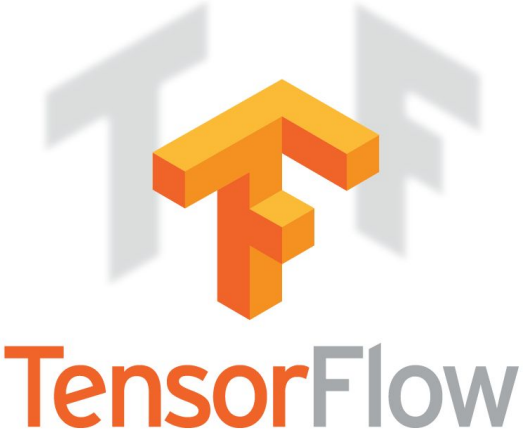
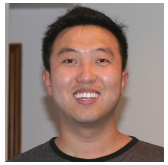
GDG DevFest 2017 Shanghai



从入门到不放弃



顾仁民
谷歌资深工程师

日程

- 人工智能，机器学习，深度学习简介
- TensorFlow简介
- TensorFlow基本概念（动手）

- TensorFlow机器学习（动手）
- TensorFlow深度学习（动手）

- TensorFlow分布式训练（动手）
- 社区和谷歌招聘介绍
- 福利时间：AlphaGo Zero

人工智能，机器学习，深度学习...

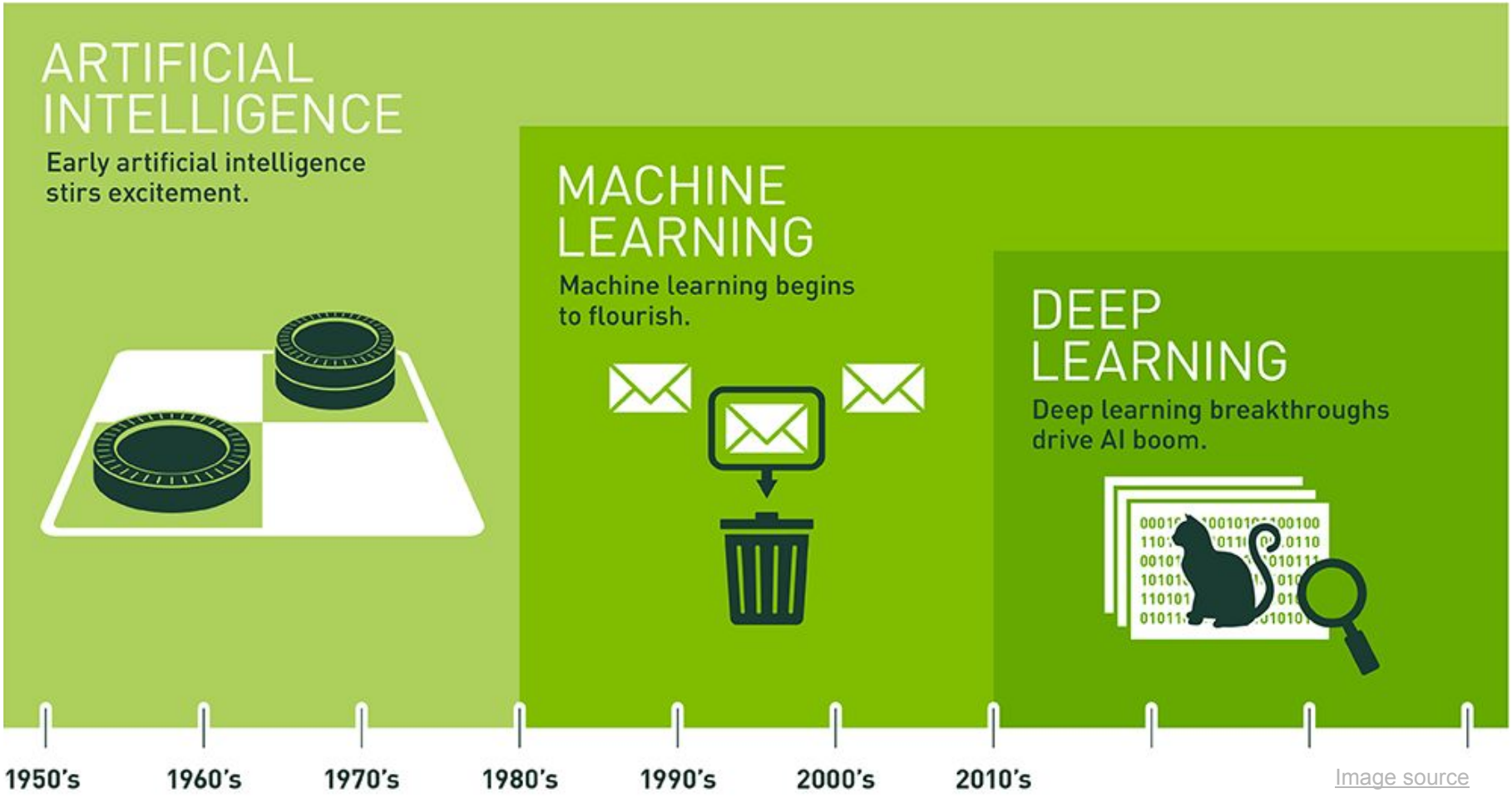
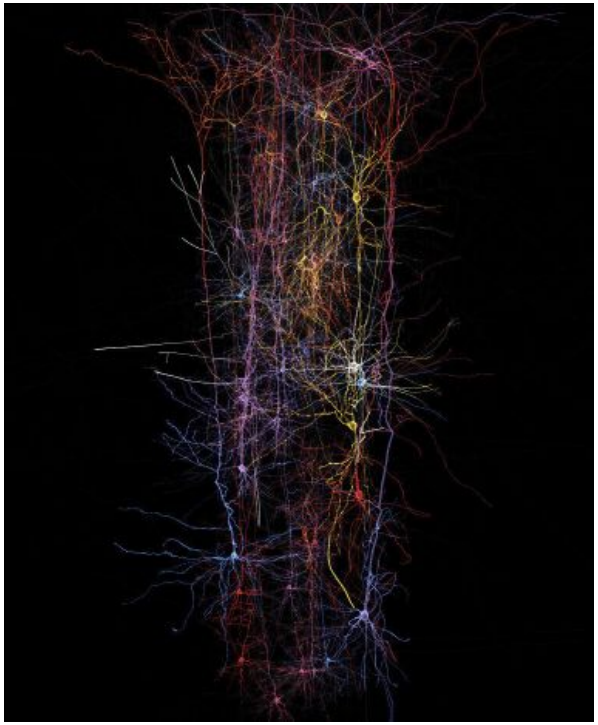


Image source

人工智能

- 各种经典算法
  - 动态规划
  - 博弈树
  - ...
- 脑模拟
  - 欧洲Blue Brain计划
  - 美国BRAIN计划
  - Numenta
  - Vicarious
  - 基于STDP的硬件实现
  - …

...



机器学习

从数据中学习

经验

归纳

新问题 → 规律 → 方案

数据

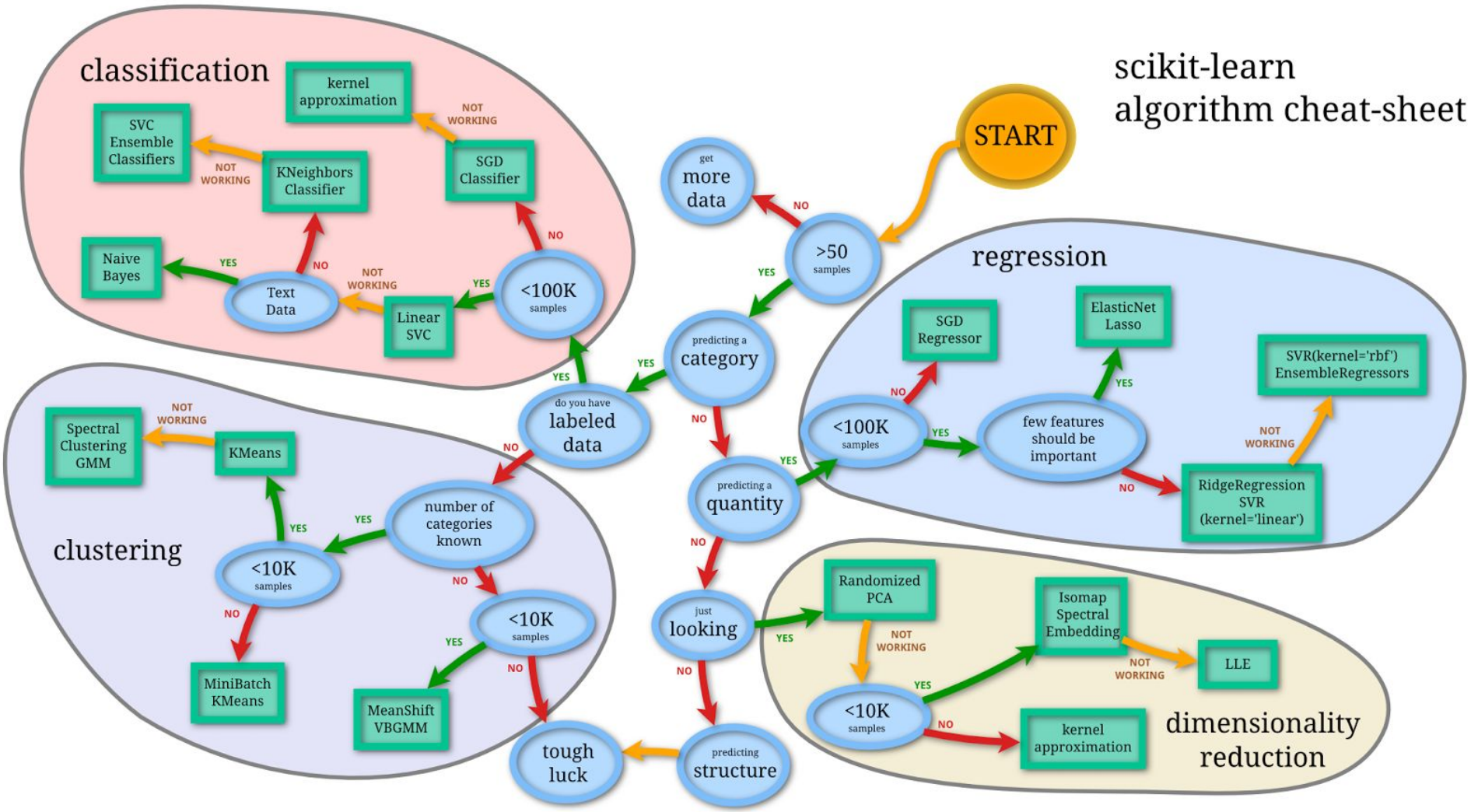
训练

新数据 → 模型 → 结论

机器学习

- 分类
  - 这个图片是猫还是狗？
  - 这个图片是什么？
  - ...
- 回归
  - 明天沪深300指数是多少？
  - 明天库房该备多少货？
  - ...
- 聚类
  - 把这些文章分成几个相似的组
  - 把客户进行分为几个相似的组
  - ...
- 其它角度
  - 是否有监督信息(老师教)
  - 是否和环境交互(自己探索)
  - ...

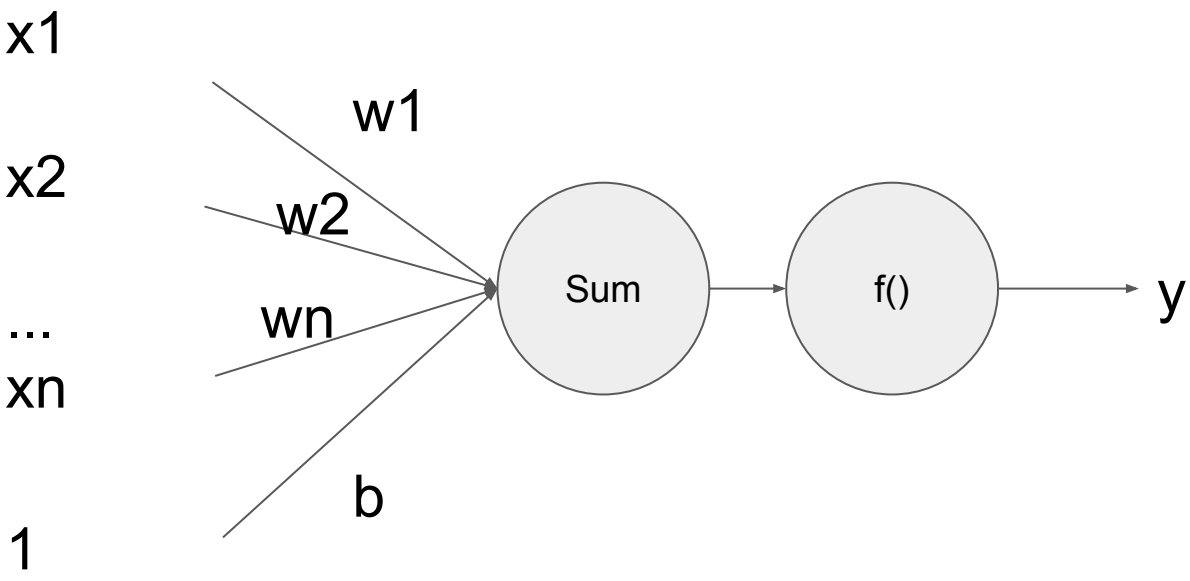
机器学习小抄



Cheat-sheets links

这只是机器学习的一小部分

深度学习



y = f(w1*x1 + w2*x2 + … + wn*xn + b)

神经网络就是将多个这样的单元以某种形式组织起来
外加卷积、回馈等各种改进技巧

深度学习小抄

RBF

DFF

CNN

RNN

LSTM

VAE

GAN

NTM

DQN

...

