深度学习：基础，前沿，应用，评价

吴 伟
lazyparser@gmail.com
中国科学院软件研究所

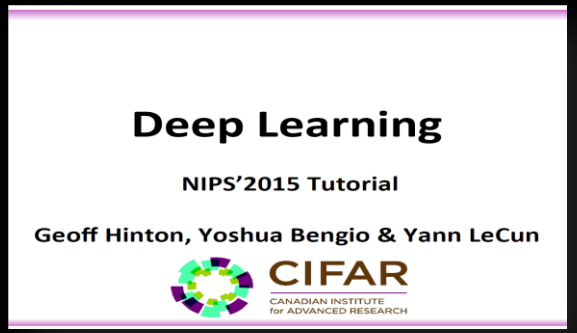
关于我

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(s) | mxnet | TK1

基础 | 前沿 | 应用 | 评价

先缩小一下入门的范围



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

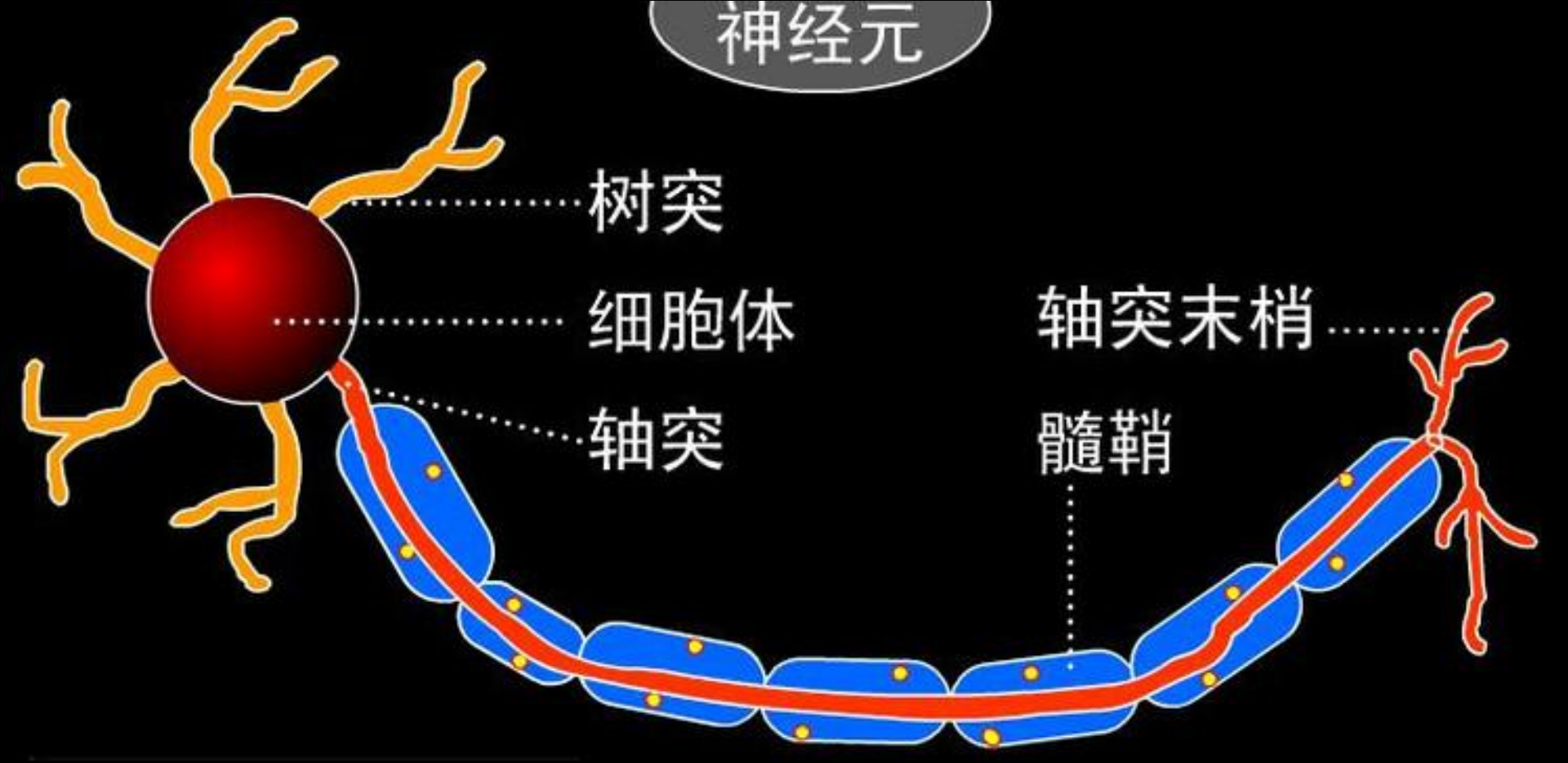
先缩小一下入门的范围

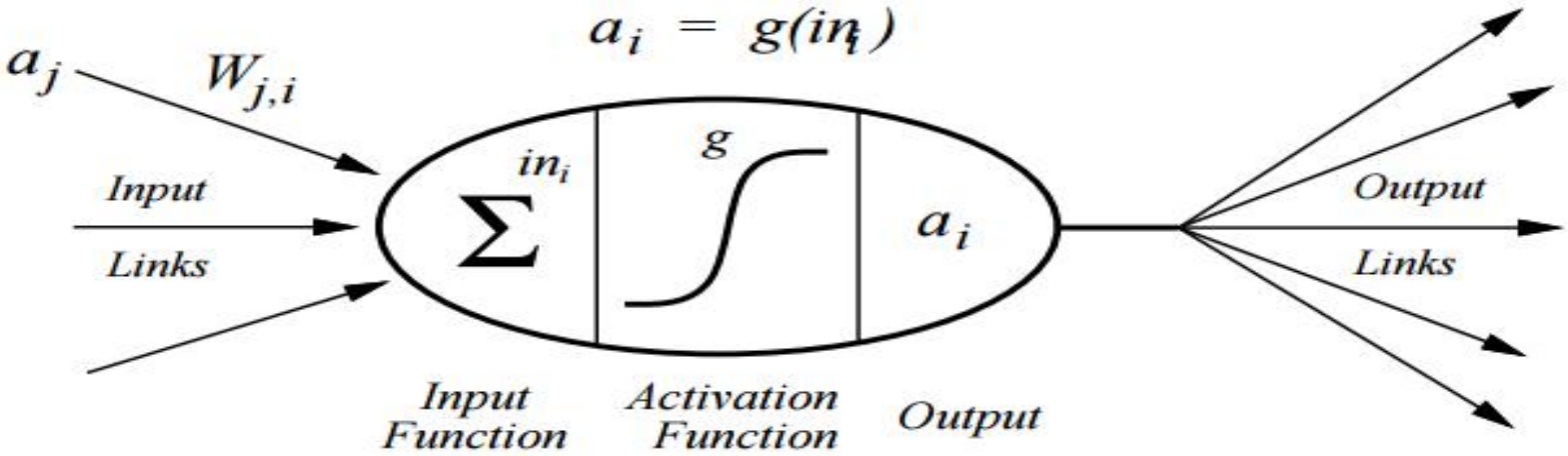
深度学习： 深度神经网络

先缩小一下入门的范围

Neural

深度学习： 深度神经网络





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

Credit: https://page.mi.fu-berlin.de/rojas/neural/chapter/K4.pdf https://appliedgo.net/perceptron/

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

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

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

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

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

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

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

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

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

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

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

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

现场编一个故事背景

某公司老板赵总，有一天突然有了一个灵感：

设计开发一个**人工智能**软件，利用当天股价的涨跌来预测第二天房价的涨跌。

现场编一个故事背景

某公司老板赵总，有一天突然有了一个灵感：

设计开发一个**人工智能**软件，利用当天股价的涨跌来预测第二天房价的涨跌。

于是找来工程师小李，要求实现这个app：

```
if 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

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

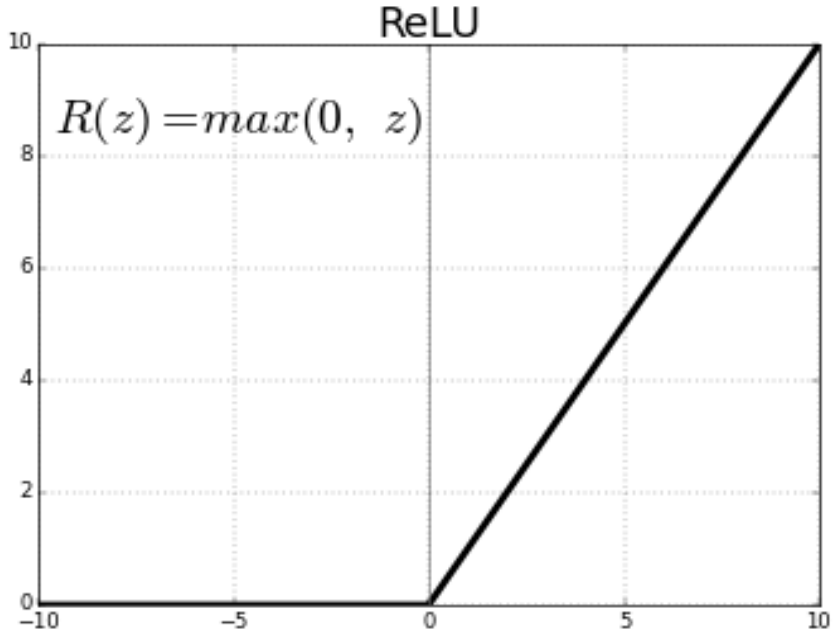
```
if (x > 0)
    y = 1
else /* x <= 0 */
    y = 0
```

产品原型（高级版）

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

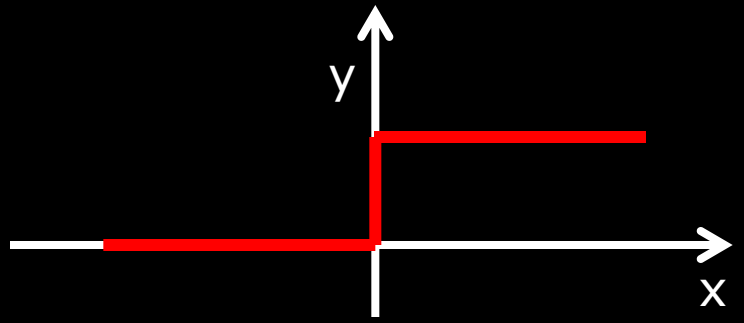
产品原型（高级版）

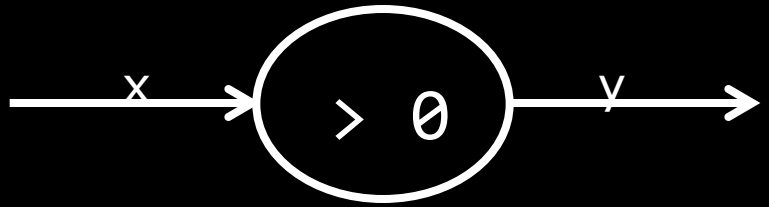
```
if (x > 0)
    y = x
else /* x
    y = 0
```



产品原型

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





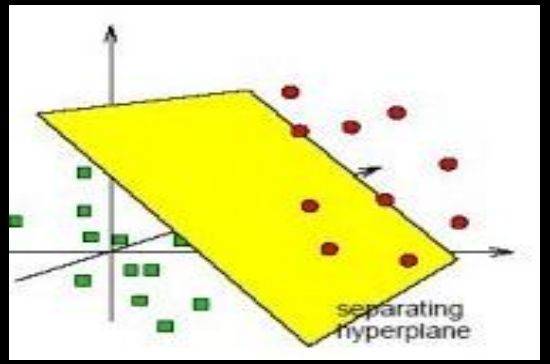
故事背景

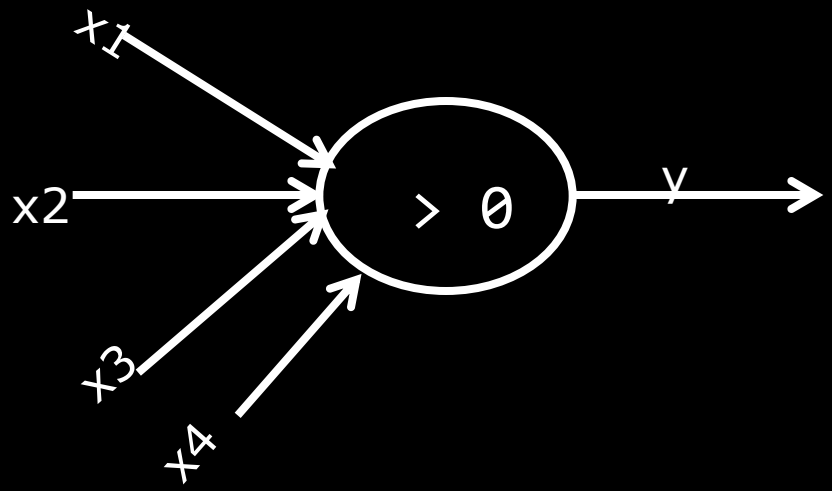
小李很快做出来了产品的原型，但是显然，这个软件是预测不准的……

赵总思考了一下，觉得应该把油价、汇率、金价都考虑进去，这样预测就准了。

产品原型

