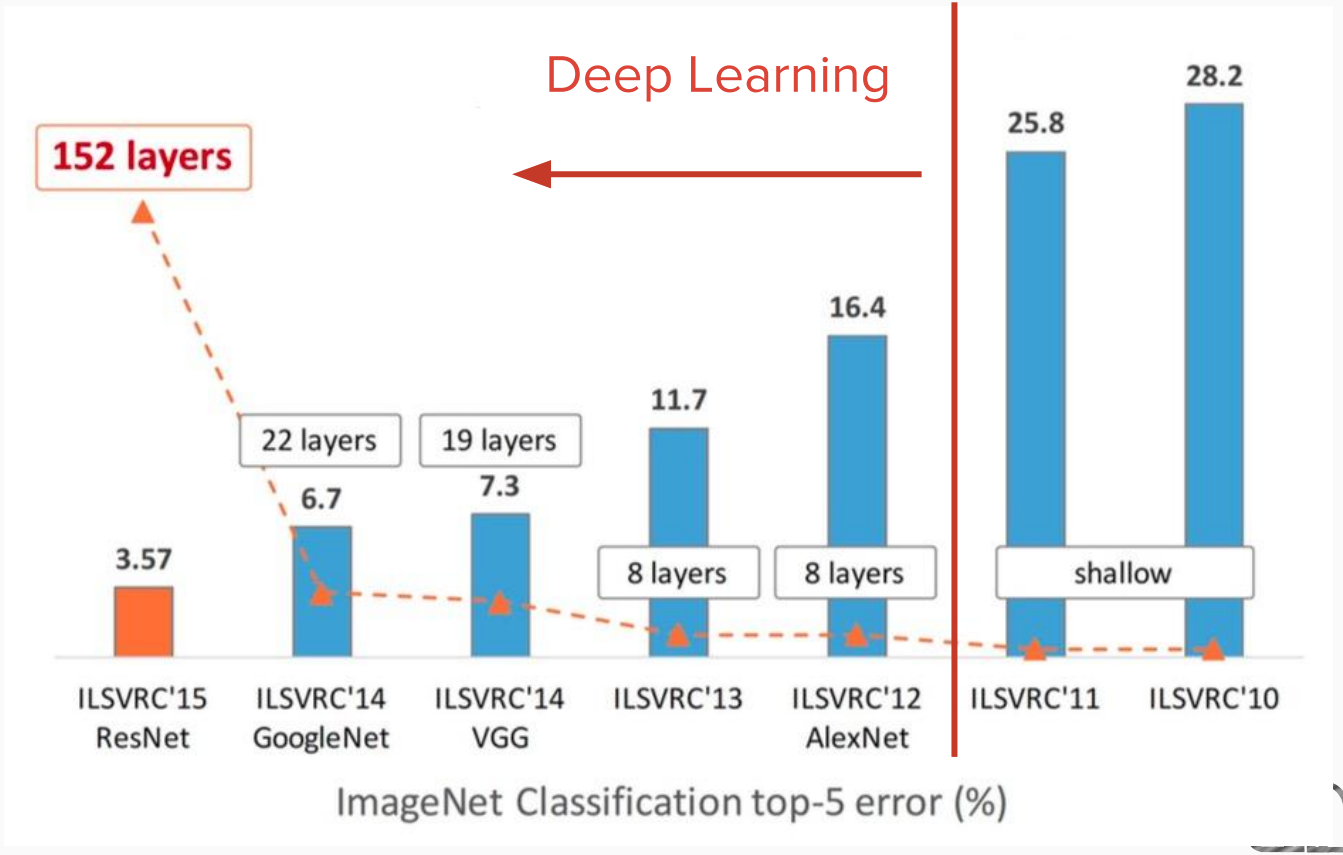
AlexNet (2012)



8 layers, ImageNet Classification top-5 error is 16.4%.

