深度学习零基础入门及前沿介绍

吴 伟
lazyparser@gmail.com
中国科学院软件研究所

关于我

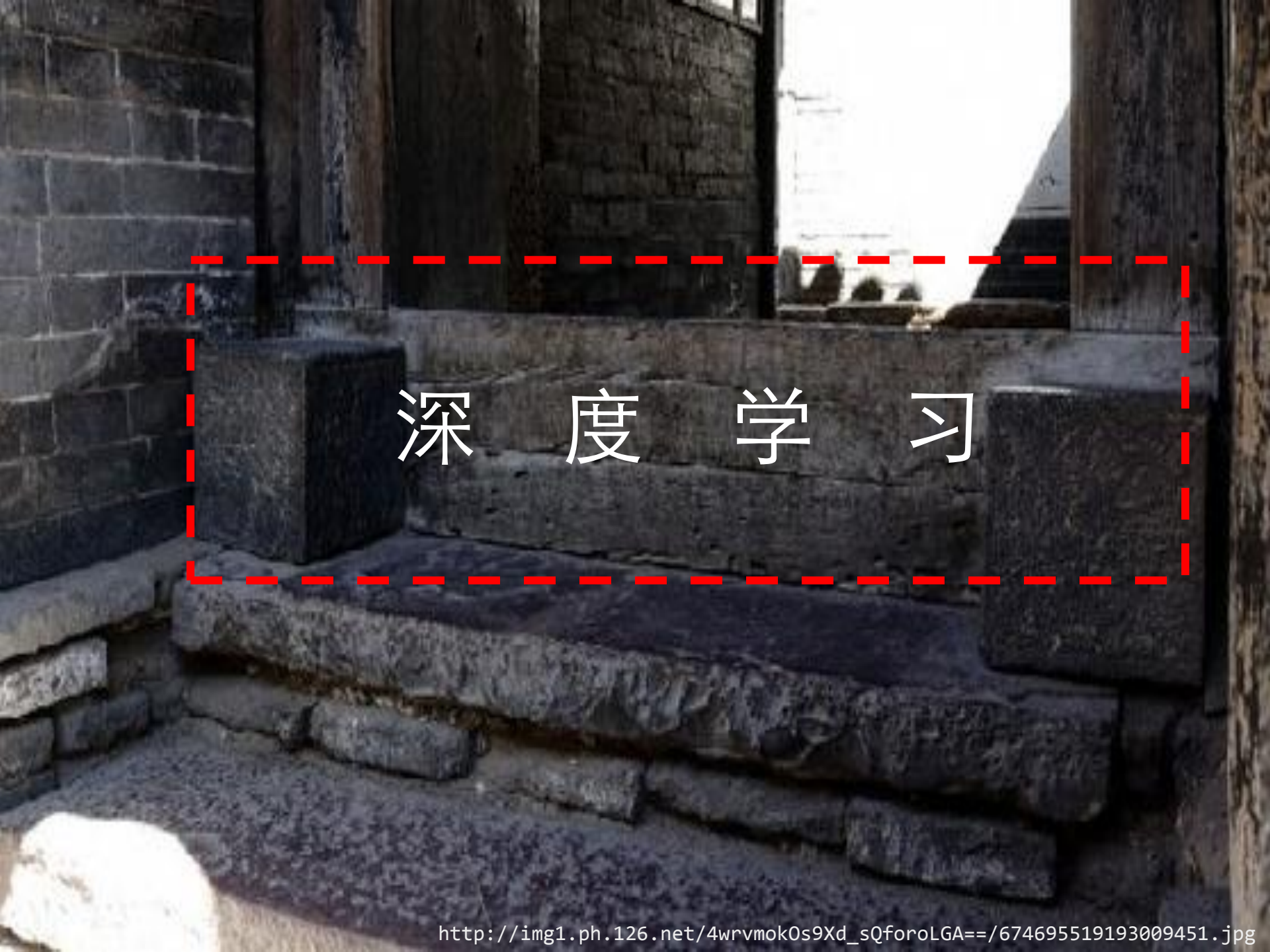
软件研究所 | 神经芯片 | 即时编译 | 机器人 | 人脸识别

开源贡献者 | HelloGCC | Mozilla | MXNet | TK1

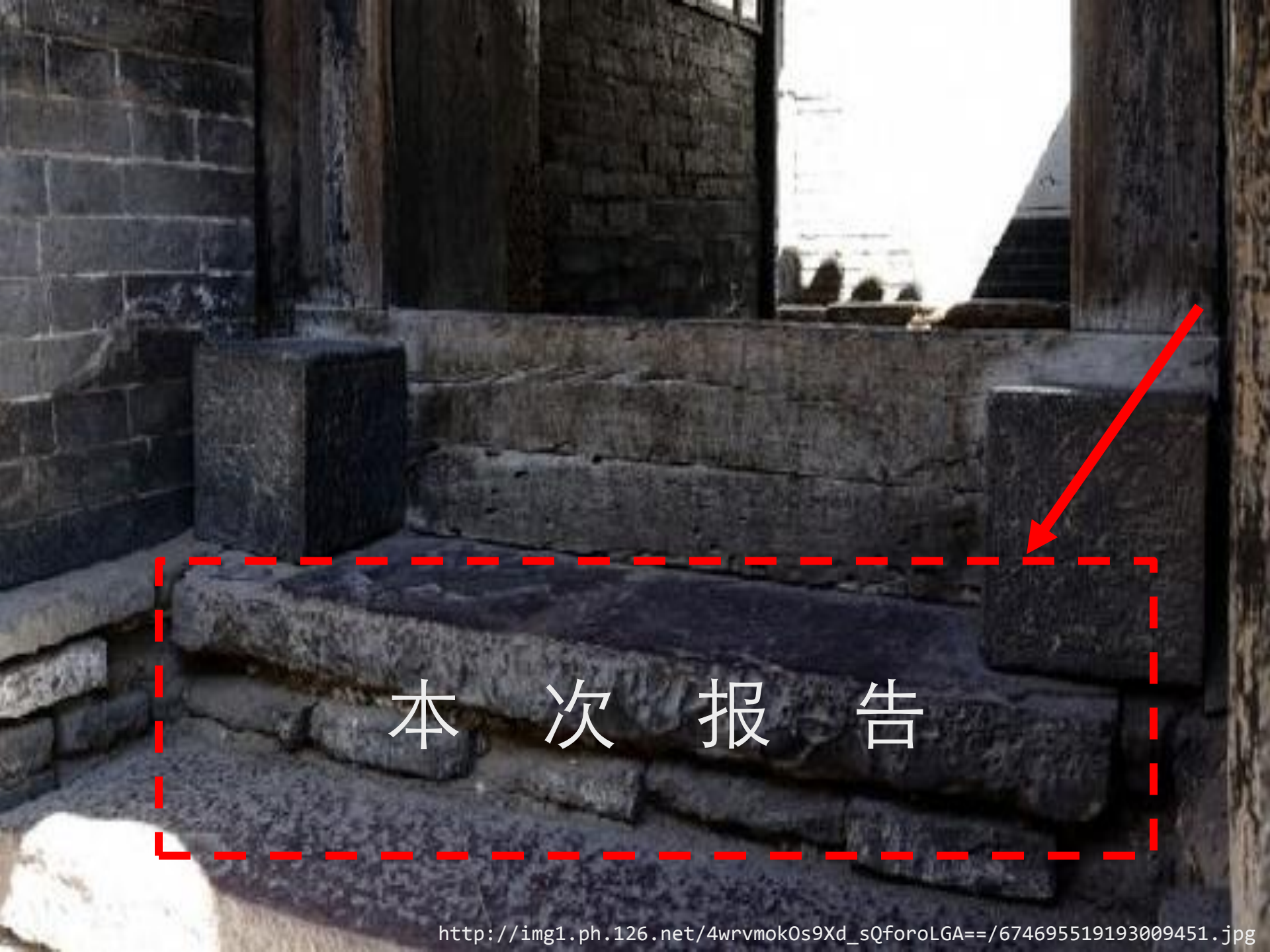
关于这次报告

数学和英语不太好的工程师，

深度学习如何入门？



http://img1.ph.126.net/4wrvmokOs9Xd_sQforoLGA==/674695519193009451.jpg

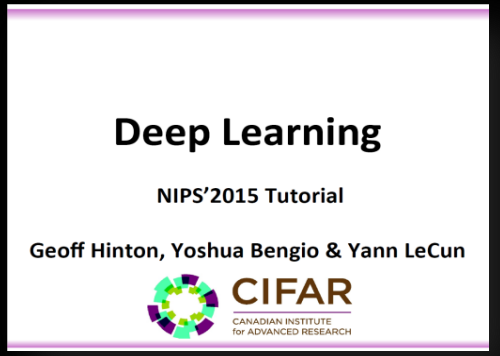


http://img1.ph.126.net/4wrvmokOs9Xd_sQforoLGA==/674695519193009451.jpg

基础 | 前沿 | 讨论

深度学习的定义随着时间在演变

Deep Learning: machine learning algorithms based on learning multiple levels of representation / abstraction.



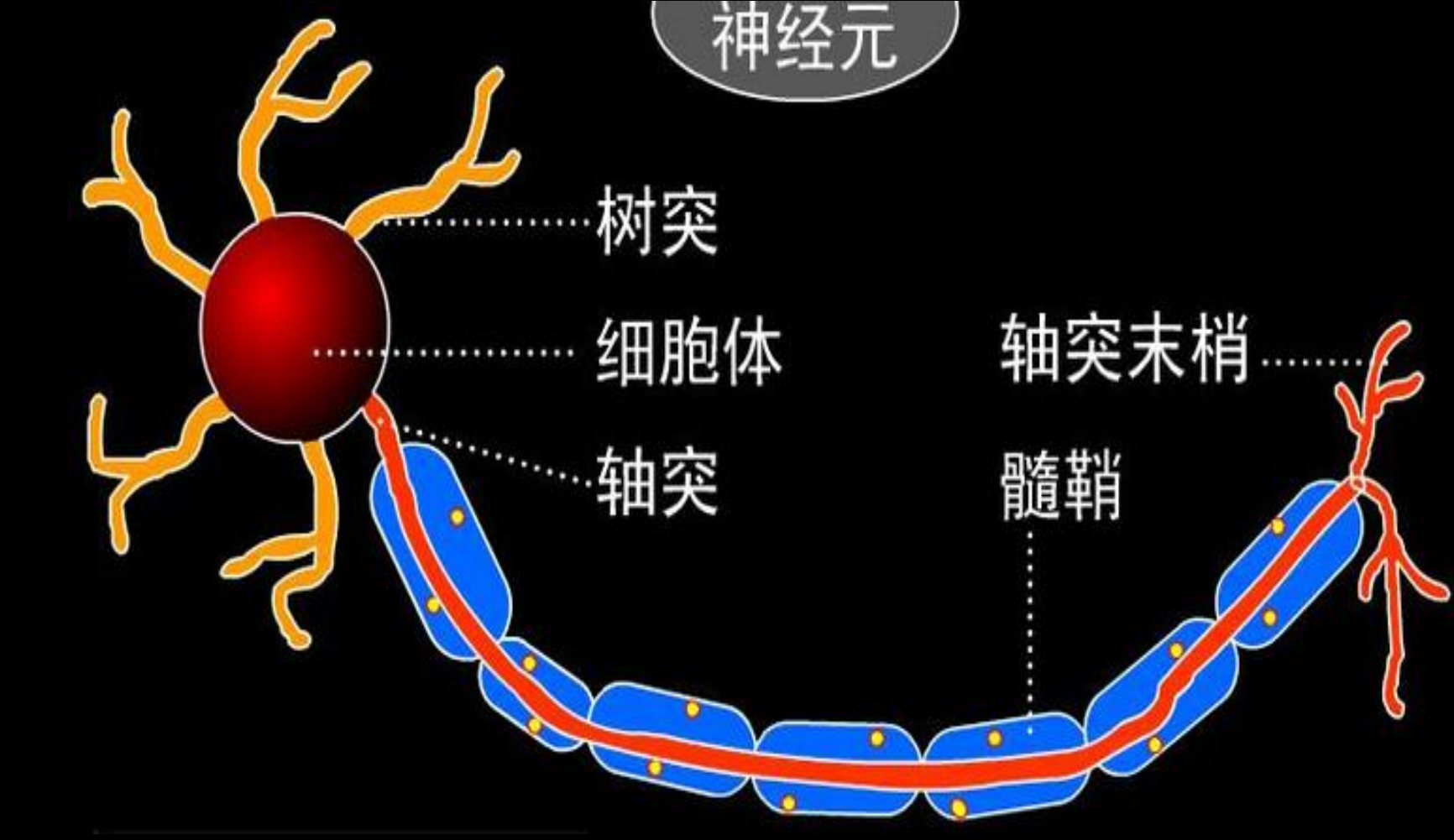
先缩小一下入门的范围

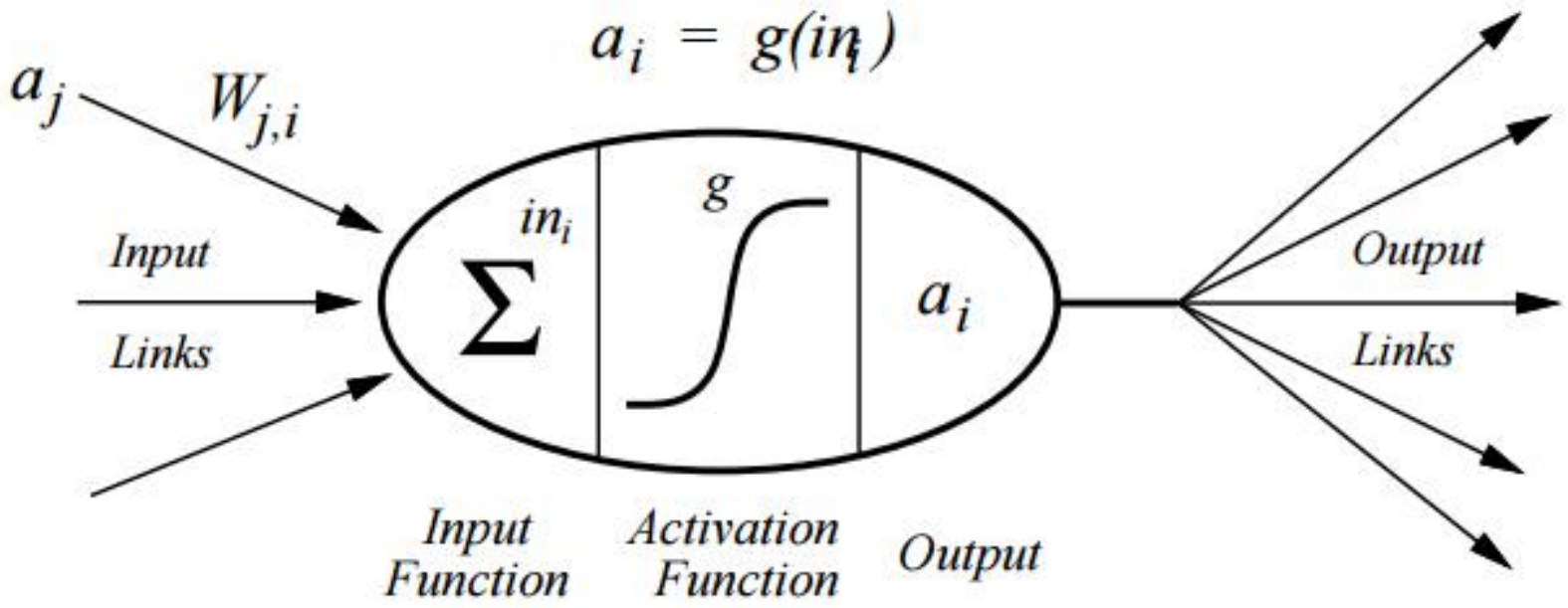
深度学习：深度神经网络

先缩小一下入门的范围

Neural Networks

深度学习： 深度神经网络





$$a_i = g(\sum_j W_{j,i} a_j)$$

https://page.mi.fu-berlin.de/rojas/neural/chapter/K4.pdf
https://appliedgo.net/perceptron/
https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15381-f01/www/handouts/110601.pdf

$$p(\mathbf{v}) = \sum_{\mathbf{h}} \frac{e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{u},\mathbf{g}} e^{-E(\mathbf{u},\mathbf{g})}}. \tag{1}$$

$$E(\mathbf{v},\mathbf{h}) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} w_{ij} v_i h_j, \tag{2}$$

$$p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h}) \quad \text{and} \quad p(v_i = 1|\mathbf{h}) = \text{sigm}\left(a_j + \sum_j h_j w_{ij}\right),$$

$$p(\mathbf{h}|\mathbf{v}) = \prod_j p(h_j|\mathbf{v}) \quad \text{and} \quad p(h_j = 1|\mathbf{v}) = \text{sigm}\left(b_j + \sum_i v_i w_{ij}\right), \tag{3}$$

$$p(v_i = x|\mathbf{h}) = \frac{1}{\sigma_i\sqrt{2\pi}} \cdot e^{-\frac{\left(x - a_i - \sigma_i \sum_j w_{ij} h_j\right)^2}{2\sigma_i^2}},$$

$$p(h_j = 1|\mathbf{v}) = \text{sigm}\left(b_j + \sum_i \frac{v_i}{\sigma_i} w_{ij}\right). \tag{4}$$

$$p(\mathbf{v}) = \sum_{\mathbf{h}} \frac{e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{u},\mathbf{g}} e^{-E(\mathbf{u},\mathbf{g})}}. \tag{1}$$

$$E(\mathbf{v},\mathbf{h}) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} w_{ij} v_i h_j, \tag{2}$$

$$p(\mathbf{v}|\mathbf{h}) \qquad\qquad ij \Big),$$

$$p(\mathbf{h}|\mathbf{v}) \qquad\qquad ij \Big), \tag{3}$$

看不懂没关系，不影响入门

其实我到现在也没看懂 ☺

$$p(v_i = x|\mathbf{h}) = \frac{1}{\sigma_i\sqrt{2\pi}} \cdot e^{-\frac{\left(x - a_i - \sigma_i \sum_j w_{ij} h_j\right)^2}{2\sigma_i^2}},$$

$$p(h_j = 1|\mathbf{v}) = \mathrm{sigm}\left(b_j + \sum_i \frac{v_i}{\sigma_i} w_{ij}\right). \tag{4}$$

故事背景

老板希望设计开发一款**人工智能**房价预测机，

利用当天股价的涨跌来预测第二天房价的涨跌。

故事背景 找来CTO实现这个app：

```
if (昨天股价涨)
then 显示房价涨;
else 显示房价不会涨;
```

产品原型

```
if (x > 0)
    y = 1
else /* x <= 0 */
    y = 0
```

产品原型

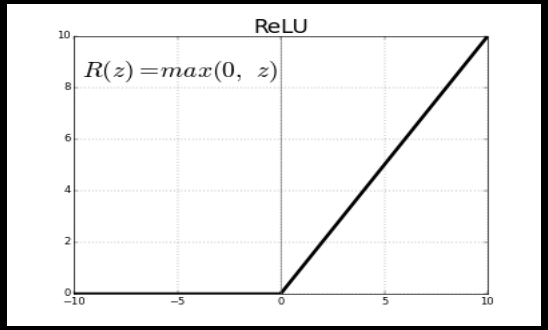
$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

```
if (x > 0)
   y = 1
else /* x <= 0 */
   y = 0
```

产品原型（高级版）

```
if (x > 0)
    y = x
else /* x <= 0 */
    y = 0
```

产品原型（高级版）

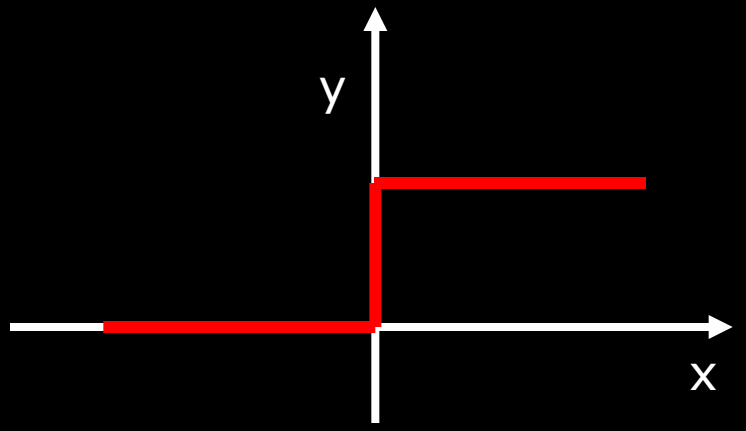


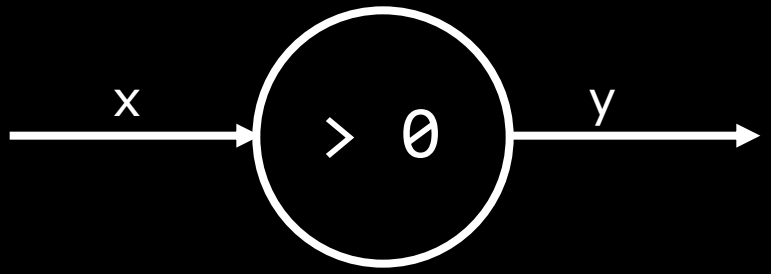

```
if (x > 0)
    y = x
else /* x <= 0 */
    y = 0
```

产品原型 1.0

Heaviside

```
if (x > 0)
    y = 1
else
    y = 0
```





故事背景

原型很快就做好了，如何评价预测的效果呢?