...

...

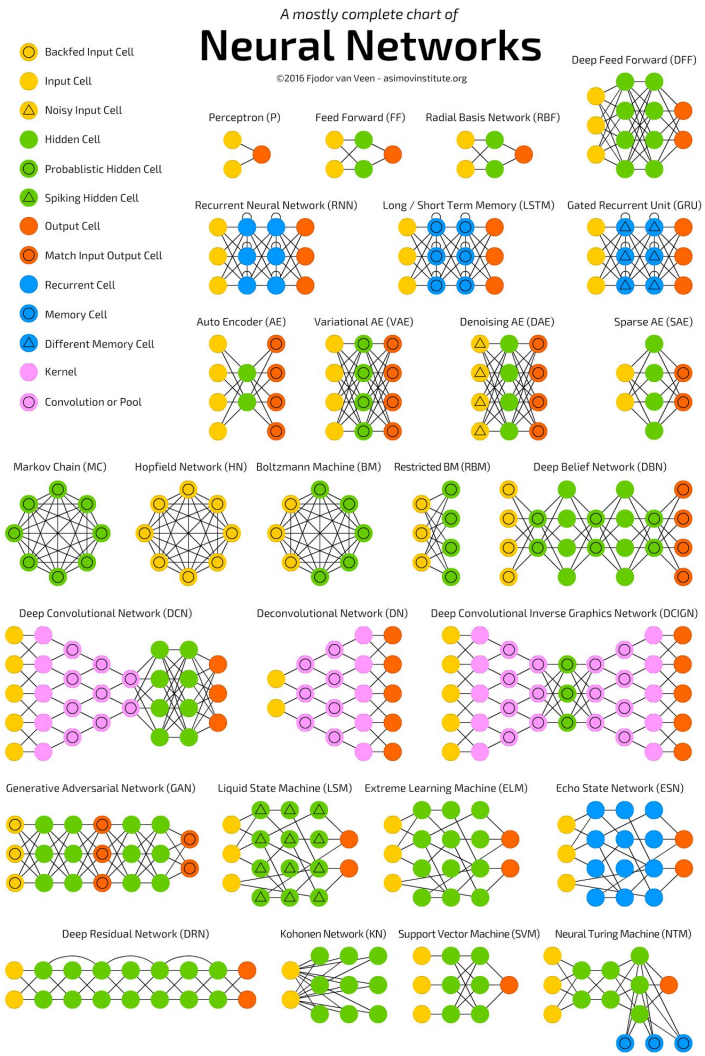


Image source

多个软件库的自我描述

猜！

A New Lightweight, Modular, and Scalable Deep Learning Framework

Tensors and Dynamic neural networks in Python with strong GPU acceleration.

A Flexible and Efficient Library for Deep Learning

An open-source software library for Machine Intelligence

TensorFlow

是一个使用**数据流图**进行**数值计算**的**开源**软件库。

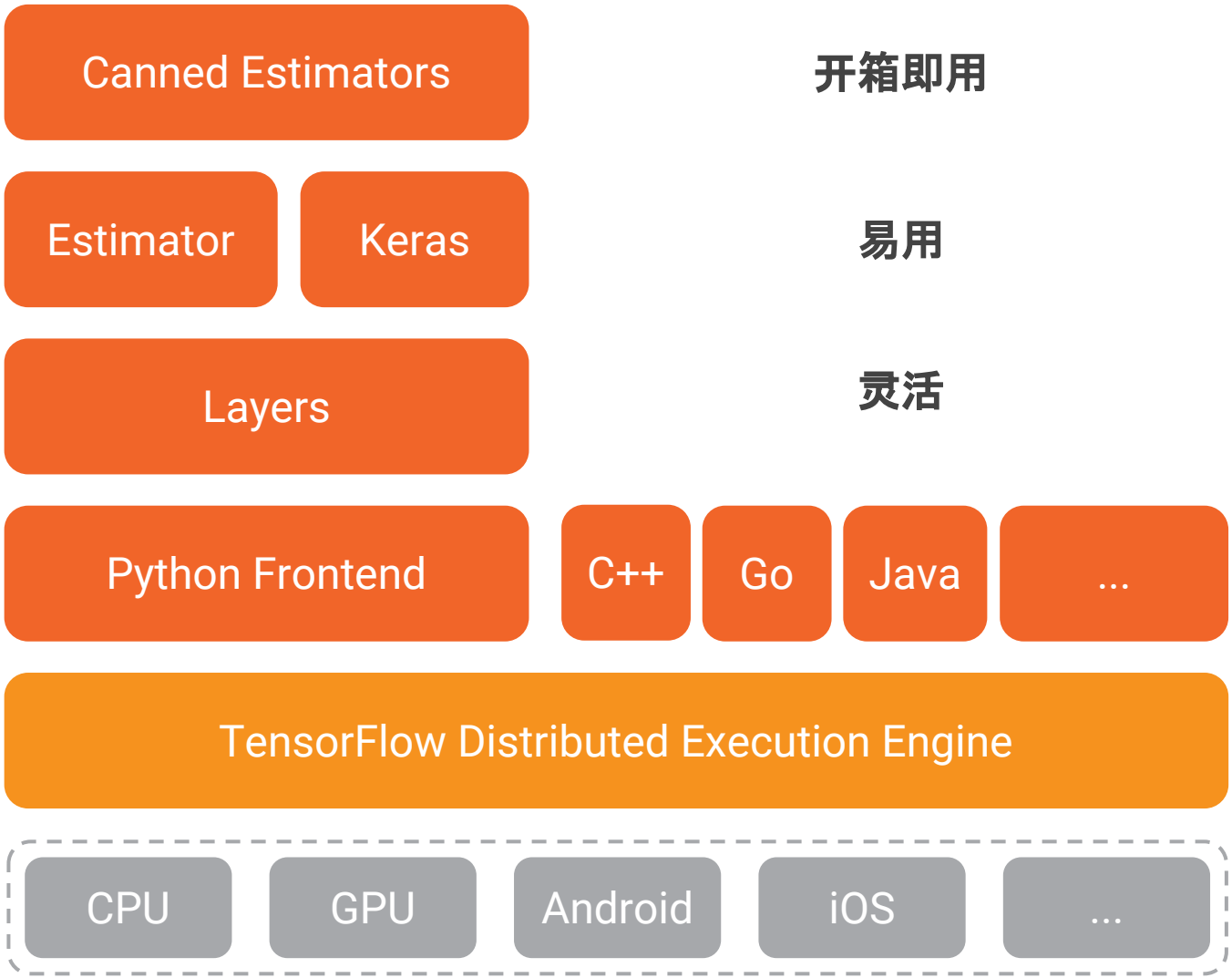
它是一个非常基础的系统，

因此也可以应用于众多领域，

包括机器学习，深度学习等。

高效，灵活，可用于生产环境

TensorFlow大体架构



边讲边练

TensorFlow 从入门到不放弃 实验环境准备

1. 在笔记本上安装Docker社区版，下载地址

2. 拉取TensorFlow镜像（本次实验以当时最新版1.4.0-rc0为例）

```
docker pull tensorflow/tensorflow:1.4.0-rc0
```

3. 启动TensorFlow镜像

```
docker run --name tflab -d \
    -p 8888:8888 -p 6006:6006 \
    -v ~/tflab:/notebooks/tflab \
    tensorflow/tensorflow:1.4.0-rc0
```

TensorFlow 从入门到不放弃 实验环境准备

4. 获取Jupyter Token并用浏览器(推荐Chrome)打开Jupyter实验环境

docker logs tflab

查看容器日志得到日志中如下URL，copy/paste到浏览器

http://localhost:8888/?token=c9428628e9cf5645fc2d0983ffb3fc72c214753610b7cc93

```
[I 04:05:39.107 NotebookApp] Serving notebooks from local directory: /notebooks
[I 04:05:39.107 NotebookApp] 0 active kernels
[I 04:05:39.107 NotebookApp] The Jupyter Notebook is running at:
[I 04:05:39.107 NotebookApp] http://[all ip addresses on your system]:8888/?token=92740eb8244b4fb851dc8a2ec927daffac159d8941b1588a
[I 04:05:39.107 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 04:05:39.108 NotebookApp]

    Copy/paste this URL into your browser when you connect for the first time,
    to login with a token:
        http://localhost:8888/?token=92740eb8244b4fb851dc8a2ec927daffac159d8941b1588a
```

TensorFlow 从入门到不放弃 实验环境准备

5. 在Jupyter中新建一个Python2的Notebook，输入如下内容并运行

```
from tensorflow.contrib.keras import datasets
datasets.mnist.load_data()
print("\n Done")
```

下载的文件会存放在容器内的/root/.keras/datasets下面

```
from tensorflow.contrib.keras import datasets

datasets.mnist.load_data()
print("\n Done")
```

```
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz
11329536/11490434 [============================>.] - ETA: 0s
 Done
```