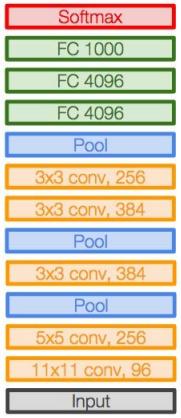
CloudMile

You Need to Go Deeper

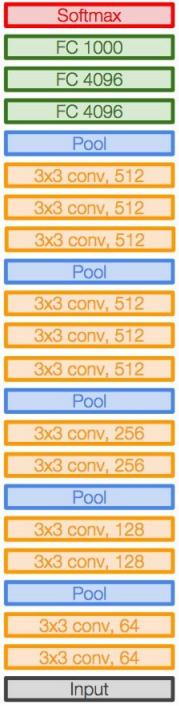


CloudMile

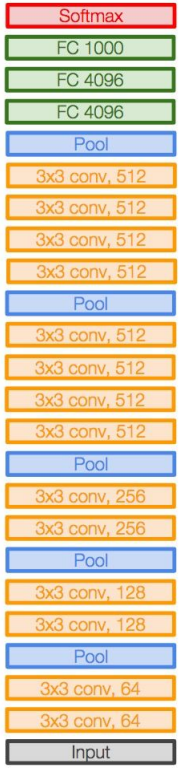
AlexNet



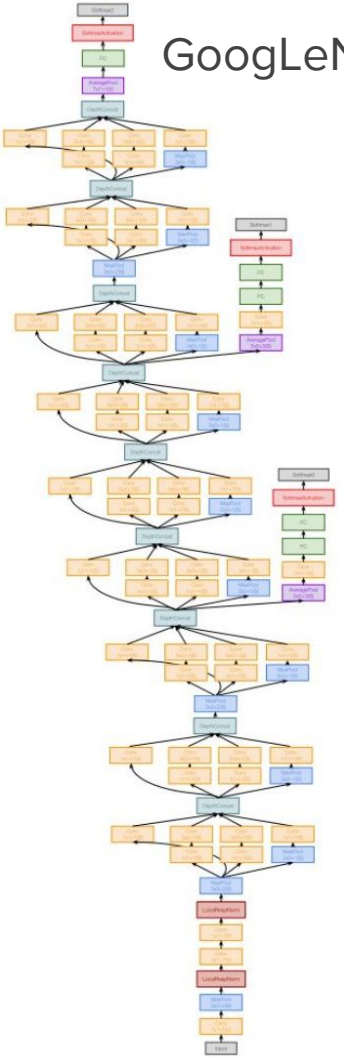
VGG16



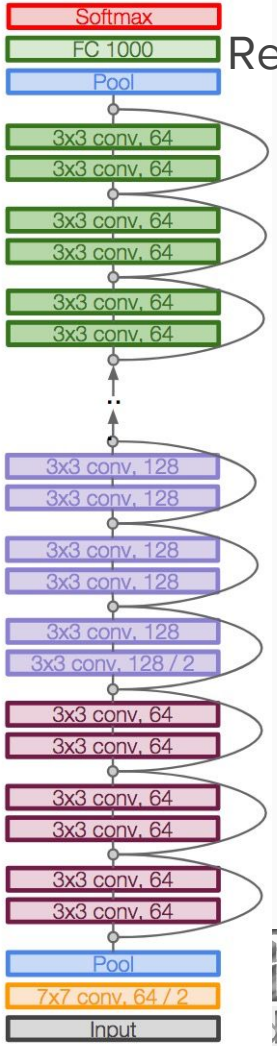
VGG19



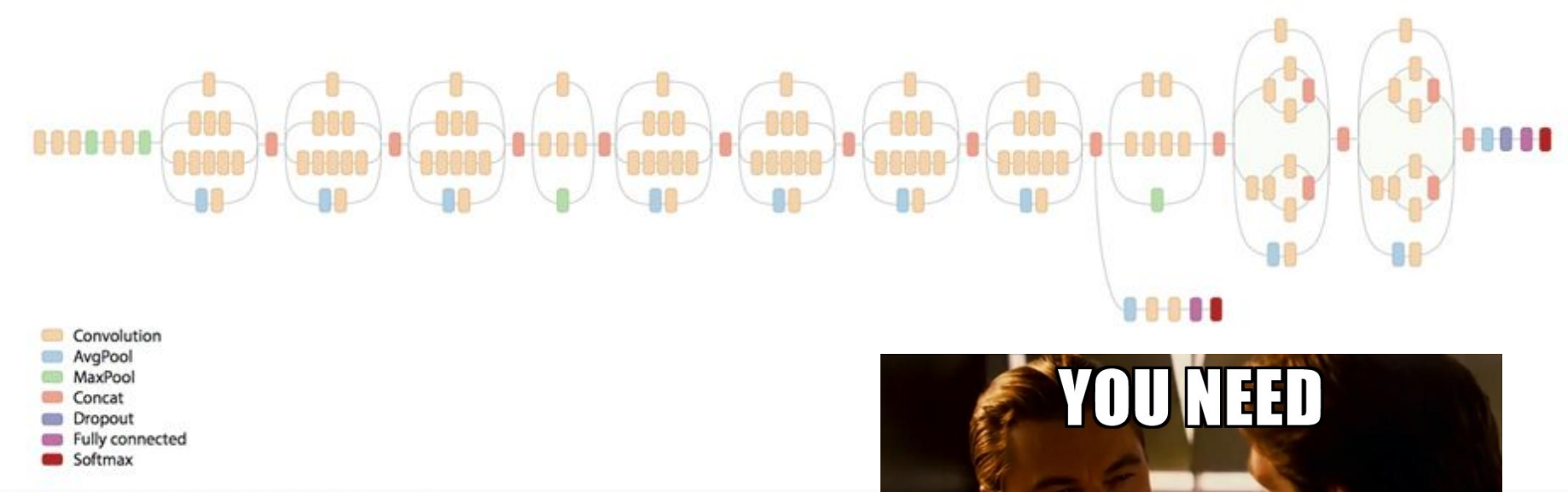
GoogLeNet



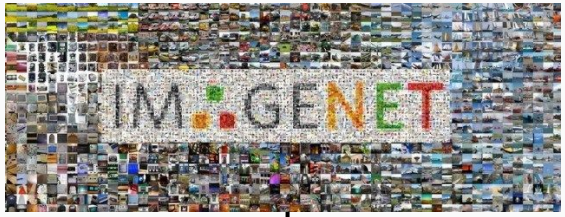
ResNet

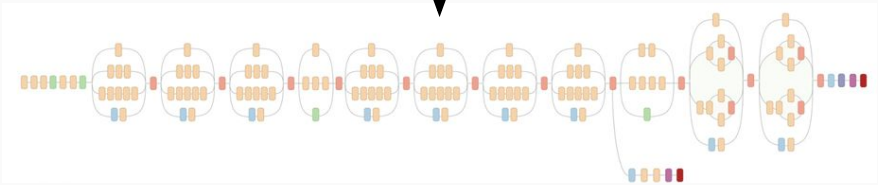


Inception V3



Transfer Learning



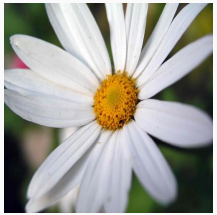


trained on ImageNet

transfer

roses(玫瑰)

sunflowers(向日葵)



daisy(雛菊)

tulips(鬱金香)

flower classifier

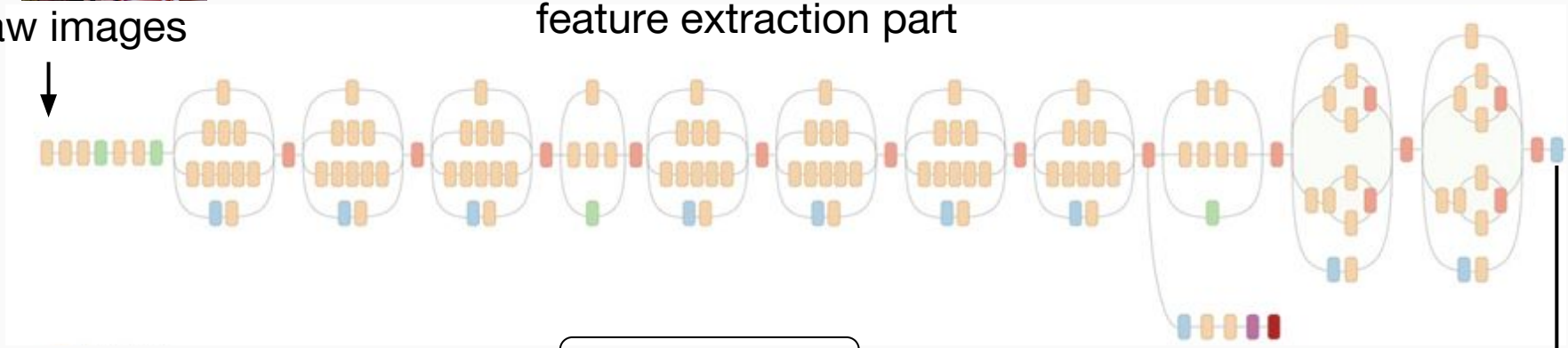
CloudMile

Transfer Learning (Cont'd)