用历史数据来评价测试准不准

11.1	11.2	11.3	11.4	……	11.30
股价	跌	跌	涨	…….	跌
房价	涨	涨	涨	…	涨

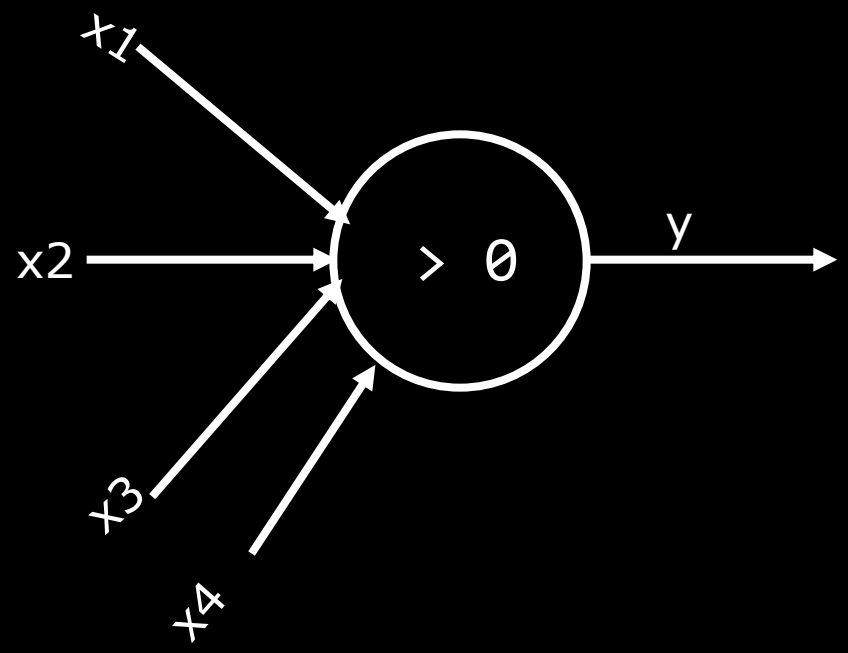
故事背景

显然，预测非常的不准 ☹

CTO觉得应该把油价、汇率、金价都考虑进去

产品原型 2.0

```
foo(x1,x2,x3,x4) {
    return x1+x2+x3+x4;
}
if (foo(x) > 0)
    y = 1
else
    y = 0
```



Credit http://taop.marchtea.com/07.02.svm.html

故事背景

产品 2.0 版还是不准

老板：股价、油价、汇率、金价对房价的影响程度不一样，也就是权重（weights）是不同的。

故事背景

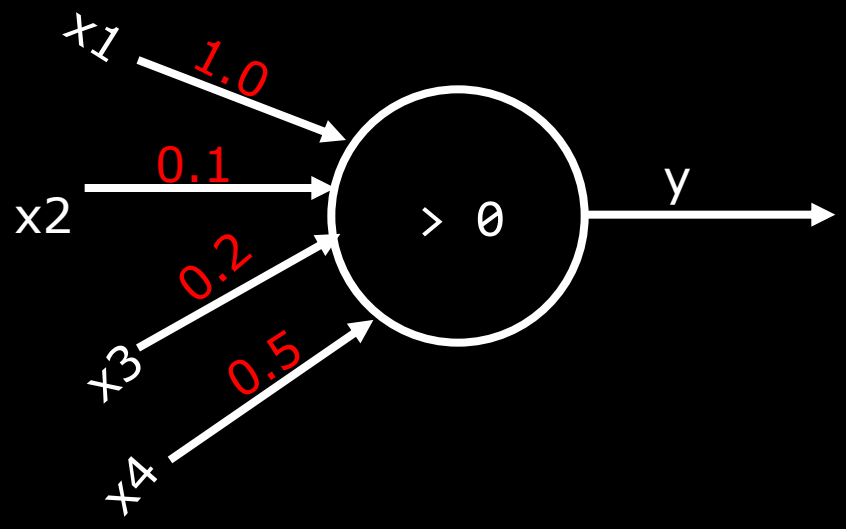
老板凭经验觉得权重可以设成

股价、油价、汇率、金价

1.0, 0.1, 0.2, 0.5

产品原型 3.0

```
foo(x1,x2,x3,x4) {
     return 1.0*x1 + 0.1*x2
          + 0.2*x3 + 0.5*x4;
}

if (foo(x) > 0)
     y = 1
else
     y = 0
```



故事背景

老板：长远看，基本面不变，房价也在缓慢涨。

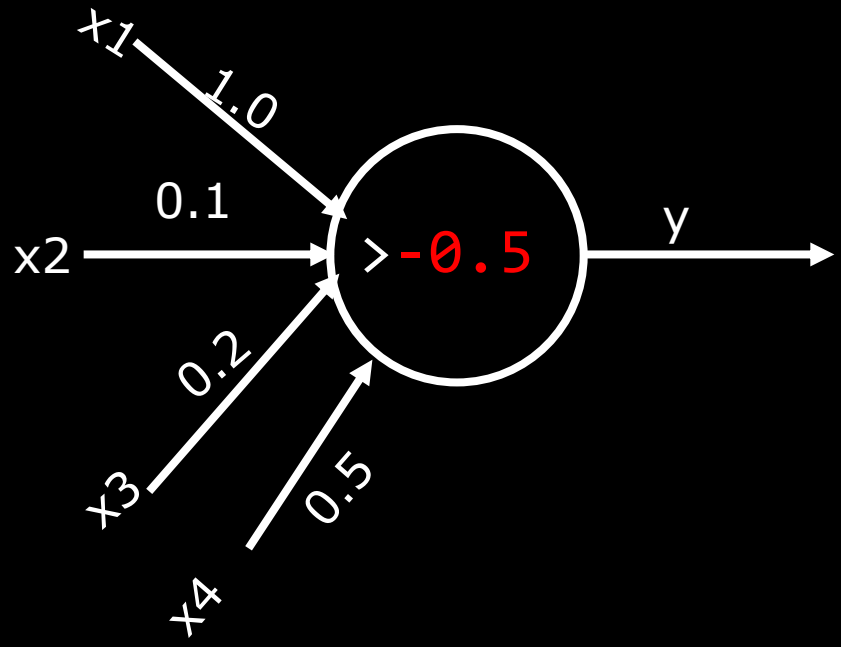
所以指标跌到一定程度，才能预测跌。

于是CTO凭经验定了个 -0.5

产品原型 4.0

```
foo(x1,x2,x3,x4) {
     return 1.0*x1 + 0.1*x2
            + 0.2*x3 + 0.5*x4;
}
```

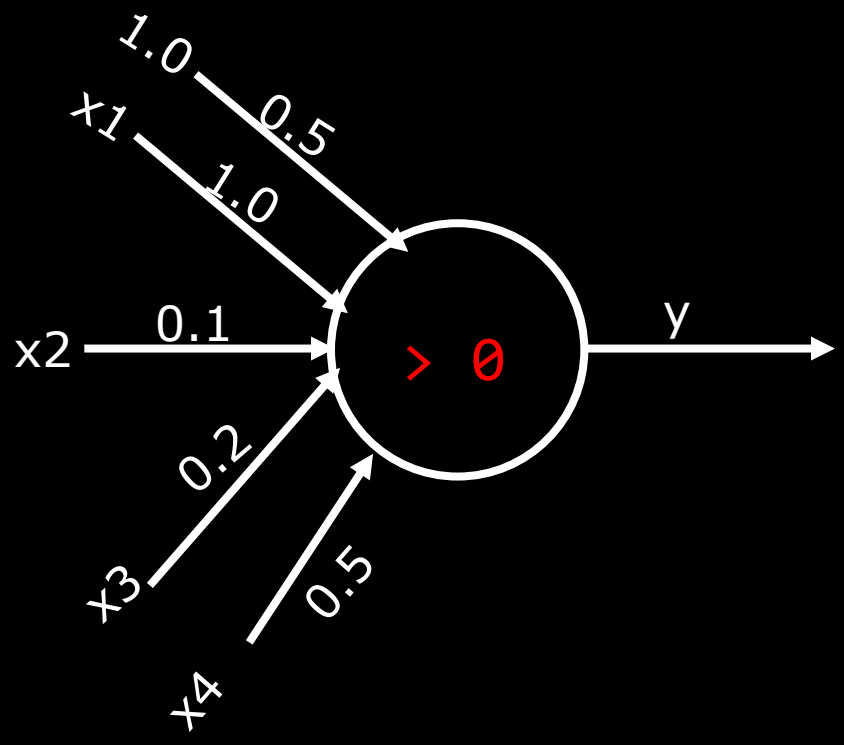
```
if (foo(x) > -0.5)
     y = 1
else
     y = 0
```



产品原型 4.1

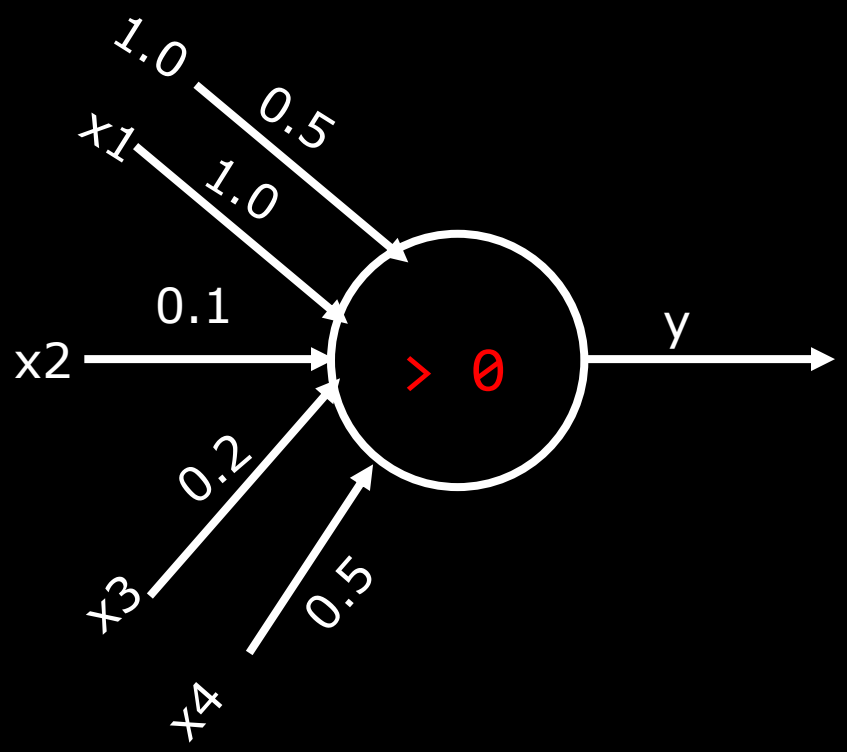
```
foo(x1,x2,x3,x4) {
     return 0.5*1 + 1.0*x1 + 0.1*x2
            + 0.2*x3 + 0.5*x4;
}
```

```
if (foo(x) > 0)
     y = 1
else
     y = 0
```



产品原型 4.2

```
w = [+0.5, 1.0, 0.1, 0.2, 0.5]
x = [   1,  x1,  x2,  x3,  x4]
foo(x) { return sum(dot_mult(w,x)) }

if (foo(x) > 0)
     y = 1
else
     y = 0
```

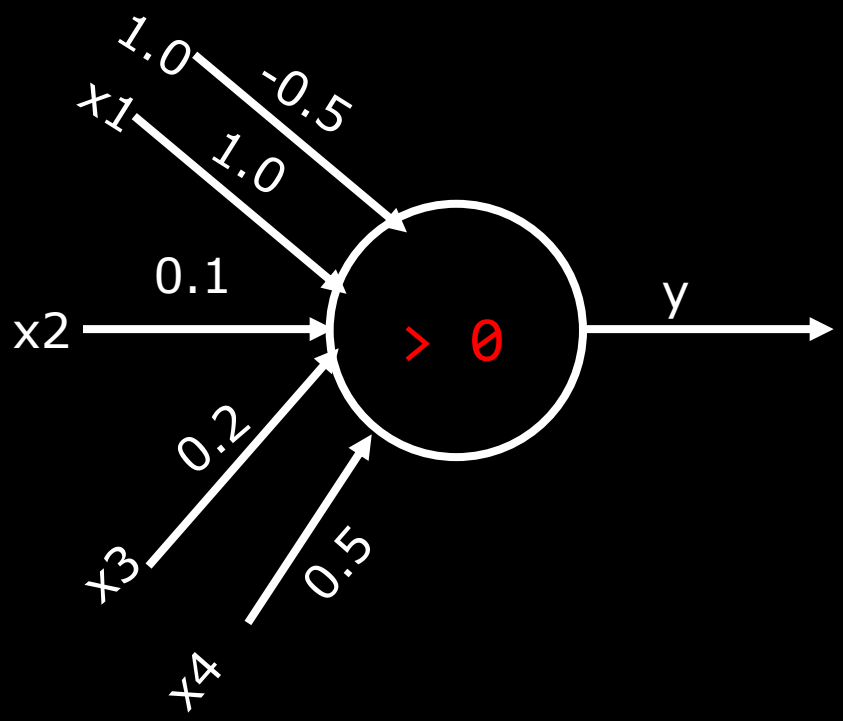


产品原型 4.2

```
w = [+0.5, 1.0, 0.1, 0.2, 0.5]
x = [   1,  x1,  x2,  x3,  x4]
foo(x) { return sum(dot_mult(w,x)) }
```

```
if (foo(x) > 0)
     y = 1
else
     y = 0
```

激活函数
Activation function



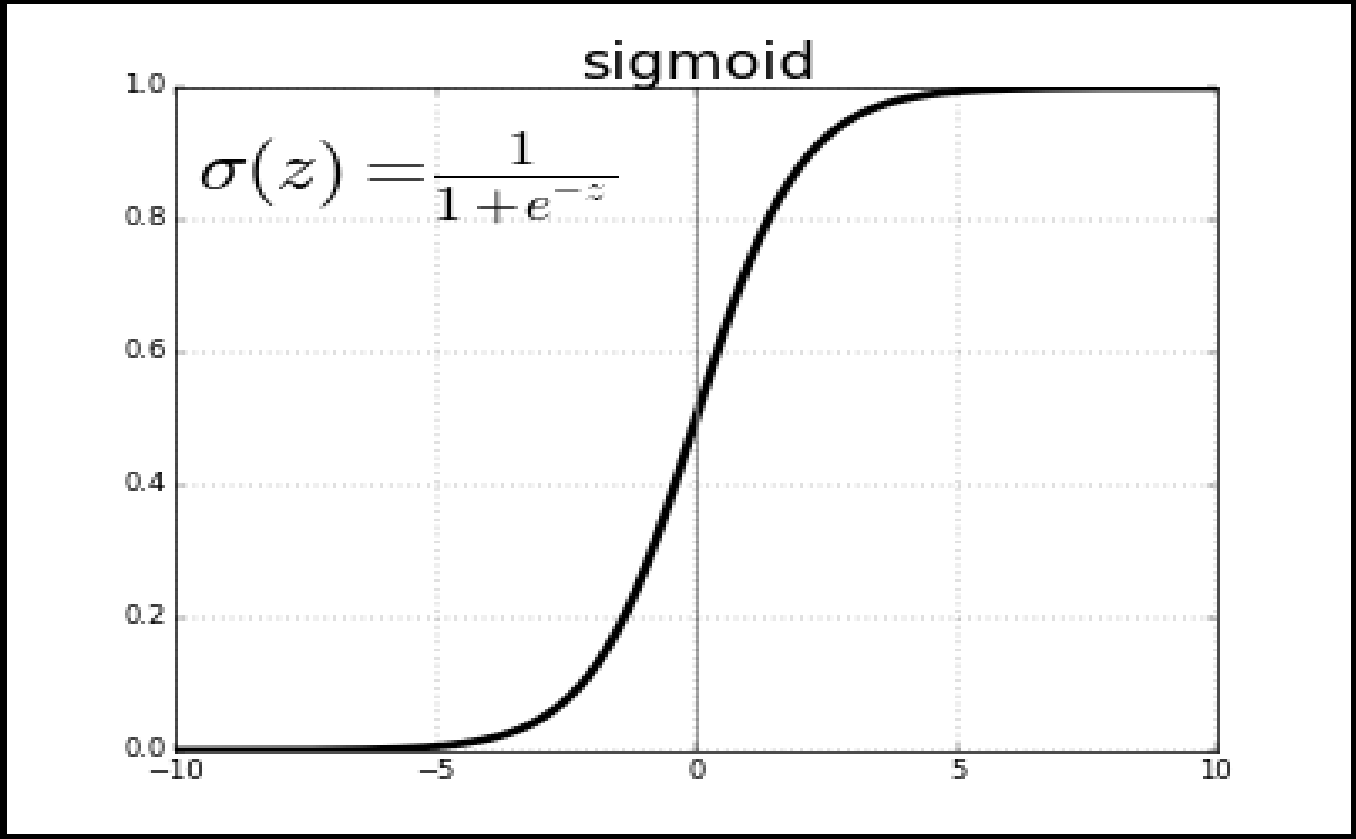
故事背景

迭代四个大版本了，还是不准。

CTO觉得可能关系复杂，线性关系不能解决，

那就试试“非线性关系”，用经典的Sigmoid

Sigmoid 函数（两种解释，类比HTML5）

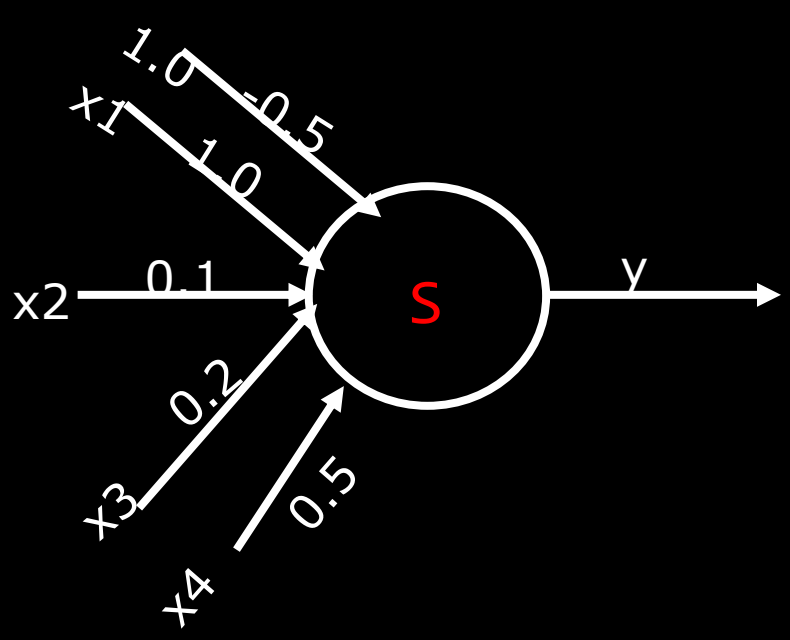


神经网络用Sigmoid函数，后面反向传播算法的时候计算微分方便

https://www.quora.com/What-is-the-sigmoid-function-and-what-is-its-use-in-machine-learnings-neural-networks

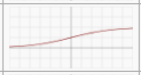
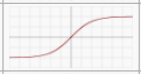
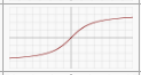
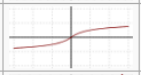
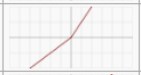
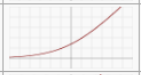
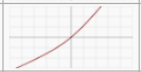
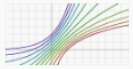
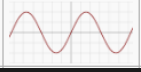
产品原型 5.0

```
w = [+0.5, 1.0, 0.1, 0.2, 0.5]
x = [   1,  x1,  x2,  x3,  x4]

foo(x) { return sum(dot_mult(w,x)) }

if (sigmoid(foo(x))>0.5)
     y = 1
else
     y = 0
```



激活（Activation）函数大家族

https://en.wikipedia.org/wiki/Activation_function

Name	Plot	Equation	Derivative (with respect to x)	Range	Order of continuity	Monotonic	Derivative Monotonic	Approximates identity near the origin
Identity		$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$	C^∞	Yes	Yes	Yes
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$	$\{0, 1\}$	C^{-1}	Yes	No	No
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	$(0, 1)$	C^∞	Yes	No	No
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$	$(-1, 1)$	C^∞	Yes	No	Yes
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$	$\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$	C^∞	Yes	No	Yes
Softsign [7][8]		$f(x) = \frac{x}{1 + \lvert x \rvert}$	$f'(x) = \frac{1}{(1 + \lvert x \rvert)^2}$	$(-1, 1)$	C^1	Yes	No	Yes
Rectifier (ReLU)[9]		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$[0, \infty)$	C^0	Yes	Yes	No
Parameteric Rectified Linear Unit (PReLU)[10]		$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$(-\infty, \infty)$	C^0	Yes iff $\alpha \geq 0$	Yes	Yes iff $\alpha = 1$
Exponential Linear Unit (ELU)[11]		$f(\alpha, x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$(-\alpha, \infty)$	C^1 when $\alpha = 1$	Yes iff $\alpha \geq 0$	Yes iff $0 \leq \alpha \leq 1$	Yes iff $\alpha = 1$
SoftPlus[12]		$f(x) = \ln(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$	$(0, \infty)$	C^∞	Yes	Yes	No
Bent identity		$f(x) = \frac{\sqrt{x^2 + 1} - 1}{2} + x$	$f'(x) = \frac{x}{2\sqrt{x^2 + 1}} + 1$	$(-\infty, \infty)$	C^∞	Yes	Yes	Yes
SoftExponential [13]		$f(\alpha, x) = \begin{cases} -\frac{\log_e(1 - \alpha(x + \alpha))}{\alpha} & \text{for } \alpha < 0 \\ x & \text{for } \alpha = 0 \\ \frac{e^{\alpha x} - 1}{\alpha} + \alpha & \text{for } \alpha > 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} \frac{1}{1 - \alpha(\alpha + x)} & \text{for } \alpha < 0 \\ e^{\alpha x} & \text{for } \alpha \geq 0 \end{cases}$	$(-\infty, \infty)$	C^∞	Yes	Yes	Yes iff $\alpha = 0$
Sinusoid		$f(x) = \sin(x)$	$f'(x) = \cos(x)$	$[-1, 1]$	C^∞	No	No	Yes

故事背景

5.0版了，还是不够准。

CTO：可能是我们的程序逻辑太简单了？

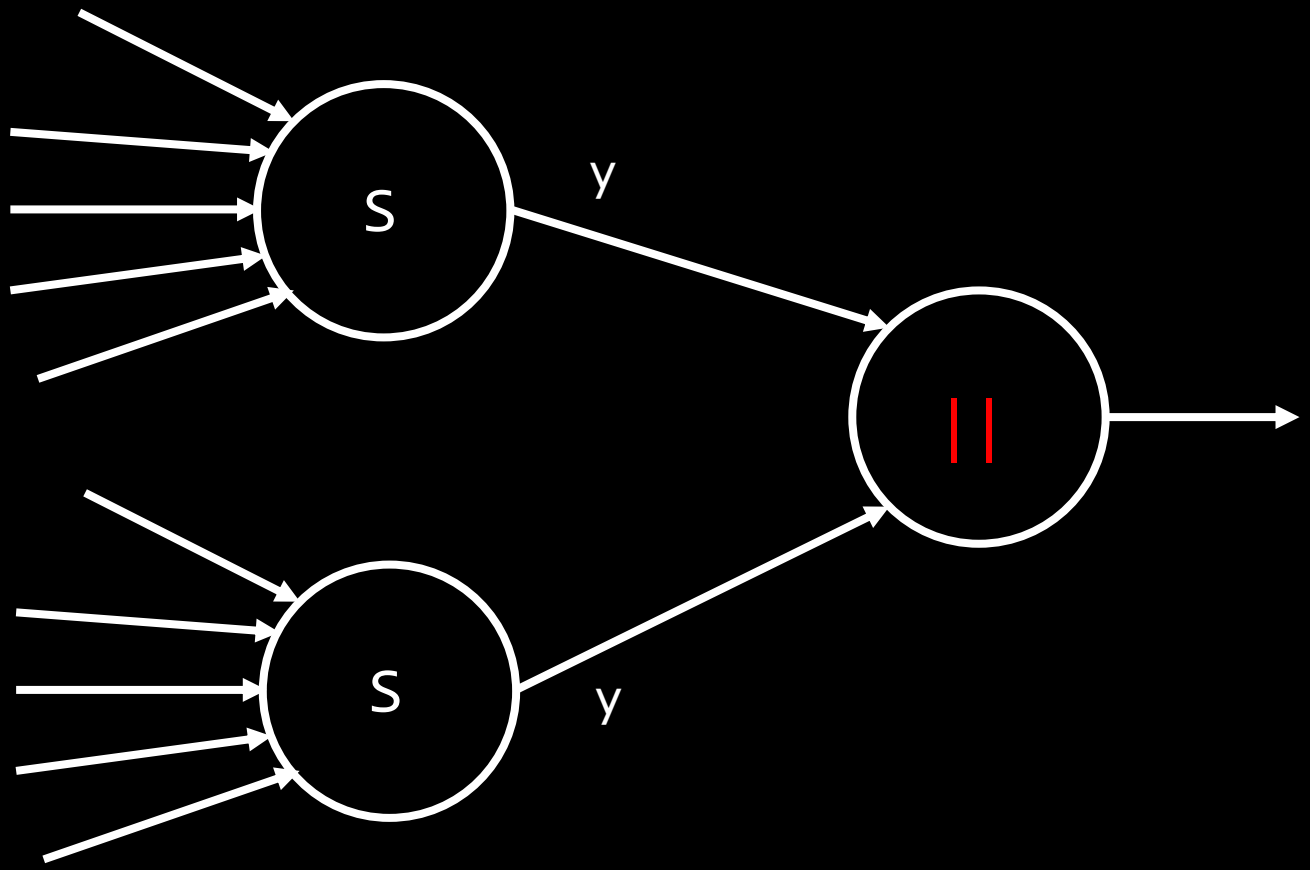
现在流行的是深度学习，动辄“一百多层”，试试？

故事背景

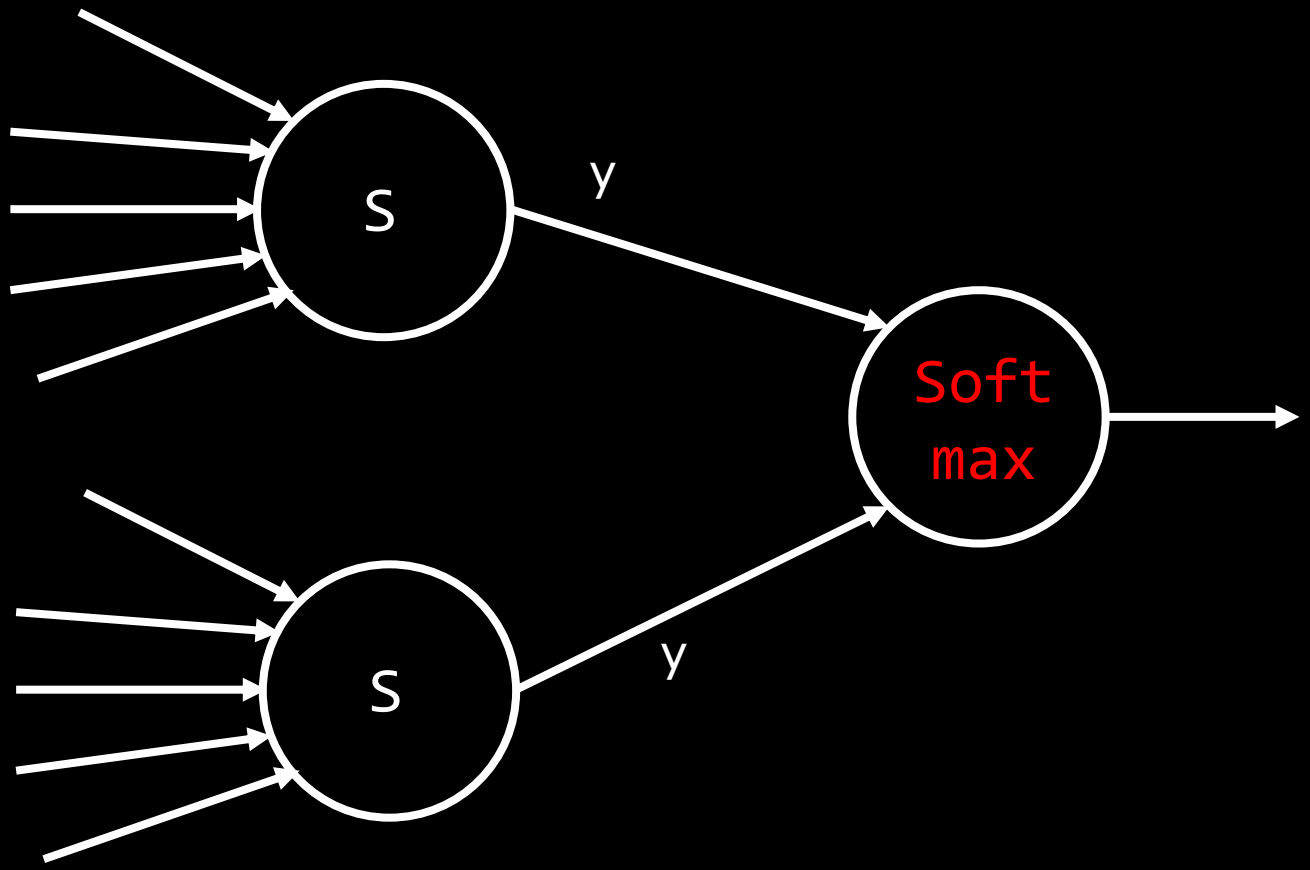
老板：大家买房会综合考虑，不会“一刀切”