6. 停掉容器，给笔记本充满电，带到会议现场

```
docker stop tflab
```

TensorFlow 从入门到不放弃 实验环境准备

7. 会议现场，开启容器，打开Jupyter

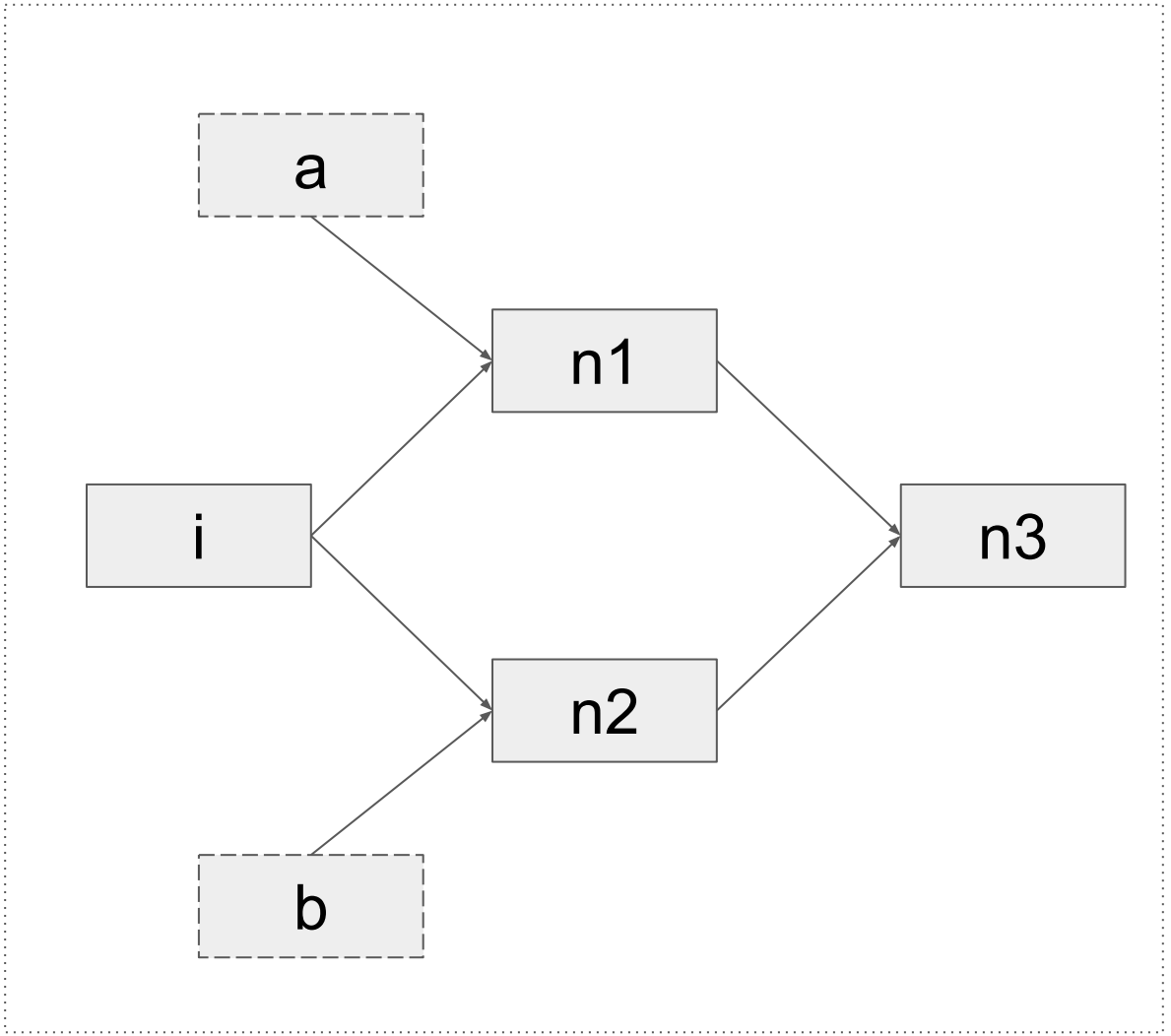
docker start tflab

http://localhost:8888

任务1 - 基本概念

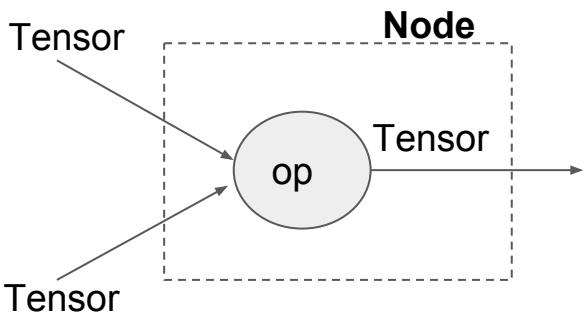
TensorFlow基本概念

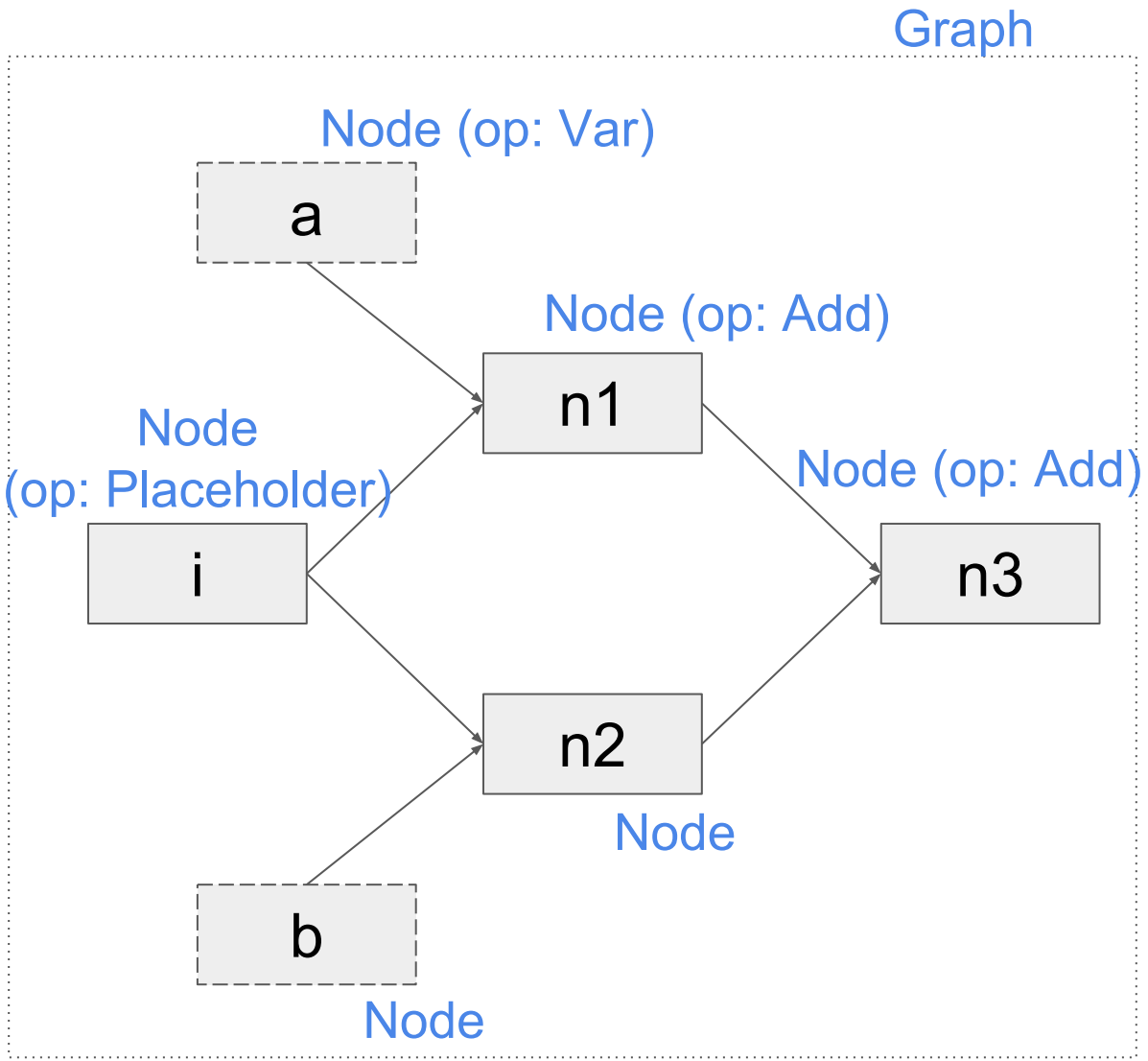
n1 = i + a
n2 = i + b
n3 = n1 + n2



TensorFlow基本概念

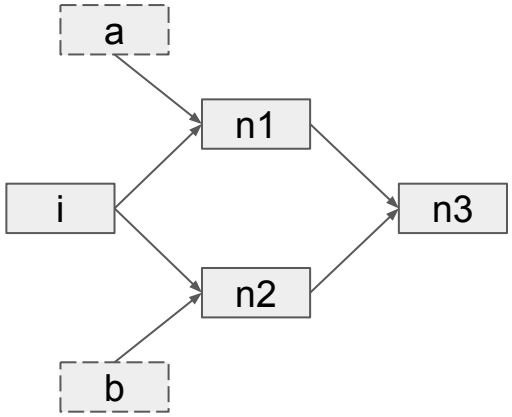
n1 = i + a
n2 = i + b
n3 = n1 + n2





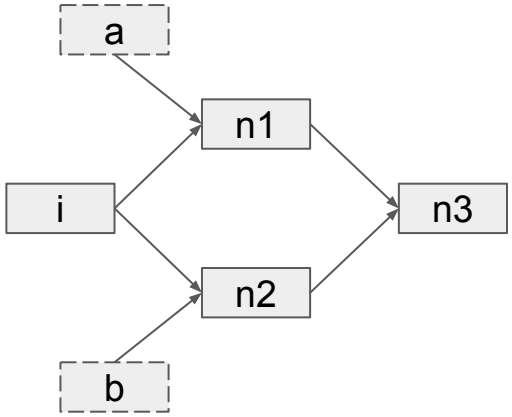
https://tensorflow.google.cn/api_docs/python/tf/GraphDef

TensorFlow基本概念




```
import tensorflow as tf


graph = tf.Graph()
with graph.as_default():
    i = tf.placeholder(tf.float32,
                       shape=(None, 1),
                       name="i")
    a = tf.Variable(tf.constant(10.0, shape=[1]), name="a")
    b = tf.Variable(tf.constant(20.0, shape=[1]), name="b")
    n1 = tf.add(i, a, name="n1")
    n2 = tf.add(i, b, name="n2")
    n3 = tf.add(n1, n2, name="n3")
```

TensorFlow基本概念



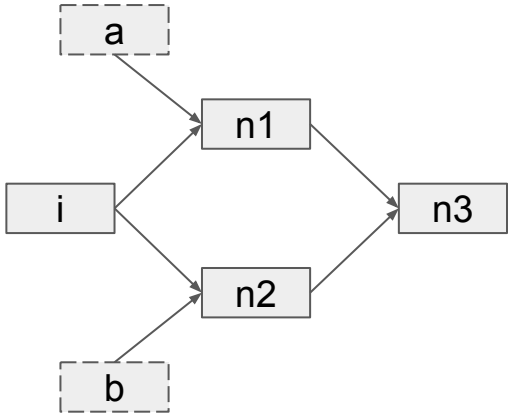

```
import tensorflow as tf


graph = tf.Graph()
with graph.as_default():
    i = tf.placeholder(tf.float32,
                       shape=(None, 1),
                       name="i")
    a = tf.Variable(tf.constant(10.0, shape=[1]), name="a")
    b = tf.Variable(tf.constant(20.0, shape=[1]), name="b")
    n1 = tf.add(i, a, name="n1")
    n2 = tf.add(i, b, name="n2")
    n3 = tf.add(n1, n2, name="n3")
```

Tensor name

Operator

Node name

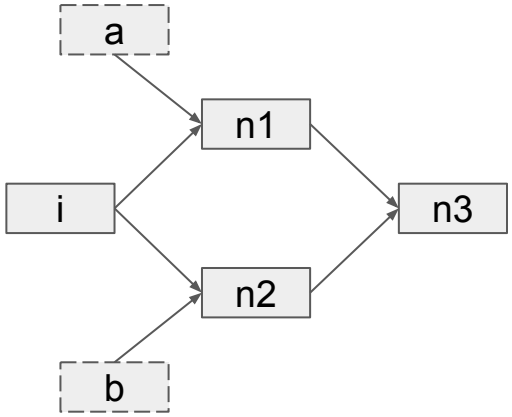
TensorFlow基本概念




```
import numpy as np
with tf.Session(graph=graph) as sess:
    sess.run(tf.global_variables_initializer())

    input_data = np.array([2]).reshape(-1, 1)
    output = sess.run(n3, feed_dict={i: input_data})
    print(output)
```

a = 10
b = 20
i = 2
n1 = i + a = 12
n2 = i + b = 22
n3 = n1 + n2 = 34

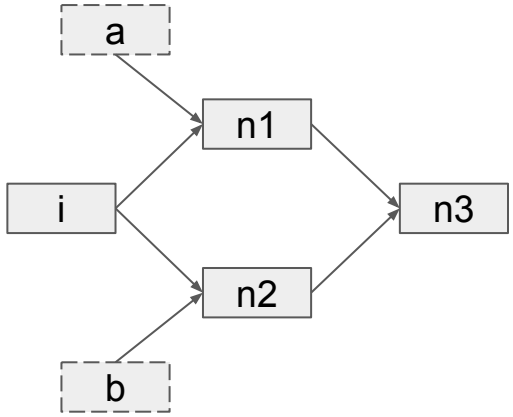
TensorFlow基本概念




```
import tensorflow as tf

graph = tf.Graph()
with graph.as_default():
    i = tf.placeholder(tf.float32,
                       shape=(None, 1),
                       name="i")
    a = tf.Variable(tf.constant(10.0, shape=[1]), name="a")
    b = tf.Variable(tf.constant(20.0, shape=[1]), name="b")
    n1 = tf.add(i, a, name="n1")
    n2 = tf.add(i, b, name="n2")
    n3 = tf.add(n1, n2, name="n3")
```

?

TensorFlow基本概念



[3] (1)

[3, 6, 8] (3)

[3]
[6]
[8]

3x1

shape=(None, 1)

shape=[1]

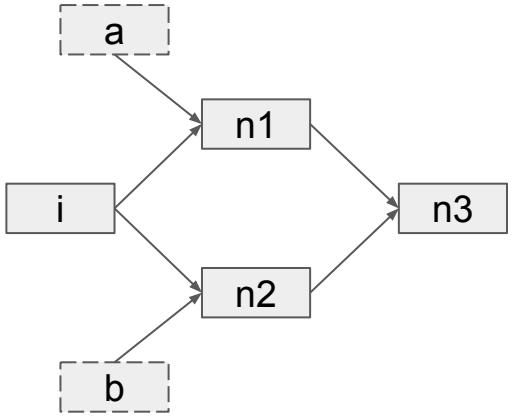
[3, 7, 4]
[6, 5, 3]

2x3

[3, 7, 4]
[6, 5, 3]

2x3x4

TensorFlow基本概念



[3] (1)

[3, 6, 8] (3)

[3]
[6]
[8]

3x1

shape=(None, 1)

shape=[1]

[3, 7, 4]
[6, 5, 3]

2x3

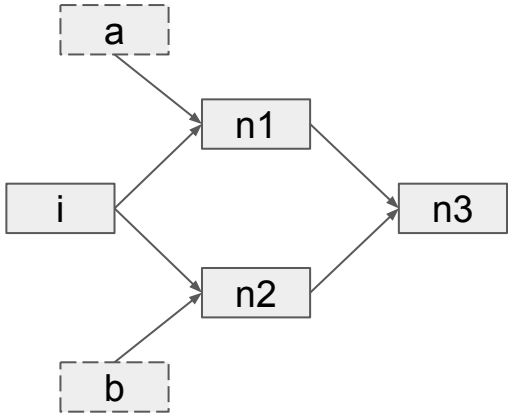
[3, 7, 4]
[6, 5, 3]