raw images

feature extraction part

pretrained on ImageNet



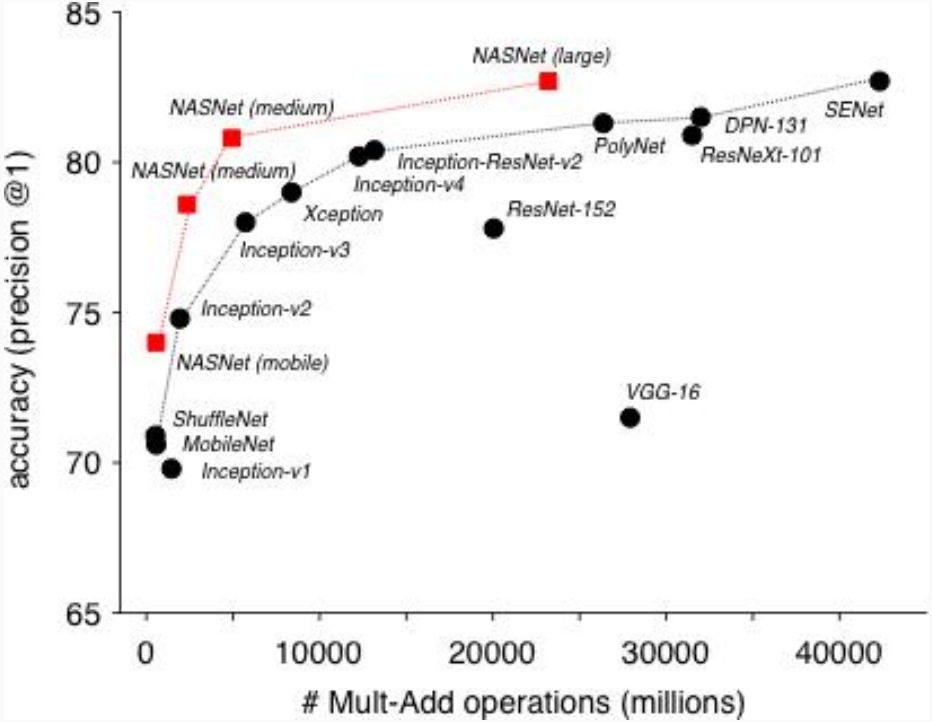
feature vector

only train this part

classification part

CloudMile

Choosing a Pretrained Model



Speed v.s. Accuracy

MobileNet

NASNet (mobile)

ResNet

Inception

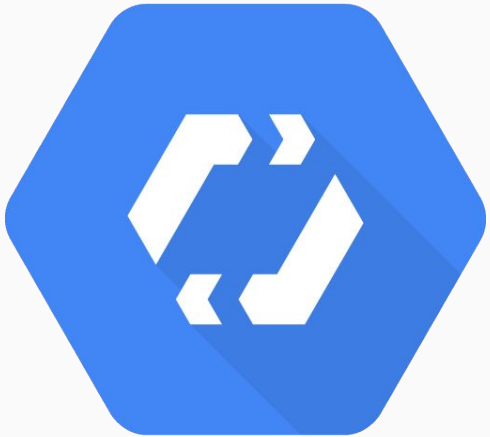
NASNet (large)

CloudMile

Cloud AutoML Vision

Simple transfer learning with efficient model training for quick demo within minutes