```
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

小李很快迭代出了产品，但是显然，

这个软件还是预测不准的……

赵总思考了一下，觉得股价、油价、汇率、金价对房价的影响是不一样的，也就是权重（weights）是不同的。

故事背景

小李明白权重的意思，但不知道具体怎么取值。

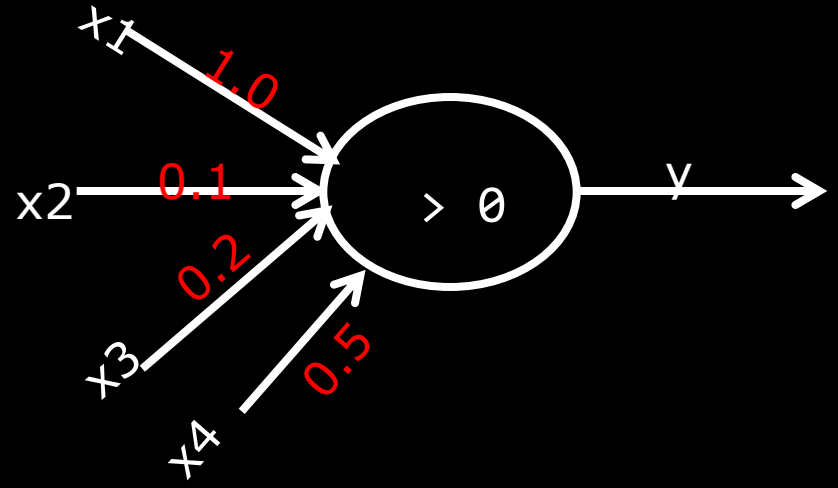
赵总根据经验，告诉小李：

股价、油价、汇率、金价，权重分别是

1.0, 0.1, 0.2, 0.5

产品原型

```
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

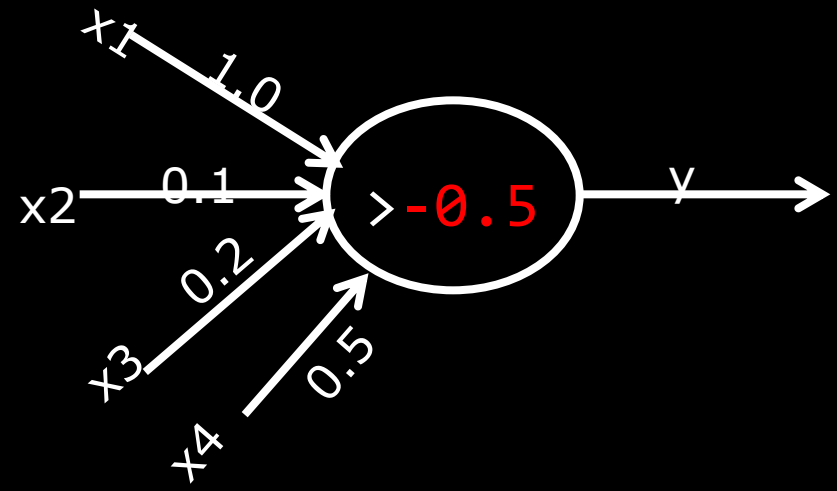
赵总进一步觉得，长远来看，就算市场基本面不变，房价也在缓慢上涨。所以只有指标看跌到一定程度，才能预测说是房价要跌。

赵总根据经验，告诉小李：

只有指标达到了-0.5了，才能算是要跌。

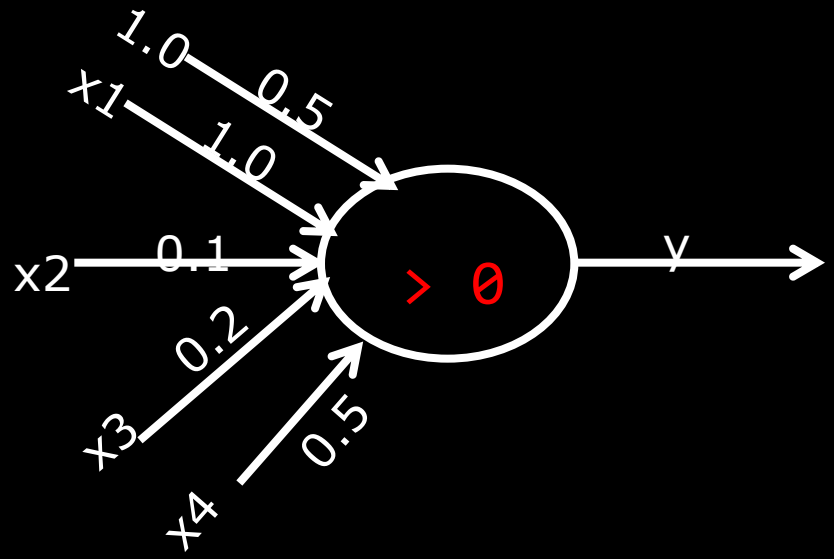
产品原型

```
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
```



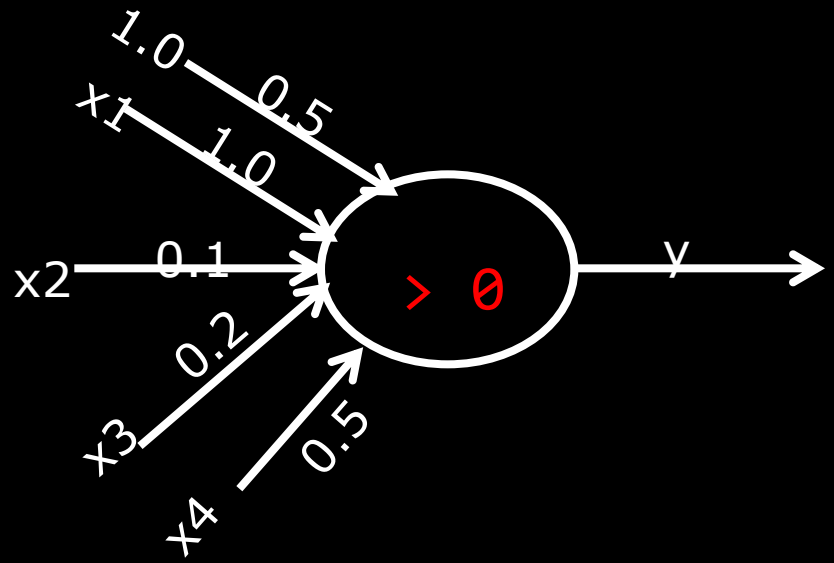
产品原型-小李重构了一下

```
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-小李重构了一下

```
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
```

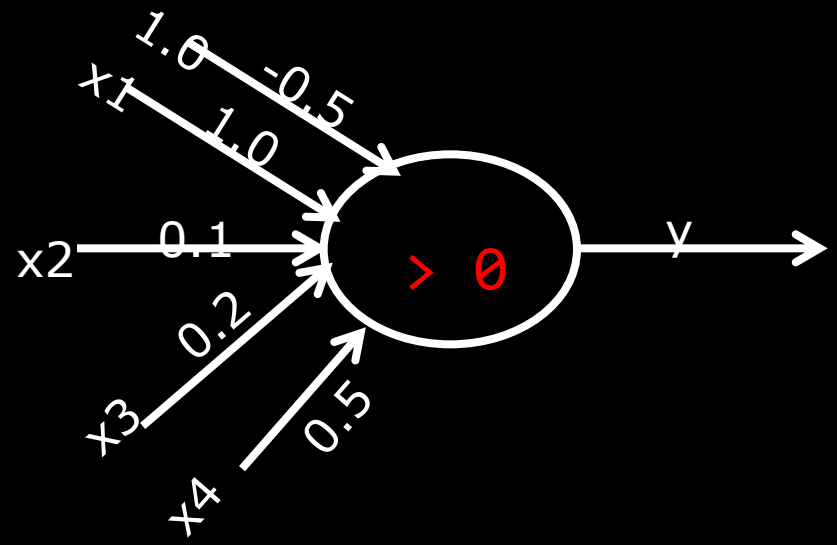


产品原型-小李重构了一下

```
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

软件迭代了四个版本了，测起来还是不准。

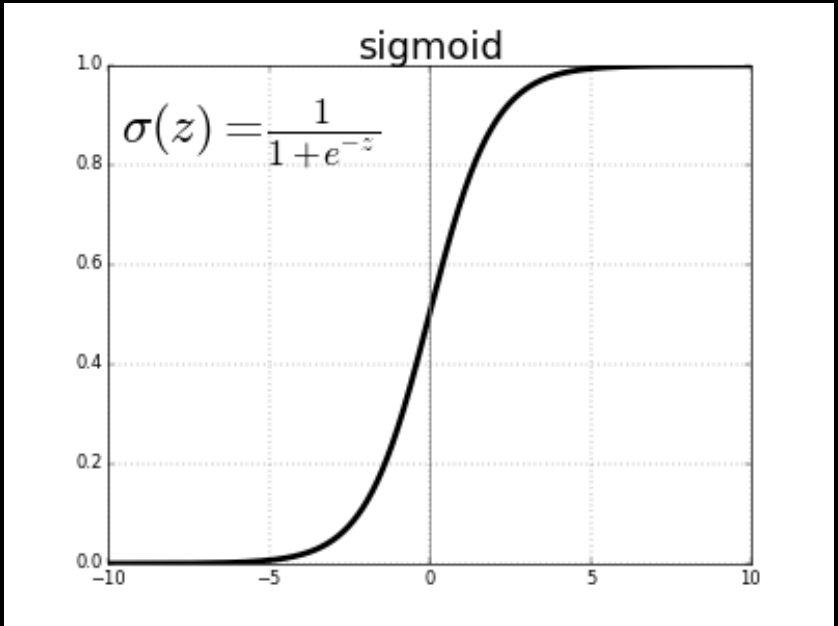
赵总读了微信公众号的深度学习文章，告诉小李：

这个关系很复杂，我们之前用的是线性关系，

不能有效的表示这种复杂的关系，

我们要用“非线性关系”，用最流行的Sigmoid。

Sigmoid 函数（两种解释，类比HTML5）

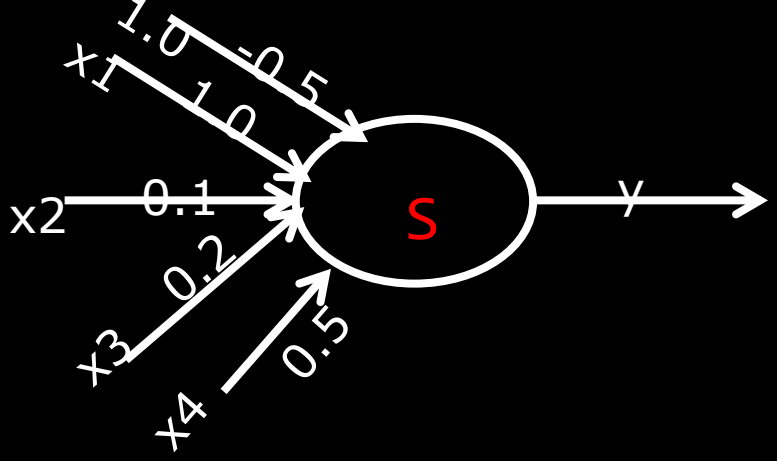


神经网络用Sigmoid函数，其实是为了后面反向传播算法的时候计算微分方便

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

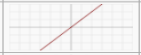
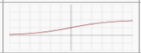
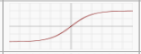
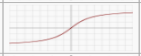
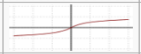
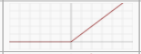
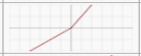
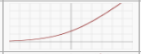
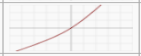
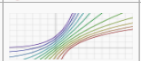
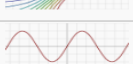
产品原型-小李重构了一下