2x3x4

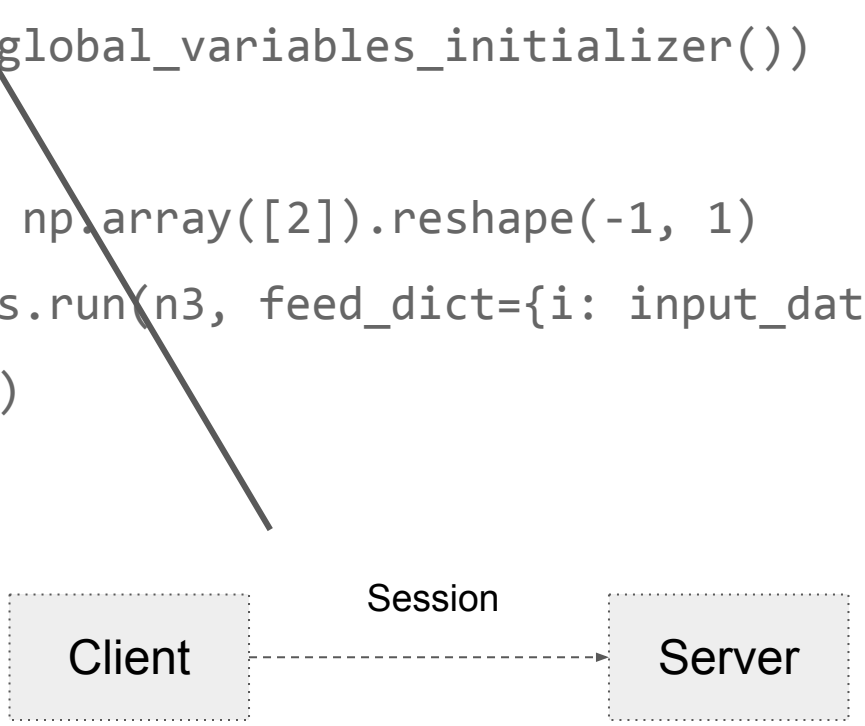
Tensor: 多维数组
Shape：维度信息

https://docs.scipy.org/doc/numpy-1.13.0/reference/arrays.ndarray.html
https://www.tensorflow.org/api_docs/python/tf/TensorShape

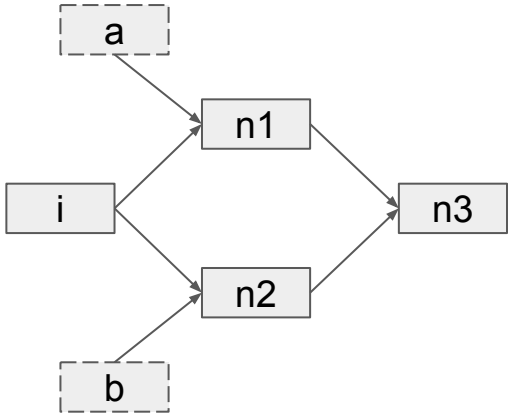
TensorFlow基本概念




```
import numpy as np
with tf.Session(graph=graph) as sess:
    sess.run(tf.global_variables_initializer())

    input_data = np.array([2]).reshape(-1, 1)
    output = sess.run(n3, feed_dict={i: input_data})
    print(output)
```



TensorFlow基本概念 回顾



Flow

Tensor

Session

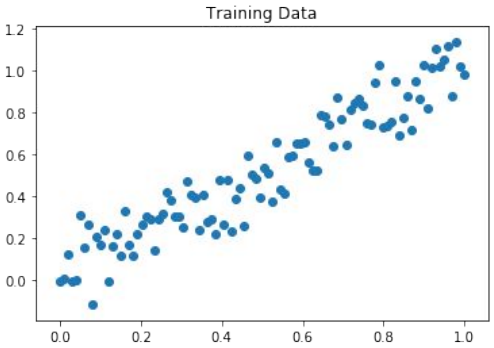
Graph

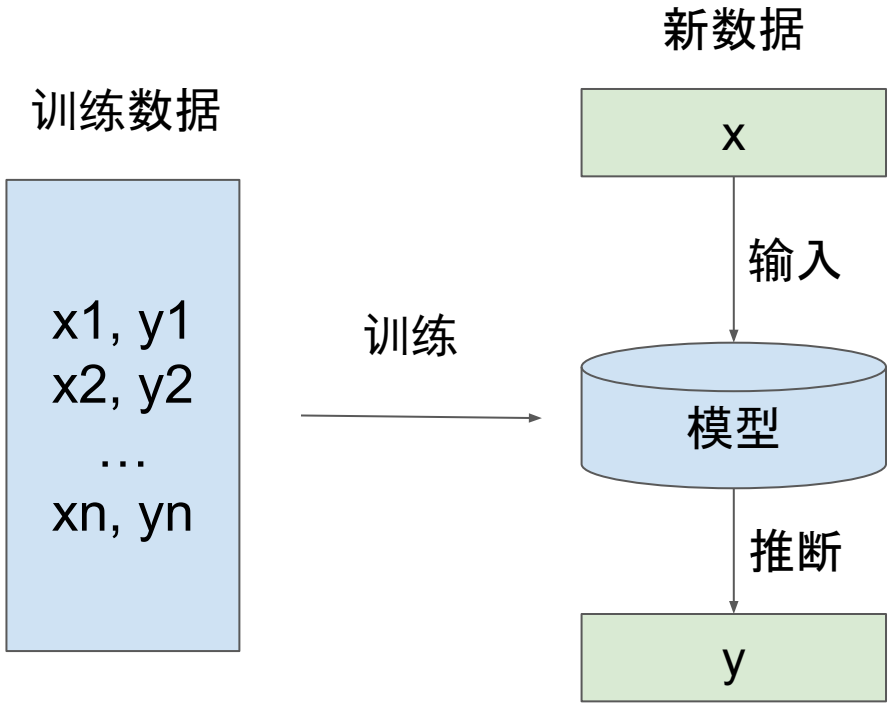
Op

Node

任务2 - **机器学习**
线性回归

任务2 大体结构





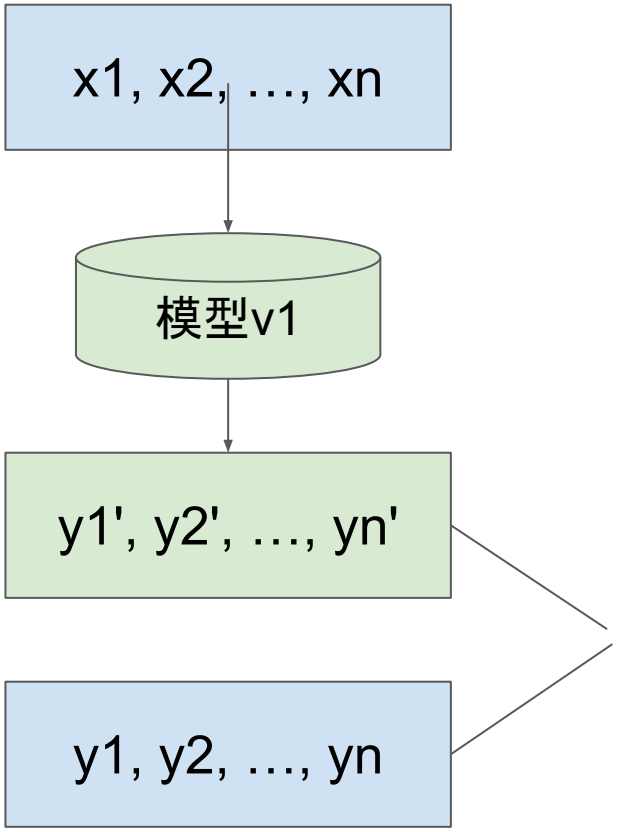
y=f(x)

根据数据，
训练出模型，
来逼近 f()

任务2 生成数据

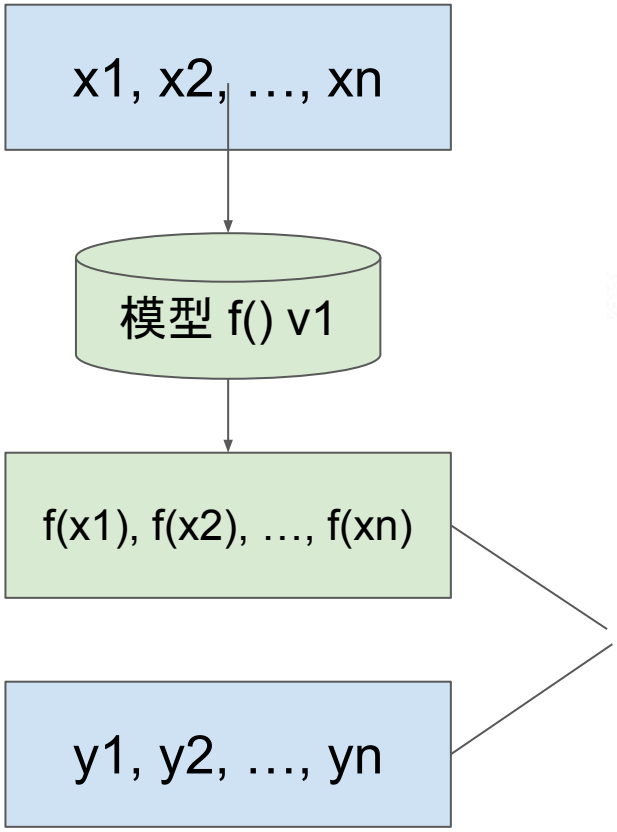
http://localhost:8888/notebooks/tflab/task2.ipynb

任务2 loss



误差多少？即loss。loss越小越好

任务2 loss

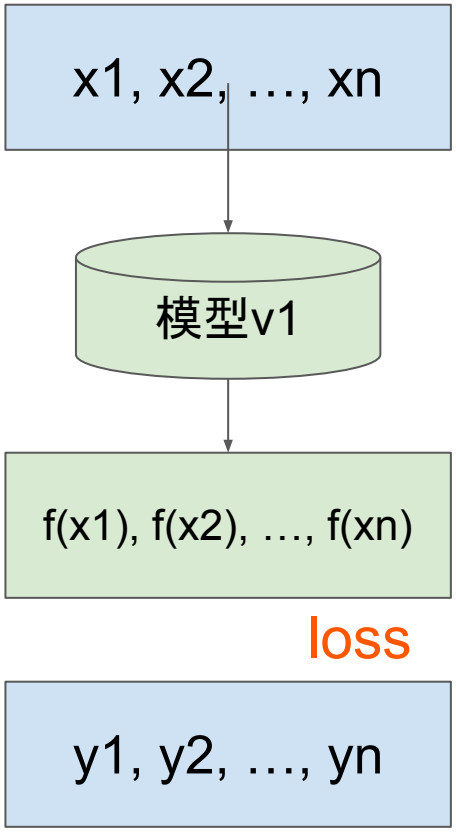


定义个loss，比如下式；
改进模型最小化这个loss即可

$$loss = (y_1\text{-}f(x_1))^2+(y_2\text{-}f(x_2))^2+\ldots+(y_n\text{-}f(x_n))^2$$

误差多少？即loss。loss越小越好

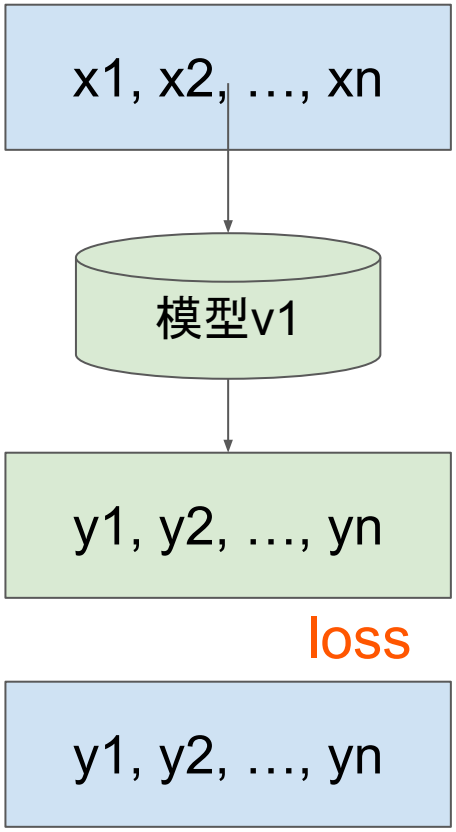
任务2 optimizer



如何改进模型最小化这个loss？

$$loss = (y_1\text{-}f(x_1))^2+(y_2\text{-}f(x_2))^2+\ldots+(y_n\text{-}f(x_n))^2$$