High accuracy model based upon learning to learn within a day



Training an Image Classifier

Train a CNN from scratch

Can be tuned for very specific cases

Requires large amounts of data

Training takes time and resources

Transfer Learning

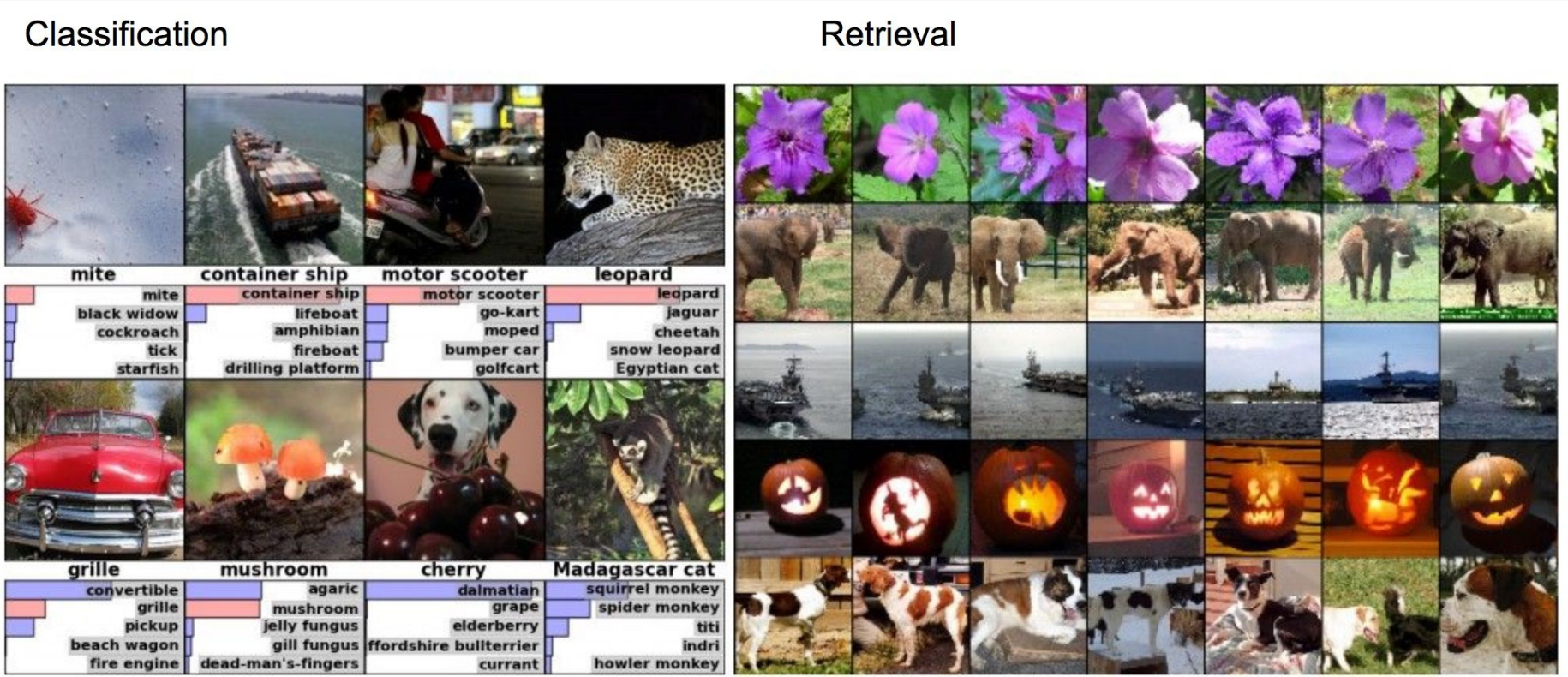
Works well with small data

Training is much faster

Less performant on very specific cases

CNNs are everywhere

Slide Credit: stanford.io/2ybZ4Ay

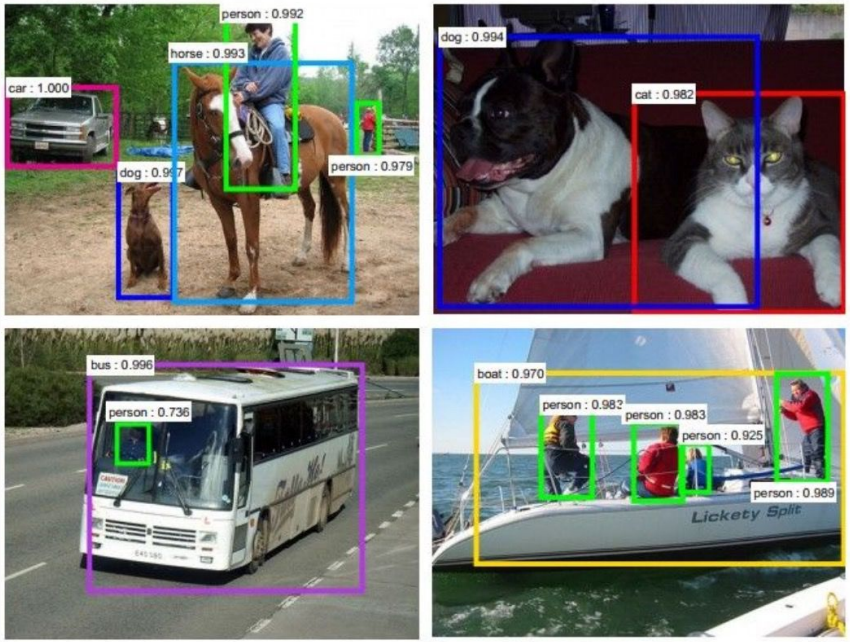


CloudMile

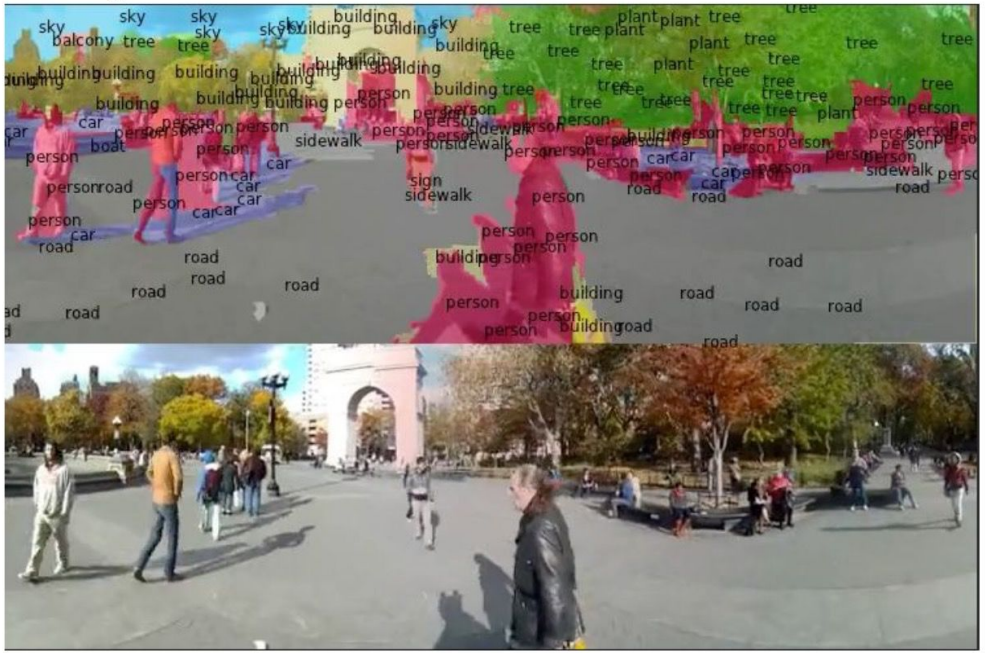
CNNs are everywhere

Slide Credit: stanford.io/2ybZ4Ay

Detection



Segmentation



CloudMile

CNNs are everywhere

Slide Credit: stanford.io/2ybZ4Ay

self-driving car

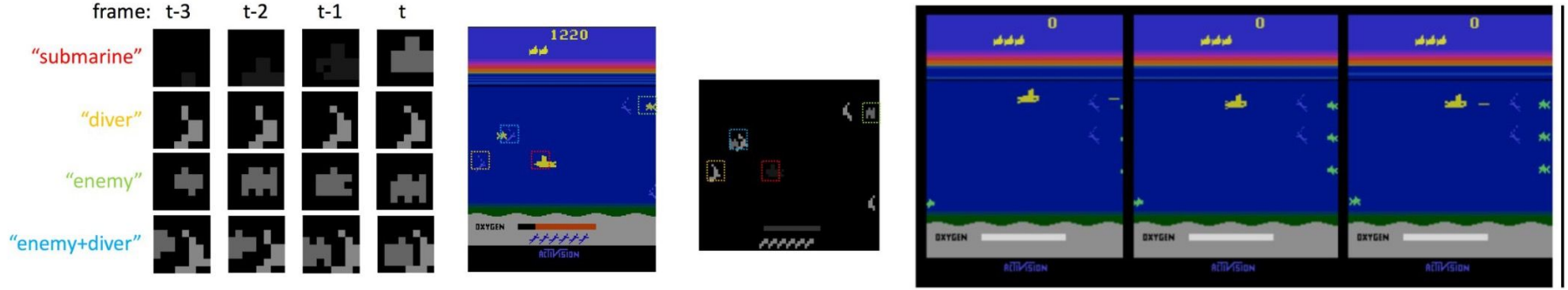
CloudMile

CNNs are everywhere

Slide Credit: stanford.io/2ybZ4Ay

Images are examples of pose estimation, not actually from Toshev & Szegedy 2014. Copyright Lane McIntosh.