```
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

软件迭代了五个版本了，测起来还是不准。

赵总觉得之所以测不准，还是因为模型太简单了。

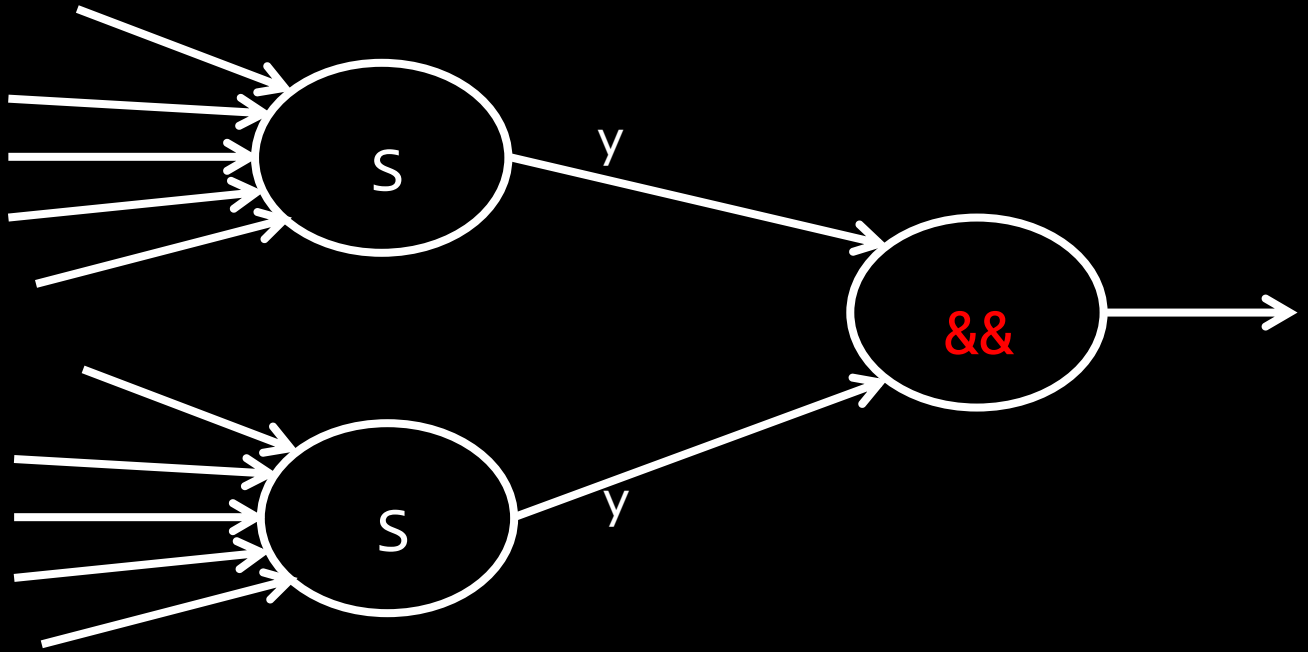
现在深度学习，动辄就说“一百多层”，看看自己，一层都没有，肯定不行。

故事背景

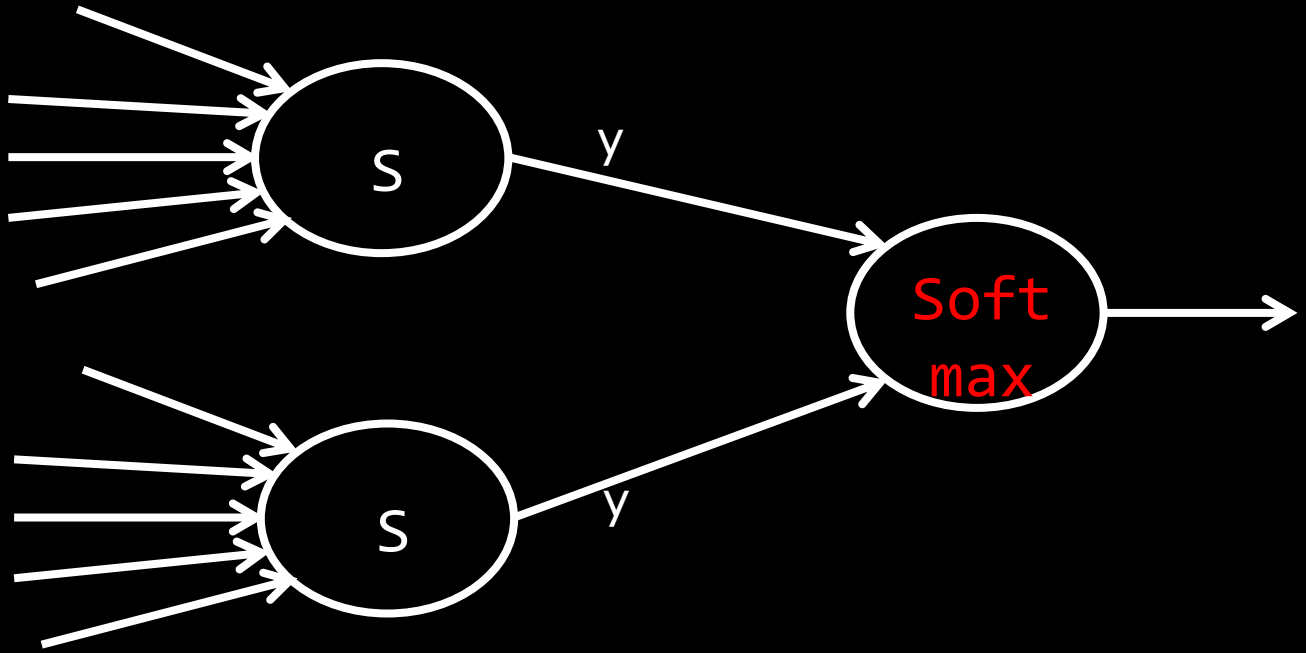
一天早晨，赵总突然想到，买房对谁都是大事，
房价预测岂能用一个公式“一刀切”？

于是把小李找来，告诉他：
我们要进一步完善，用多个不同的计算公式，
综合最终的结果给出预测结果。

小李很快复制粘贴出来了结果， 全说涨才涨



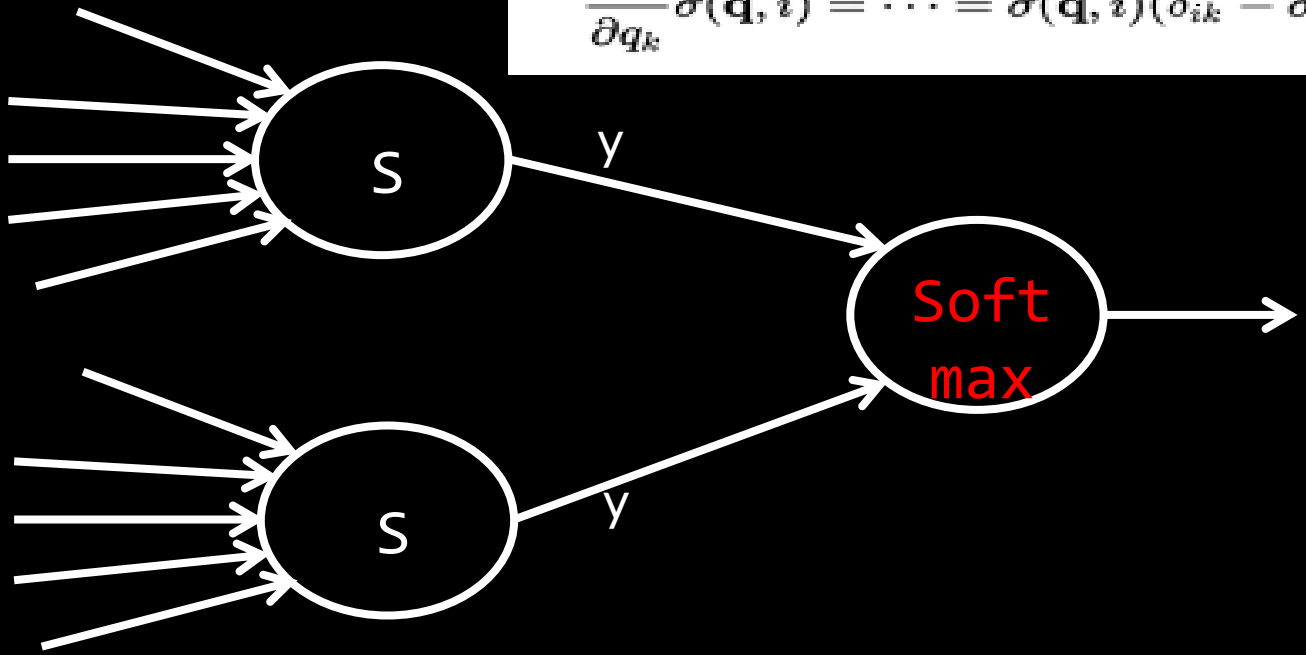
照猫画虎将最后的投票函数改成了 softmax



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

照猫画虎将最后的投票函数改成了 softmax

$$\frac{\partial}{\partial q_k}\sigma(\mathbf{q}, i) = \cdots = \sigma(\mathbf{q}, i)(\delta_{ik} - \sigma(\mathbf{q}, k))$$



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

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……

赵总将这个问题交给了小李，让小李自己去完成。

故事背景

现在小李手里有神经网络（模型），有房价、股价、油价、汇率和金价的历史数据（训练集），也有赵总给的经验权重（初始值）。

小李编了几个权重，效果都不好。接下来怎么办，小李也不知道了……

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].

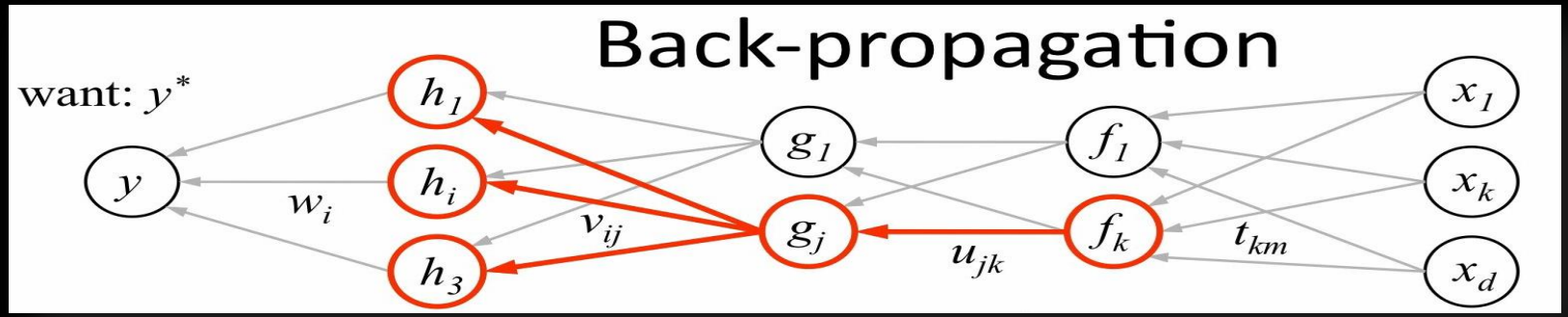
more difficult when we introduce hidden units whose actual or desired states are not specified by the task. (In perceptrons, there are 'feature analysers' between the input and output that are not true hidden units because their input connections are fixed by hand, so their states are completely determined 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.