任务2 optimizer



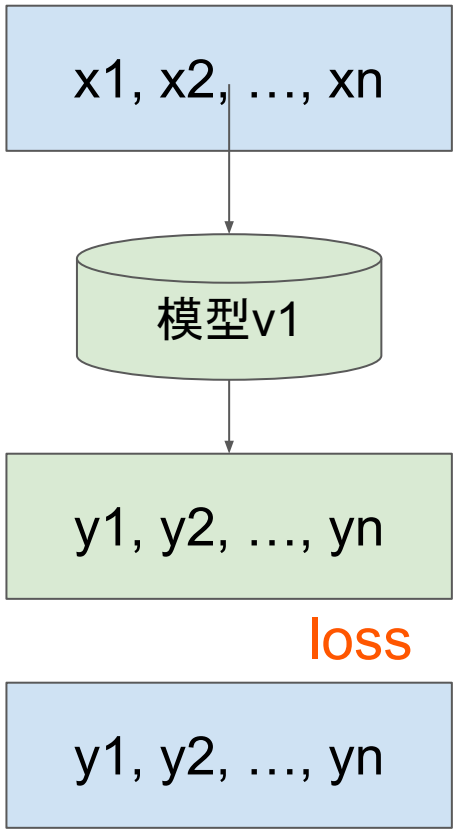
如何改进模型最小化这个loss？

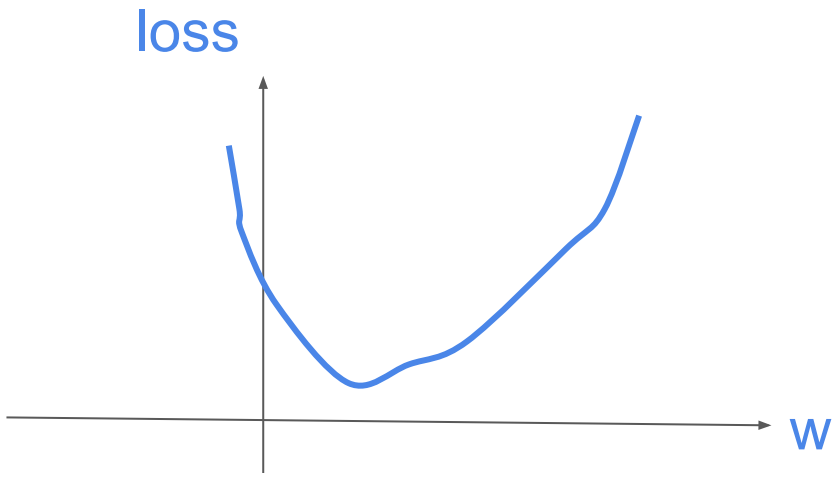
$$loss = (y_1\text{-}f(x_1))^2+(y_2\text{-}f(x_2))^2+\ldots+(y_n\text{-}f(x_n))^2$$

假设模型v1，即f(x)，是以w为参数的函数;
比如 f(x) = wx;

则 loss 实际上是以 w 为参数的函数;

任务2 optimizer

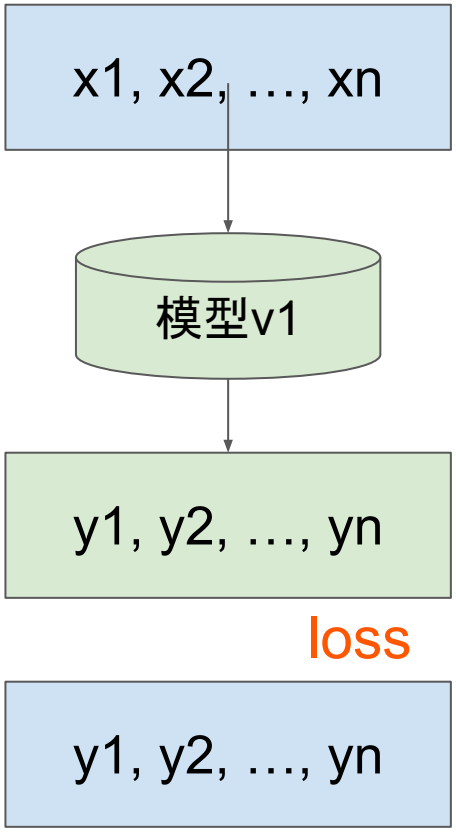


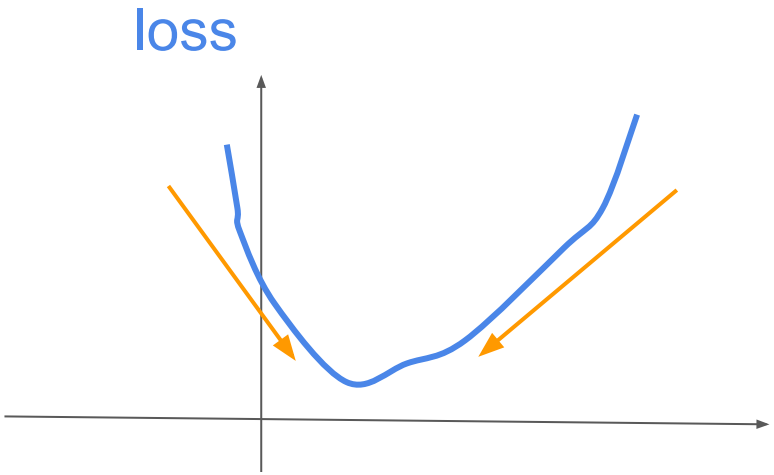


假设模型v1，即f(x)，是以w为参数的函数;
比如 f(x) = wx;

则 loss 实际上是以 w 为参数的函数;

任务2 optimizer





怎么最小化呢？沿着梯度下降的方向前进；

TensorFlow可以自动做这些事情

任务2 optimizer

loss

y1, y2, …, yn

TensorFlow可以自动做这些事情

任务2 回顾

Canned Estimators

Estimator | Keras

task2.ipynb

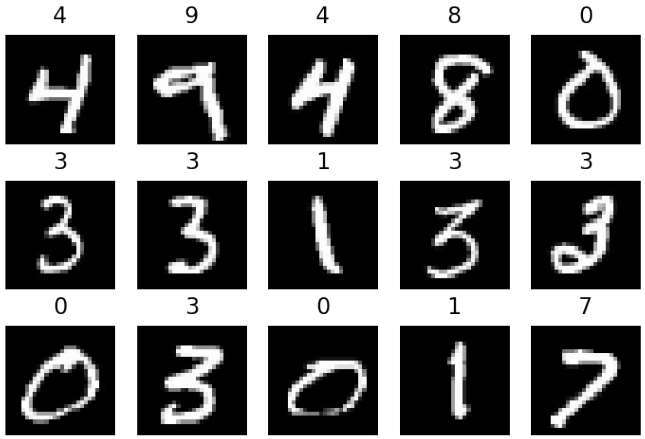
Layers

Python Frontend | C++ | Go | Java | ...

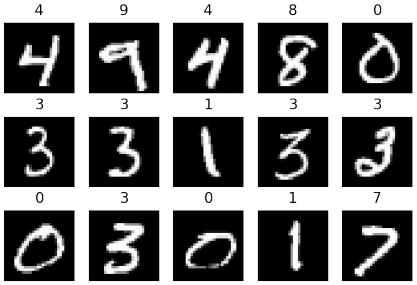
TensorFlow Distributed Execution Engine

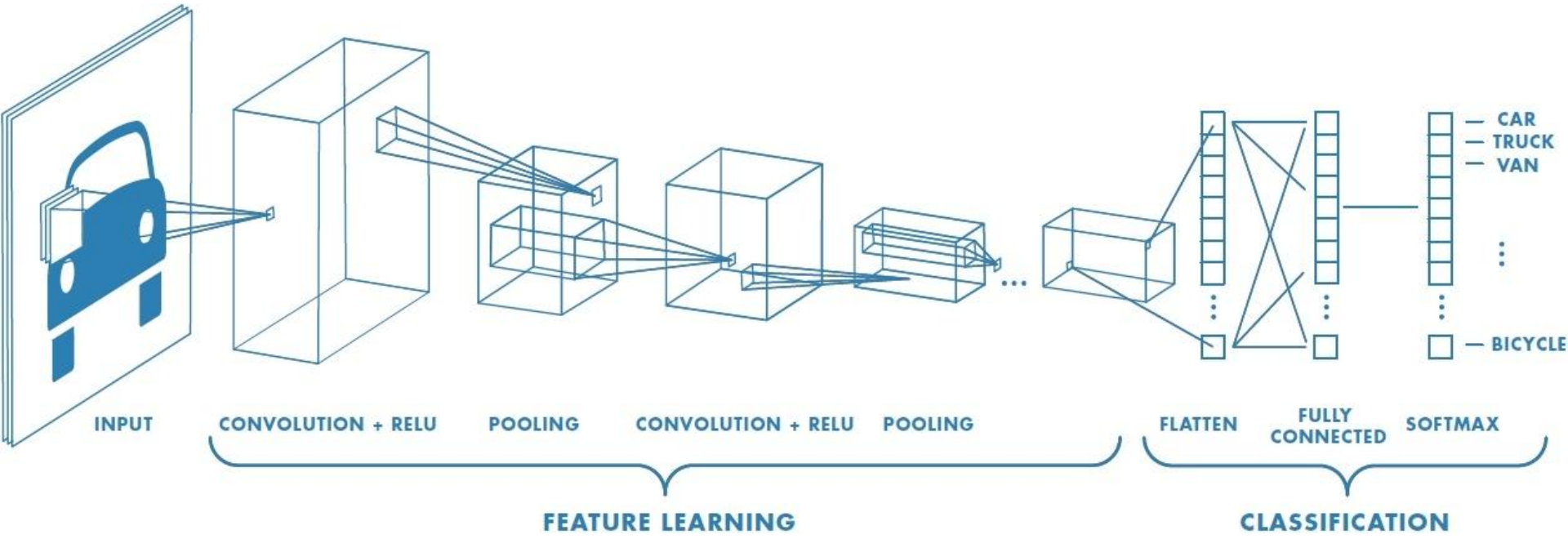
CPU | GPU | Android | iOS | ...

任务3 - **深度学习**
手写体数字识别



深度学习





卷积

DropOut

全连接

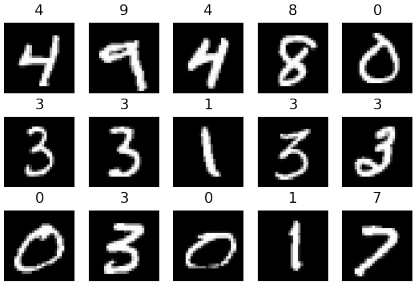
池化

激活

SoftMax

Image source

卷积 Conv



Input Volume (+pad 1) (7x7x3)

x[:,:,0]

0	0	0	0	0	0	0
0	0	1	2	0	1	0
0	0	0	2	0	0	0
0	0	1	2	2	0	0
0	2	2	1	2	2	0
0	1	2	1	1	2	0
0	0	0	0	0	0	0

x[:,:,1]

0	0	0	0	0	0	0
0	0	1	0	0	1	0
0	2	1	2	2	0	0
0	2	2	0	1	2	0
0	1	1	0	1	1	0
0	0	1	0	2	0	0
0	0	0	0	0	0	0

x[:,:,2]

0	0	0	0	0	0	0
0	1	0	2	1	1	0
0	1	2	2	1	1	0
0	0	0	2	2	2	0
0	1	0	0	0	1	0
0	1	1	0	1	0	0
0	0	0	0	0	0	0

Filter W0 (3x3x3)

w0[:,:,0]

1	0	0
-1	1	0
-1	1	1

w0[:,:,1]

1	1	0
1	1	-1
-1	-1	1

w0[:,:,2]

1	-1	-1
1	1	0
1	0	1

Bias b0 (1x1x1)

b0[:,:,0]

1

Filter W1 (3x3x3)

w1[:,:,0]

0	0	1
-1	0	-1
0	-1	-1

w1[:,:,1]

0	0	0
0	-1	0
0	0	1

w1[:,:,2]

-1	1	-1
0	1	-1
-1	0	-1

Bias b1 (1x1x1)

b1[:,:,0]

0

Output Volume (3x3x2)

o[:,:,0]

2	9	4
4	8	6
2	3	8

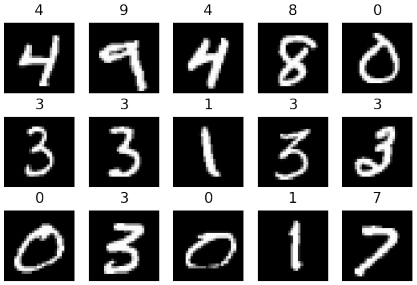
o[:,:,1]