是不是用多个不同的计算公式，综合给出预测结果？

CTO很快复制粘贴出来了6.0版， 有说涨就涨



还照猫画虎用上了softmax



https://en.wikipedia.org/wiki/Softmax_function

☺ https://www.zhihu.com/question/23765351

故事背景

有神经元，有层，是神经网络了！但是……

两个输入的神经元，权重一模一样……

老板让CTO自己也编一套放进去

故事背景

现在CTO手里有神经网络（模型），有房价、股价、油价、汇率和金价的历史数据（训练集），和老板给的经验权重（初始值）。

CTO编了几个权重，预测都不准。接下来怎么办？

思路：用历史数据来修正权重

11.1	11.2	11.3	11.4	……	11.30
股价	涨	跌	涨	……	涨
汇率	跌	涨	跌	……	跌
金价	涨	跌	涨	……	涨
油价	跌	涨	跌	……	跌
房价（有监督）	**涨**	**涨**	**涨**	**涨**	**涨**

大神出场：反向传播算法&BP神经网络

Learning representations by back-propagating errors

David E. Rumelhart*, Geoffrey E. Hinton† & Ronald J. Williams*

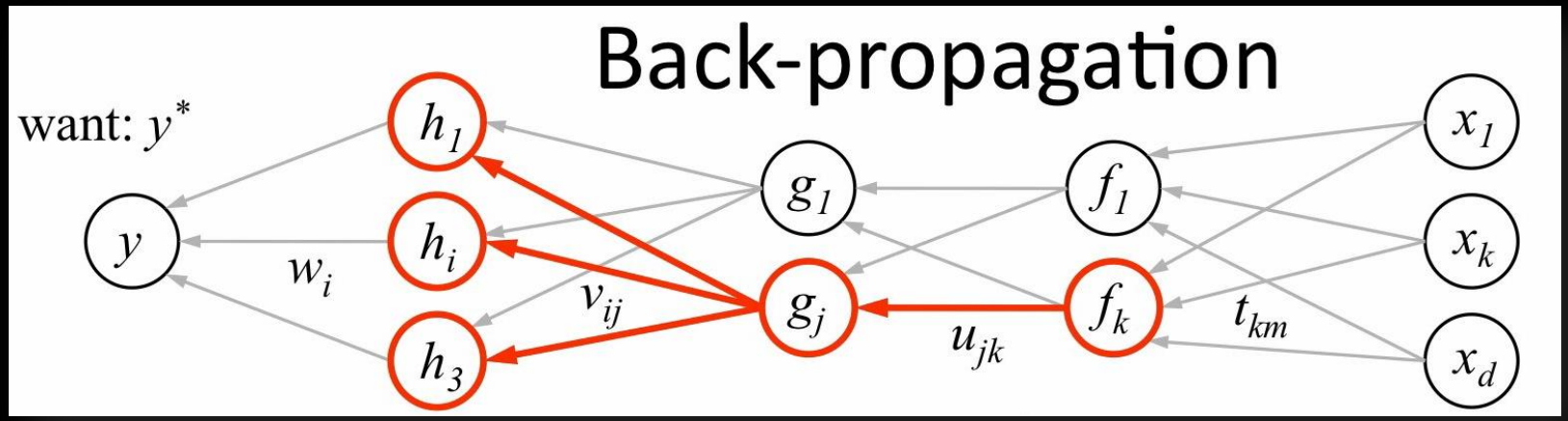
* Institute for Cognitive Science, C-015, University of California, San Diego, La Jolla, California 92093, USA
† Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Philadelphia 15213, USA

We describe a new learning procedure, back-propagation, for networks of neurone-like units. The procedure repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector. As a result of the weight adjustments, internal 'hidden' units which are not part of the input or output come to represent important features of the task domain, and the regularities in the task are captured by the interactions of these units. The ability to create useful new features distinguishes back-propagation from earlier, simpler methods such as the perceptron-convergence procedure[1].

more difficult when we introduce hidden units whose actual or desired states are not specified by the task. (In perceptrons, there are 'feature analysers' between the input and output that are not true hidden units because their input connections are fixed by hand, so their states are completely determined by the input vector: they do not learn representations.) The learning procedure must decide under what circumstances the hidden units should be active in order to help achieve the desired input–output behaviour. This amounts to deciding what these units should represent. We demonstrate that a general purpose and relatively simple procedure is powerful enough to construct appropriate internal representations.

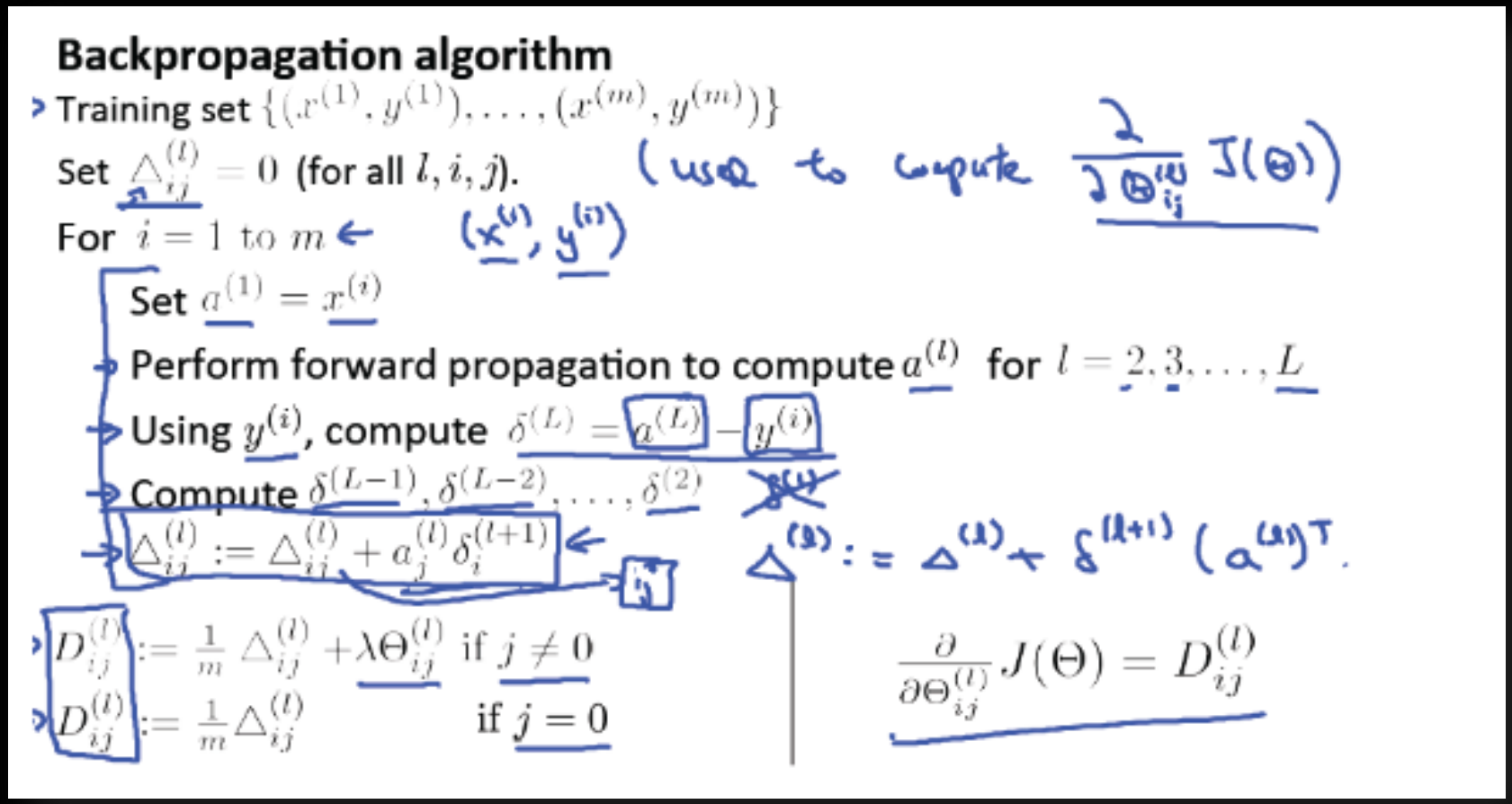
The simplest form of the learning procedure is for layered networks which have a layer of input units at the bottom; any number of intermediate layers; and a layer of output units at the top. Connections within a layer or from higher to lower layers are forbidden, but connections can skip intermediate layers. An input vector is presented to the network by setting the states of the input units. Then the states of the units in each layer are determined by applying equations (1) and (2) to the connections coming from lower layers. All units within a layer have their states set in parallel, but different layers have their states set sequentially, starting at the bottom and working upwards until the states of the output units are determined.

（不理解没关系，该算法已经不用自己实现了）



https://www.youtube.com/watch?v=An5z8lR8asY
https://i.ytimg.com/vi/An5z8lR8asY/maxresdefault.jpg

进一步学习：吴恩达在Coursera上的机器学习课程

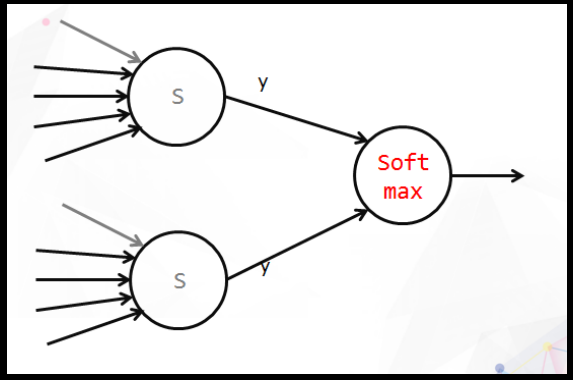


故事背景

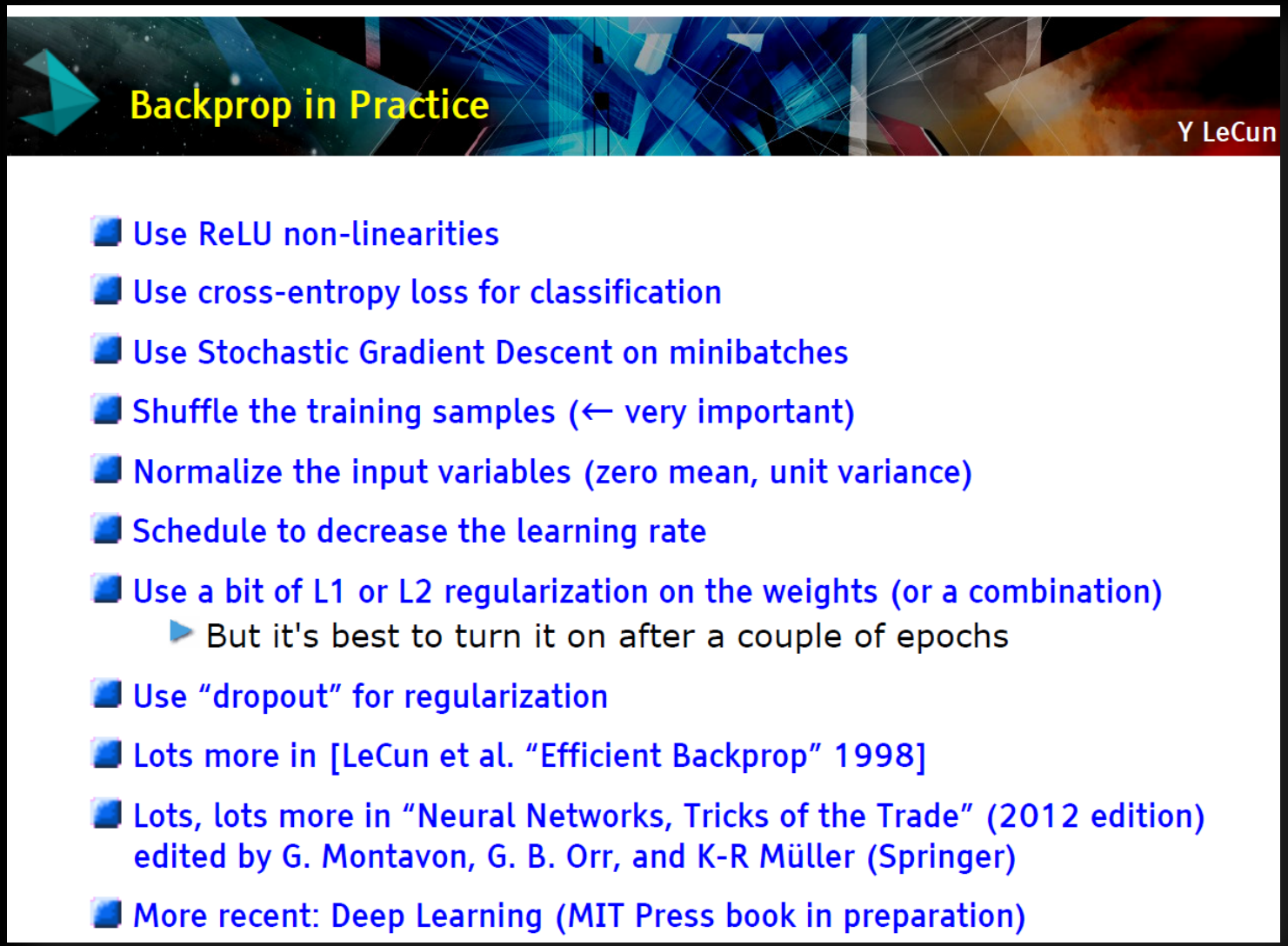
训练完的效果确实比以前好了很多，但是……

权重还是都是一样的嘛！

不起眼但是很重要：随机初始化



随机化在训练中经常需要被使用到



Happy Ending

准确率终于上去了。老板很开心，问CTO权重的含义。

CTO不知道，请教各路大神.....

“说不好”

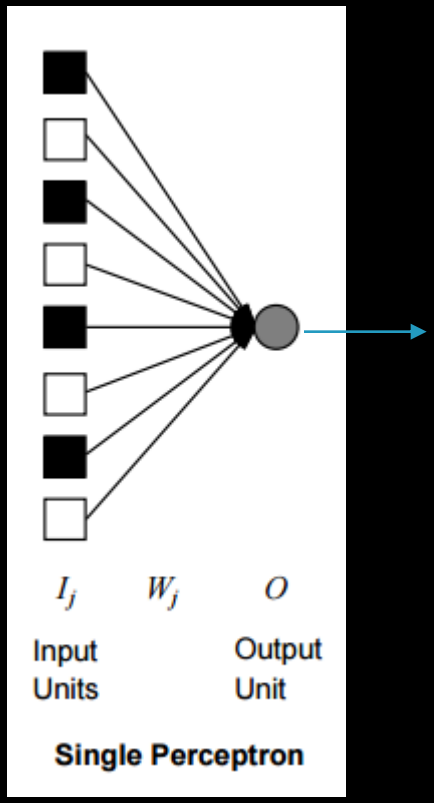
"很复杂"

"不好说"

"有点难"

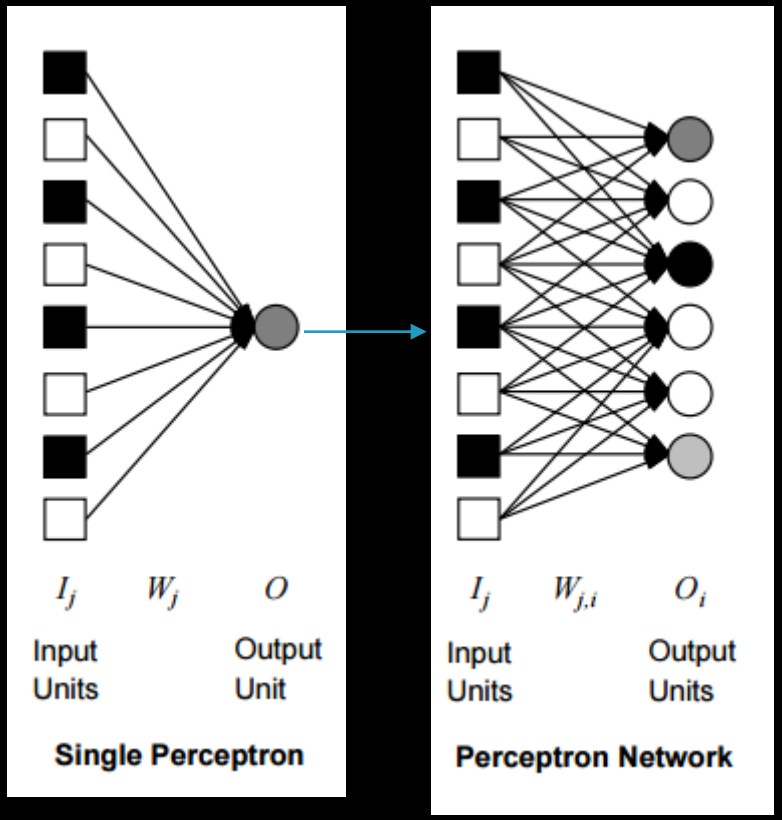
"问错人了吧"

反向传播算法在手，层数你有



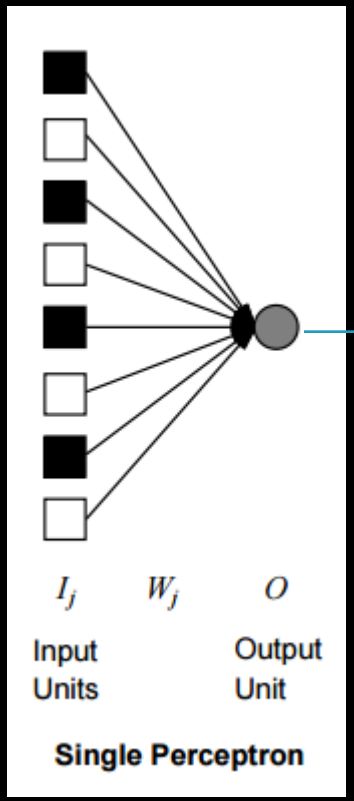
https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15381-f01/www/handouts/110601.pdf

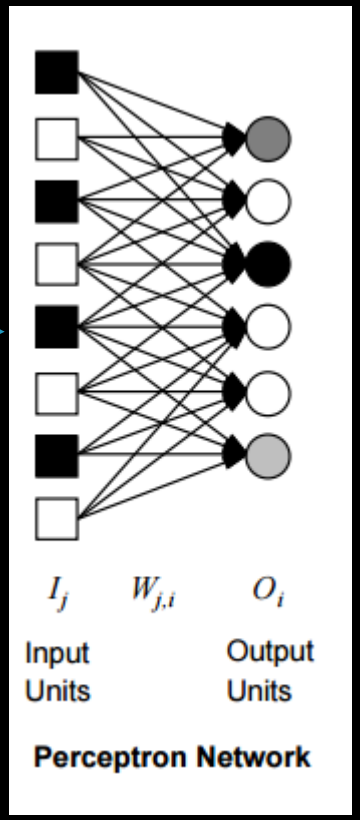
反向传播算法在手，层数你有

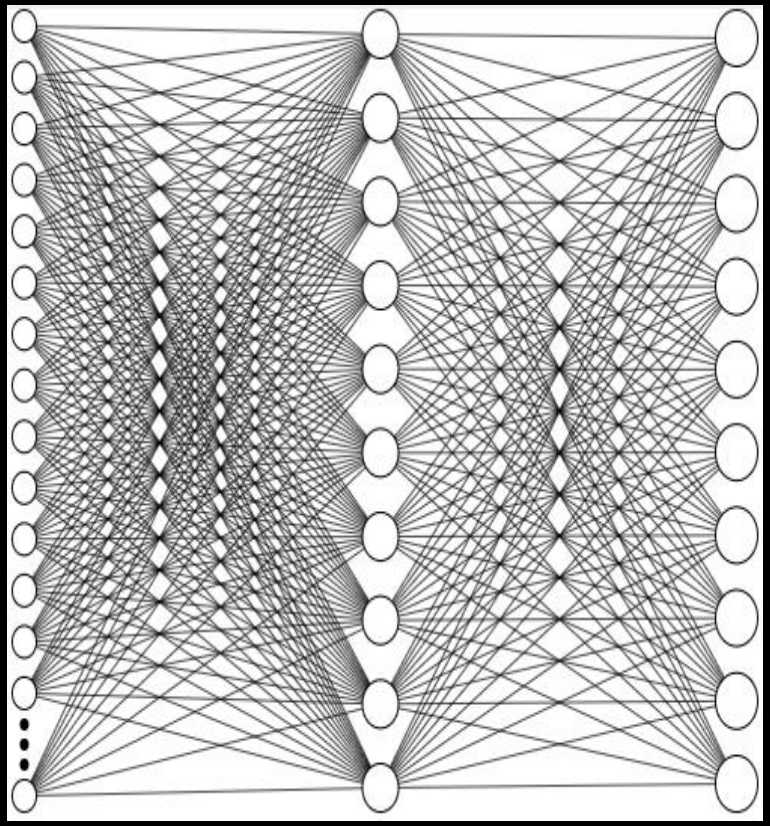


https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15381-f01/www/handouts/110601.pdf

反向传播算法在手，层数你有

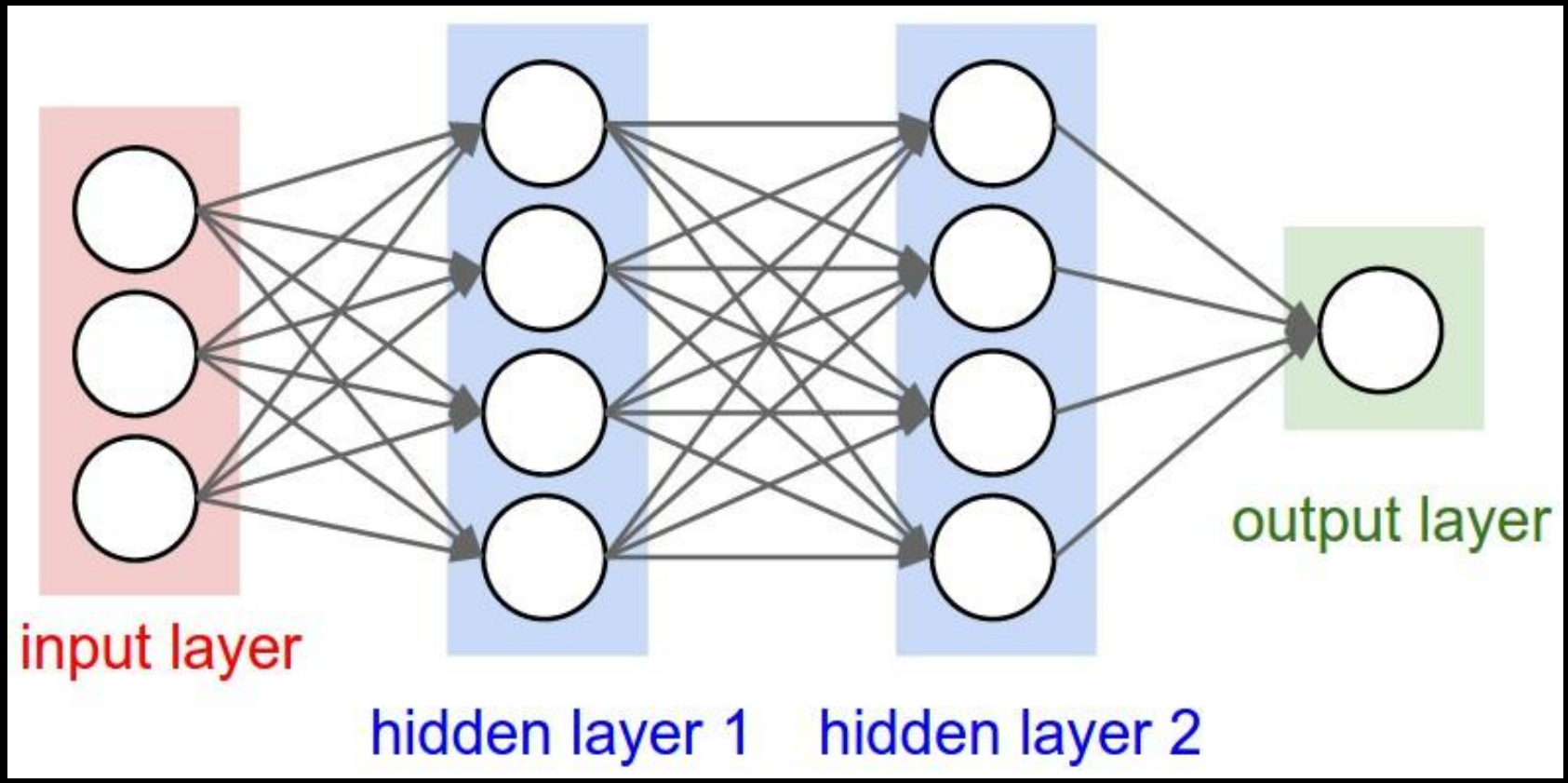






https://en.wikipedia.org/wiki/Artificial_neural_network
https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15381-f01/www/handouts/110601.pdf