思路：用历史数据来修正权重

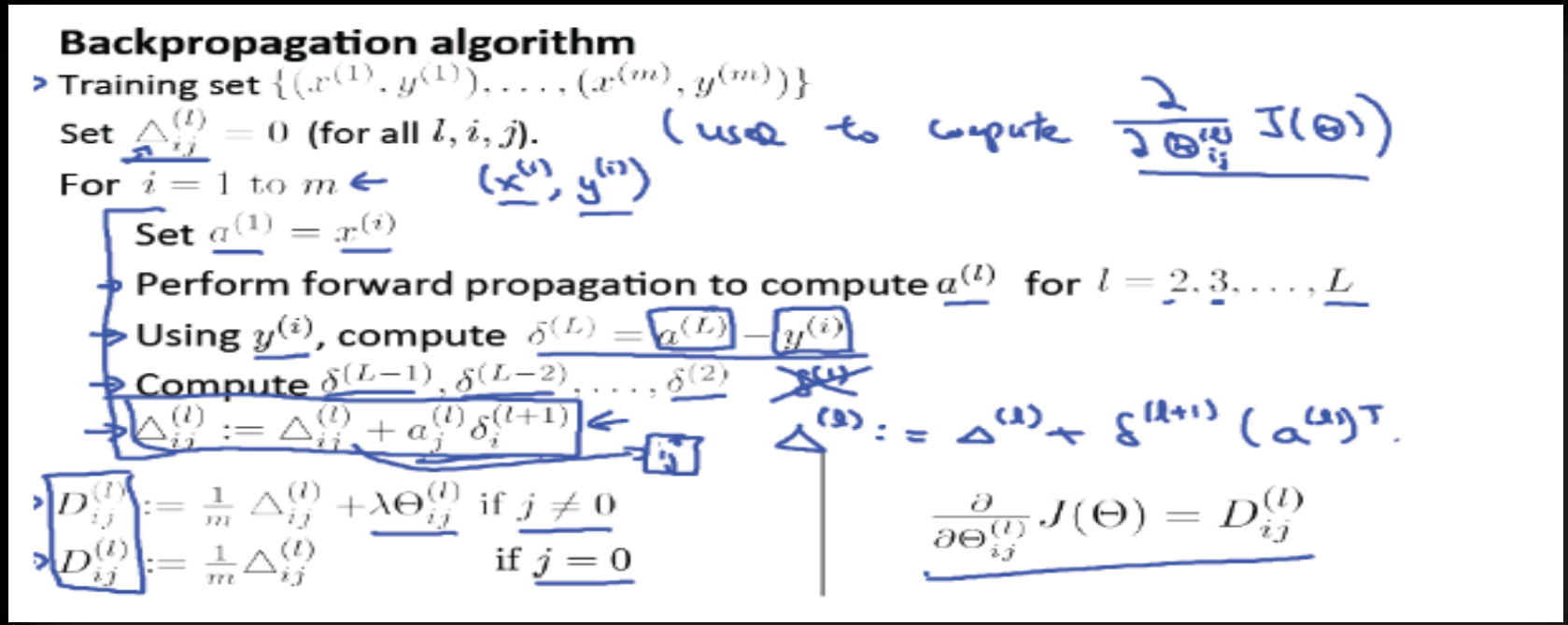
11.1	11.2	11.3	11.4	……	11.30
股价	涨	跌	涨	……	涨
汇率	跌	涨	跌	……	跌
金价	涨	跌	涨	……	涨
油价	跌	涨	跌	……	跌
房价（有监督）	**涨**	**涨**	**涨**	**涨**	**涨**

（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

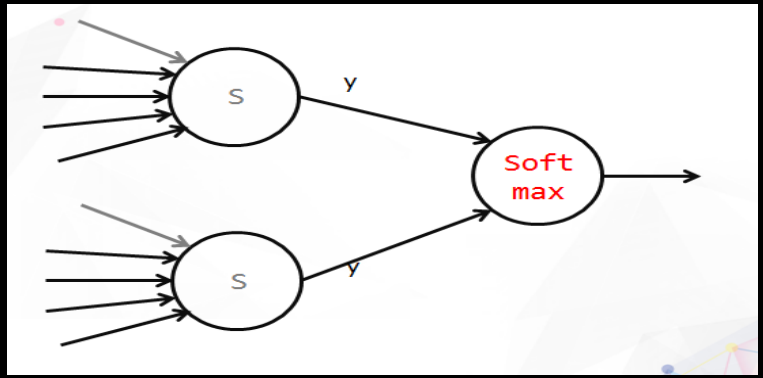


故事背景

小李激动的训练完，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

Backprop in Practice

Y LeCun

- Use ReLU non-linearities
- Use cross-entropy loss for classification
- Use Stochastic Gradient Descent on minibatches
- Shuffle the training samples (← very important)
- Normalize the input variables (zero mean, unit variance)
- Schedule to decrease the learning rate
- Use a bit of L1 or L2 regularization on the weights (or a combination)
  - But it's best to turn it on after a couple of epochs
- Use “dropout” for regularization
- Lots more in [LeCun et al. “Efficient Backprop” 1998]
- Lots, lots more in “Neural Networks, Tricks of the Trade” (2012 edition) edited by G. Montavon, G. B. Orr, and K-R Müller (Springer)
- More recent: Deep Learning (MIT Press book in preparation)

故事背景

小李用了反向传播算法，准确率上去了。

老板很开心，让他加层，并问小李里面权重的意思。

小李表示不知道，于是去问各路大神.....

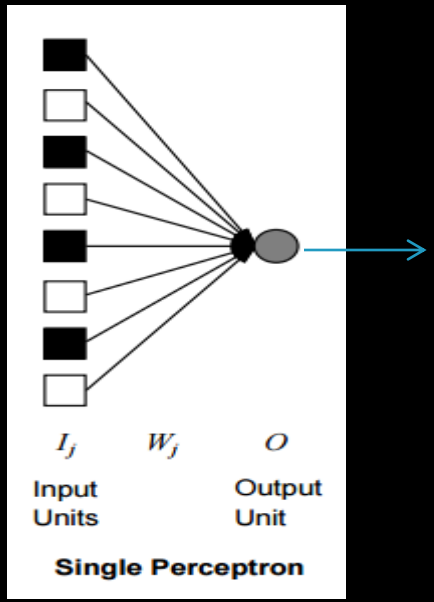
"不知道"

"很复杂"

"不好说"

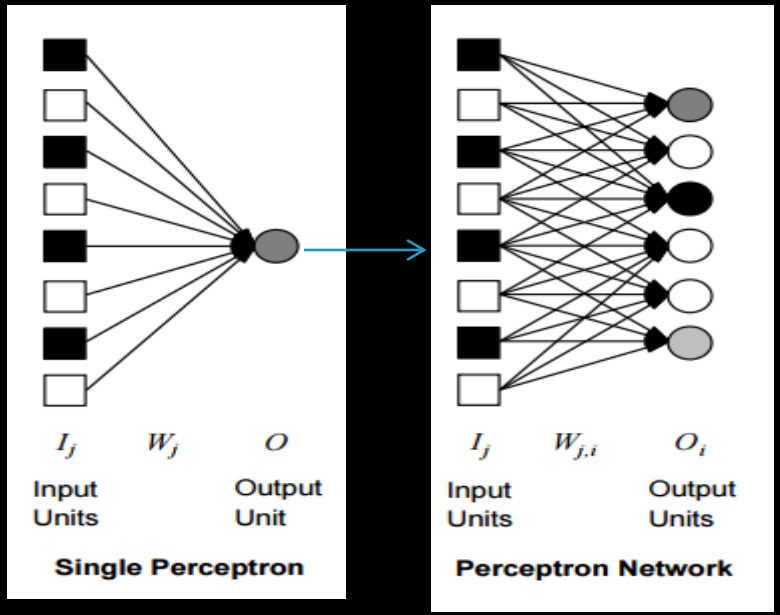
"有点难"

算法在手，层数你有



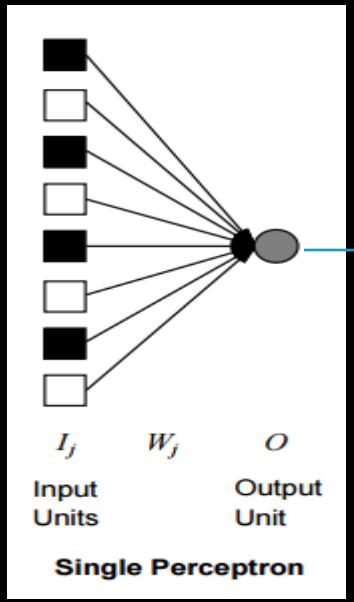
https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15381-f01/www/handouts/110601.pdf

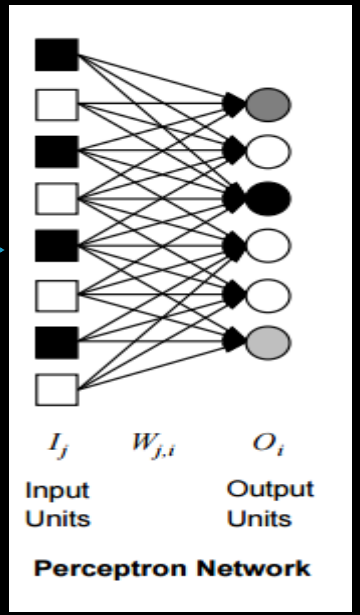
算法在手，层数你有

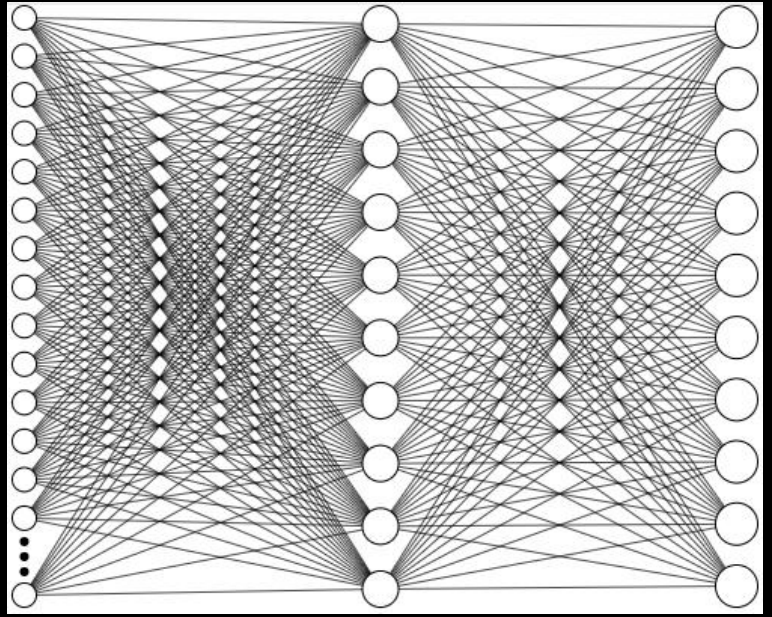


https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15381-f01/www/handouts/110601.pdf