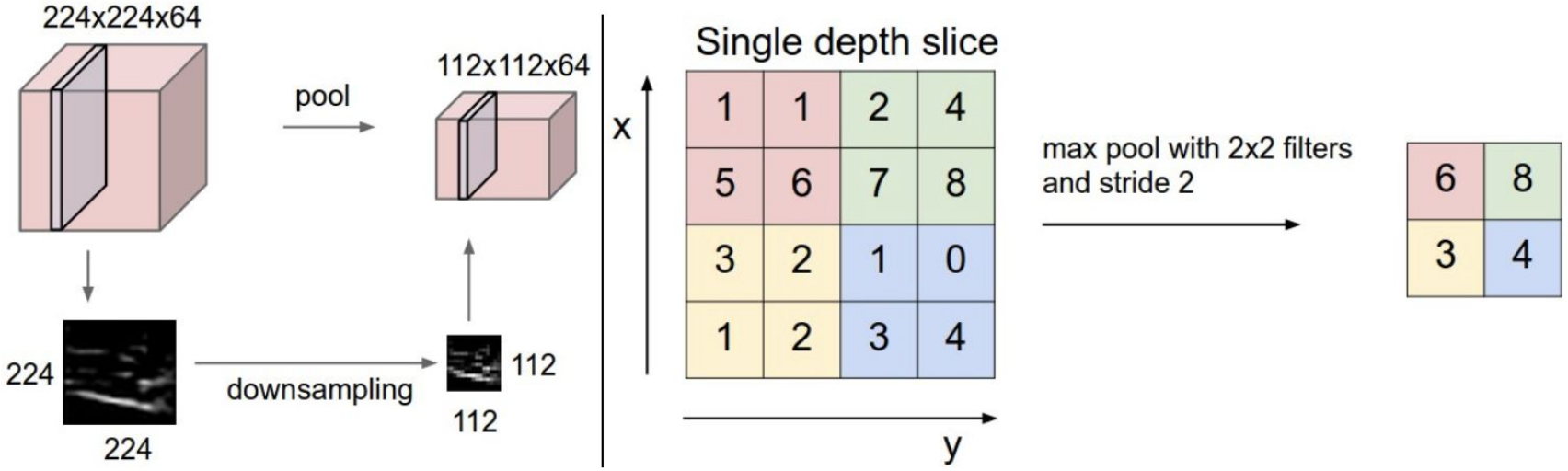
-1	-3	-1
-7	-6	-4
1	-2	0

toggle movement

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

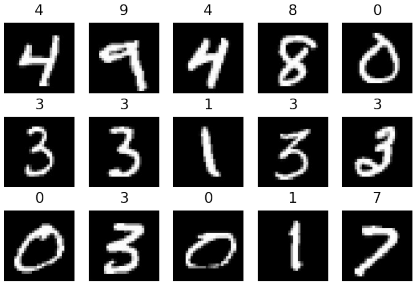
池化 Pooling

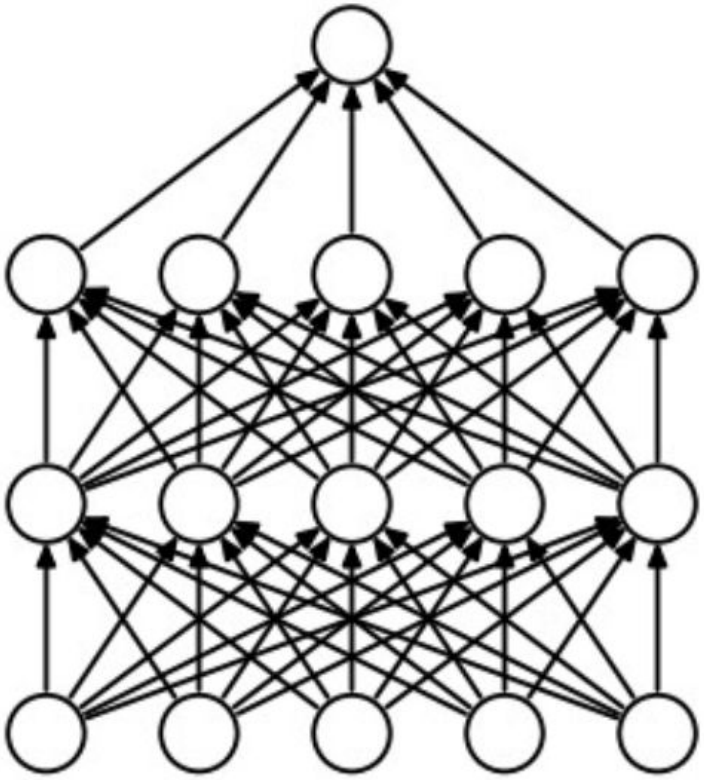




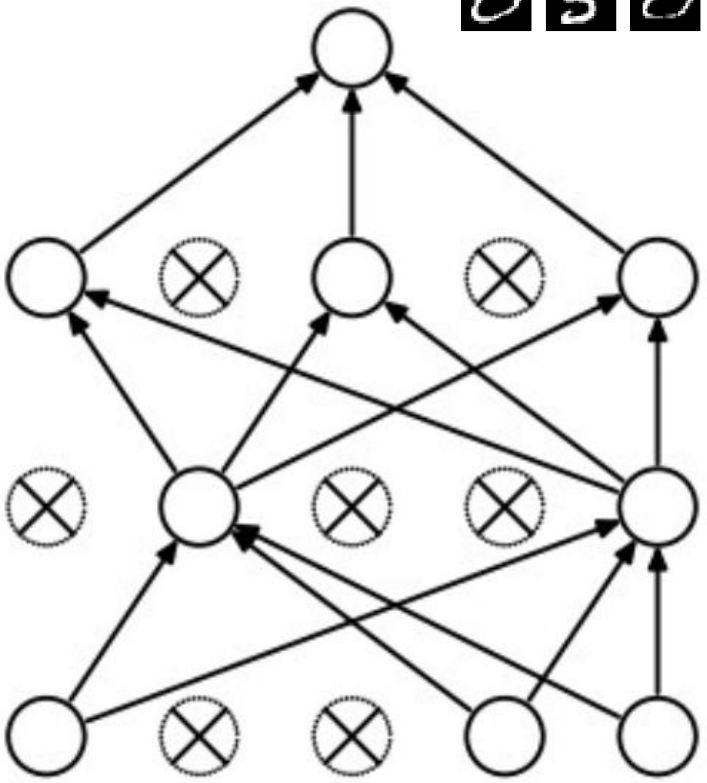
http://cs231n.github.io/convolutional-networks/

丢弃 Dropout



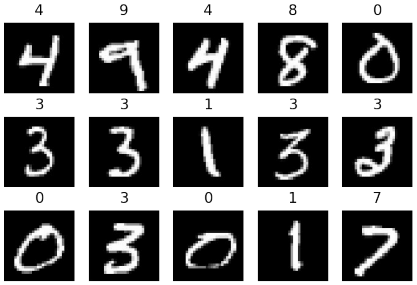


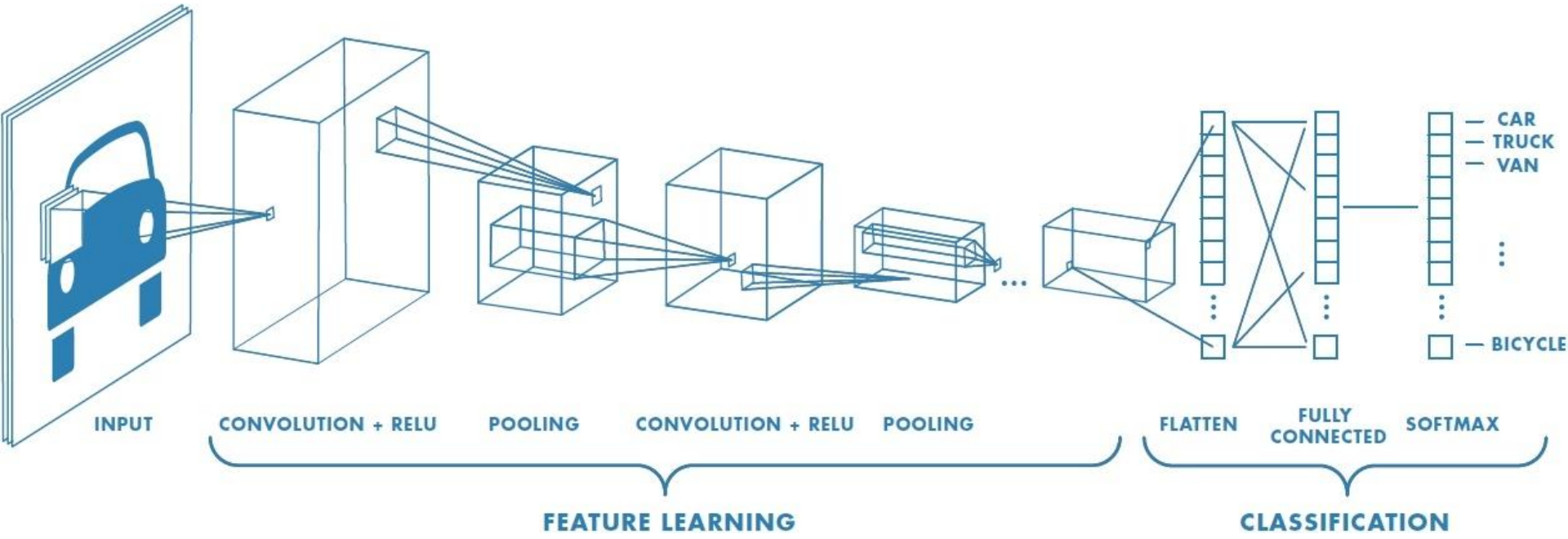
(a) Standard Neural Net



(b) After applying dropout.

深度学习回顾





卷积

DropOut

全连接

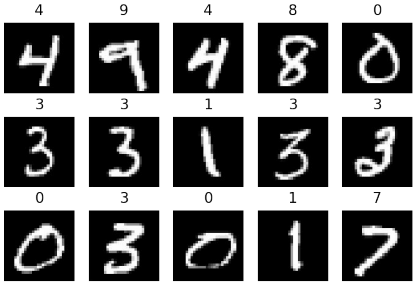
池化

激活

SoftMax

Image source

深度学习回顾



那就写点代码吧

http://localhost:8888/notebooks/tflab/task3.ipynb

卷积

DropOut

全连接

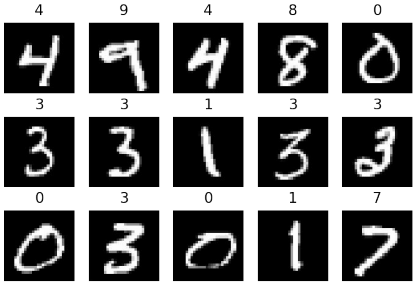
池化

激活

SoftMax

Image source

TensorFlow深度学习回顾



Canned Estimators

Estimator ← Keras

task3.ipynb

Layers

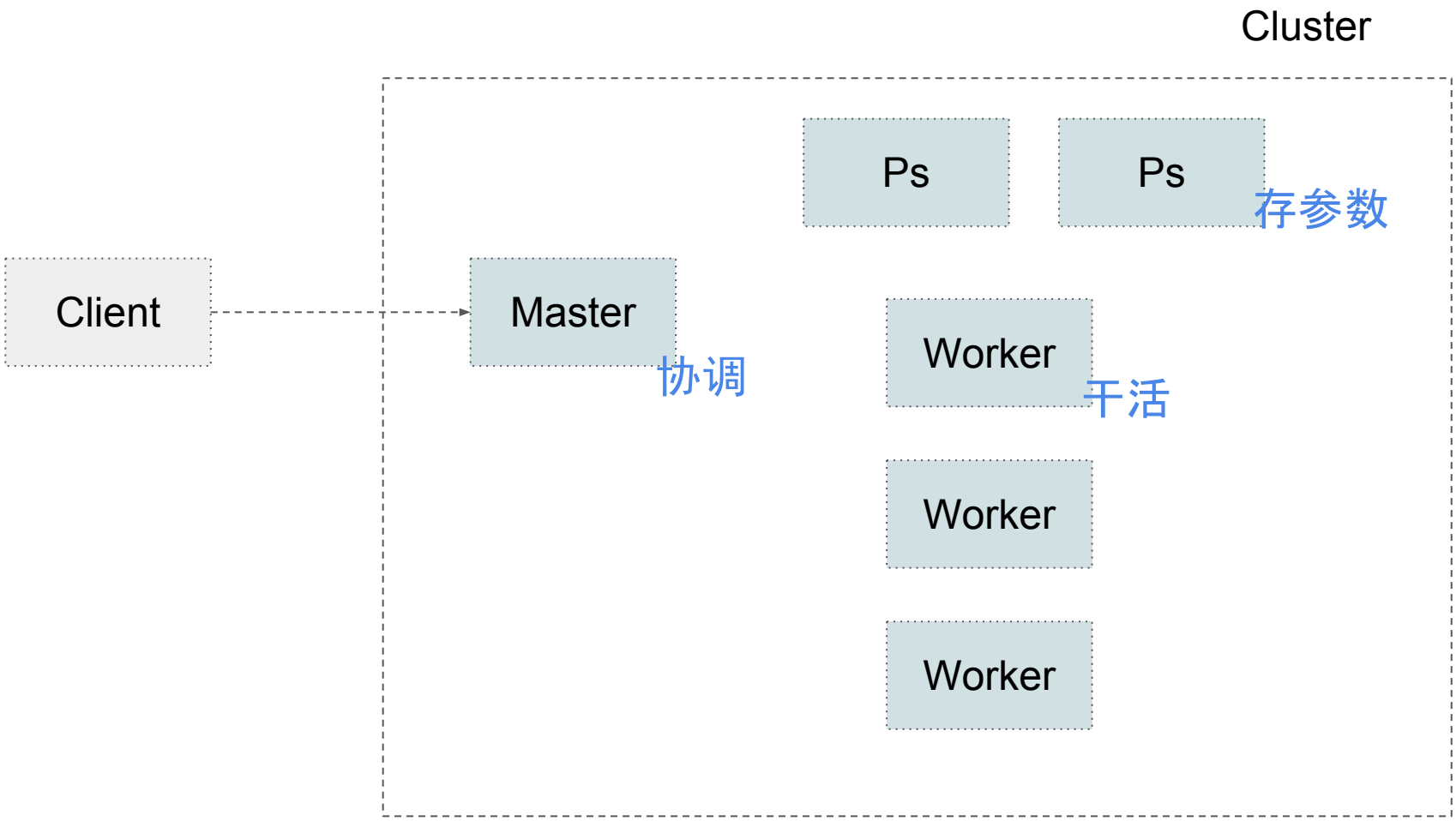
Python Frontend | C++ | Go | Java | ...