[Toshev, Szegedy 2014]



CloudMile

CNNs are everywhere

Slide Credit: stanford.io/2ybZ4Ay

No errors

A white teddy bear sitting in the grass

Minor errors

A man in a baseball uniform throwing a ball

Somewhat related

A woman is holding a cat in her hand

A man riding a wave on top of a surfboard

A cat sitting on a suitcase on the floor

A woman standing on a beach holding a surfboard

Image Captioning

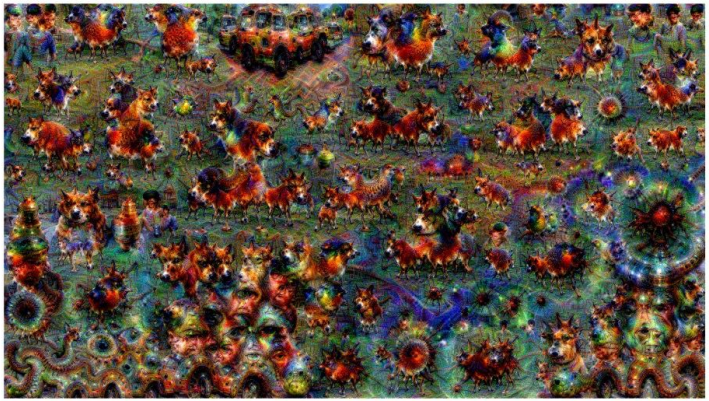
[Vinyals et al., 2015]
[Karpathy and Fei-Fei, 2015]

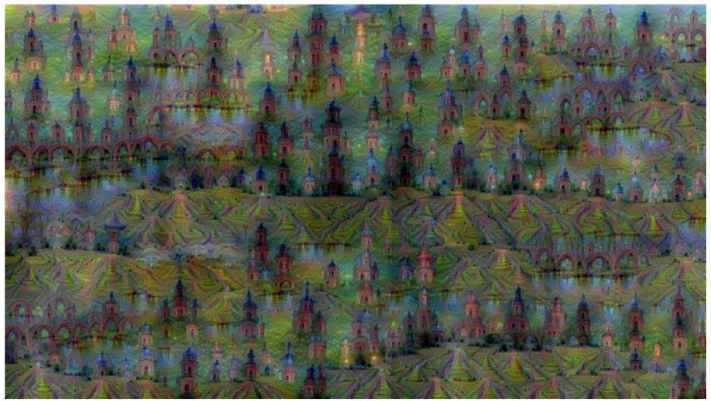
All images are CC0 Public domain:
https://pixabay.com/en/luggage-antique-cat-1643010/
https://pixabay.com/en/teddy-plush-bears-cute-teddy-bear-1623436/
https://pixabay.com/en/surf-wave-summer-sport-litoral-1668716/
https://pixabay.com/en/woman-female-model-portrait-adult-983967/
https://pixabay.com/en/handstand-lake-meditation-496008/
https://pixabay.com/en/baseball-player-shortstop-infield-1045263/

Captions generated by Justin Johnson using Neuraltalk2

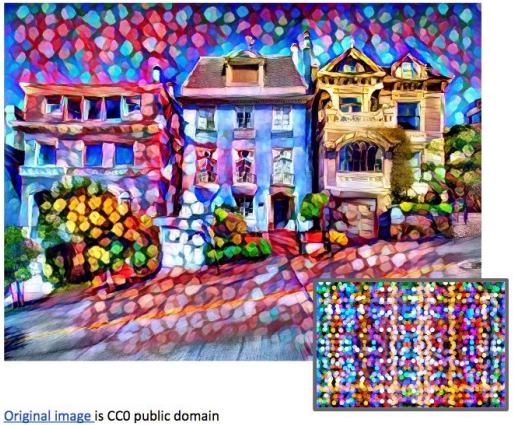
CNNs are everywhere

Slide Credit: stanford.io/2ybZ4Ay

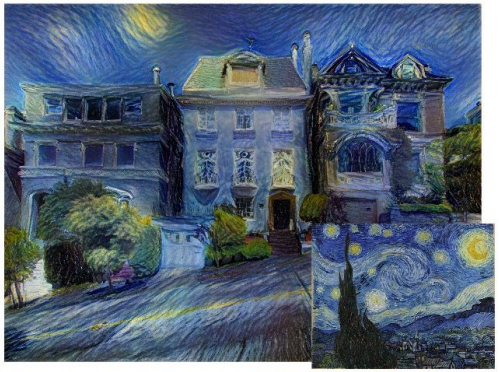


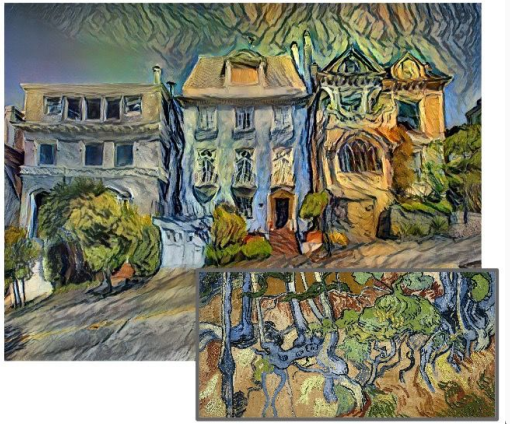


Figures copyright Justin Johnson, 2015. Reproduced with permission. Generated using the Inceptionism approach from a blog post by Google Research.



Original image is CC0 public domain
Starry Night and Tree Roots by Van Gogh are in the public domain
Bokeh image is in the public domain
Stylized images copyright Justin Johnson, 2017;
reproduced with permission





Gatys et al, "Image Style Transfer using Convolutional Neural Networks", CVPR 2016
Gatys et al, "Controlling Perceptual Factors in Neural Style Transfer", CVPR 2017

Build your first CNN!