算法在手，层数你有

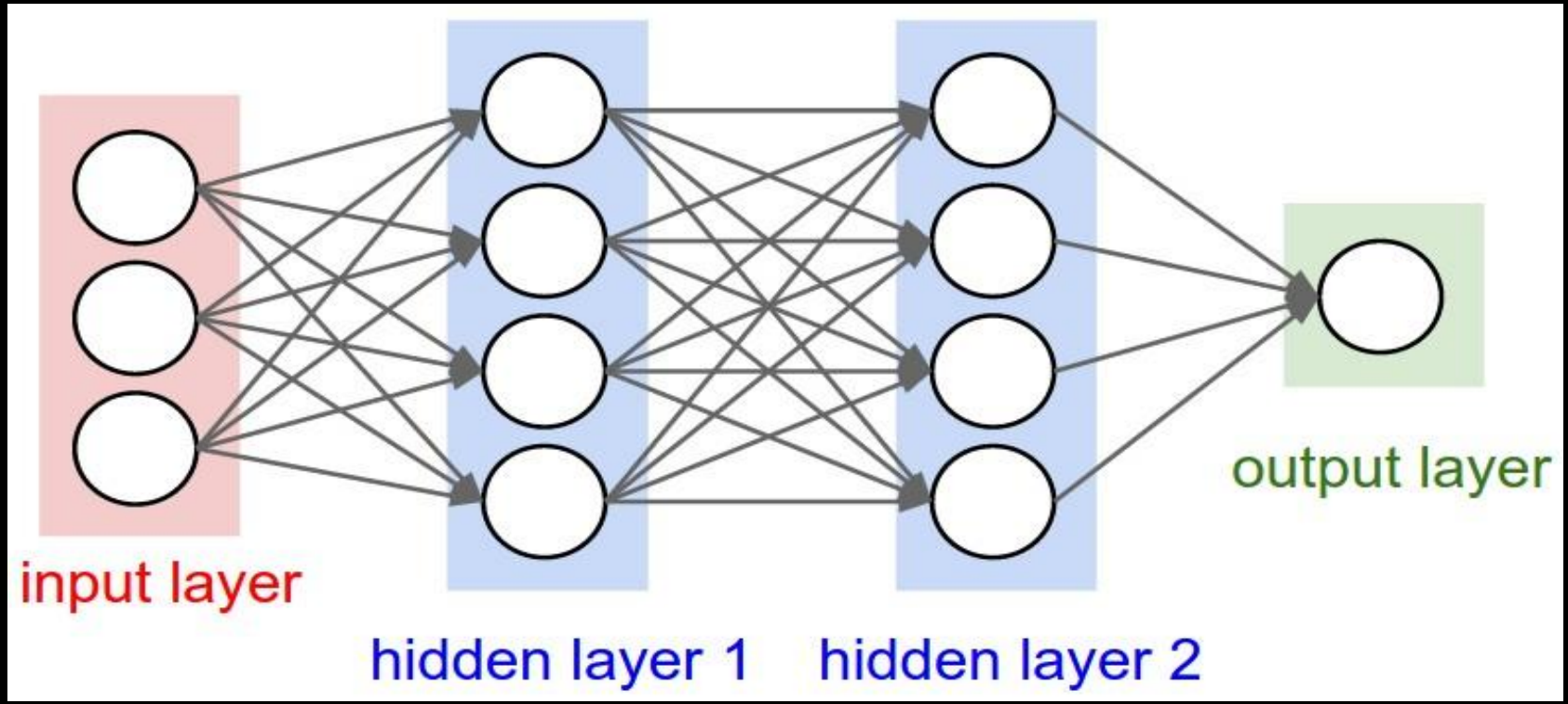






https://en.wikipedia.org/wiki/Artificial_neural_network
https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15381-f01/www/handouts/110601.pdf

算法在手，层数你有



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

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

关于过学习（Overfit）和欠学习（Underfit）

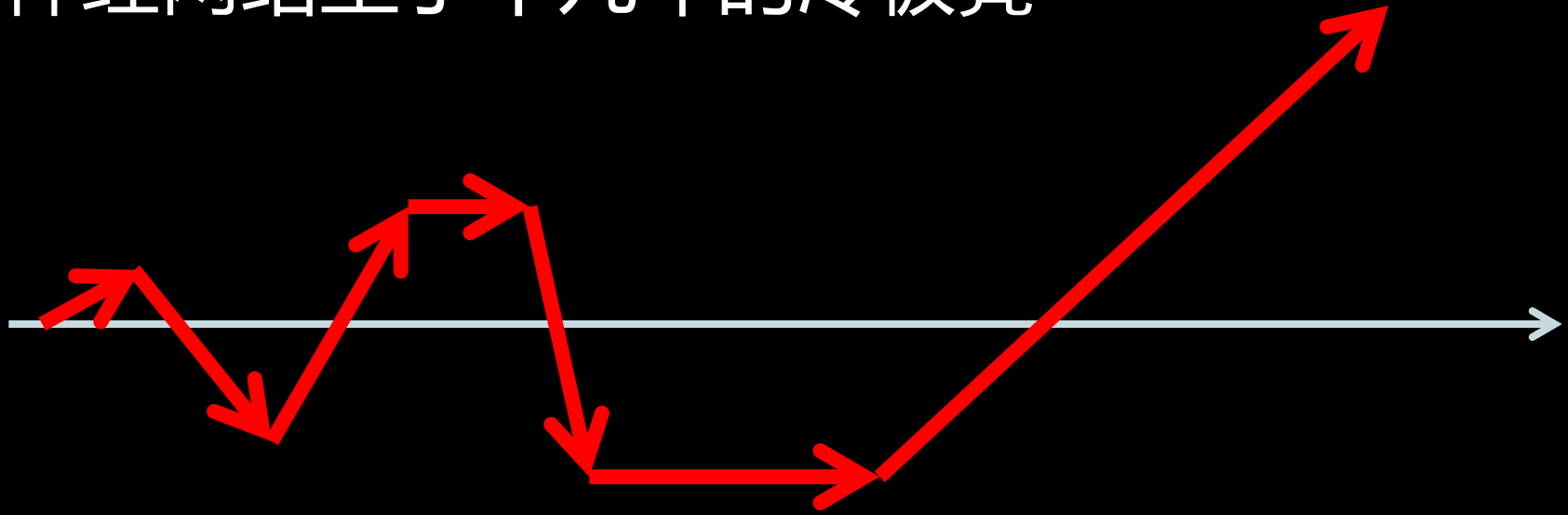
给我四个参数，我能拟合出来一头大象。

给我五个，我就能让大象的鼻子动起来。

—— 冯·诺依曼

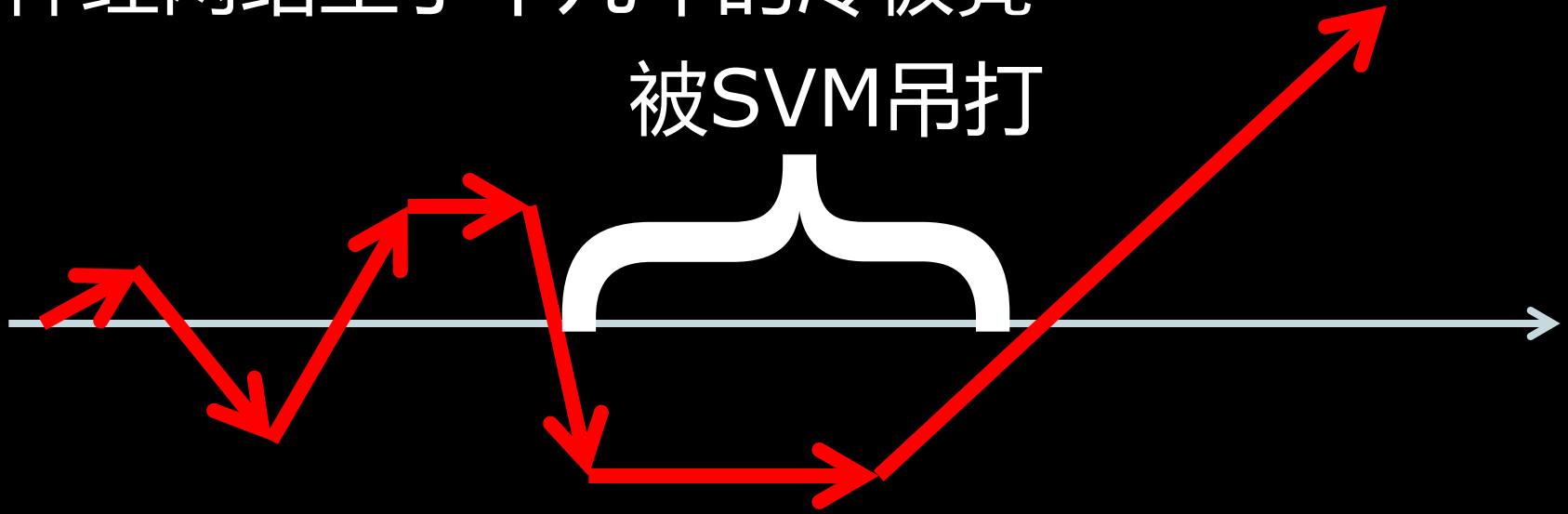
http://blog.sciencenet.cn/blog-3779-803730.html