TensorFlow Distributed Execution Engine

CPU | GPU | Android | iOS | ...

任务4 - 分布式训练

TensorFlow分布式训练 - 概念

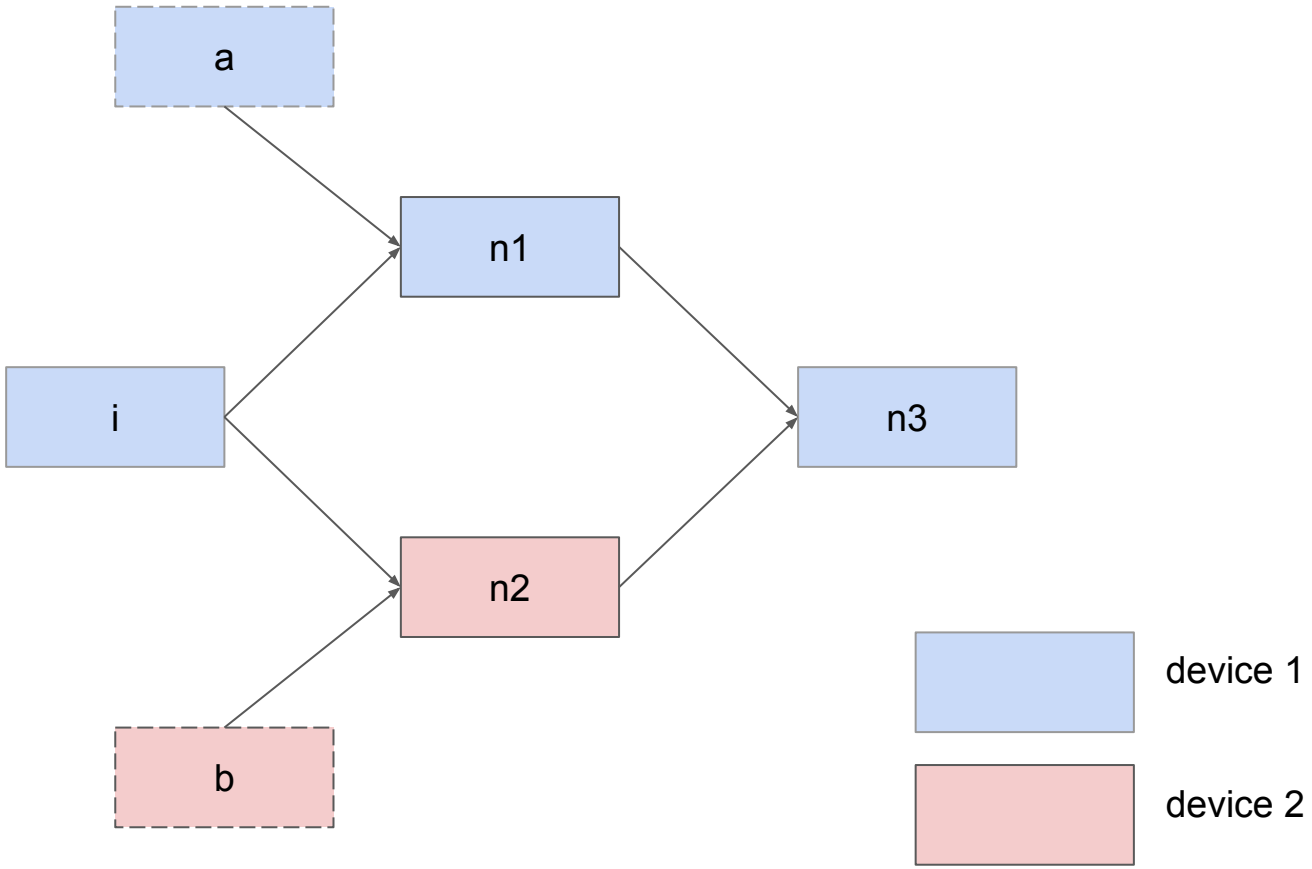


Device: /job:worker/task:2/gpu:0
/job:ps/task:1/cpu:0

TensorFlow分布式训练 - 方法

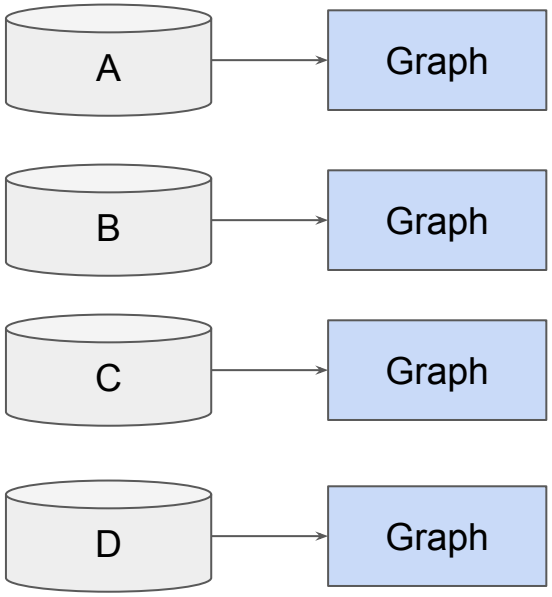
- 图并行 (graph-parallelism)
- 数据并行 (data-parallelism)
  - 图内复制 (In-graph replica)
  - 图间复制 (Between-graph replica)

TensorFlow分布式训练 - 图并行



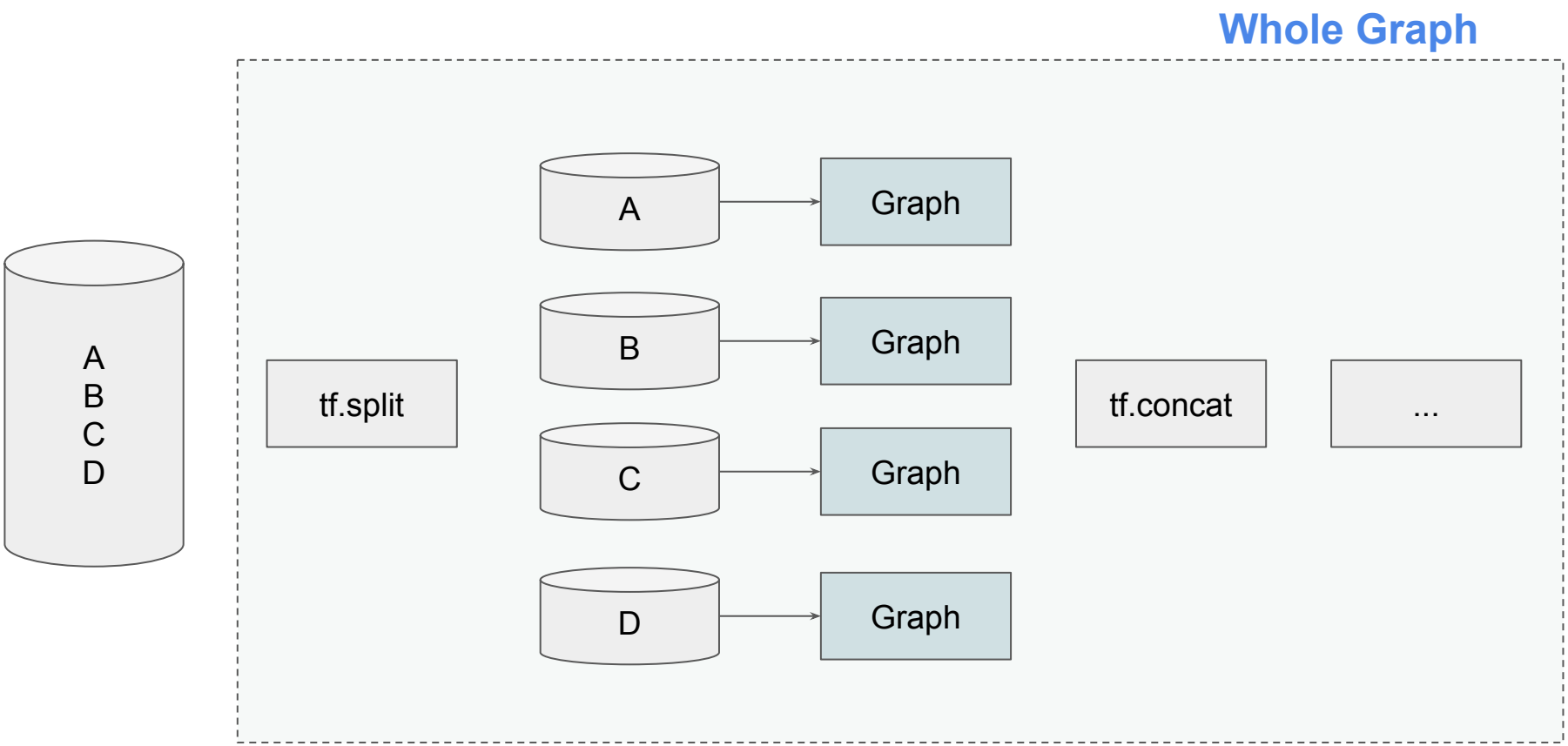
图的各个部分在不同的device上被执行

TensorFlow分布式训练 - 数据并行



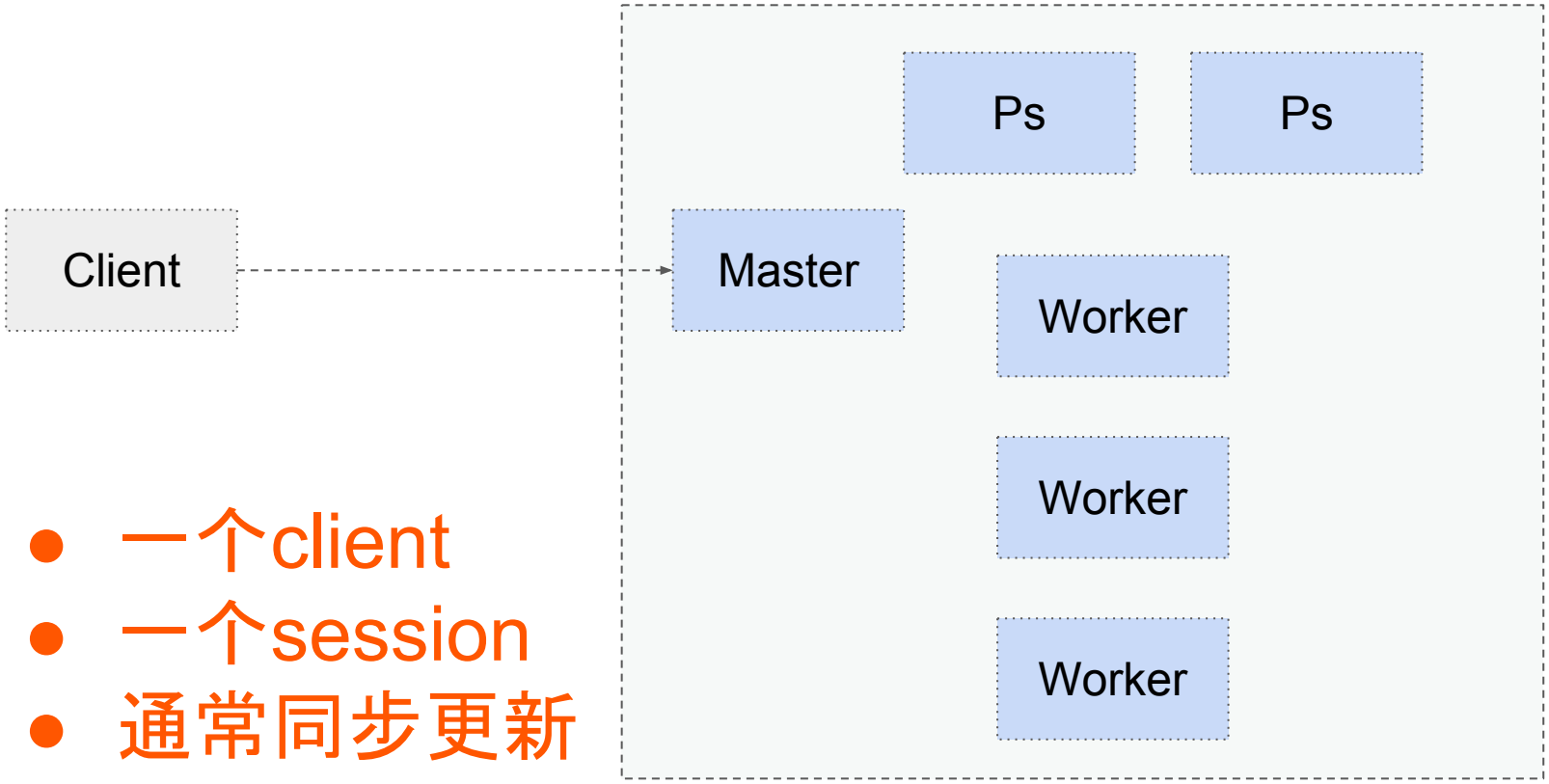
- 输入数据被分片
- 共享的图权重
- 一样的图结构

TensorFlow分布式训练 - 数据并行 - In-Graph



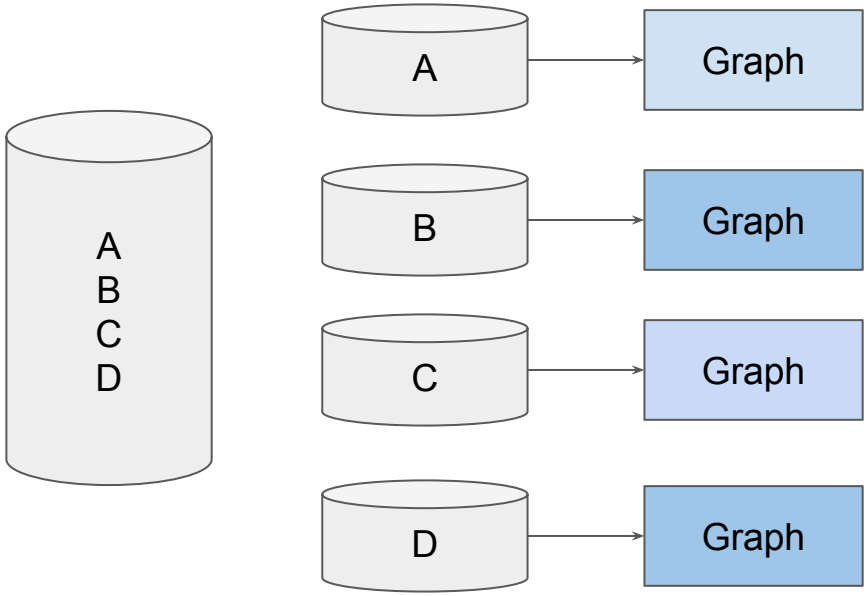
- Inside the whole graph, replica key part graph
- One client, one session