Lab 2-i

Topic : Build a basic CNN classification model

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Train a CNN classification model
Duration	~ 10 mins

Lab 2-i

Conv2D

```
keras.layers.Conv2D(filters, kernel_size, strides=(1, 1),
padding='valid', activation=None)
```

Filters: the dimensionality of the output space

Kernel_size: height and width of the 2D convolution window

Strides: the stride of the combolution along the height and width

Padding: one of ‘valid’ and ‘same’

Activation: activation function

Lab 2-i

MaxPooling2D

```
keras.layers.MaxPooling2D(pool_size=(2, 2), strides=None,
padding='valid'
```

Pool_size : factors by which to downscale

Strides : Stride values. If none, it will default to pool_size

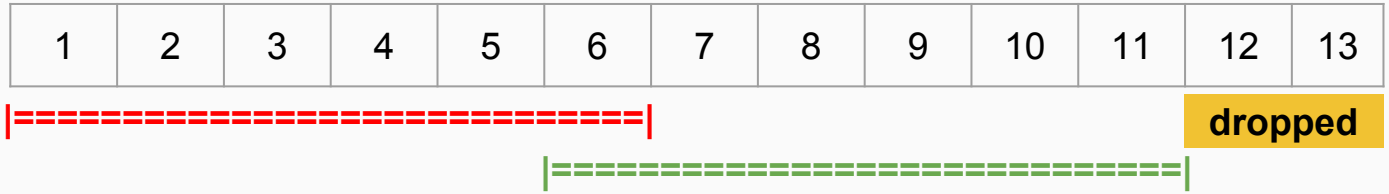
Padding : one of ‘valid’ and ‘same’

What is padding?

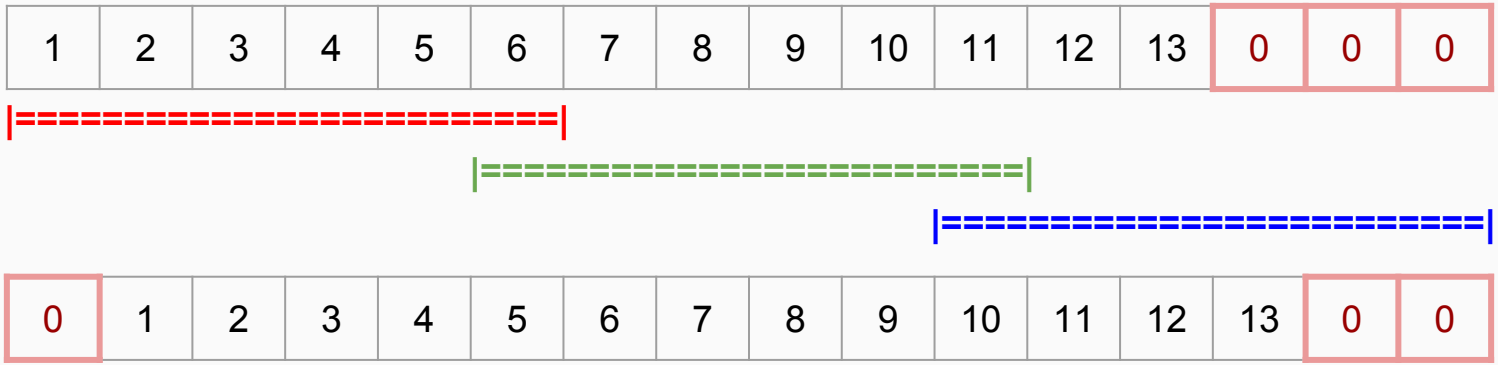
In this example:

- Input width = 13
- Filter width = 6
- Stride = 5

Padding = 'VALID' → **Without padding**



Padding = 'SAME' → **With zero padding**



CloudMile

Lab 2-i

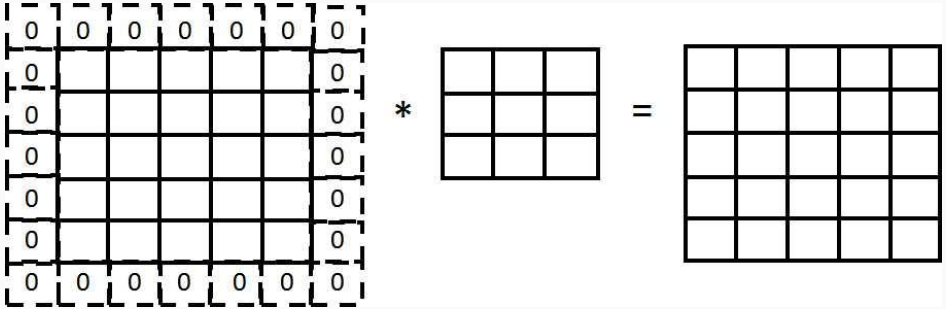
```
K.clear_session()

model_basic_cnn = keras.Sequential()

model_basic_cnn.add(Conv2D(filters=32,
                           kernel_size=(3,3),
                           input_shape=(28,28,1),
                           padding='same',
                           activation='relu'))

model_basic_cnn.add(MaxPool2D(pool_size=(2,2),
                              strides=(2,2)))

model_basic_cnn.add(Conv2D(filters=64,
                           kernel_size=(3,3),
                           padding='same',
                           activation='relu'))

model_basic_cnn.add(MaxPool2D(pool_size=(2,2),
                              strides=(2,2)))

model_basic_cnn.add(Conv2D(filters=128,
                           kernel_size=(3,3),
                           padding='same',
                           activation='relu'))

model_basic_cnn.add(MaxPool2D(pool_size=(2,2),
                              strides=(2,2)))

model_basic_cnn.add(Flatten())

model_basic_cnn.add(Dense(10, activation='softmax'))

model_basic_cnn.summary()
```

Padding → **'same'** : padding the input such that the output has the same length as the original input if the stride = 1

→ **'valid'** : no padding

Stride → specify the stride length of the convolution

Padding(same)