神经网络坐了十几年的冷板凳

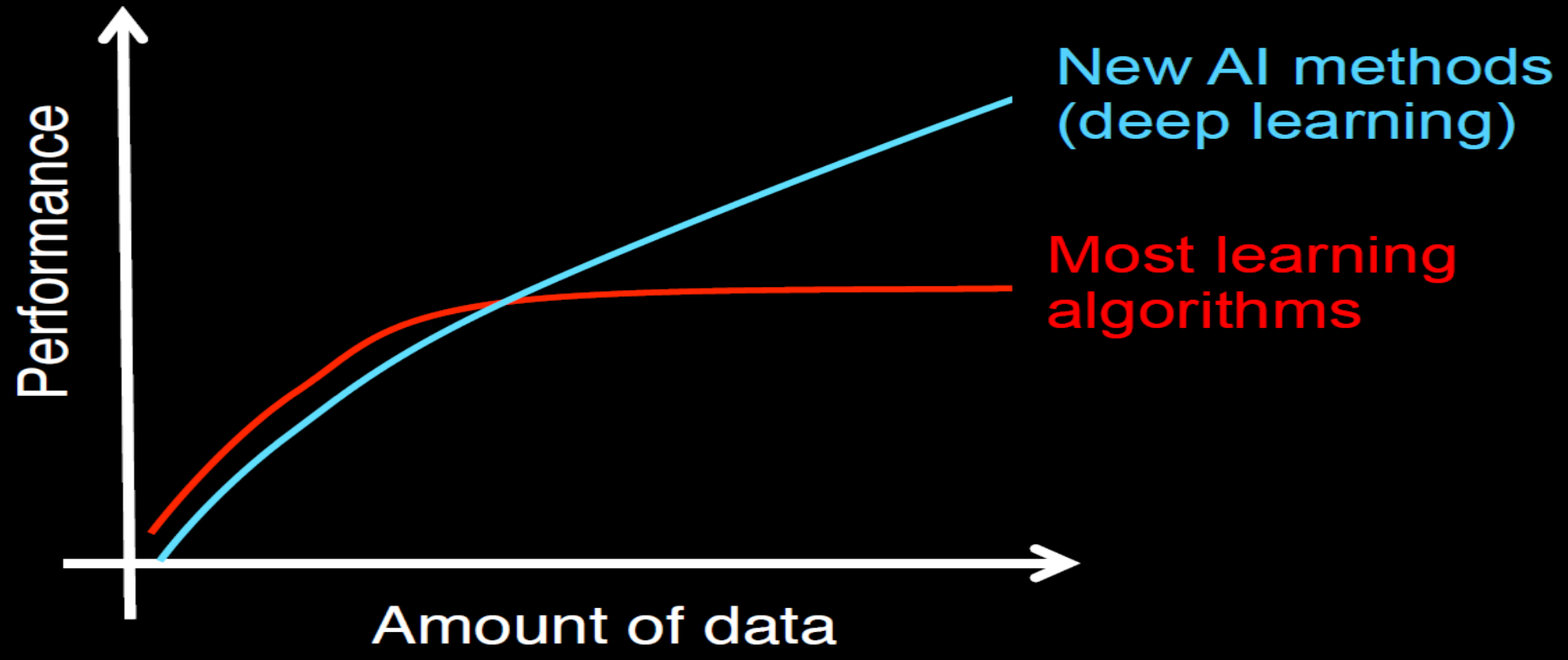


神经网络坐了十几年的冷板凳

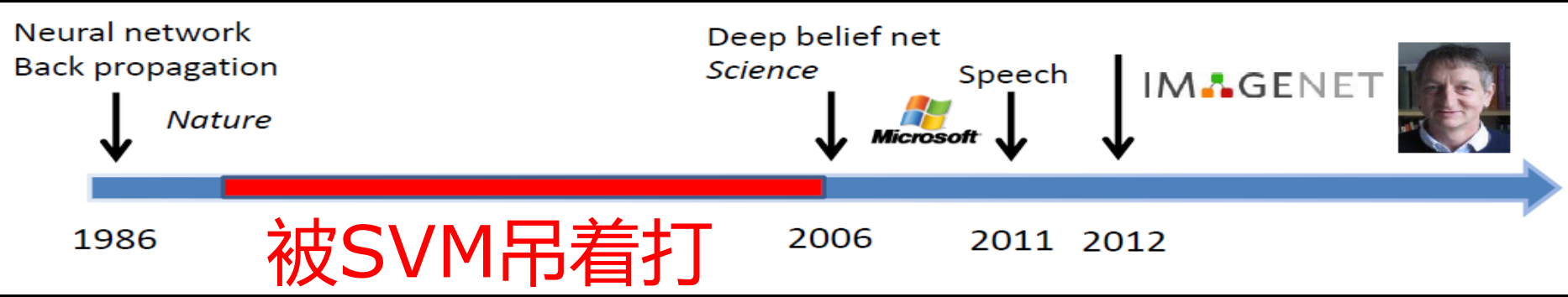
被SVM吊打



Data and machine learning



Andrew Ng



Credit: Xiaogang Wang, The Chinese University of Hong Kong

Deep belief net
Science

Speech

IMAGENET

2006 2011 2012

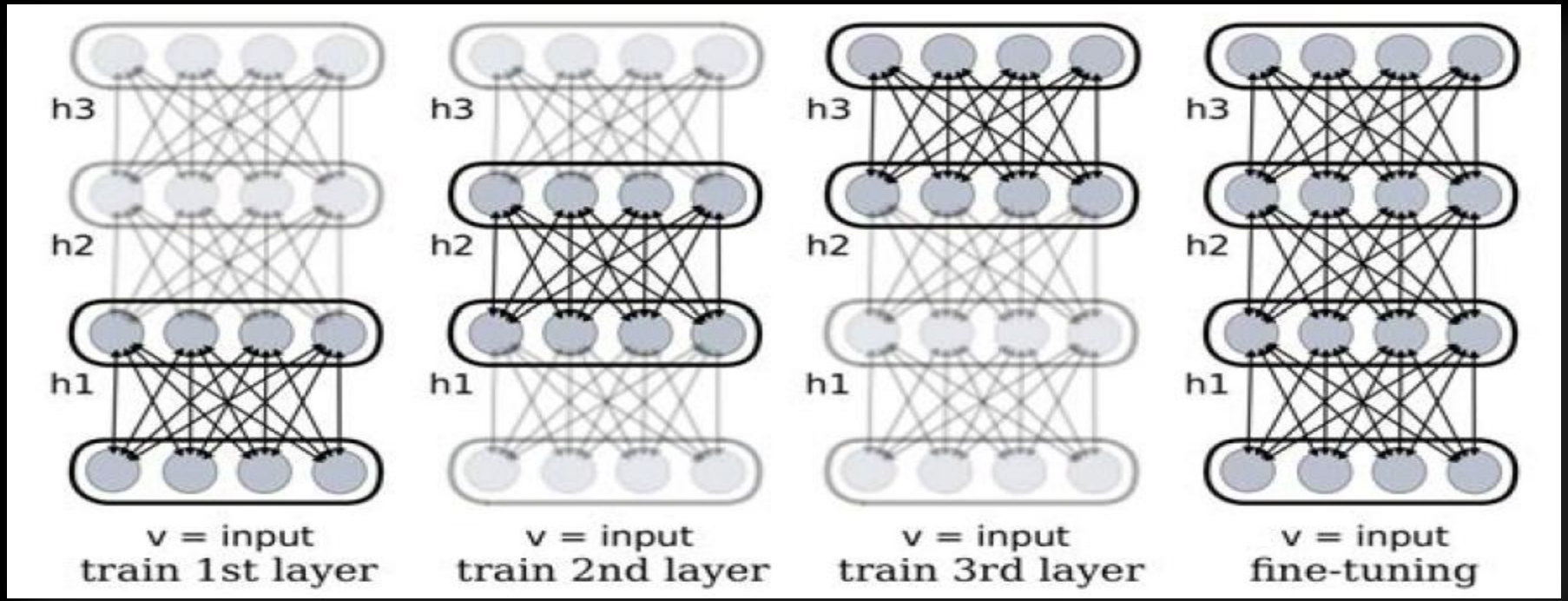
面壁十年图破壁

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.