反向传播算法在手，层数你有



http://cs231n.github.io/neural-networks-1/

反向传播算法在手，层数你。。。额，不行了

http://neuralnetworksanddeeplearning.com/chap6.html

不行了：反向传播算法的局限性

梯度消失

局部最优

过学习

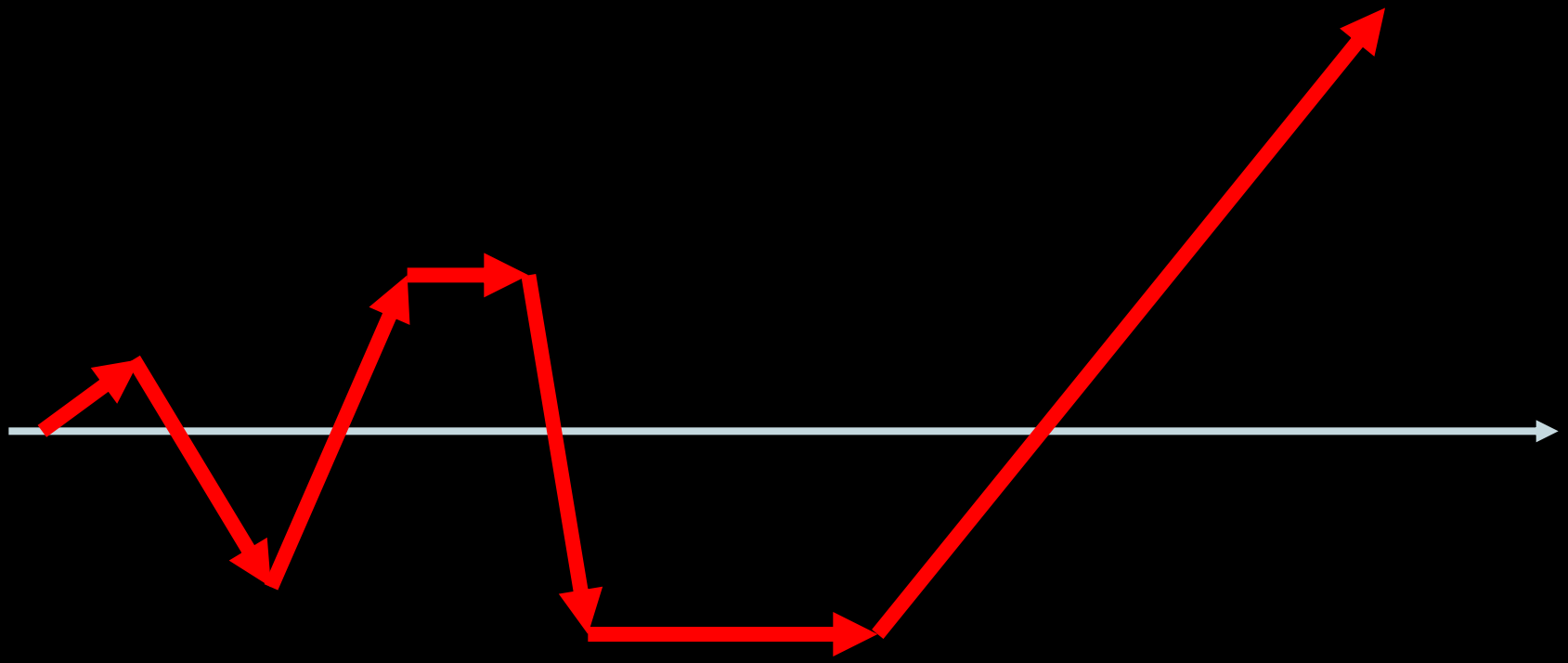
过学习（Overfit）和欠学习（Underfit）

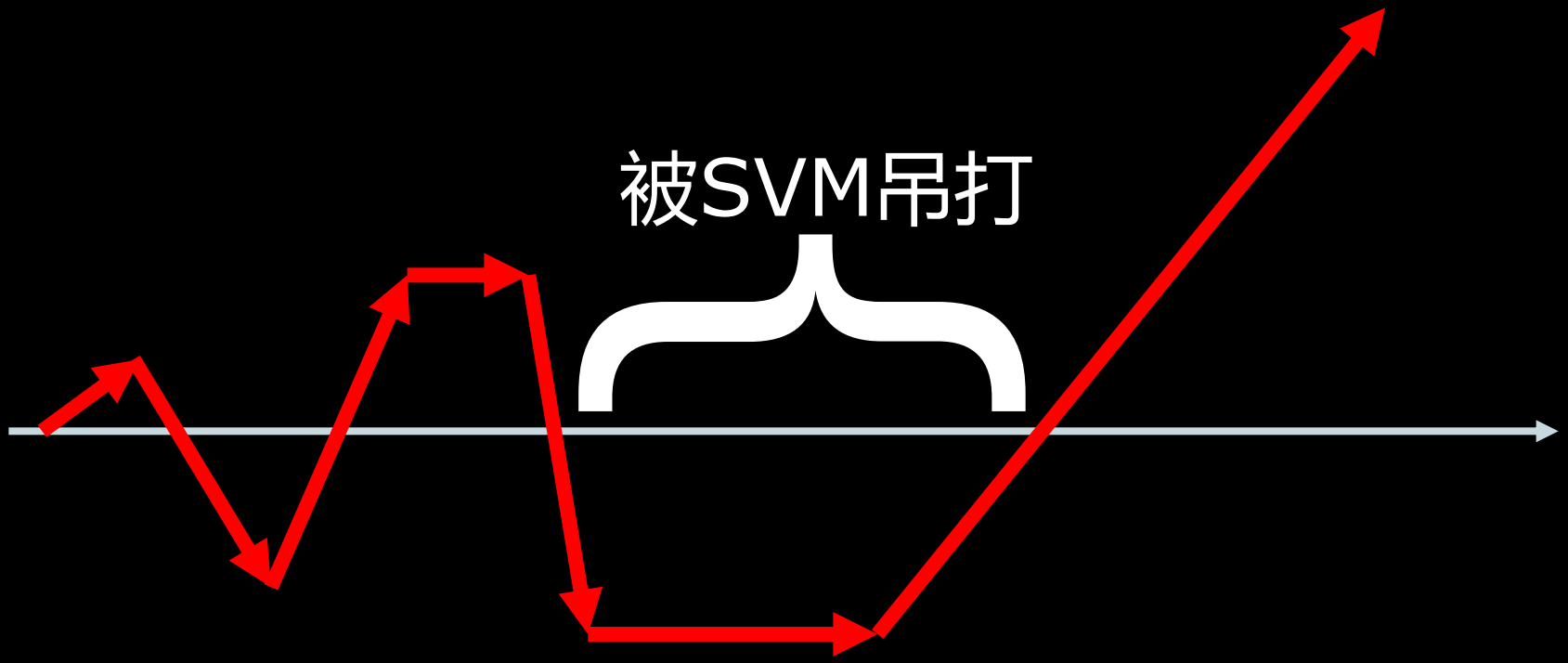
给我四个参数，我能拟合出来一头大象。

给我五个，我就能让大象的鼻子动起来。

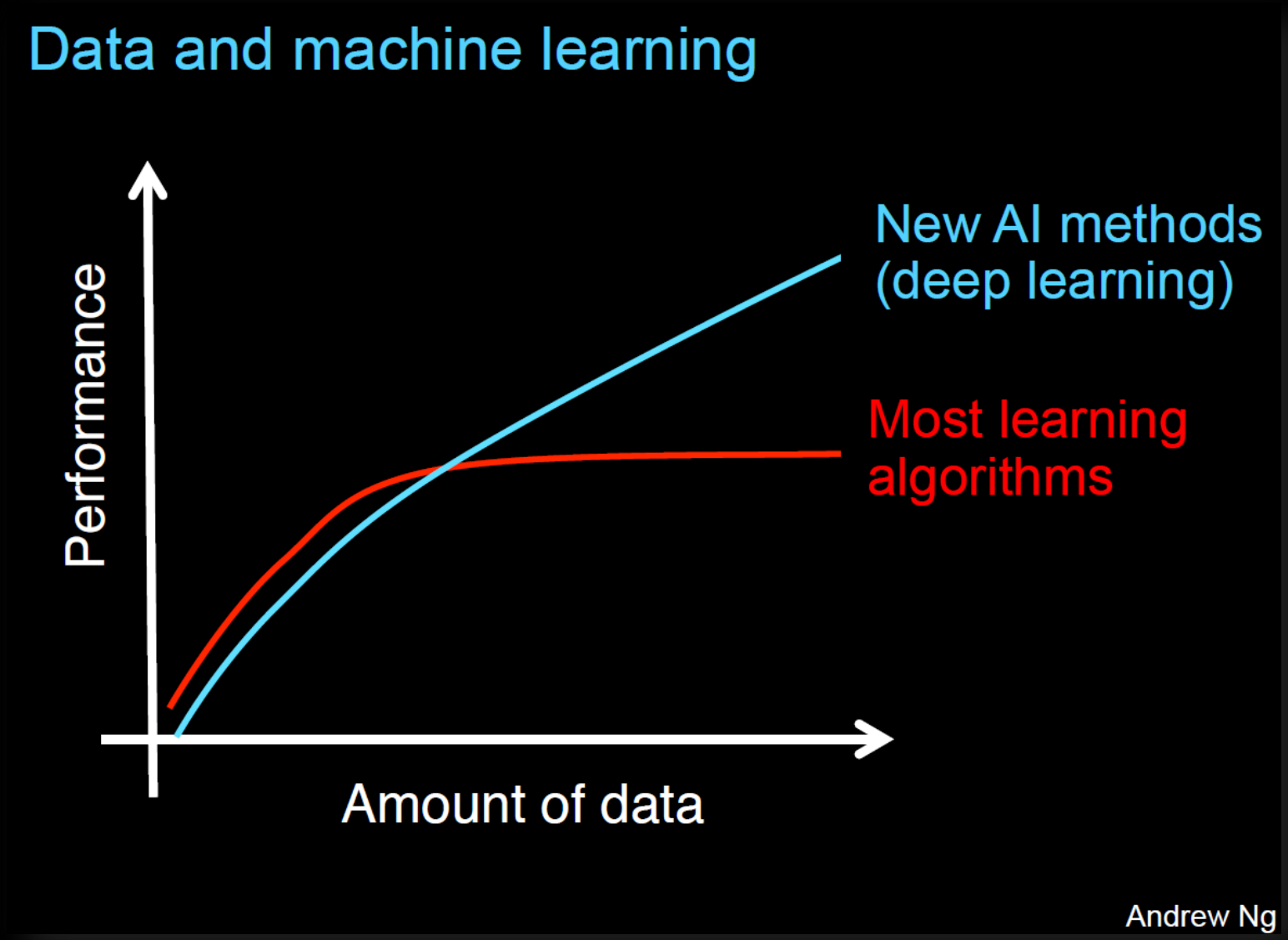
—— 冯·诺依曼

神经网络坐了十几年的冷板凳




被SVM吊打

那么卢瑟为什么能够逆袭了呢？



Neural network
Back propagation

Nature

Deep belief net

Science

Microsoft

Speech

IMAGENET

1986

被SVM吊着打

2006

2011

2012

Credit: Xiaogang Wang, The Chinese University of Hong Kong

Neural network
Back propagation

Deep belief net
Science

Speech

IMAGENET

2006 2011 2012

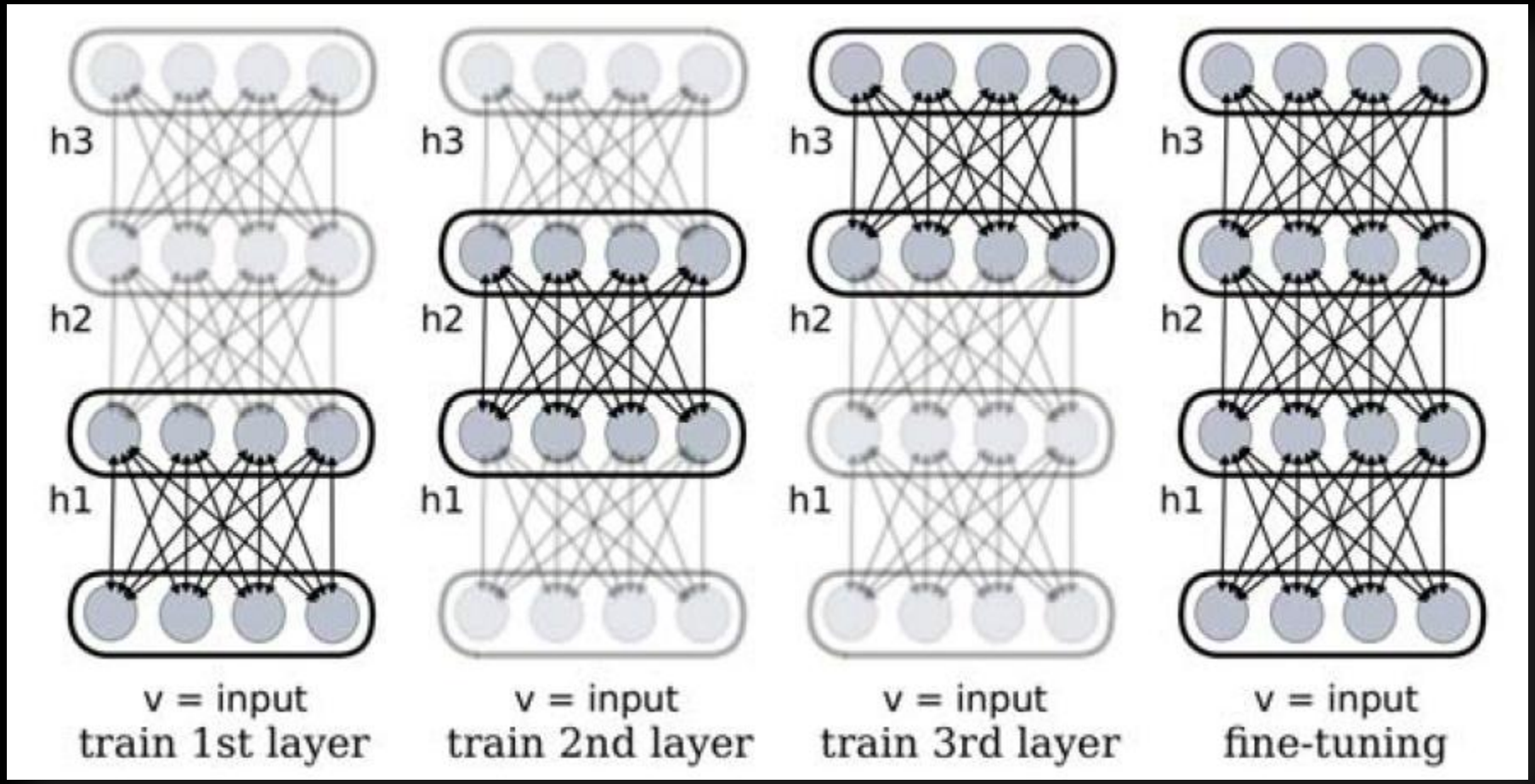
面壁十年图破壁

我Hinton悟到了

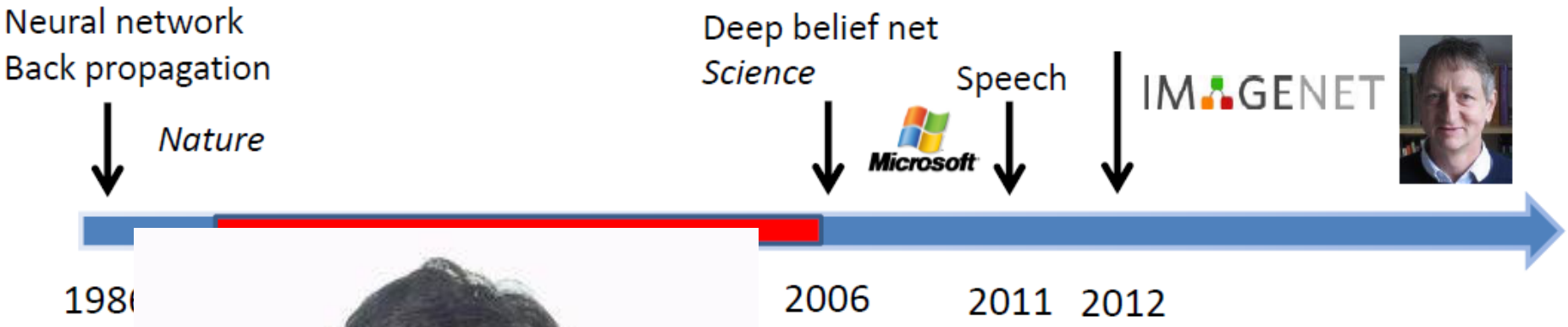
一！层！一！层！来！

Credit: Xiaogang Wang, The Chinese University of Hong Kong

一层一层先预训练参数，最后反向传播



Available from http://www.i6doc.com/en/livre/?GCOI=28001100817300.

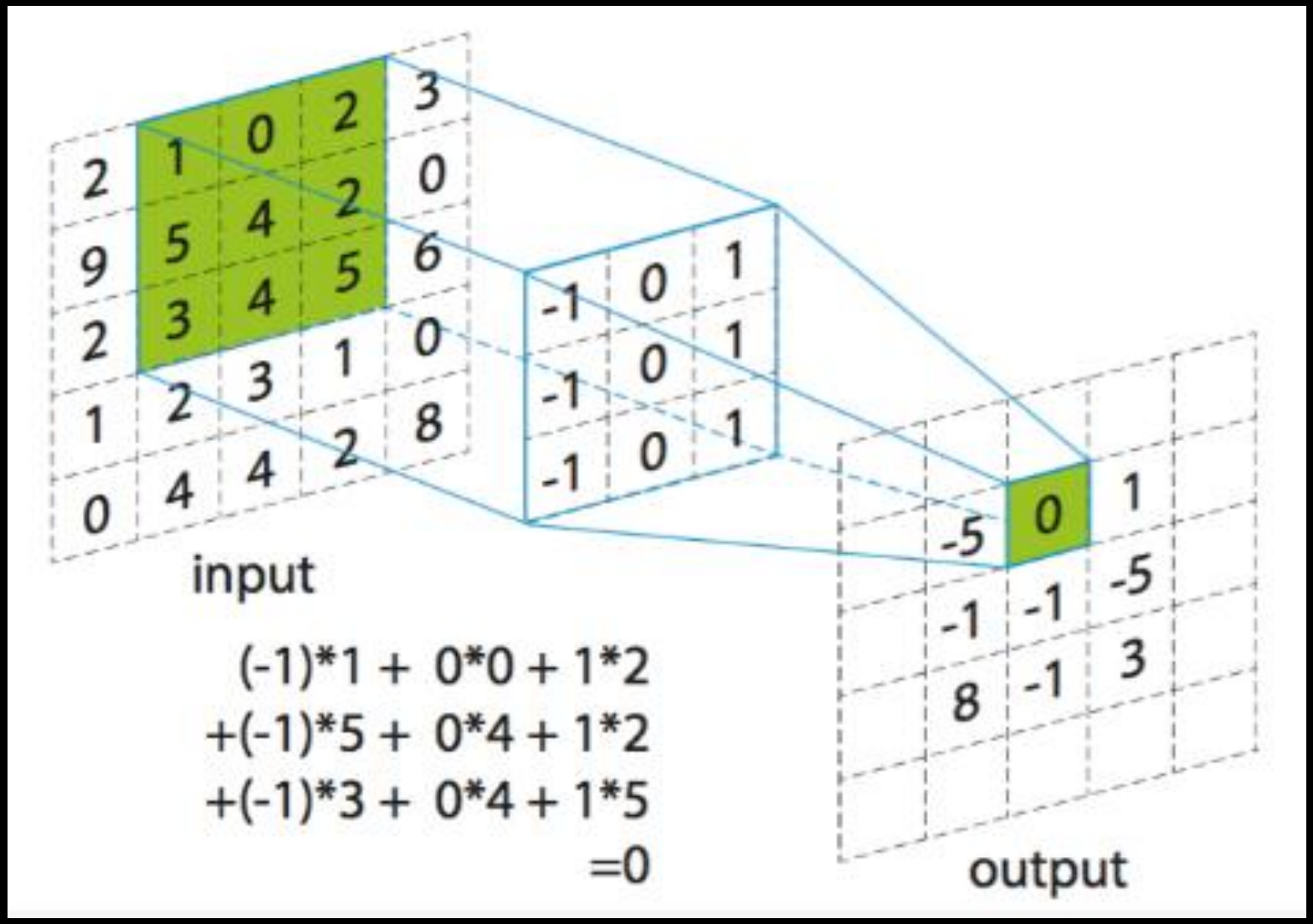


减少训练权重的规模

试试我乐康卷☺

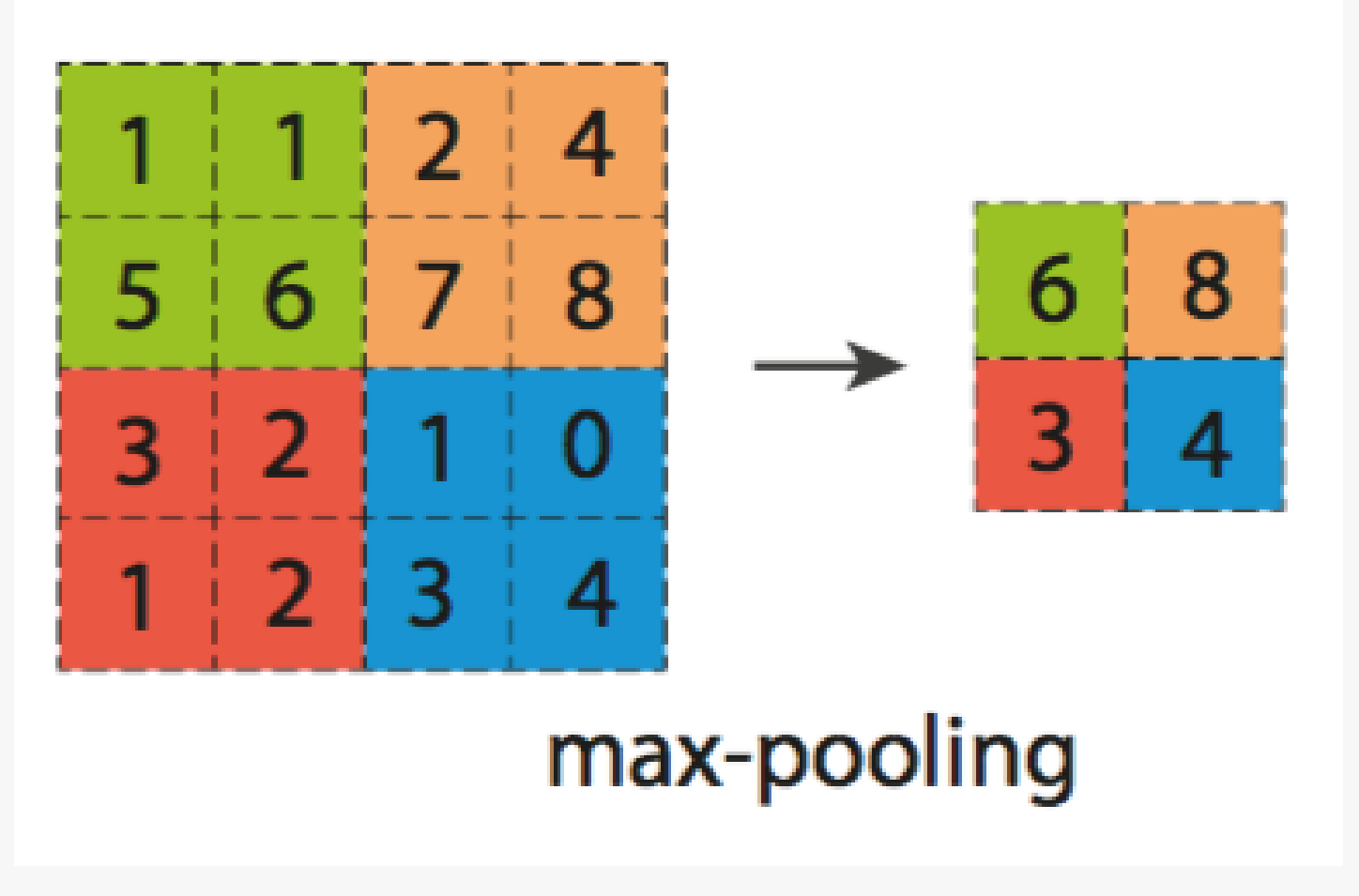
Credit: Xiaogang Wang, The Chinese University of Hong Kong