CloudMile

Lab 2-i

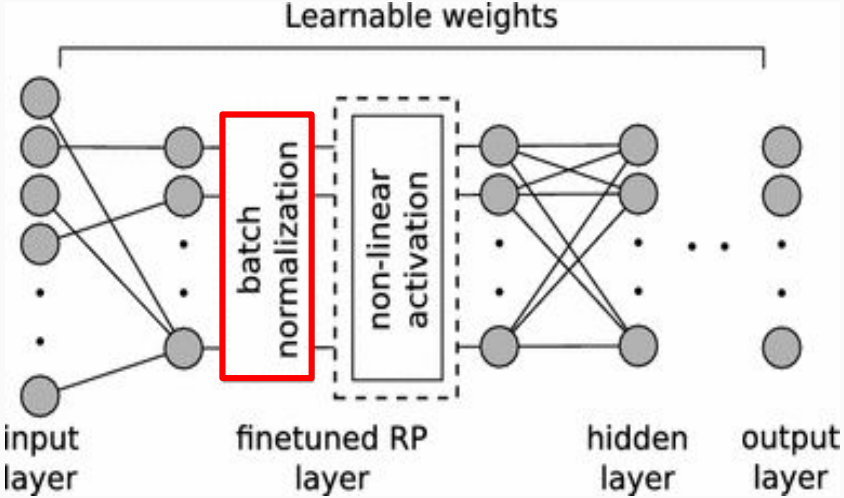
Topic : Build a basic CNN classification model

Start coding !

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Train a CNN classification model
Duration	~ 10 mins

Batch normalization

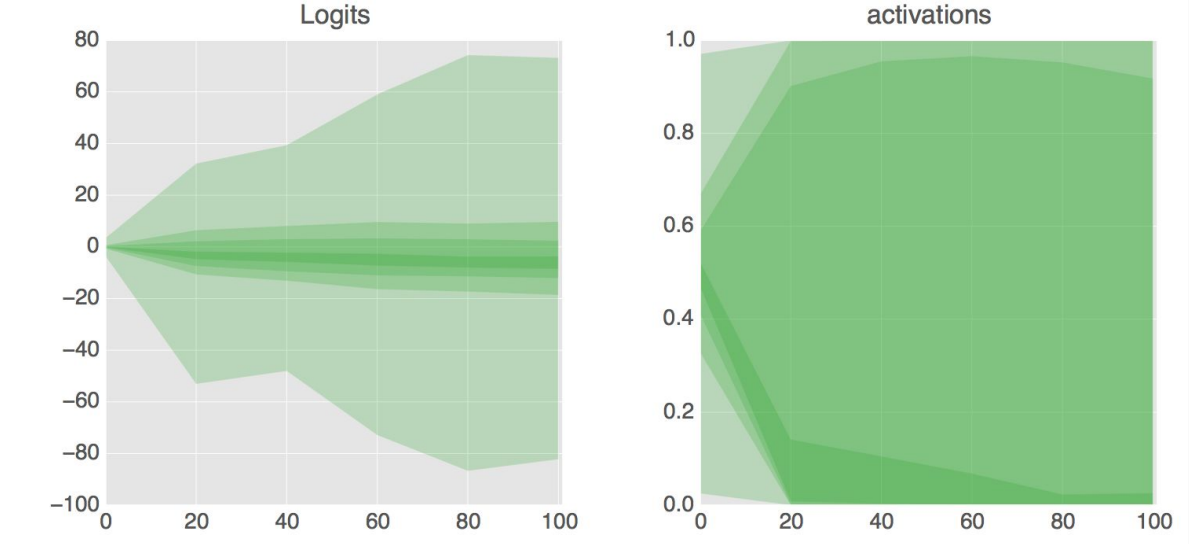
- Before feeding the outputs of one layer into the inputs of the next, we center and rescale them.