TensorFlow分布式训练 - 数据并行 - In-Graph



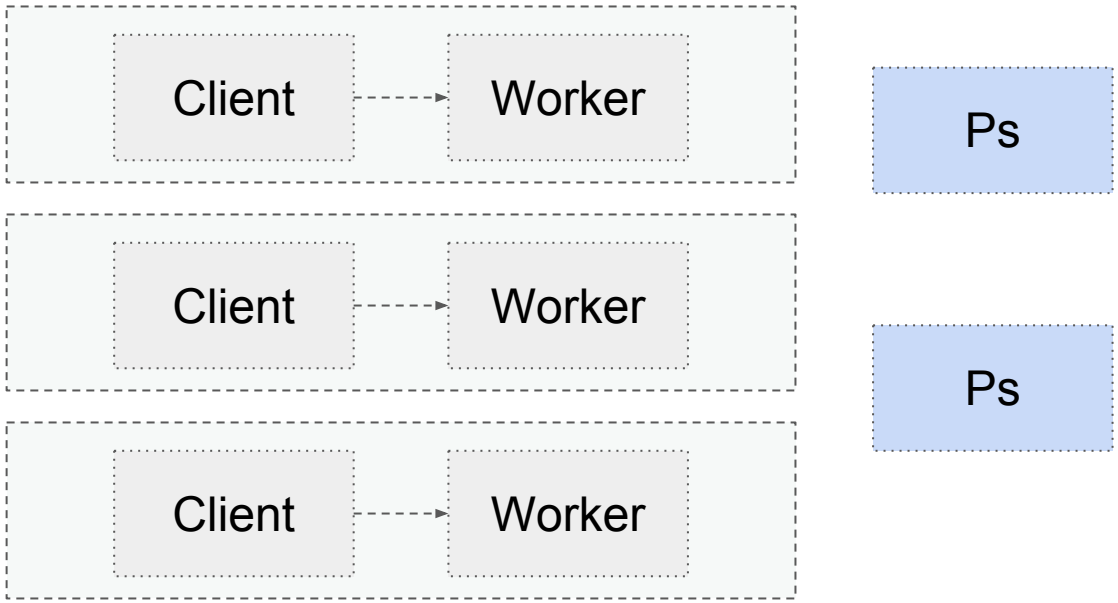
- 一个client
- 一个session
- 通常同步更新

TensorFlow分布式训练 - 数据并行 - Between-Graph



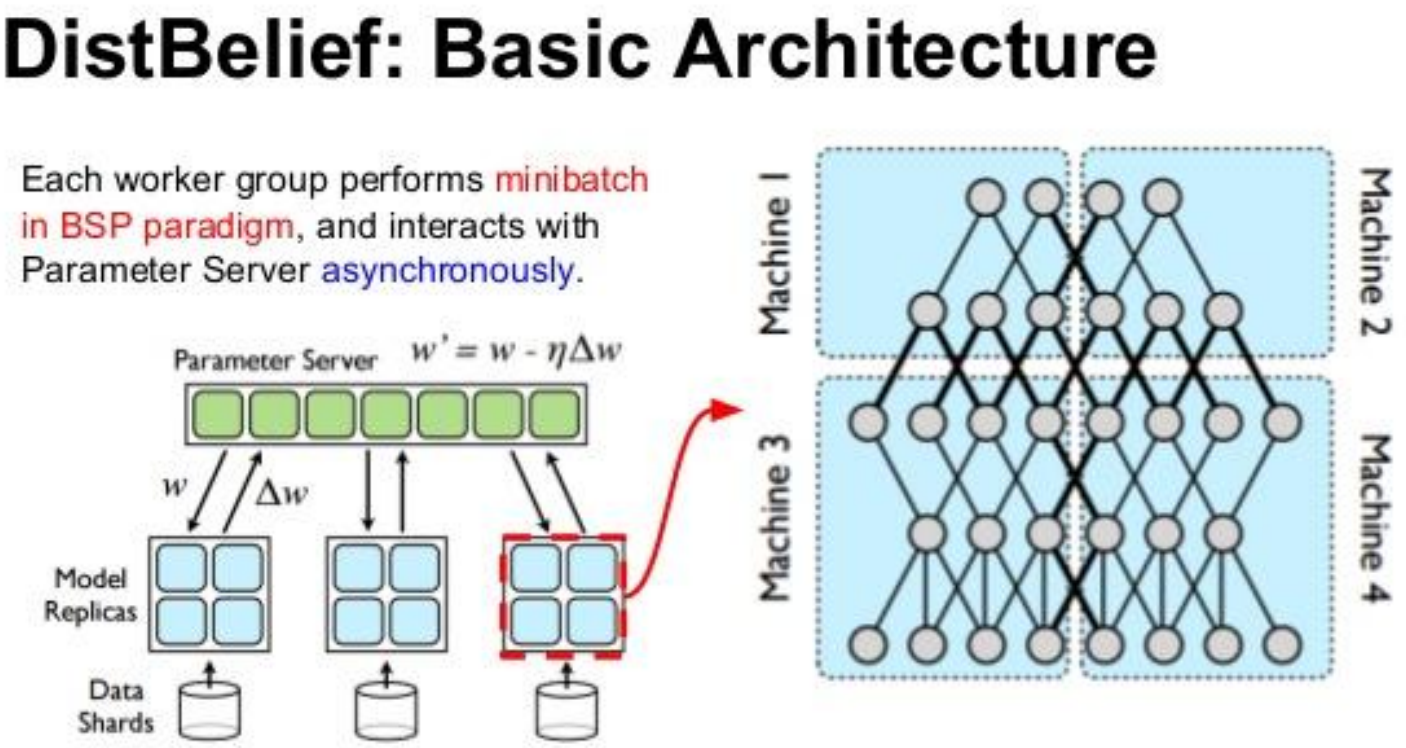
- Replica whole graph
- Multi client, multi session

TensorFlow分布式训练 - 数据并行 - Between-Graph



- 多个client
- 多个session
- 同步更新或异步更新

TensorFlow分布式训练



- 图并行
- Between-graph数据并行
- 异步更新

DistBelief是TensorFlow的上一代

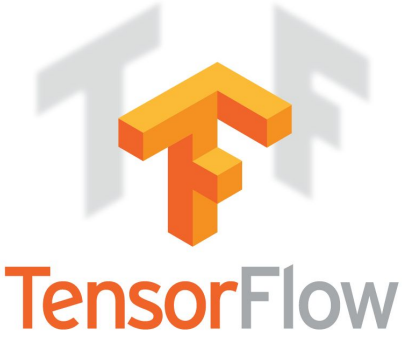
TensorFlow分布式训练

- 图并行
- Between-graph数据并行
- 异步更新

DistBelief是TensorFlow的上一代

社区及谷歌招聘介绍

社区介绍



developer.google.cn

chinagdg.org

tensorflow.google.cn

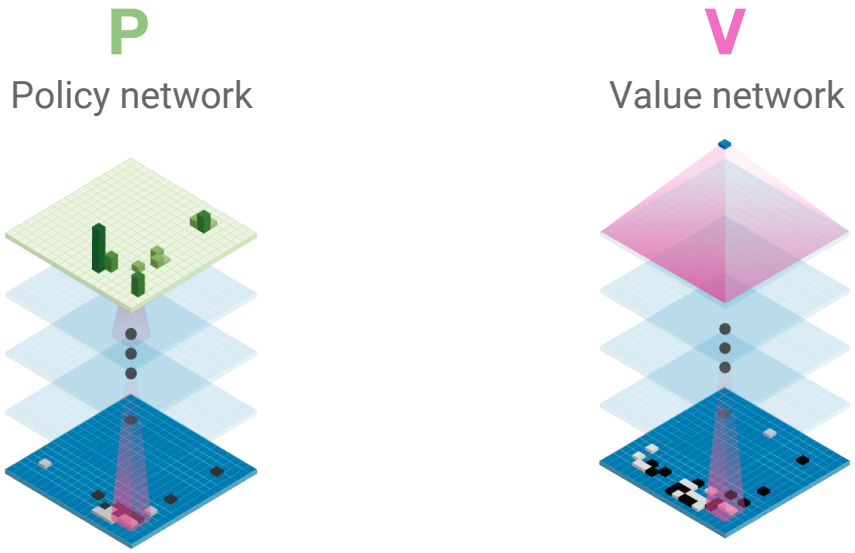
tensorflow-china@googlegroups.com

谷歌招聘介绍

http://careers.google.cn/

福利时间 AlphaGo Zero

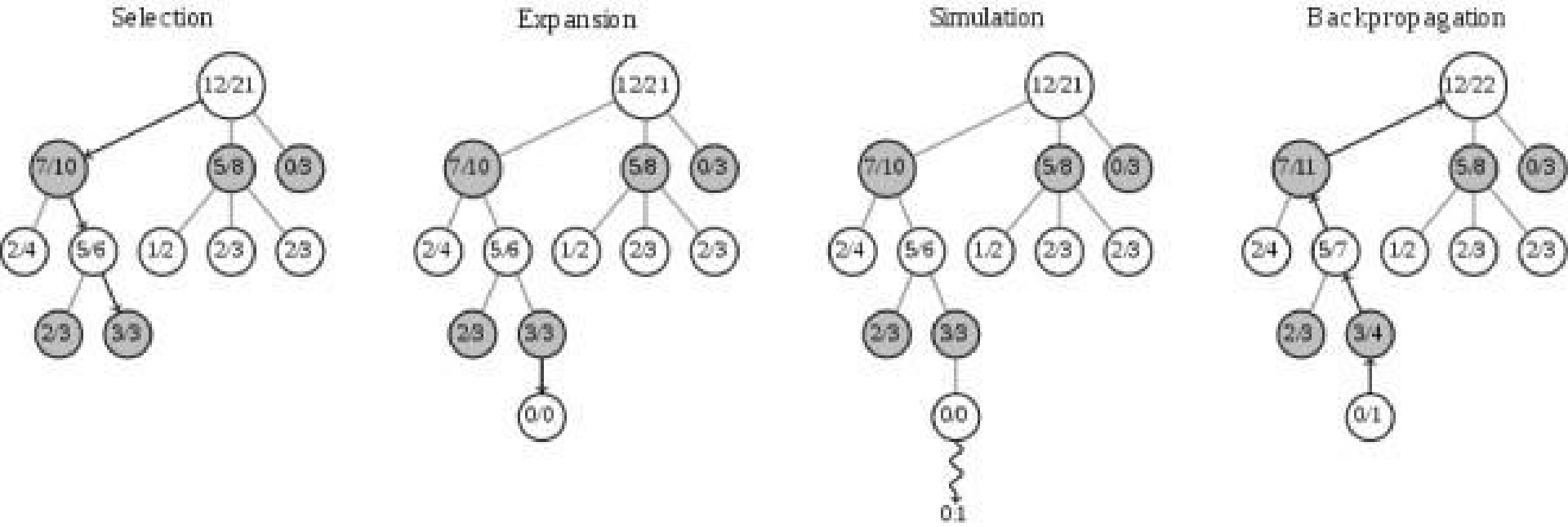
AlphaGo Zero 评估走法和评估局势



合并成一个网络

$$(\mathbf{p}, v) = f_\theta(s)$$

AlphaGo Zero 自我博弈



AlphaGo Zero