乐康卷：卷积是个啥？Conv？



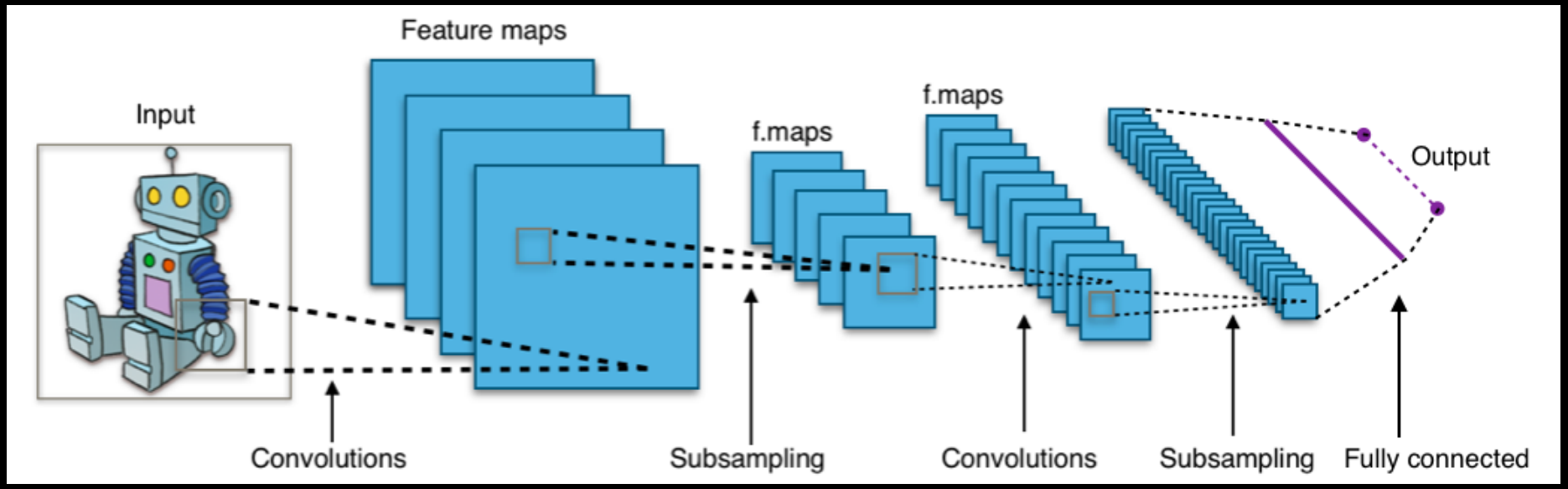
http://www.36dsj.com/archives/70111

乐康卷：pooling又是啥？



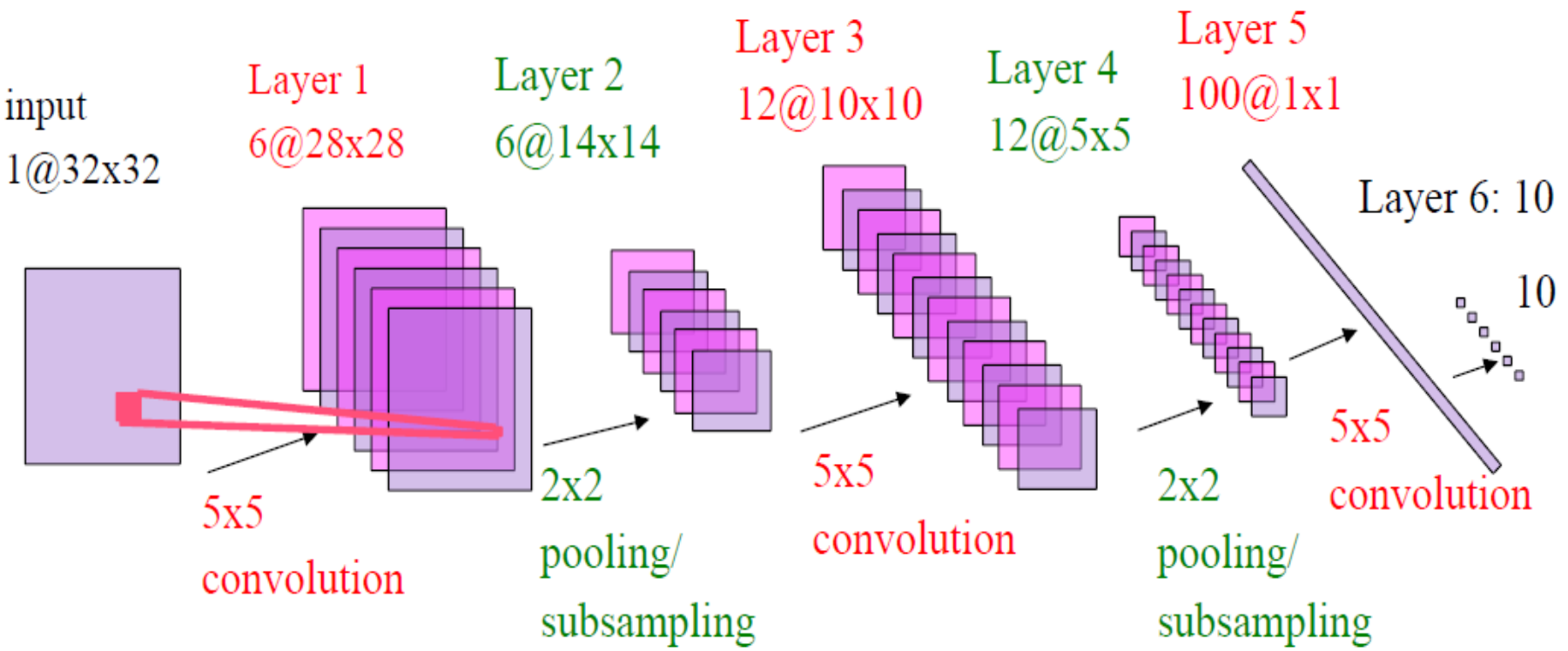
http://www.36dsj.com/archives/70111

乐康卷：极大的减少了权重规模



https://en.wikipedia.org/wiki/Convolutional_neural_network#/media/File:Typical_cnn.png

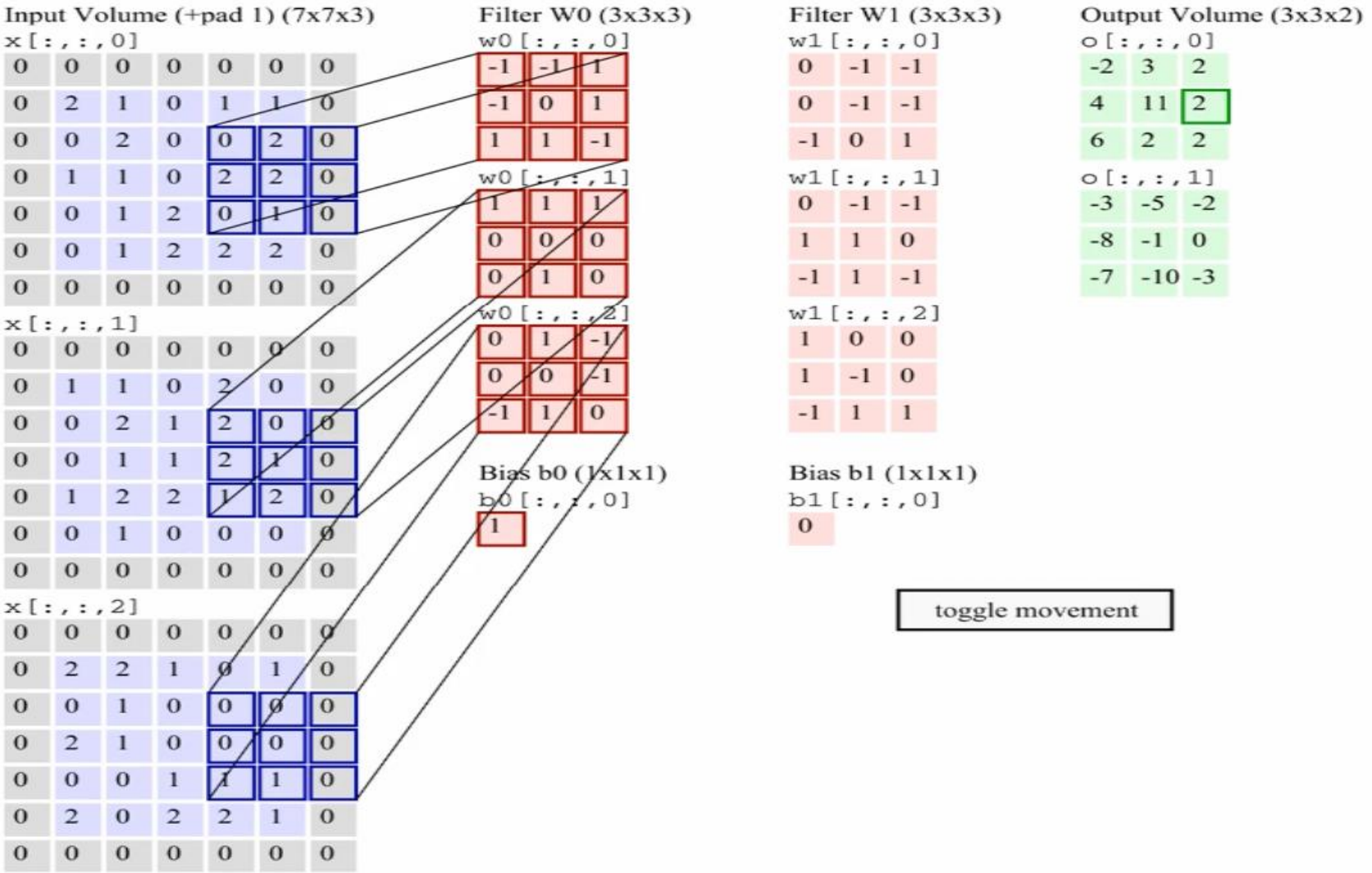
乐康卷：参数配置 LeNet5



Multiple Convolutions

有很多非常好的教程可以参考

Y LeCun



Animation: Andrej Karpathy http://cs231n.github.io/convolutional-networks/

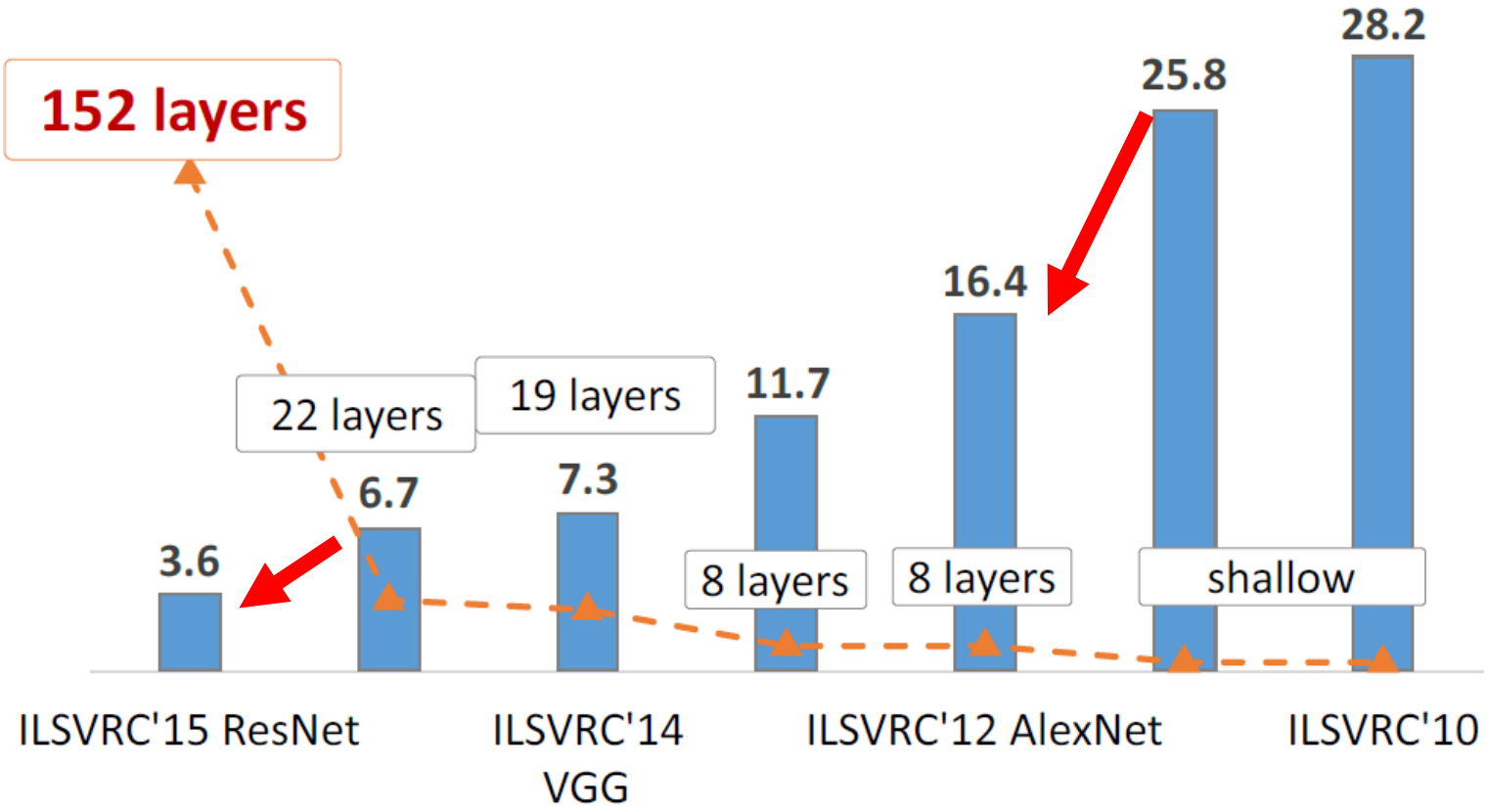
~~不行了：反向传播算法的局限性~~

梯度消失 – 逐层预训练

局部最优 – 多次随机化

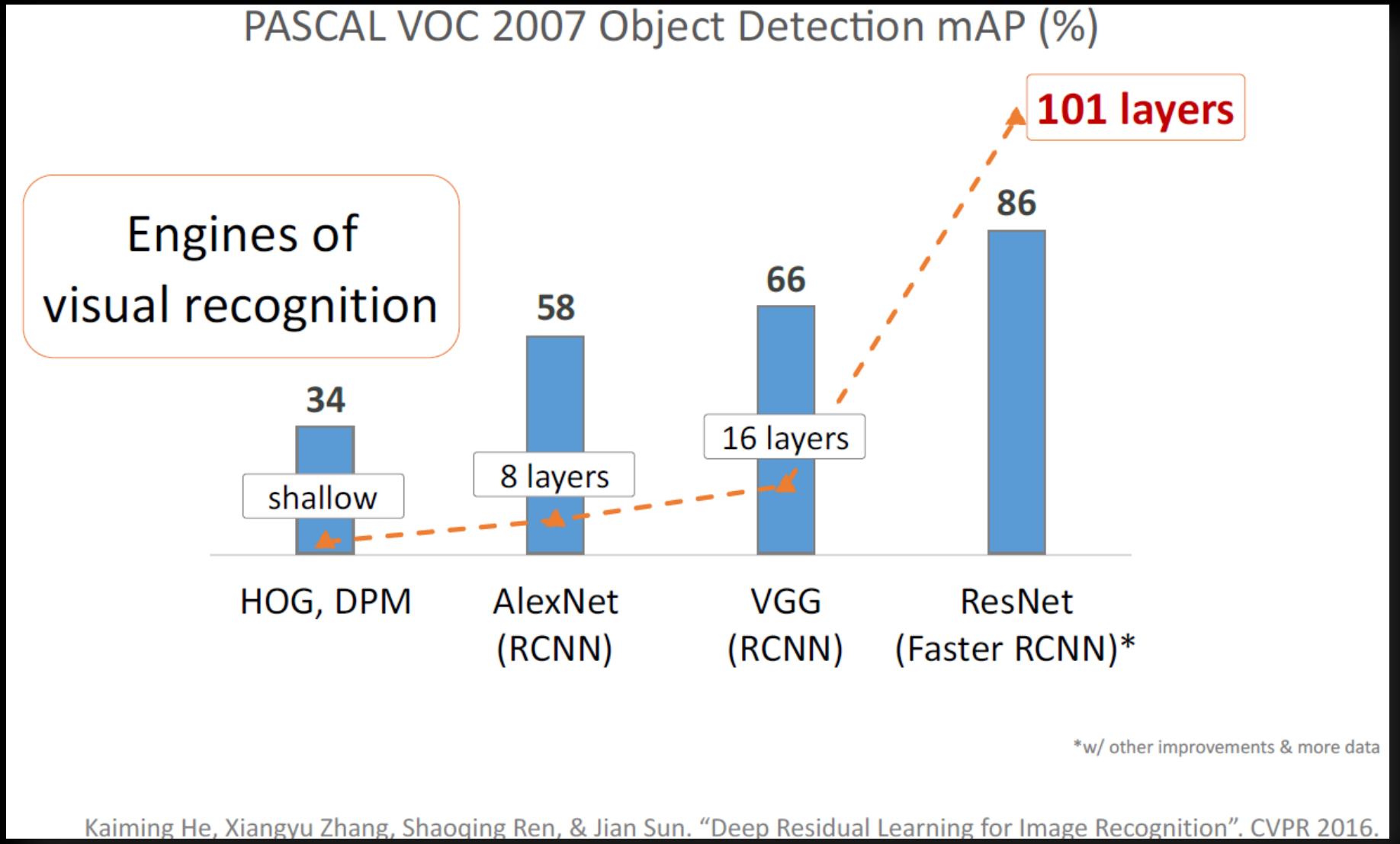
过学习 – 大数据集

算法和模型都完备后，PhD们的心开始躁动起来



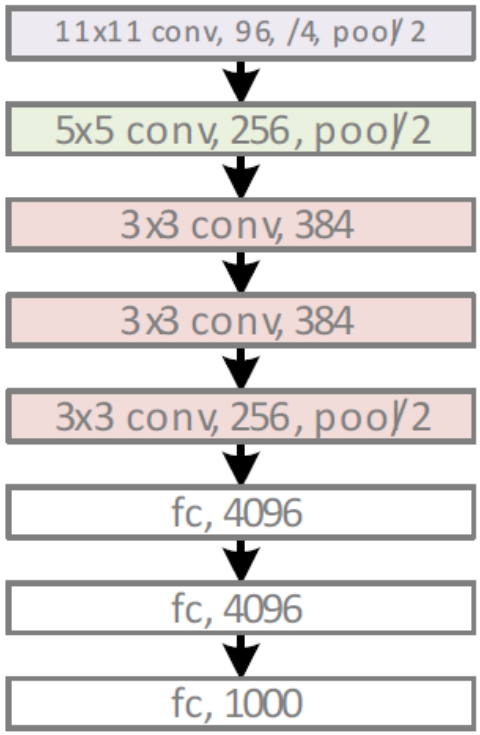
Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.

算法和模型都完备后，PhD们的心开始躁动起来



算法和模型都完备后，PhD们的心开始躁动起来

AlexNet, 8 layers (ILSVRC 2012)

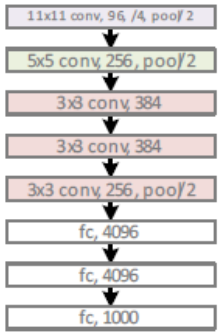


Caffe、mxnet中有现成模型

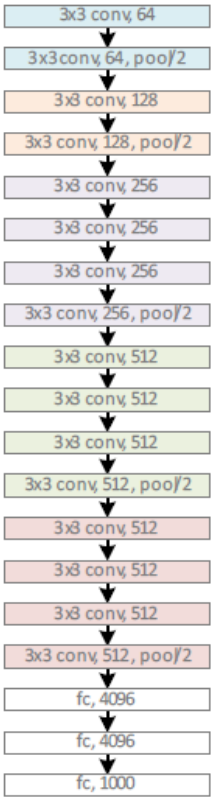
Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. “Deep Residual Learning for Image Recognition”. CVPR 2016.

算法和模型都完备后，PhD们的心开始躁动起来

AlexNet, 8 layers (ILSVRC 2012)



VGG, 19 layers (ILSVRC 2014)

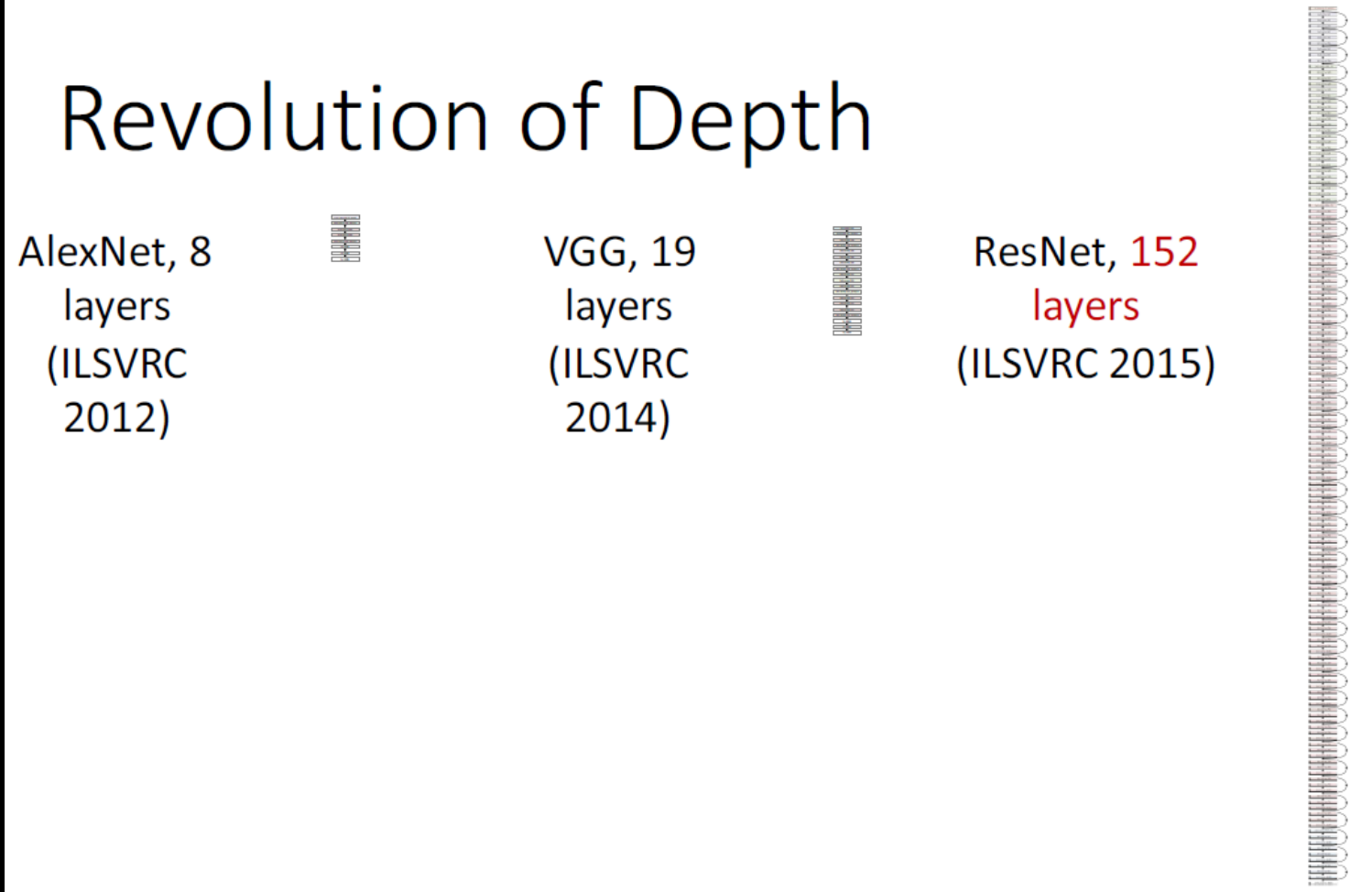


GoogleNet, 22 layers (ILSVRC 2014)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. “Deep Residual Learning for Image Recognition”. CVPR 2016.

算法和模型都完备后，PhD们的心开始躁动起来

Revolution of Depth



Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.

算法和模型都完备后，PhD们的心开始躁动起来

README.md

Deep Residual Networks with 1K Layers

1001层DRN

By Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun.

Microsoft Research Asia (MSRA).

Table of Contents

Introduction

This repository contains re-implemented code for the paper "Identity Mappings in Deep Residual Networks" (http://arxiv.org/abs/1603.05027). This work enables training quality **1k-layer** neural networks in a super simple way.

Acknowledgement: This code is re-implemented by Xiang Ming from Xi'an Jiaotong Univeristy for the ease of release.

Seel Also: Re-implementations of **ResNet-200 [a] on ImageNet** from Facebook AI Research (FAIR): https://github.com/facebook/fb.resnet.torch/tree/master/pretrained

https://github.com/KaimingHe/resnet-1k-layers

Happy Ending

老板（激动地）：我们加到两千层试试！

CTO：.....