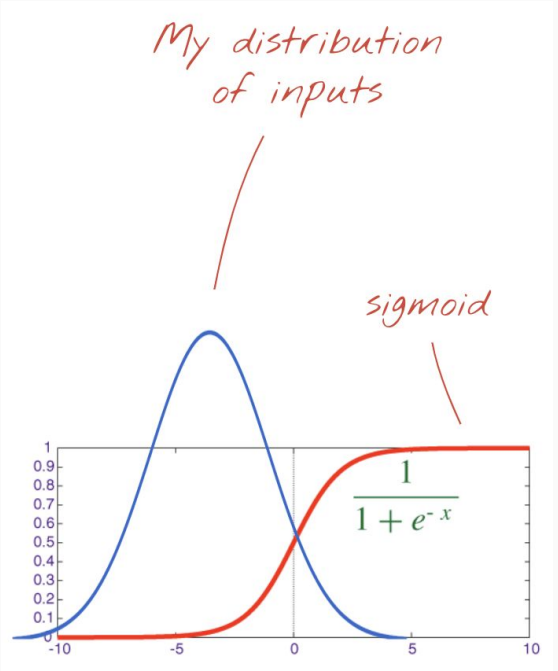


$$\hat{x} = \frac{x - avg_{batch}(x)}{stdev_{batch}(x) + \epsilon}$$

Batch normalization

Before batch normalization

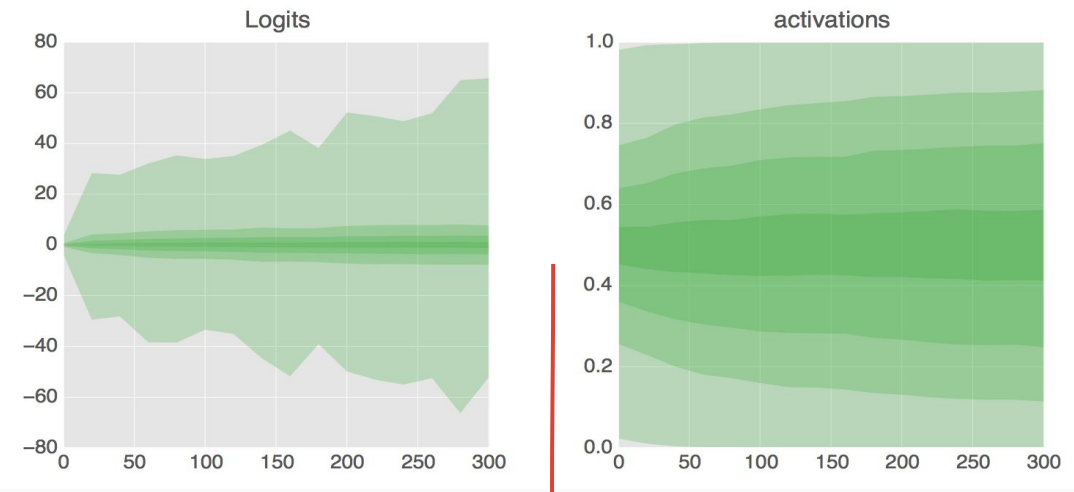




CloudMile

Batch normalization

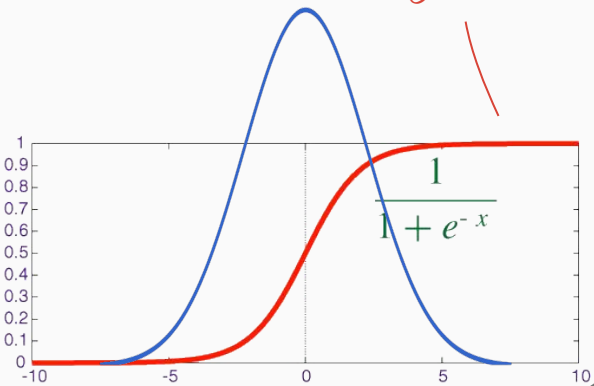
After batch normalization



Batch norm

My distribution of inputs

sigmoid



CloudMile

Lab 2-e

Topic : Build an advanced CNN classification model

Start coding !

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Train a CNN classification model
Duration	~ 10 mins

Why not pretrained model?

- In keras,

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	88 MB	0.790	0.945	22,910,480	126
VGG16	528 MB	0.713	0.901	138,357,544	23
VGG19	549 MB	0.713	0.900	143,667,240	26
ResNet50	99 MB	0.749	0.921	25,636,712	168
InceptionV3	92 MB	0.779	0.937	23,851,784	159
InceptionResNetV2	215 MB	0.803	0.953	55,873,736	572
MobileNet	16 MB	0.704	0.895	4,253,864	88
MobileNetV2	14 MB	0.713	0.901	3,538,984	88
DenseNet121	33 MB	0.750	0.923	8,062,504	121
DenseNet169	57 MB	0.762	0.932	14,307,880	169
DenseNet201	80 MB	0.773	0.936	20,242,984	201
NASNetMobile	23 MB	0.744	0.919	5,326,716	-
NASNetLarge	343 MB	0.825	0.960	88,949,818	-

Keras Functional API

```
from keras.layers import Input, Dense
from keras.models import Model

# This returns a tensor
inputs = Input(shape=(784,))

# a layer instance is callable on a tensor, and returns a tensor
x = Dense(64, activation='relu')(inputs)
x = Dense(64, activation='relu')(x)
predictions = Dense(10, activation='softmax')(x)

# This creates a model that includes
# the Input layer and three Dense layers
model = Model(inputs=inputs, outputs=predictions)
model.compile(optimizer='rmsprop',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
model.fit(data, labels)  # starts training
```

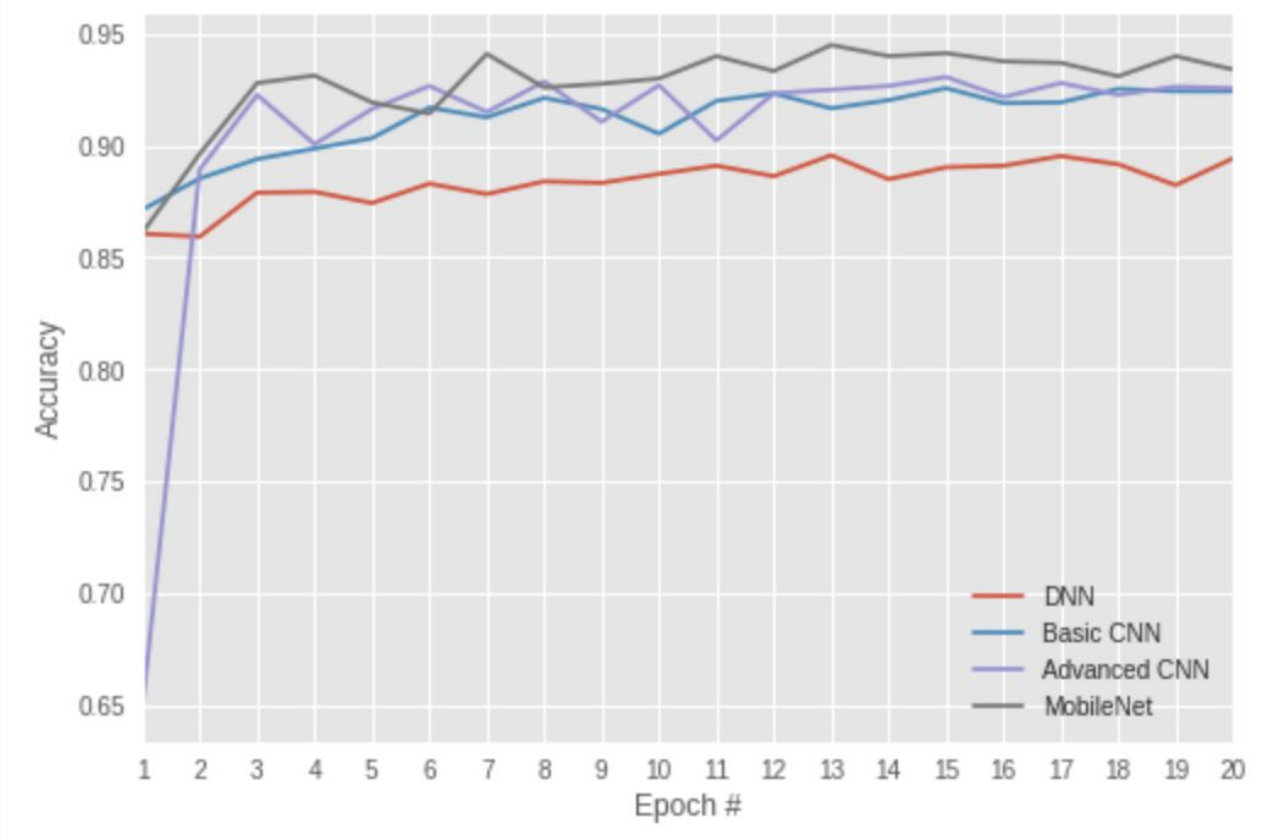
Lab 2-f

Topic : MobileNet

Start coding !

Filename	lab2.ipynb
Data	Fashion MNIST
Target	➔ Use Mobilenet to classify fashion MNIST
Duration	~ 10 mins

Lab 2-f Result



CloudMile