Neural network
Back propagation

198

Deep belief net
Science

Microsoft

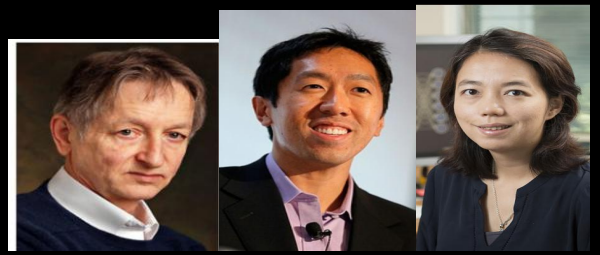
Speech

IMAGENET

2006 2011 2012

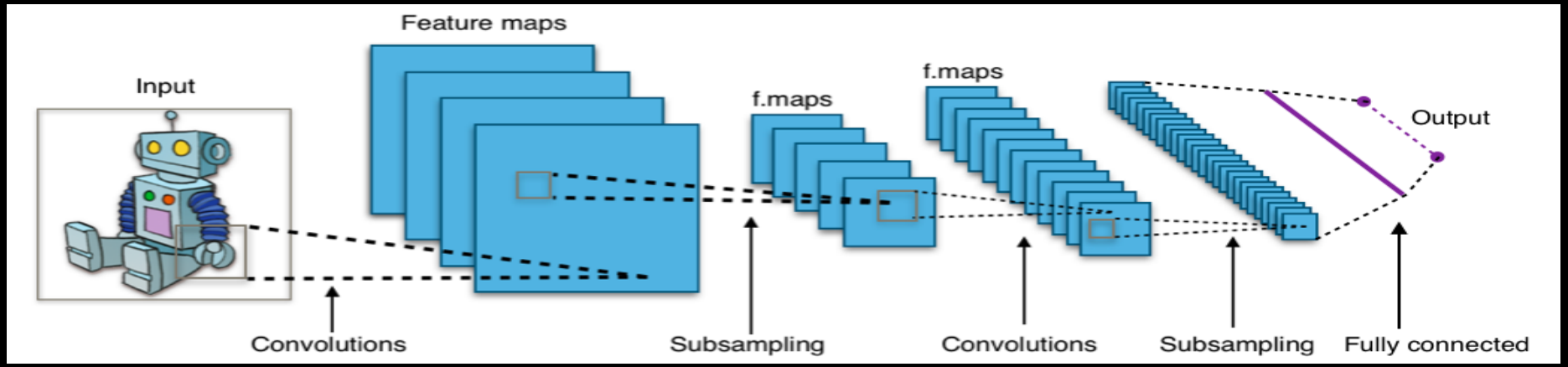
减少训练权重的规模

试试我乐康卷 ☺



Credit: Xiaogang Wang, The Chinese University of Hong Kong

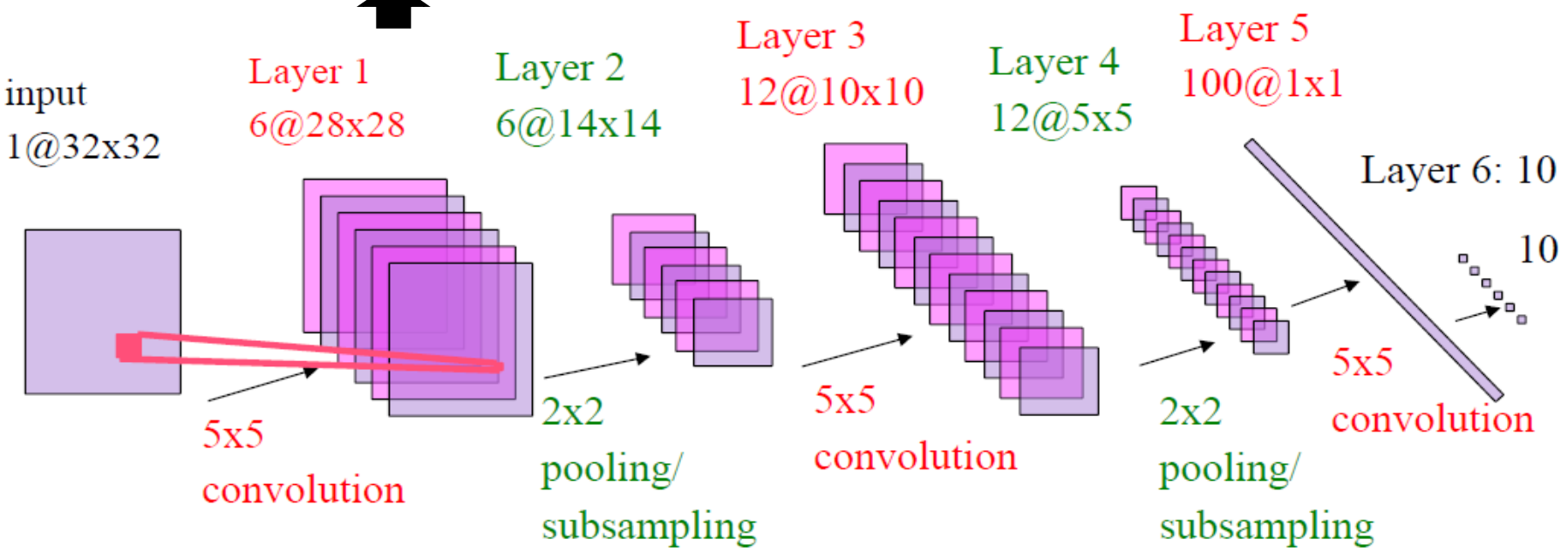
乐康卷：大体架构



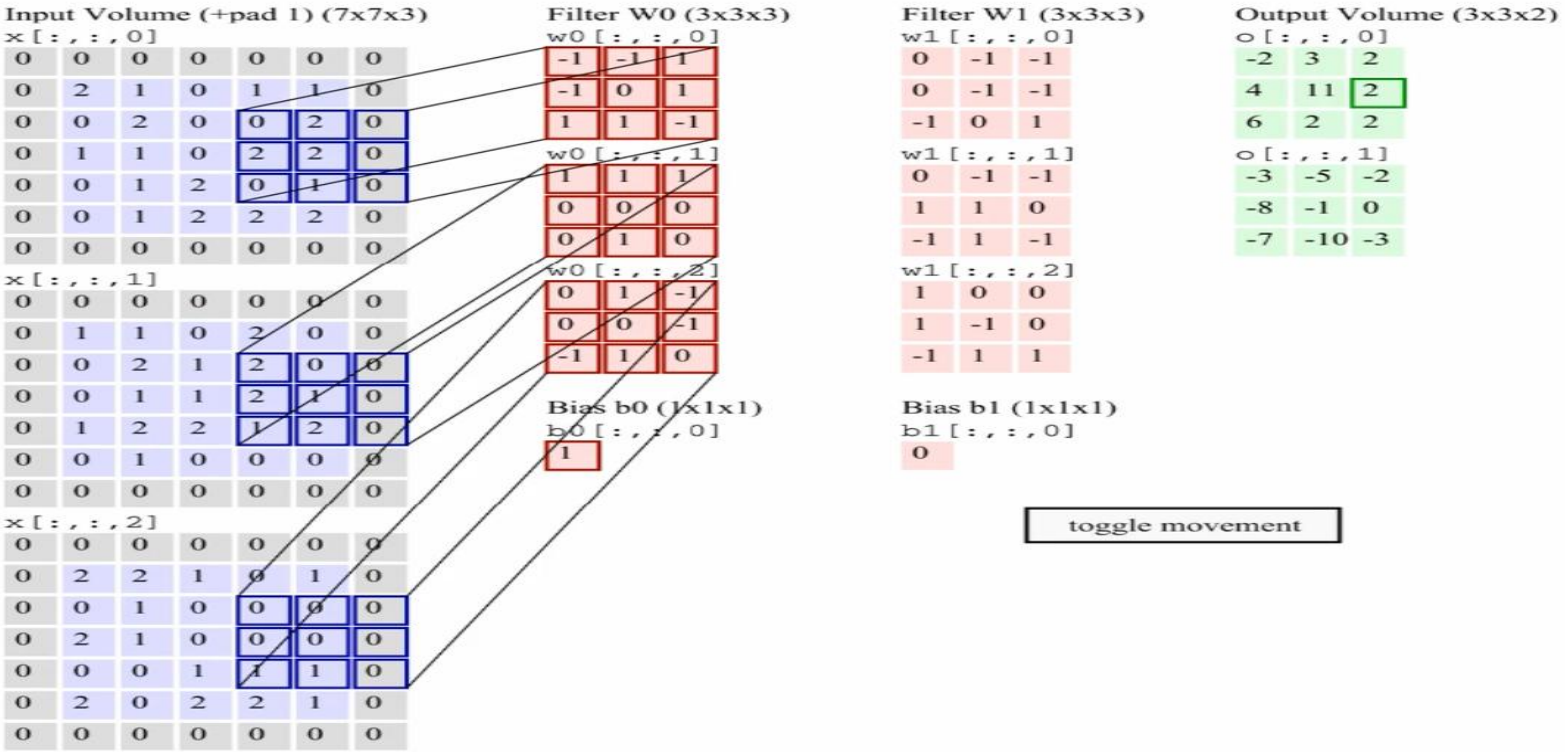
https://en.wikipedia.org/wiki/Convolutional_neural_network#/media/File:Typical_cnn.png