小结与思考

单纯的加层就可以提高预测的效果么？

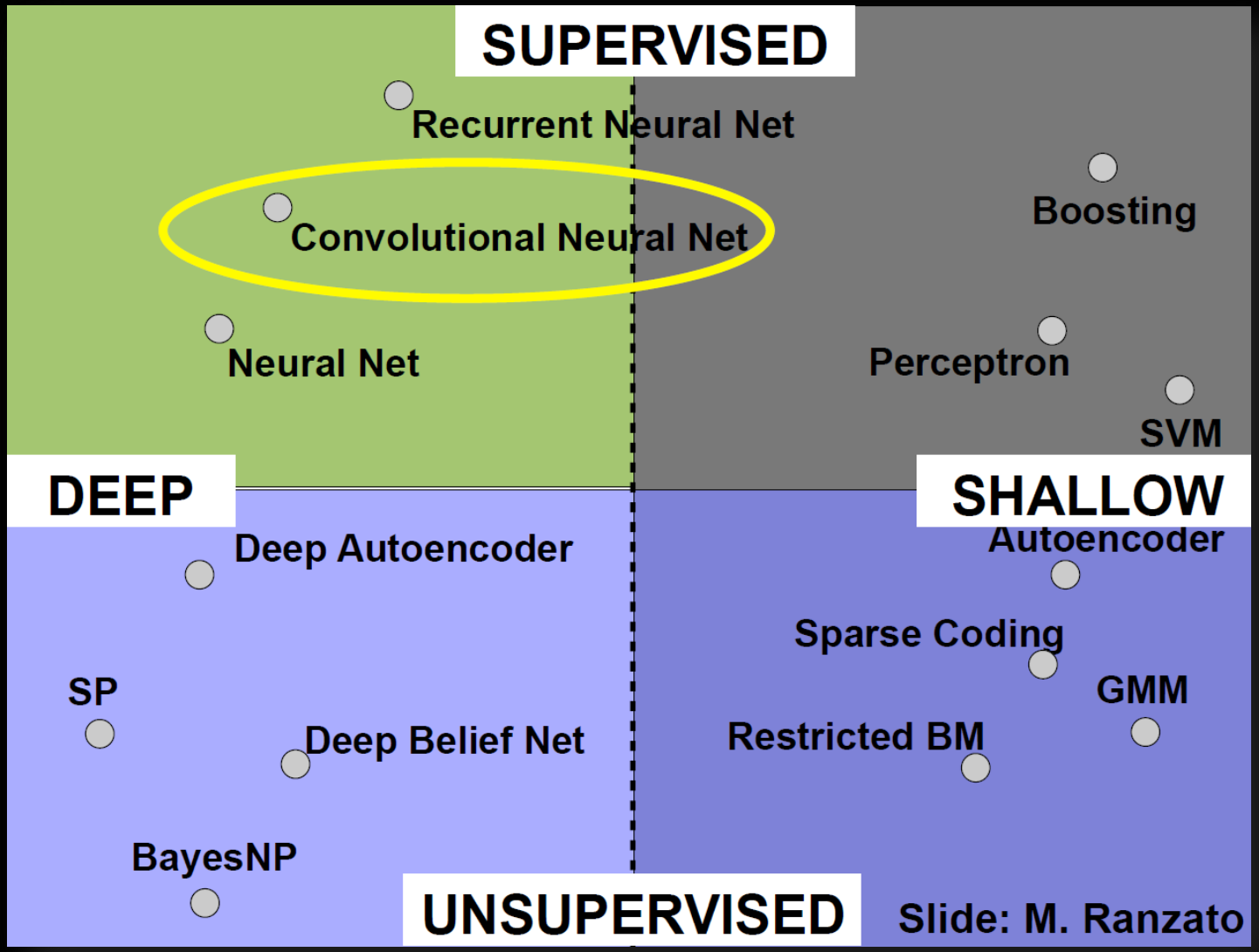
扩大训练数据的数据规模就可以提高预测的效果么？

为什么深度学习会在图像、音频、NLP领域成功？

最重要的，如果一开始就针对的错误的问题……

如果你能看到故事中有的地方技术上好像不太对，那么恭喜你已经入门☺

没有提到的跟机器学习有关的内容



没有提到的跟机器学习有关的内容

$$p(\mathbf{v}) = \sum_{\mathbf{h}} \frac{e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{u},\mathbf{g}} e^{-E(\mathbf{u},\mathbf{g})}}. \tag{1}$$

$$E(\mathbf{v},\mathbf{h}) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} w_{ij} v_i h_j, \tag{2}$$

$$p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h}) \quad \text{and} \quad p(v_i = 1|\mathbf{h}) = \text{sigm}\left(a_j + \sum_j h_j w_{ij}\right),$$

$$p(\mathbf{h}|\mathbf{v}) = \prod_j p(h_j|\mathbf{v}) \quad \text{and} \quad p(h_j = 1|\mathbf{v}) = \text{sigm}\left(b_j + \sum_i v_i w_{ij}\right), \tag{3}$$

$$p(v_i = x|\mathbf{h}) = \frac{1}{\sigma_i\sqrt{2\pi}} \cdot e^{-\frac{\left(x - a_i - \sigma_i \sum_j w_{ij} h_j\right)^2}{2\sigma_i^2}},$$

$$p(h_j = 1|\mathbf{v}) = \text{sigm}\left(b_j + \sum_i \frac{v_i}{\sigma_i} w_{ij}\right). \tag{4}$$

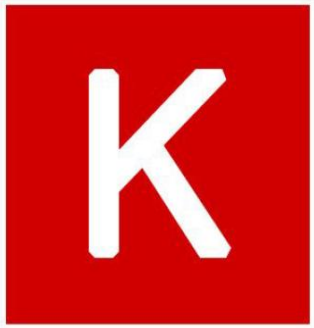
想要学习深度学习，如何上手？

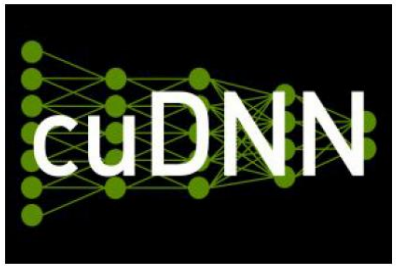
现在学习深度学习是相对容易的

入门阶段已经不需要自己写算法

Deep-Learning Package Zoo

- Torch
- Caffe
- Theano (Keras, Lasagne)
- CuDNN
- Tensorflow
- Mxnet
- Etc.





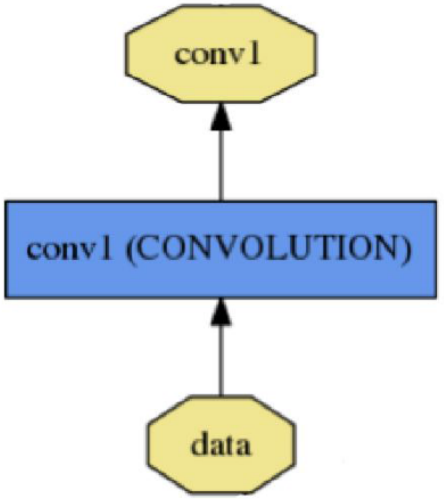
以两个最常用的包为例

A Caffe Layer

```
name: "conv1"
type: CONVOLUTION
bottom: "data"
top: "conv1"
convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
        type: "xavier"
    }
}
```

name, type, and the connection structure (input blobs and output blobs)

layer-specific parameters



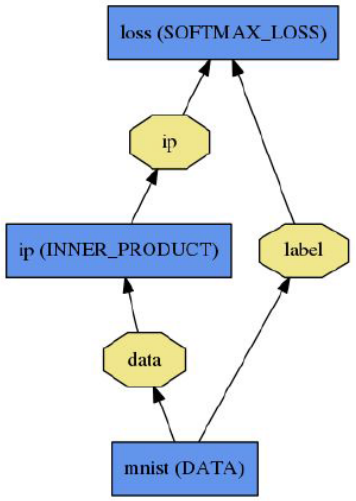
*Link to the Google Protobuffer Documentation

from the tutorial by Evan Shelhamer , Evan Shelhamer, Jeff Donahue, Yangqing Jia, and Ross Girshick

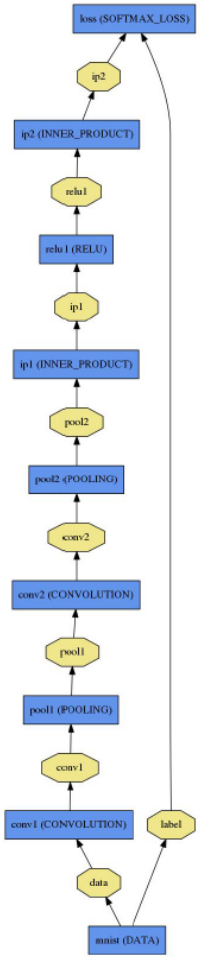
A Caffe Network

- A network is a set of layers connected as a DAG:

```
name: "dummy-net"
layers { name: "data" …}
layers { name: "conv" …}
layers { name: "pool" …}
    … more layers …
layers { name: "loss" …}
```



LogReg ↑



LeNet →

from the tutorial by Evan Shelhamer , Evan Shelhamer, Jeff Donahue, Yangqing Jia, and Ross Girshick

MNIST：深度学习领域的 Hello World

THE MNIST DATABASE

of handwritten digits

Yann LeCun, Courant Institute, NYU
Corinna Cortes, Google Labs, New York
Christopher J.C. Burges, Microsoft Research, Redmond

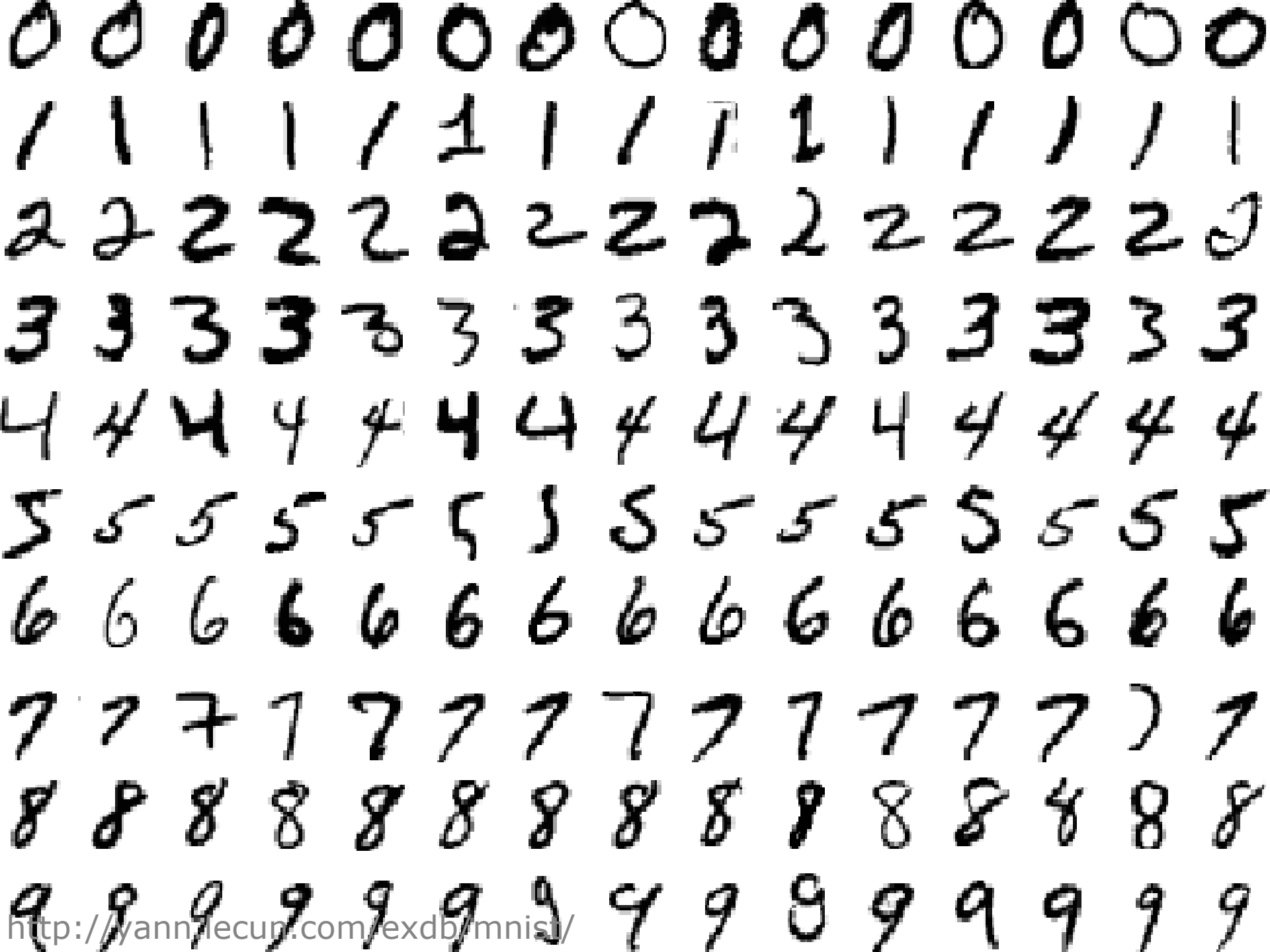
The MNIST database of handwritten digits, available from this page, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

It is a good database for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on preprocessing and formatting.

Four files are available on this site:

```
train-images-idx3-ubyte.gz:  training set images (9912422 bytes)
train-labels-idx1-ubyte.gz:  training set labels (28881 bytes)
t10k-images-idx3-ubyte.gz:   test set images (1648877 bytes)
t10k-labels-idx1-ubyte.gz:   test set labels (4542 bytes)
```

http://yann.lecun.com/exdb/mnist/


http://yann.lecun.com/exdb/mnist/

Misclassified examples

Credit: Barnabás Póczos & Aarti Singh, CMU-10701

Caffe

Deep learning framework by the BVLC

Created by
Yangqing Jia
Lead Developer
Evan Shelhamer

View On GitHub

Training LeNet on MNIST with Caffe

We will assume that you have Caffe successfully compiled. If not, please refer to the Installation page. In this tutorial, we will assume that your Caffe installation is located at `CAFFE_ROOT`.

Prepare Datasets

You will first need to download and convert the data format from the MNIST website. To do this, simply run the following commands:

```
cd $CAFFE_ROOT
./data/mnist/get_mnist.sh
./examples/mnist/create_mnist.sh
```

If it complains that `wget` or `gunzip` are not installed, you need to install them respectively. After running the script there should be two datasets, `mnist_train_lmdb`, and `mnist_test_lmdb`.

LeNet: the MNIST Classification Model

Before we actually run the training program, let's explain what will happen. We will use the LeNet network, which is known to work well on digit classification tasks. We will use a slightly different version from the original LeNet implementation, replacing the sigmoid activations with Rectified Linear Unit (ReLU) activations for the neurons.

Caffe

Deep learning framework by the BVLC

Created by
Yangqing Jia
Lead Developer
Evan Shelhamer

View On GitHub

Caffe Model Zoo

Lots of researchers and engineers have made Caffe models for different tasks with all kinds of architectures and data. These models are learned and applied for problems ranging from simple regression, to large-scale visual classification, to Siamese networks for image similarity, to speech and robotics applications.

To help share these models, we introduce the model zoo framework:

- A standard format for packaging Caffe model info.
- Tools to upload/download model info to/from Github Gists, and to download trained `.caffemodel` binaries.
- A central wiki page for sharing model info Gists.

Where to get trained models

First of all, we bundle BVLC-trained models for unrestricted, out of the box use.
See the BVLC model license for details. Each one of these can be downloaded by running `scripts/download_model_binary.py <dirname>` where `<dirname>` is specified below:

如何上手：推荐直接用现有的模型

先下载和编译好 Caffe；

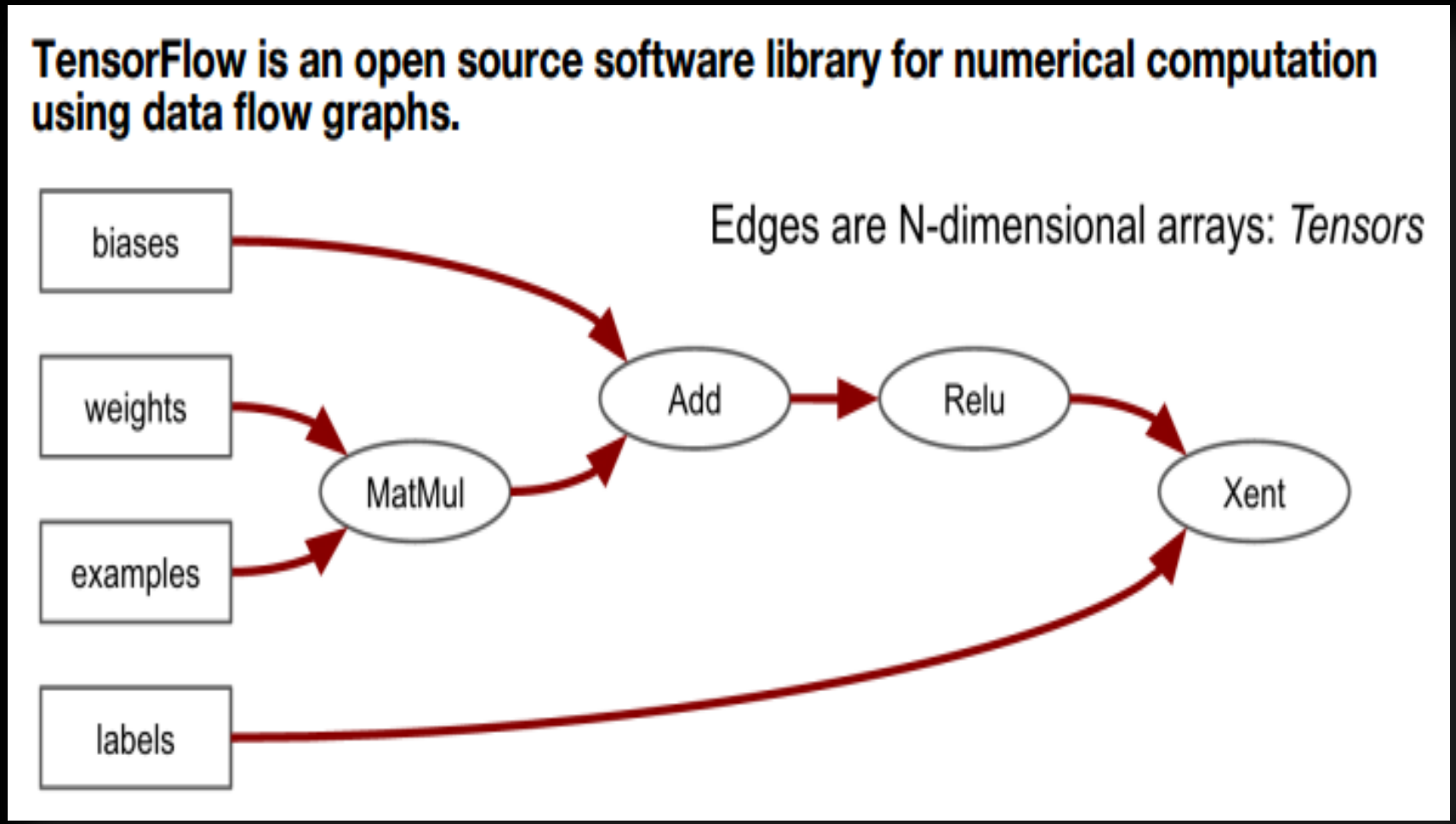
训练一个针对 MNIST 数据集的 LeNet 模型；

用用 AlexNet，跑起来 Web demo;

下载 Inception、VGG、的权重，跑起来模型；

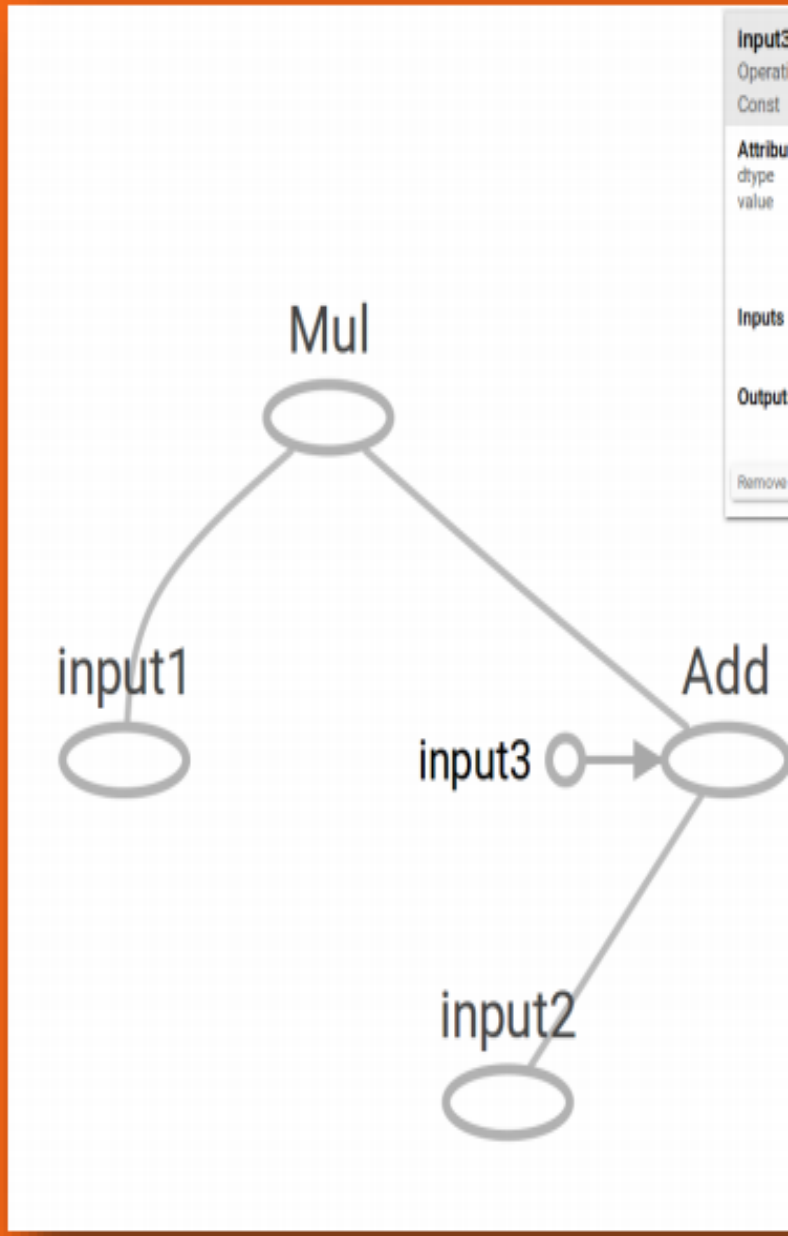
跑起来 OpenFace 的 Web Demo；

TensorFlow：搞清楚Tensor和Flow就行



https://indico.cern.ch/event/433556/contributions/1930588/attachments/1231746/1806025/tensorflow_introduction.pdf

```
import tensorflow as tf

input1 = tf.placeholder(tf.float32)
input2 = tf.placeholder(tf.float32)
input3 = tf.constant(5.0)
intermed = tf.add(input2, input3)
mul = tf.mul(input1, intermed)

with tf.Session() as sess:
  result = sess.run(
    [mul, intermed],
    feed_dict={input1:[1.0],
               input2:[2.0]})
  print(result)
```



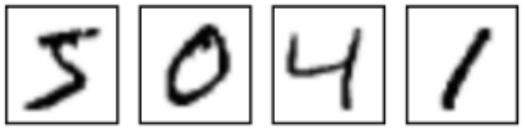
http://flarecast.eu/wp-content/uploads/2015/04/TensoFlow.pdf

MNIST For ML Beginners

This tutorial is intended for readers who are new to both machine learning and TensorFlow. If you already know what MNIST is, and what softmax (multinomial logistic) regression is, you might prefer this faster paced tutorial. Be sure to install TensorFlow before starting either tutorial.

When one learns how to program, there's a tradition that the first thing you do is print "Hello World." Just like programming has Hello World, machine learning has MNIST.

MNIST is a simple computer vision dataset. It consists of images of handwritten digits like these:



It also includes labels for each image, telling us which digit it is. For example, the labels for the above images are 5, 0, 4, and 1.

In this tutorial, we're going to train a model to look at images and predict what digits they are. Our goal isn't to train a really elaborate model that achieves state-of-the-art performance -- although we'll give you code to do that later! -- but rather to dip a toe into using TensorFlow. As such, we're going to start with a very simple model, called a Softmax Regression.

The actual code for this tutorial is very short, and all the interesting stuff happens in just three lines. However, it is very important to understand the ideas behind it: both how TensorFlow works and the core machine learning concepts. Because of this, we are going to very carefully work through the code.

README.md

TensorFlow Models

This repository contains machine learning models implemented in TensorFlow. The models are maintained by their respective authors.

To propose a model for inclusion please submit a pull request.

Models

- autoencoder -- various autoencoders
- inception -- deep convolutional networks for computer vision
- namignizer -- recognize and generate names
- neural_gpu -- highly parallel neural computer
- privacy -- privacy-preserving student models from multiple teachers
- resnet -- deep and wide residual networks
- slim -- image classification models in TF-Slim
- swivel -- the Swivel algorithm for generating word embeddings
- syntaxnet -- neural models of natural language syntax
- textsum -- sequence-to-sequence with attention model for text summarization.
- transformer -- spatial transformer network, which allows the spatial manipulation of data within the network
- im2txt -- image-to-text neural network for image captioning.

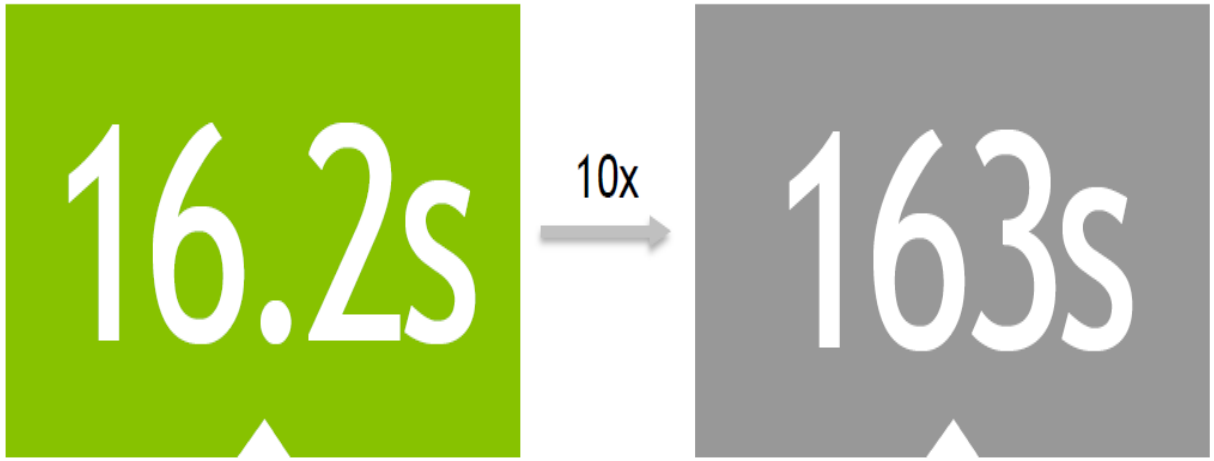
深度学习怎么学，取决于你多有钱

GPU Acceleration

And FPGA、TPU、寒武纪

-gpu N flag tells caffe which gpu to use

Alternatively, specify solver_mode: GPU in solver.prototxt



Tesla K40m, cuDNN v3-RC
ECC off, autoboost on

Intel® Xeon® E5-2698 v3 @ 2.60GHz, 3.6GHz turbo, (16 cores total), HT off

Julien Demouth, Senior Engineer

Benchmark: Train Caffenet model, 20 iterations, 256x256 images, mini-batch size 256

基础 | 前沿 | 讨论

前沿 | ~~NPU~~ | ~~GAN~~ | ~~自编程~~

语言处理 | 语音识别 | 图像识别 | 视频理解

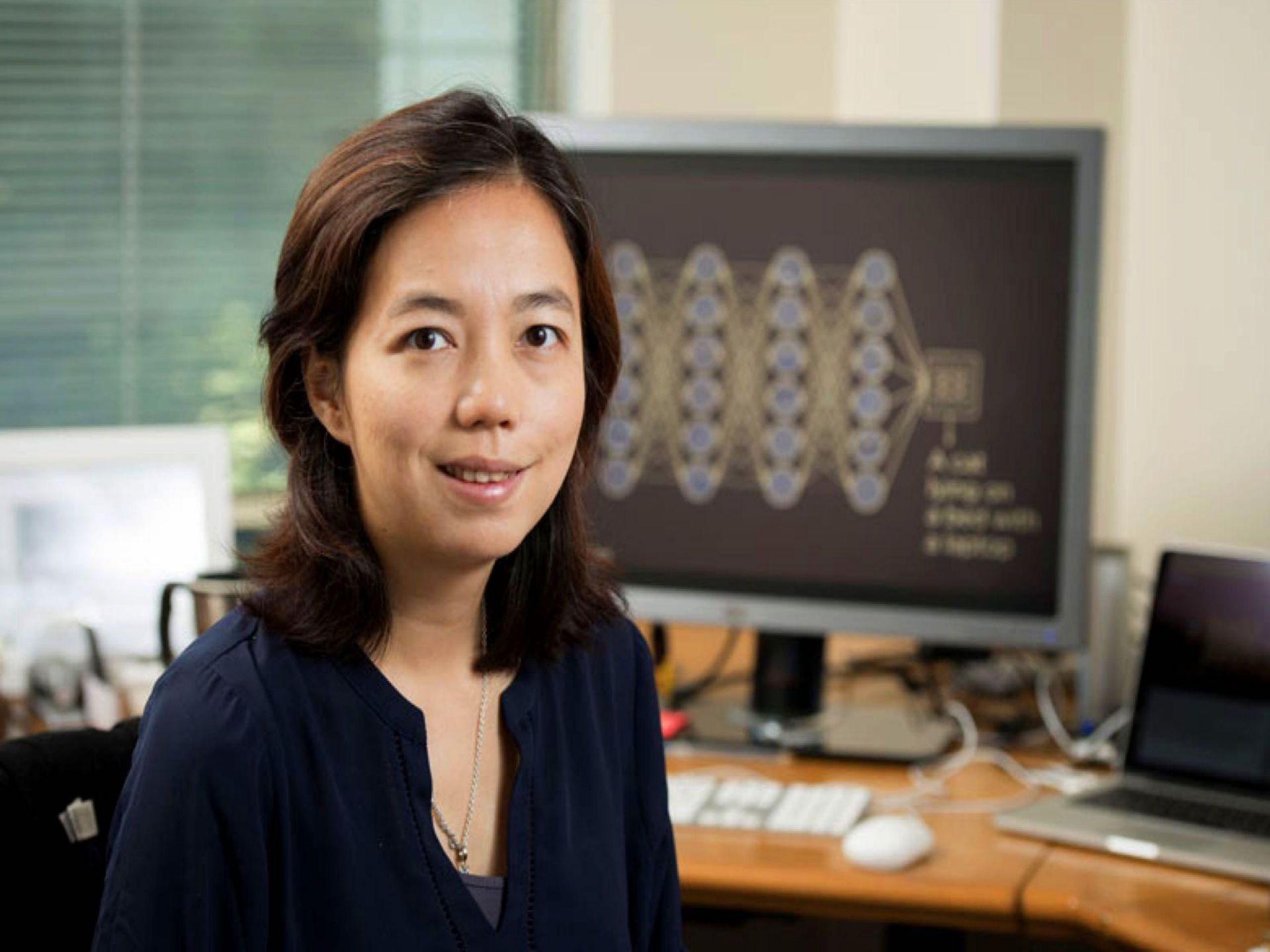
图像识别 | 成果最突出显著

图像识别 | MNIST


http://yann.lecun.com/exdb/mnist/

图像识别 | IMAGENET

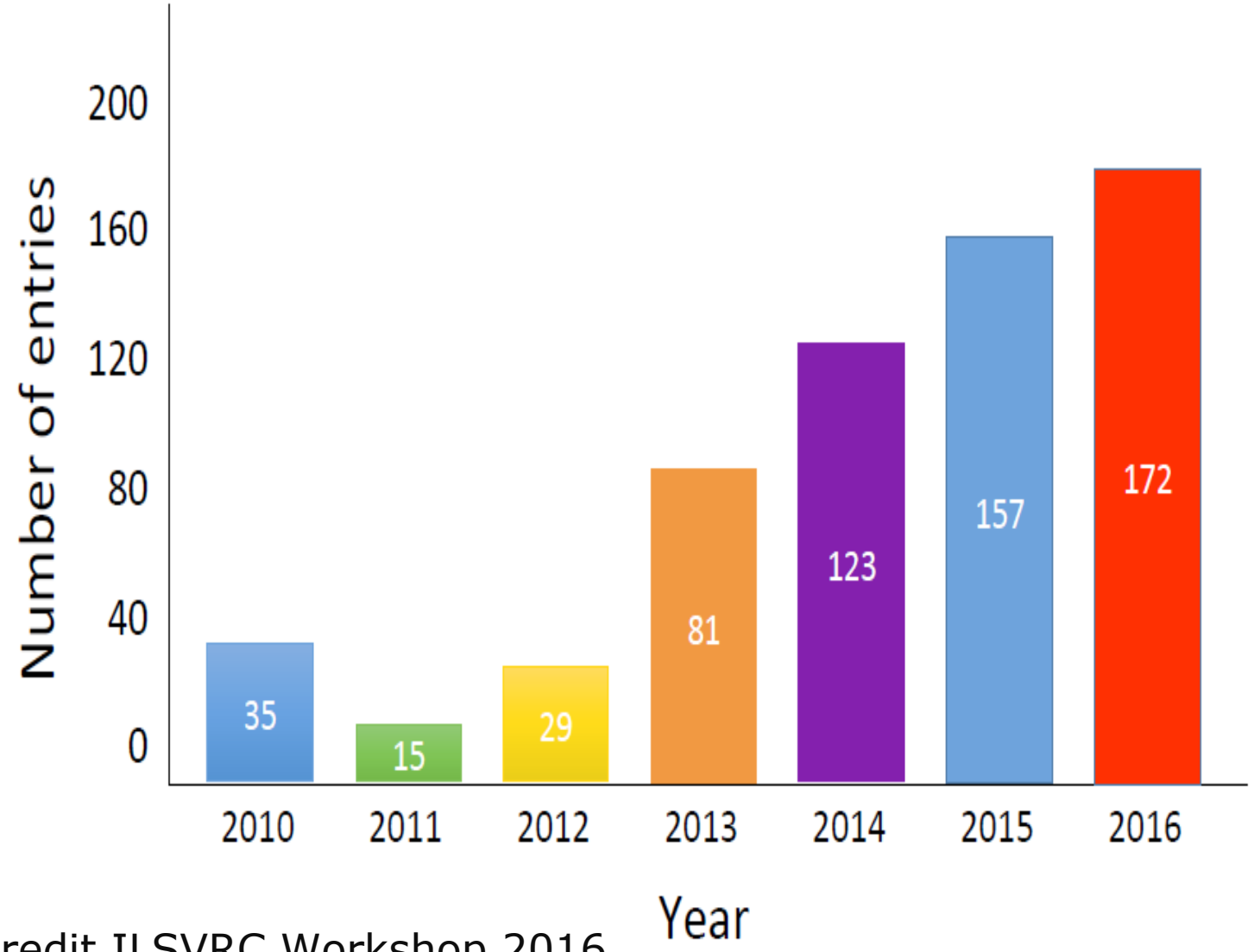
WordNet | 21K | 14M | 1M



图像识别 | ILSVRC

始于2010, 里程碑

Participation in ILSVRC over the years

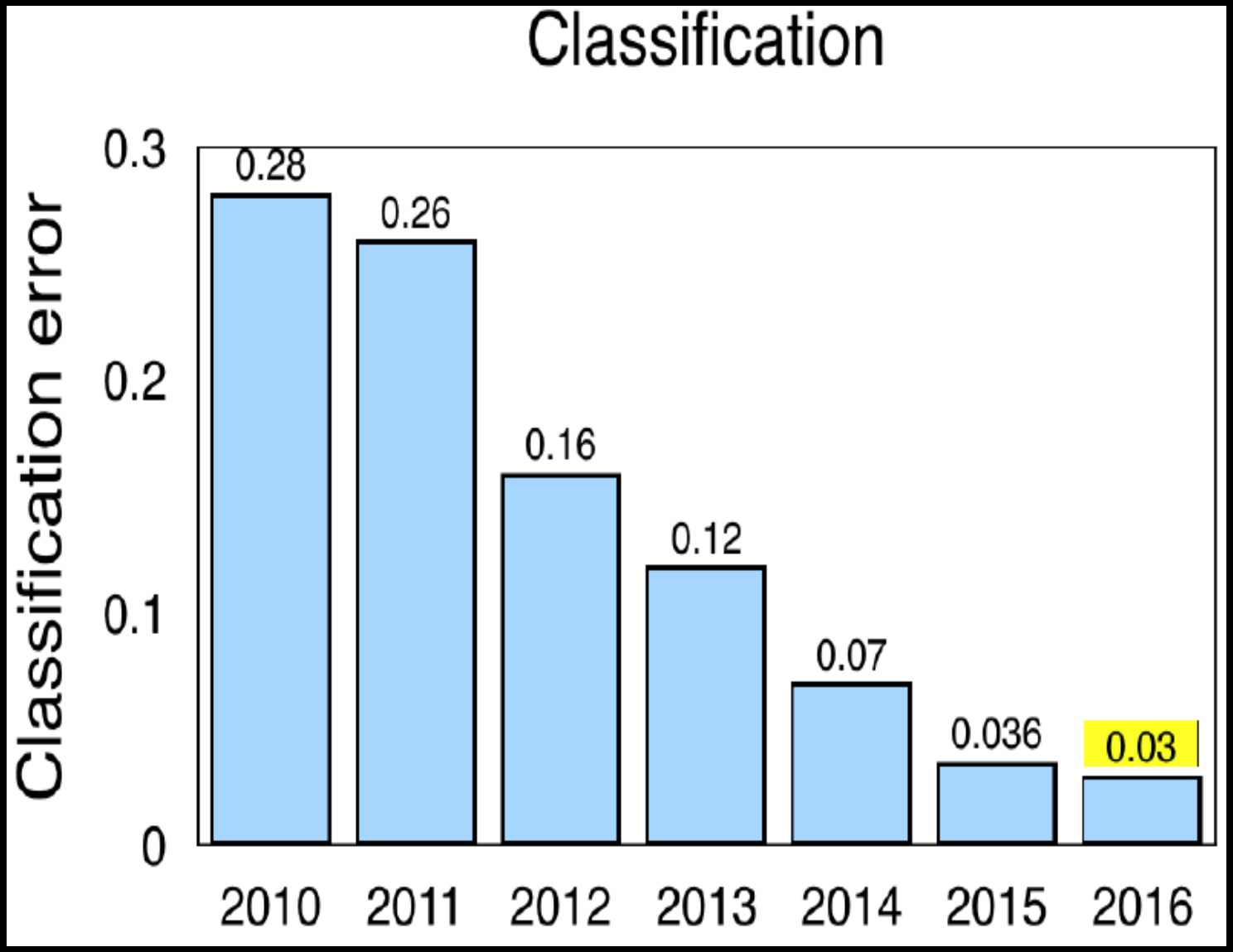


TODO: Credit ILSVRC Workshop 2016

挑战赛项目1：图片内容判定

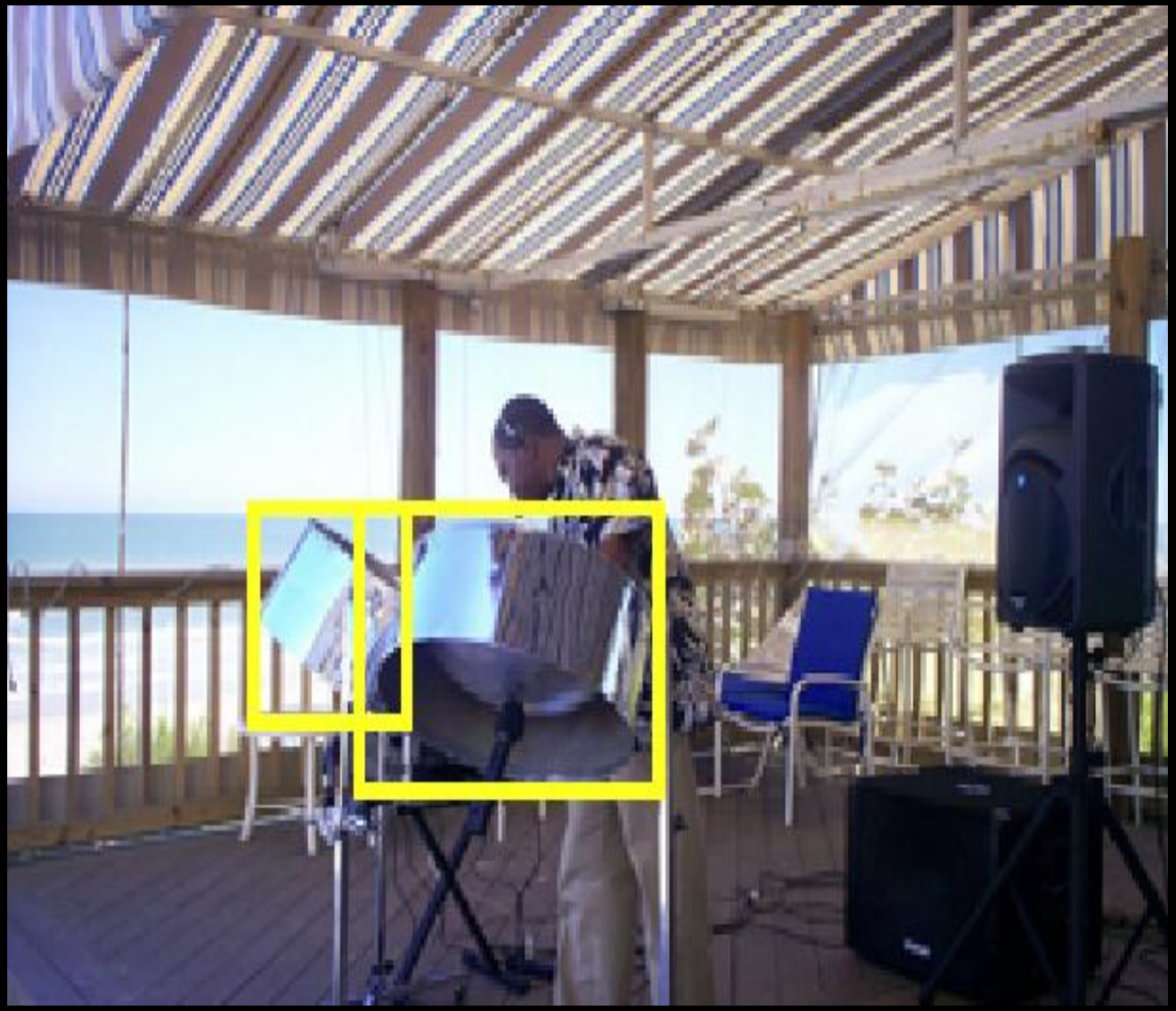
TODO: Credit ILSVRC Workshop 2016

挑战赛项目1：图片内容判定



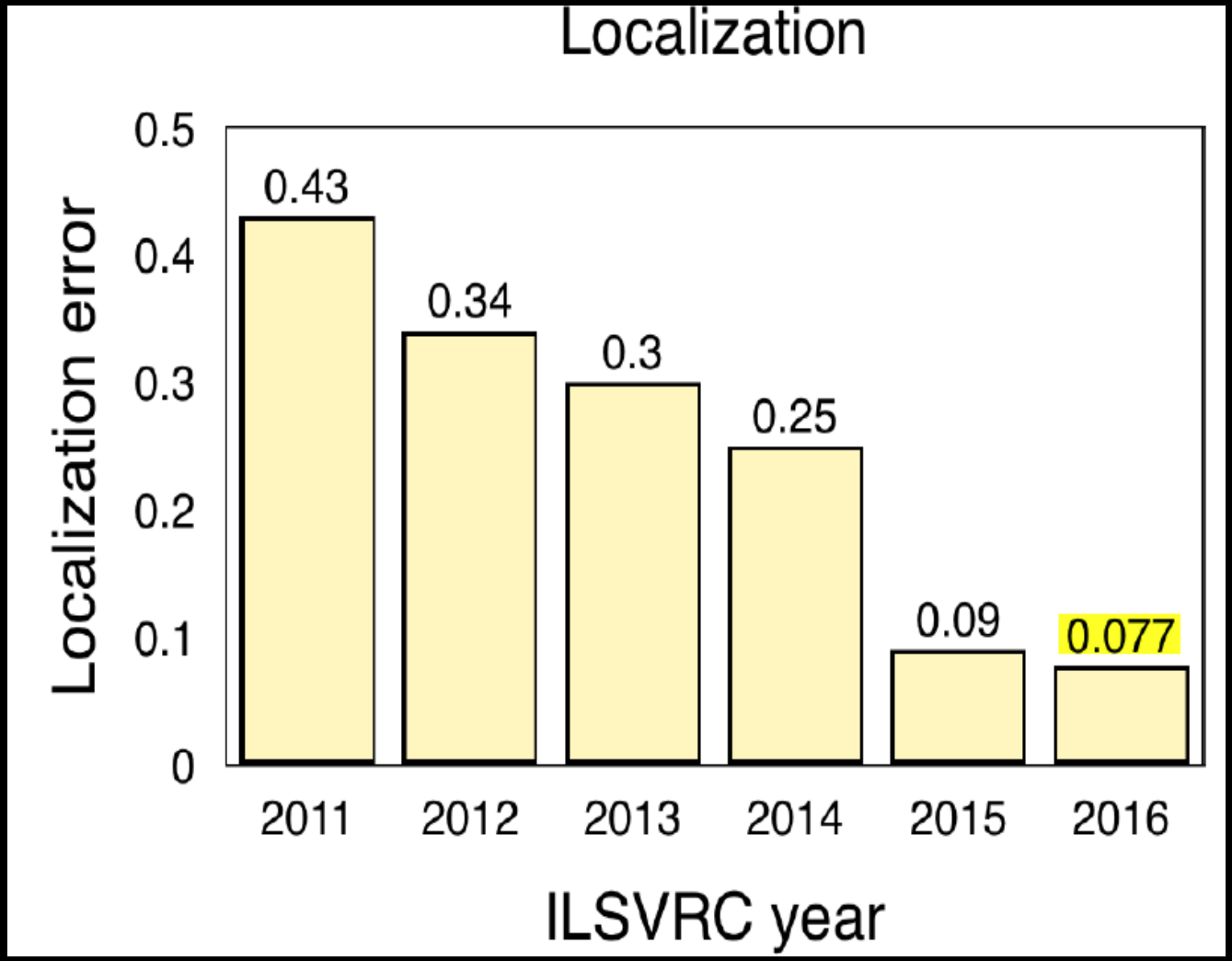
TODO: Credit ILSVRC Workshop 2016

挑战赛项目2: 图片内容定位



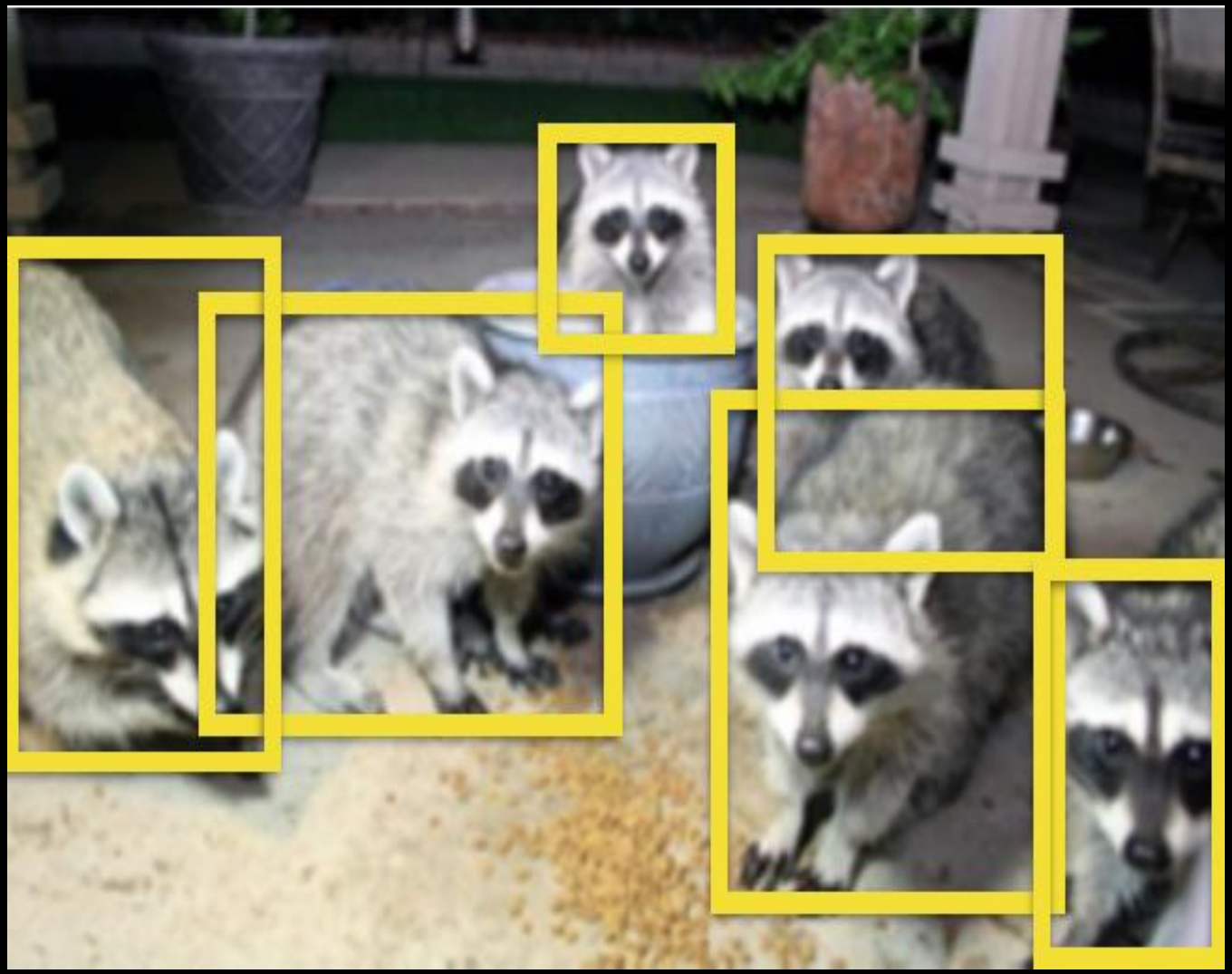
TODO: Credit ILSVRC Workshop 2016

挑战赛项目2：图片内容定位



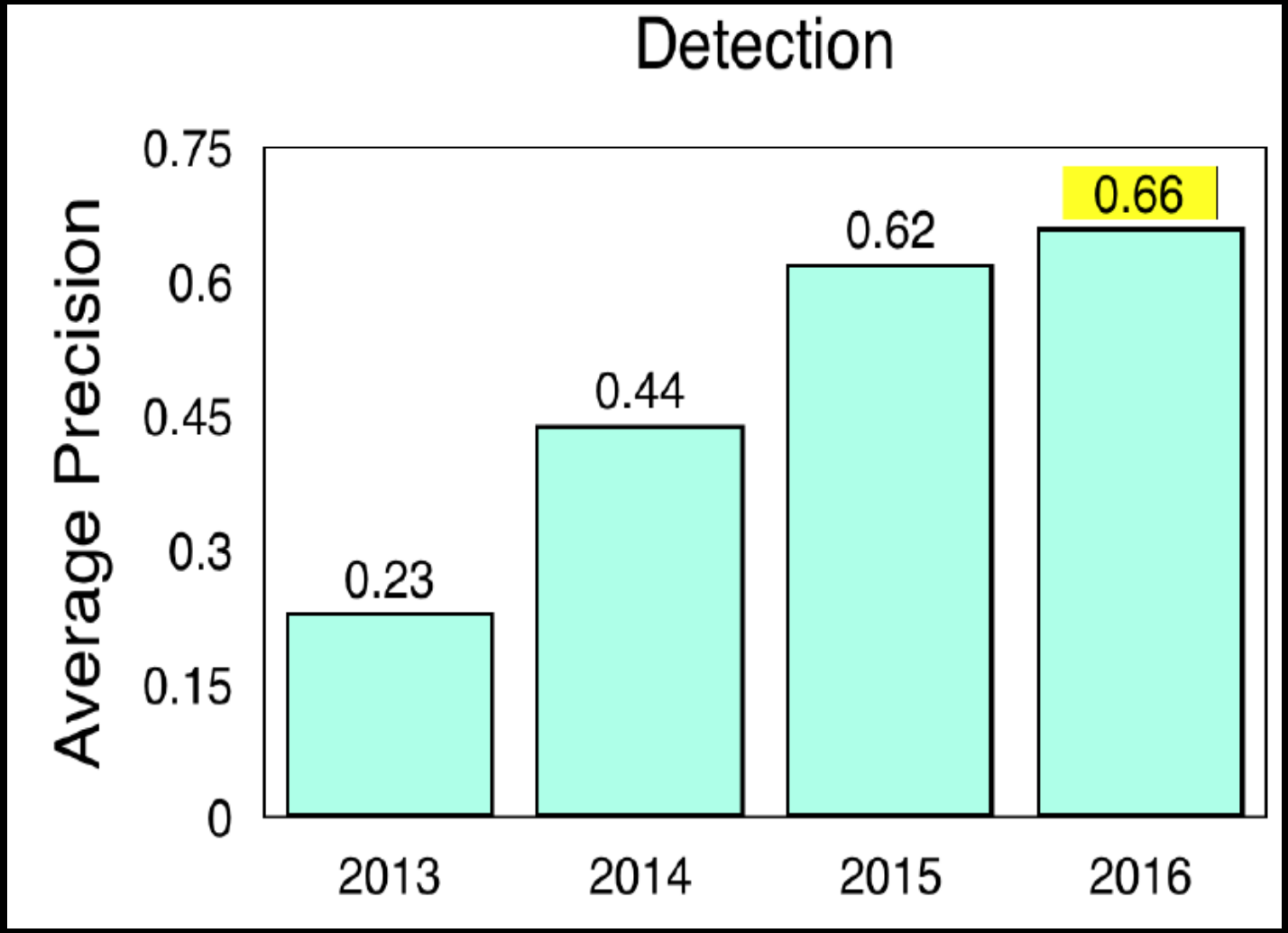
TODO: Credit ILSVRC Workshop 2016

挑战赛项目3: 图片对象检测



TODO: Credit ILSVRC Workshop 2016

挑战赛项目3：图片对象检测



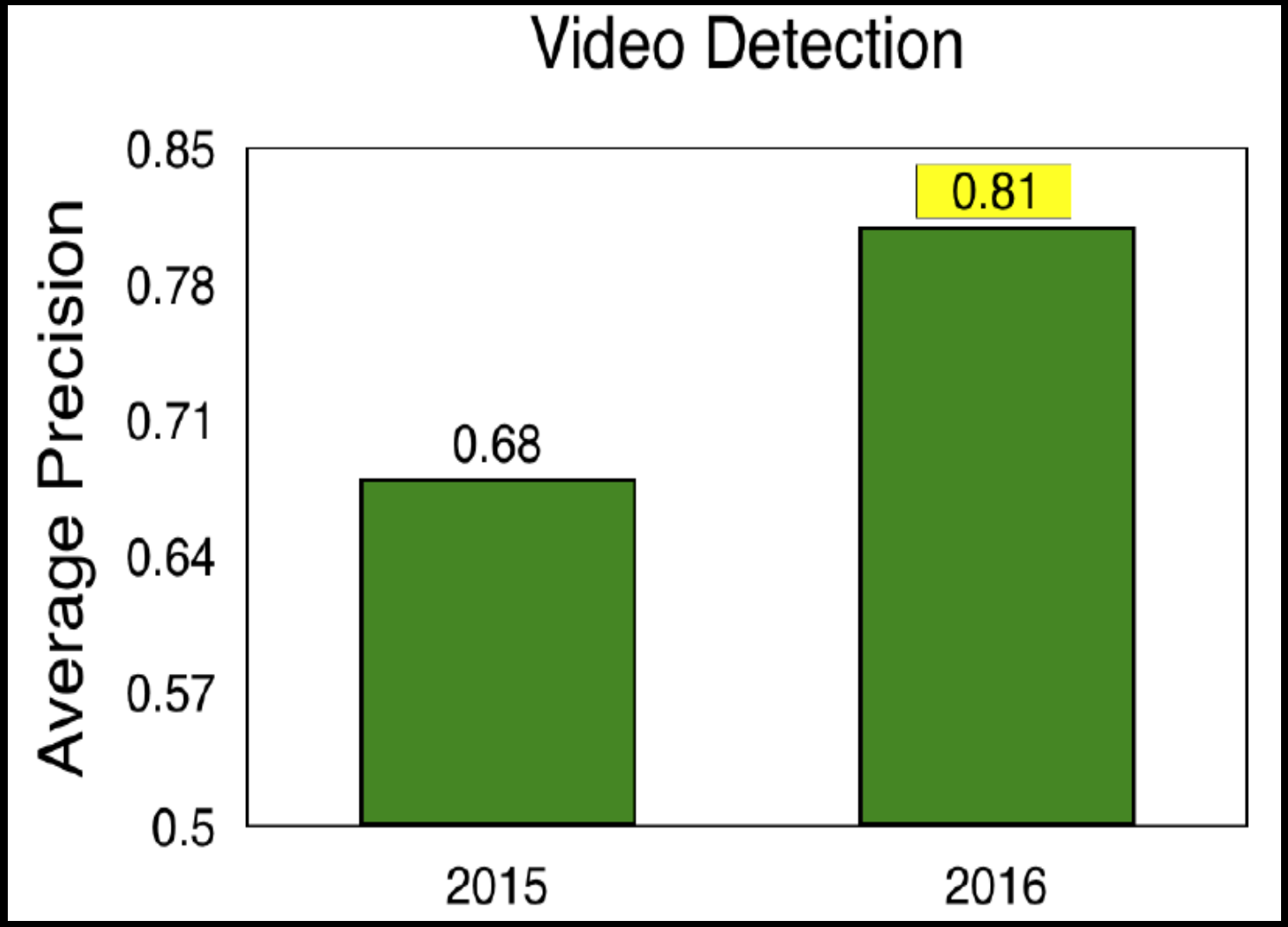
TODO: Credit ILSVRC Workshop 2016

挑战赛项目4: 视频中对象检测

Fully annotated 30 object classes across 6,278 snippets (train+test)

TODO: Credit ILSVRC Workshop 2016

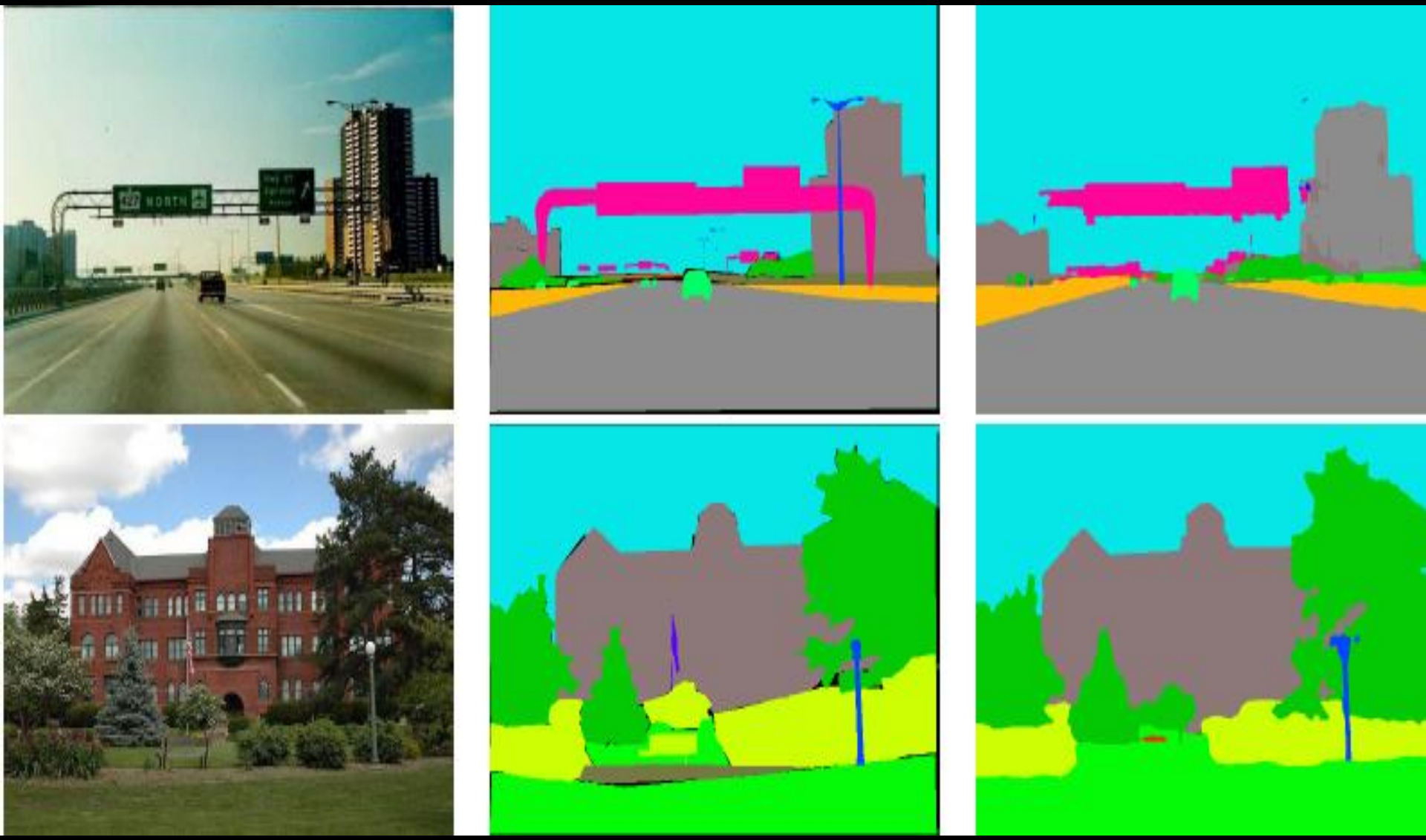
挑战赛项目4: 视频中对象检测



TODO: Credit ILSVRC Workshop 2016

图像识别 | Beyond ImageNet

Sense Classification & Parsing



TODO: Credit ILSVRC Workshop 2016

Image Captioning

A person riding a motorcycle on a dirt road.

Two dogs play in the grass.

TODO: Credit ILSVRC Workshop 2016

人脸识别 | 混战


智造平安城市！ 旷视智能安防解决方案——天眼系统
Face++ 旷视
Face++ 旷视
SKYNET
FACE RECOGNITION
FACE RECOGNITION
FACE RECOGNITION
HUMAN SEGMENTATION


SenseFace-TX1
硬件
NVIDIA Tegra X1
核心板仅一张信用卡大小
质量约75g
功耗6.5 ~ 15W
功能
多人脸检测
人脸关键点提取
人脸质量评估
动态跟踪
性能
1080P监控视频
25hz实时处理
NVIDIA
NVIDIA JETSON TX1 系统模组
专为智能设备设计的全功能模组
英伟达携手商汤科技共建人工智能城市
开启万物感知新时代


SENSEFACE
抓拍人数 1009
告警人数 46
视频路数 4
属性
风格
定位 展会1号相机
任务 人脸分析系统1
11:05
男
19岁
黄种人
睁眼
闭嘴
男
27岁
眼镜
黄种人
睁眼
闭嘴
正在播放
定位 展会1号相机
任务 人脸分析系统1
定位 展会2号相机
任务 人脸分析系统2
定位 展会3号相机
任务 人脸分析系统3
姓名 王晶
性别 男
安博展员工库
梁博
男


SENSEFACE
抓拍人数 1031
告警人数 48
视频路数 4
2016-10-25
属性
风格
姓名 梁博
性别 男
所在库 安博展员工库

Facebook公司公开了DeepFace模型

DeepFace: Closing the Gap to Human-Level Performance in Face Verification

Yaniv Taigman Ming Yang Marc'Aurelio Ranzato Lior Wolf

Facebook AI Research
Menlo Park, CA, USA
{yaniv, mingyang, ranzato}@fb.com

Tel Aviv University
Tel Aviv, Israel
wolf@cs.tau.ac.il

Abstract

In modern face recognition, the conventional pipeline consists of four stages: detect ⇒ align ⇒ represent ⇒ classify. We revisit both the alignment step and the representation step by employing explicit 3D face modeling in order to apply a piecewise affine transformation, and derive a face representation from a nine-layer deep neural network. This deep network involves more than 120 million parameters using several locally connected layers without weight sharing, rather than the standard convolutional layers. Thus we trained it on the largest facial dataset to-date, an identity labeled dataset of four million facial images belonging to more than 4,000 identities. The learned representations coupling the accurate model-based alignment with the large facial database generalize remarkably well to faces in

toward tens of thousands of appearance features in other recent systems [5, 7, 2].

The proposed system differs from the majority of contributions in the field in that it uses the deep learning (DL) framework [3, 21] in lieu of well engineered features. DL is especially suitable for dealing with large training sets, with many recent successes in diverse domains such as vision, speech and language modeling. Specifically with faces, the success of the learned net in capturing facial appearance in a robust manner is highly dependent on a very rapid 3D alignment step. The network architecture is based on the assumption that once the alignment is completed, the location of each facial region is fixed at the pixel level. It is therefore possible to learn from the raw pixel RGB values, without any need to apply several layers of convolutions as is done in many other networks [19, 21]

Google随即公开了FaceNet

FaceNet: A Unified Embedding for Face Recognition and Clustering

Florian Schroff
fschroff@google.com
Google Inc.

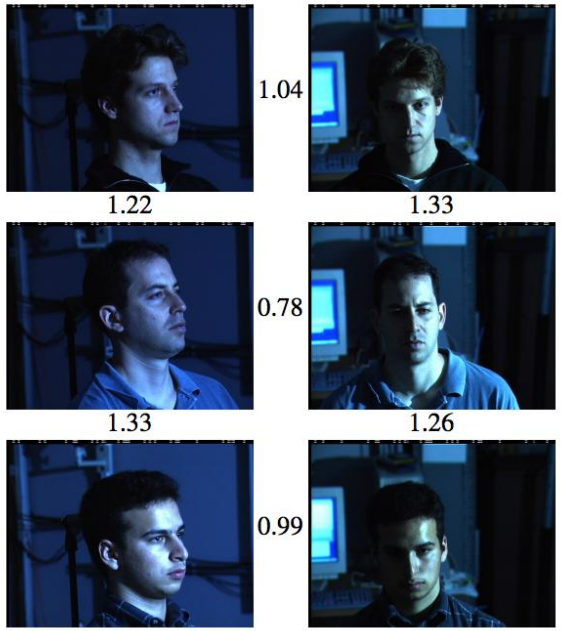
Dmitry Kalenichenko
dkalenichenko@google.com
Google Inc.

James Philbin
jphilbin@google.com
Google Inc.

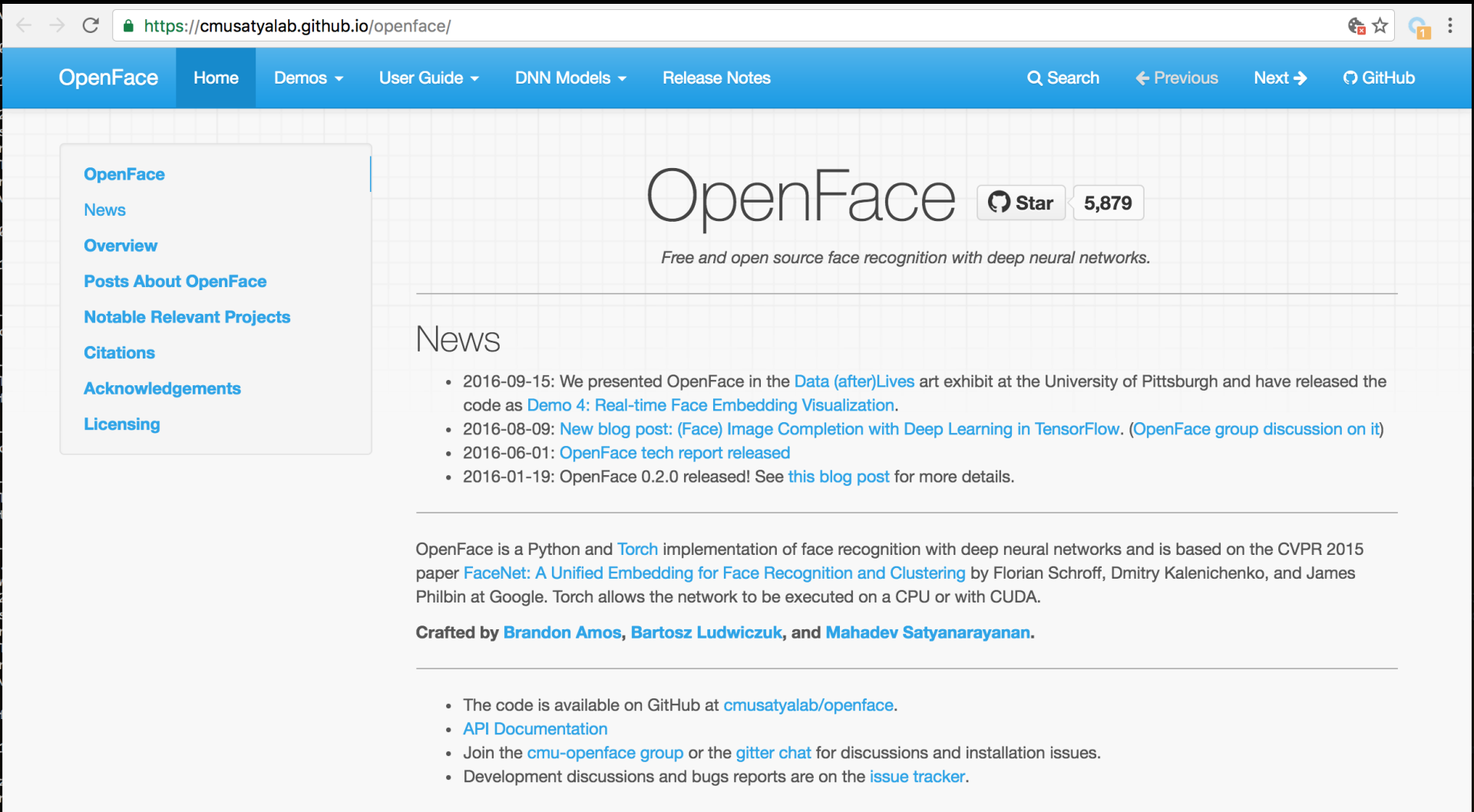
Abstract

Despite significant recent advances in the field of face recognition [10, 14, 15, 17], implementing face verification and recognition efficiently at scale presents serious challenges to current approaches. In this paper we present a system, called FaceNet, that directly learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity. Once this space has been produced, tasks such as face recognition, verification and clustering can be easily implemented using standard techniques with FaceNet embeddings as feature vectors.

Our method uses a deep convolutional network trained to directly optimize the embedding itself, rather than an intermediate bottleneck layer as in previous deep learning approaches. To train, we use triplets of roughly aligned matching / non-matching face patches generated using a novel online triplet mining method. The benefit of our approach is much greater representational efficiency: we achieve state-of-the-art face recognition performance using



开源研究项目填补了预训练权重问题



以及更多的更完善的开源研究项目

GitHub, Inc. [US] https://github.com/davidsandberg/facenet

README.md

FaceNet implementation in Tensorflow

This is a TensorFlow implementation of the face recognizer described in the paper "FaceNet: A Unified Embedding for Face Recognition and Clustering". The project also uses ideas from the paper "A Discriminative Feature Learning Approach for Deep Face Recognition" as well as the paper "Deep Face Recognition" from the Visual Geometry Group at Oxford.

The accuracy measured on the LFW test set is ~0.98 when trained on a combination of FaceScrub and CASIA-WebFace.

Inspiration

The code is heavly inspired by the OpenFace implementation.

Training data

The FaceScrub dataset and the CASIA-WebFace dataset have been used for training. This training set consists of total of 536 685 images over 11105 identities.

Pre-processing

基础 | 前沿 | 讨论

评价 | 泡泡越大，失望越大 (AGI)

评价 | 能解决的问题是有限的

评价丨自动编程，呵呵哒

评价丨警惕伪科学和各类歧视

评价 | 警惕伪科学和各类歧视

Automated Inference on Criminality using Face Images

Xiaolin Wu
McMaster University
Shanghai Jiao Tong University
xwu510@gmail.com

Xi Zhang
Shanghai Jiao Tong University
zhangxi_19930818@sjtu.edu.cn

Abstract

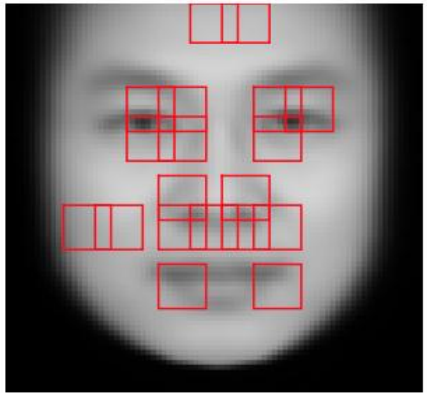
We study, for the first time, automated inference on criminality based solely on still face images, which is free of any biases of subjective judgments of human observers. Via supervised machine learning, we build four classifiers (logistic regression, KNN, SVM, CNN) using facial images of 1856 real persons controlled for race, gender, age and facial expressions, nearly half of whom were convicted criminals, for discriminating between criminals and non-criminals. All four classifiers perform consistently well and empirically establish the validity of automated face-induced

management science, criminology, etc.

In all cultures and all periods of recorded human history, people share the belief that the face alone suffices to reveal innate traits of a person. Aristotle in his famous work Prior Analytics asserted, "It is possible to infer character from features, if it is granted that the body and the soul are changed together by the natural affections". Psychologists have known, for as long as a millennium, the human tendency of inferring innate traits and social attributes (e.g., the trustworthiness, dominance) of a person from his/her facial appearance, and a robust consensus of individuals' inferences . These are the facts found through numerous studies

(a) Three samples in criminal ID photo set S_c.

(b) Three samples in non-criminal ID photo set S_n



(a) (b)

	Mean		Variance	
	criminal	non-criminal	criminal	non-criminal
ρ	0.5809	0.4855	0.0245	0.0187
d	0.3887	0.4118	0.0202	0.0144
θ	0.2955	0.3860	0.0185	0.0130

评价 | 新交互开启新攻击面

感谢各位抽出时间一起学习讨论

扫一扫上面的二维码图案，加我微信