乐康卷：参数配置

LeNet5

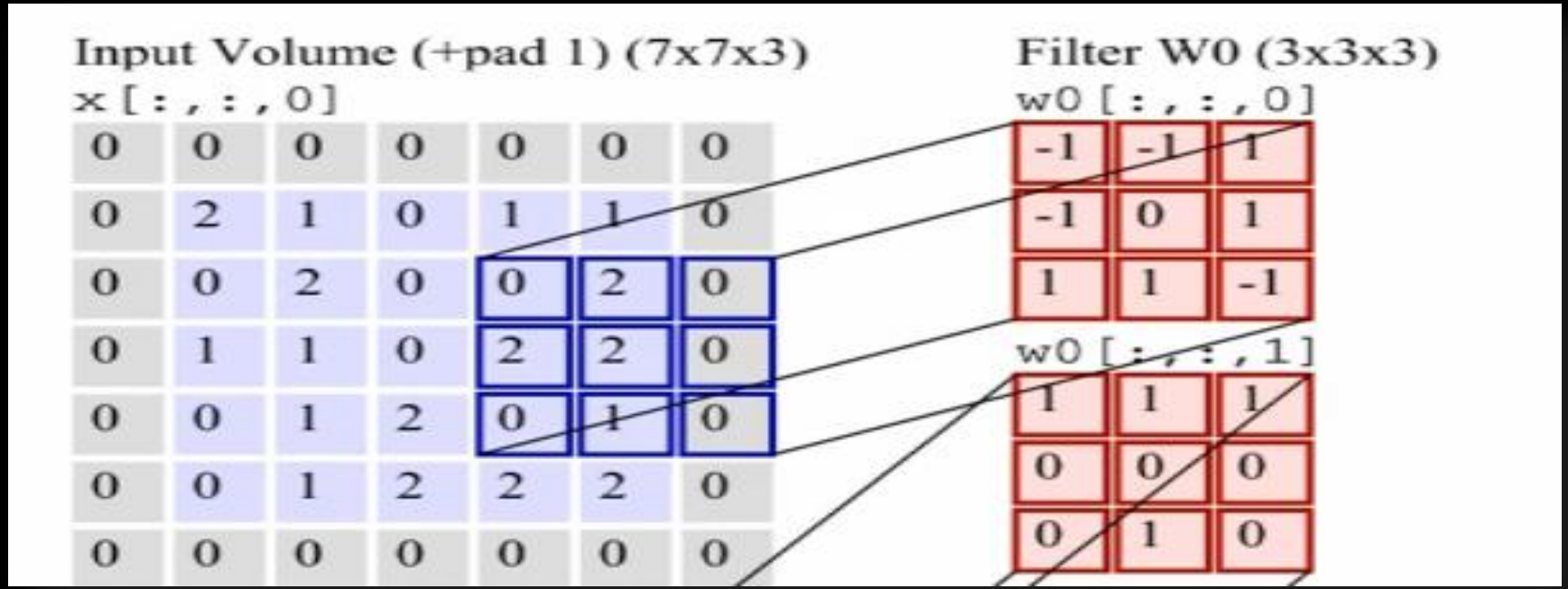


乐康卷：具体实例（来自LeCun大神）



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

乐康卷：具体实例（来自LeCun大神）



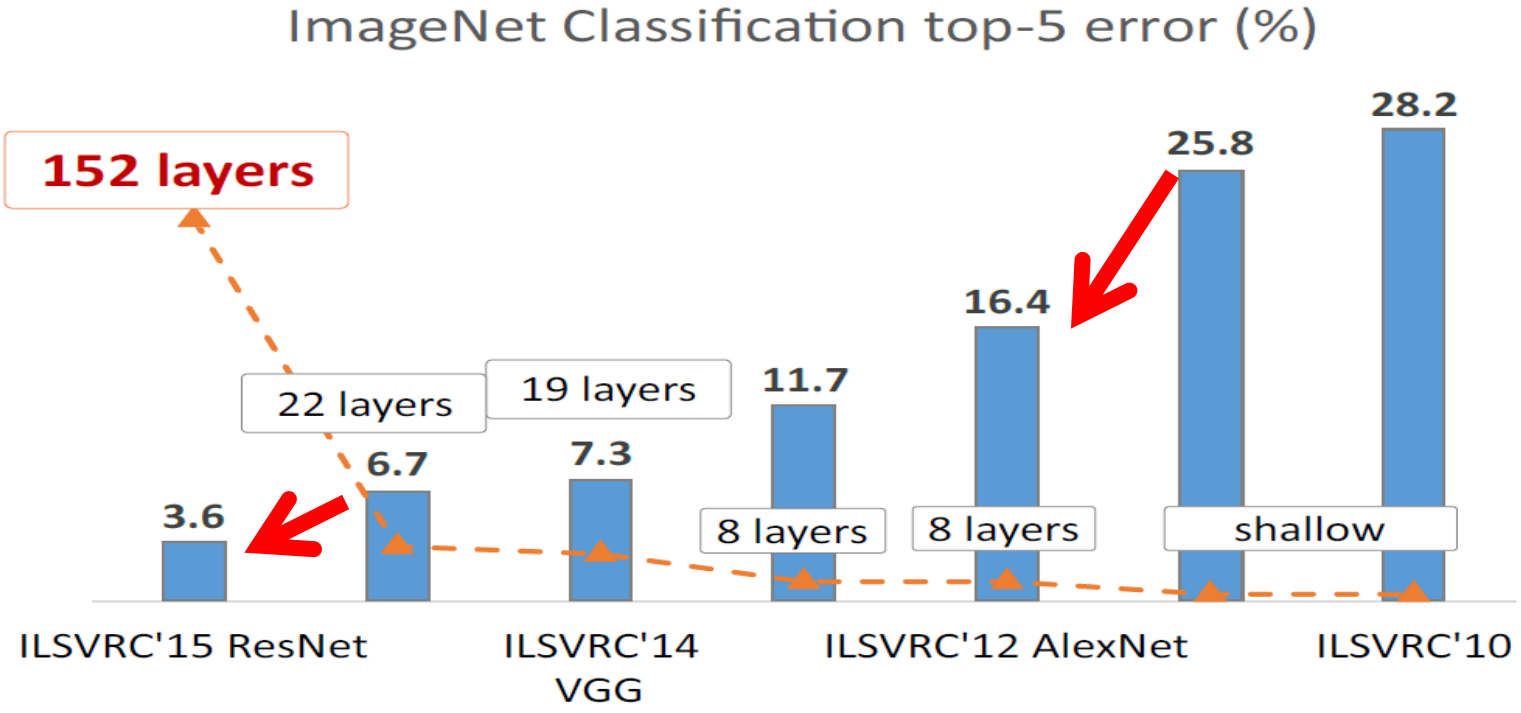
~~不行了：反向传播算法的局限性~~

梯度消失 – 逐层预训练

局部最优 – 多次随机化

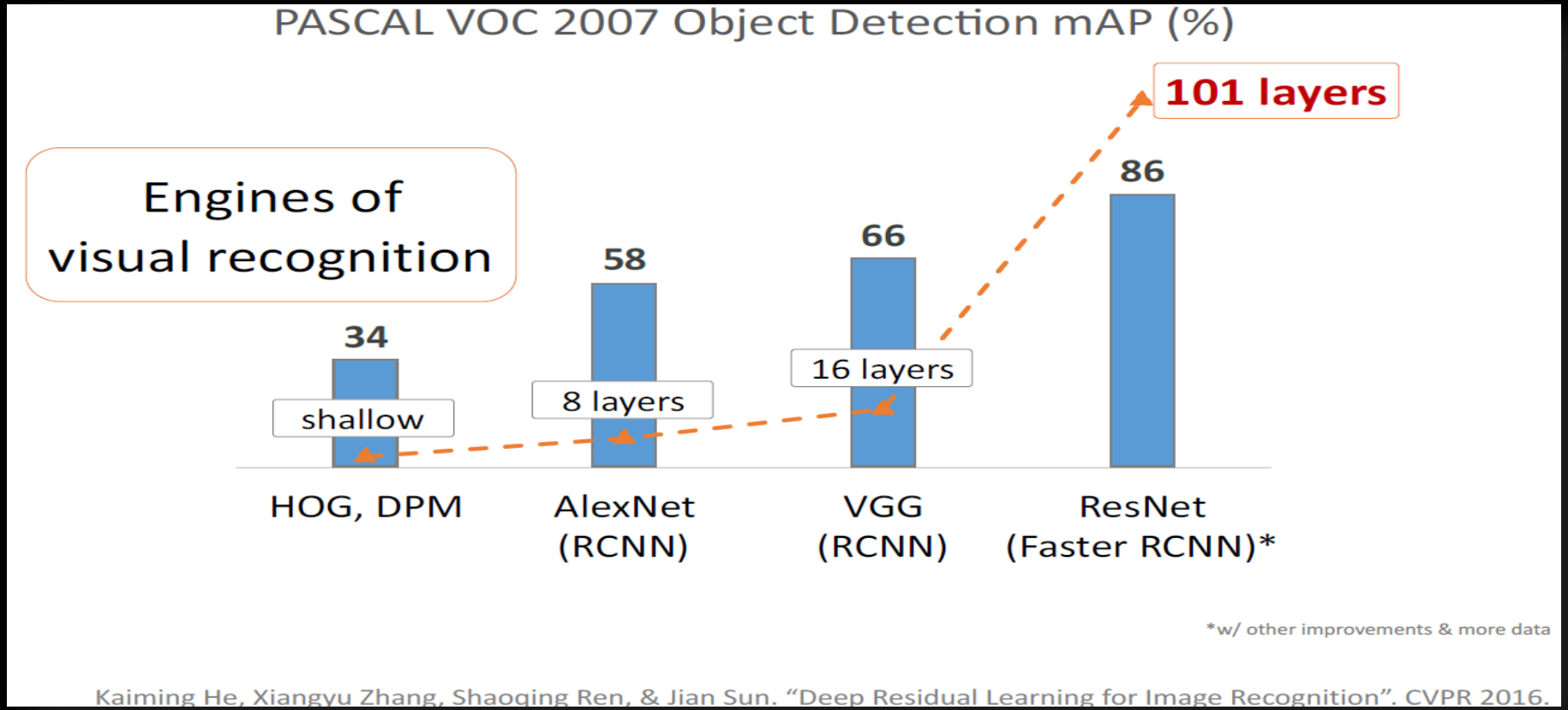
过学习 – 大数据集

算法和模型都完备后，PhD们的心开始躁动起来



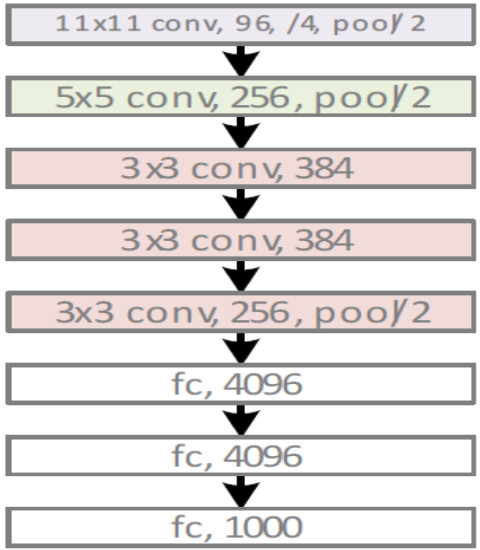
Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.

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)

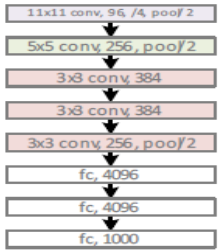


Caffe、mxnet中有现成模型

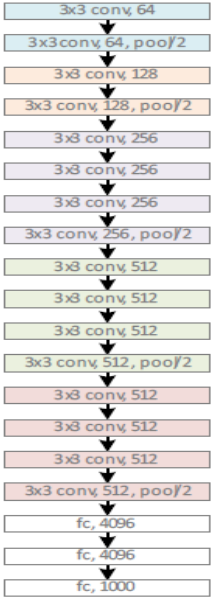
Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. “Deep Residual Learning for Image Recognition”. CVPR 2016.

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

AlexNet, 8 layers (ILSVRC 2012)

VGG, 19 layers (ILSVRC 2014)

ResNet, 152 layers (ILSVRC 2015)

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

1. Introduction
2. Notes
3. Usage

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

故事背景

赵总看到这里很激动：之所以预测不好，

一定是因为层数不够。直接加到两千层试试！

小李：……

小结与思考

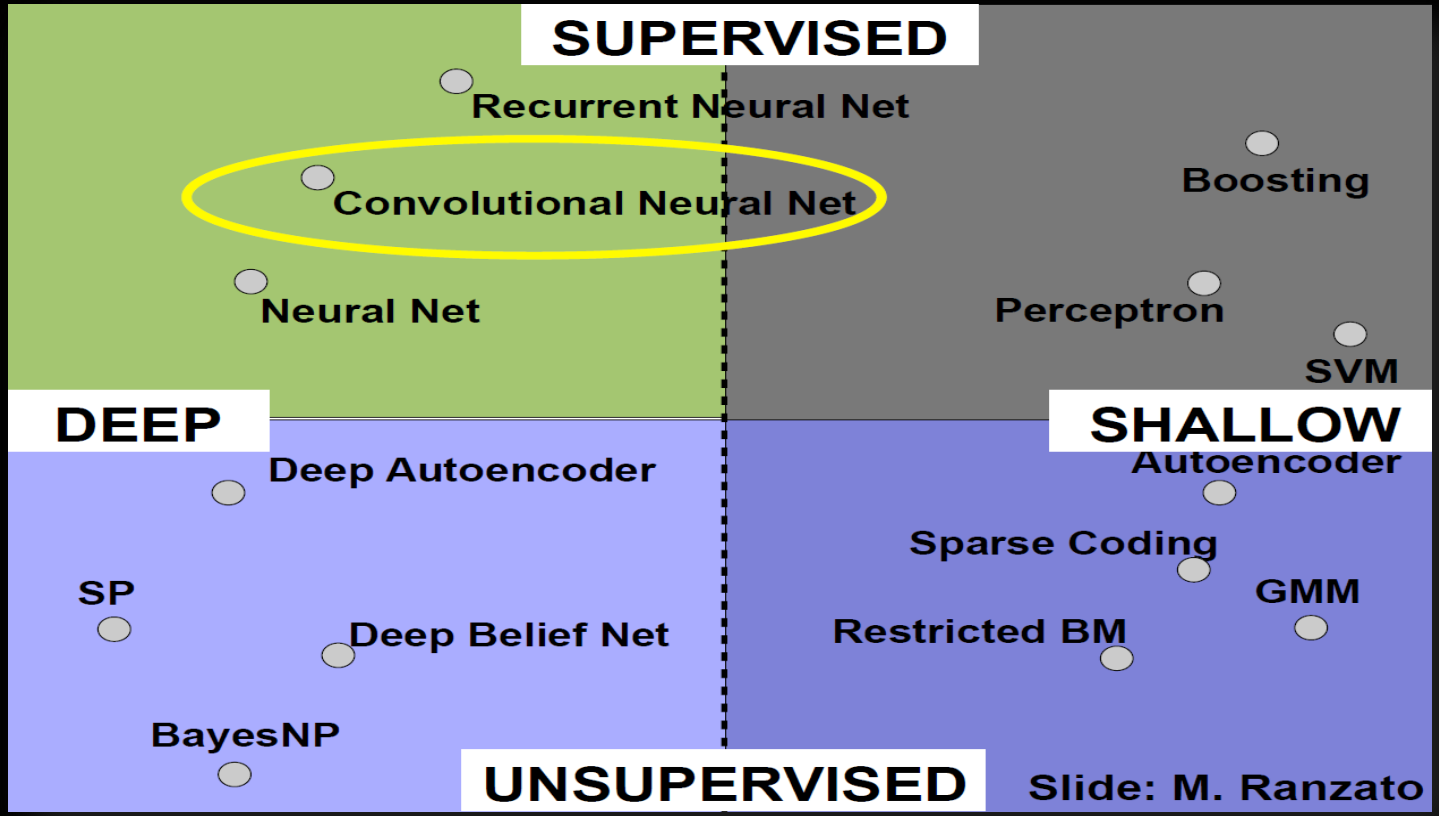
为什么深度学习会在图像、音频、NLP领域成功？

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？

最重要的，如果一开始就针对的错误的问题……

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

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

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

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

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

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

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

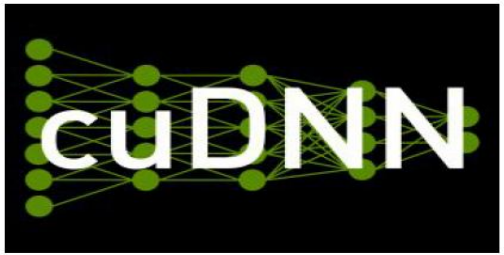
An Introduction to Deep Learning Available from http://www.i6doc.com/en/livre/?GCOI=28001100817300.

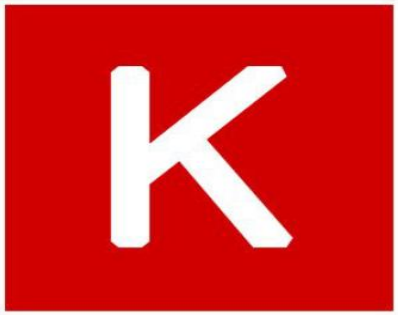
想要学习深度学习，如何上手?

在2016年上手学深度学习是非常容易的

Deep-Learning Package Zoo

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



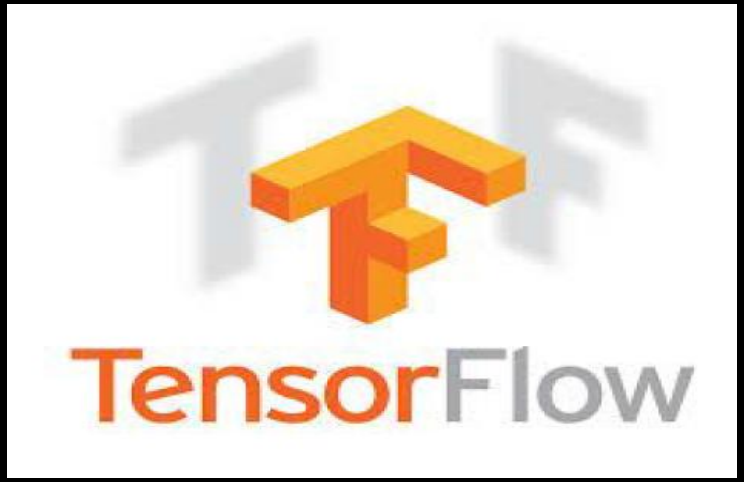


入门阶段已经不需要自己写算法，写模型即可

Deep-Learning Package Design Choices

- Model specification: **Configuration file** (e.g. Caffe, DistBelief, CNTK) versus **programmatic generation** (e.g. Torch, Theano, Tensorflow)
- For programmatic models, choice of high-level language: Lua (Torch) vs. Python (Theano, Tensorflow) vs others.
- We chose to work with **python** because of rich community and library infrastructure.

选取目前最常用的两个包进行介绍

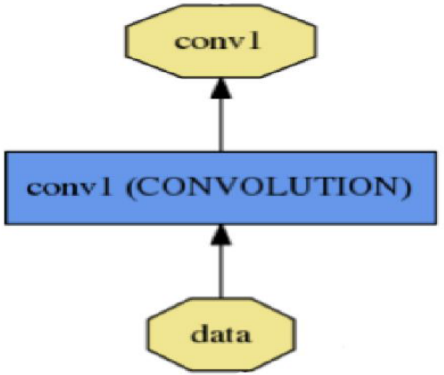


A Caffe Layer

```
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



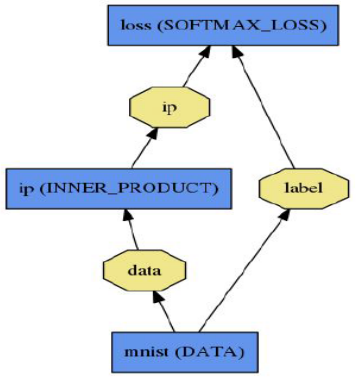
*Link to the Google Protobuffer Documentation

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

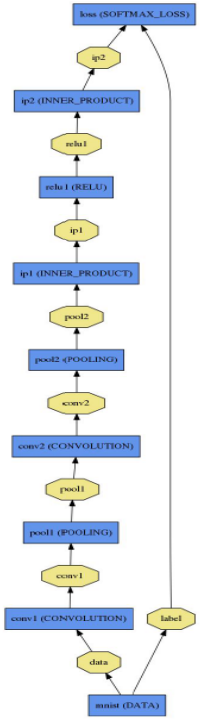
A Caffe Network

- 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.

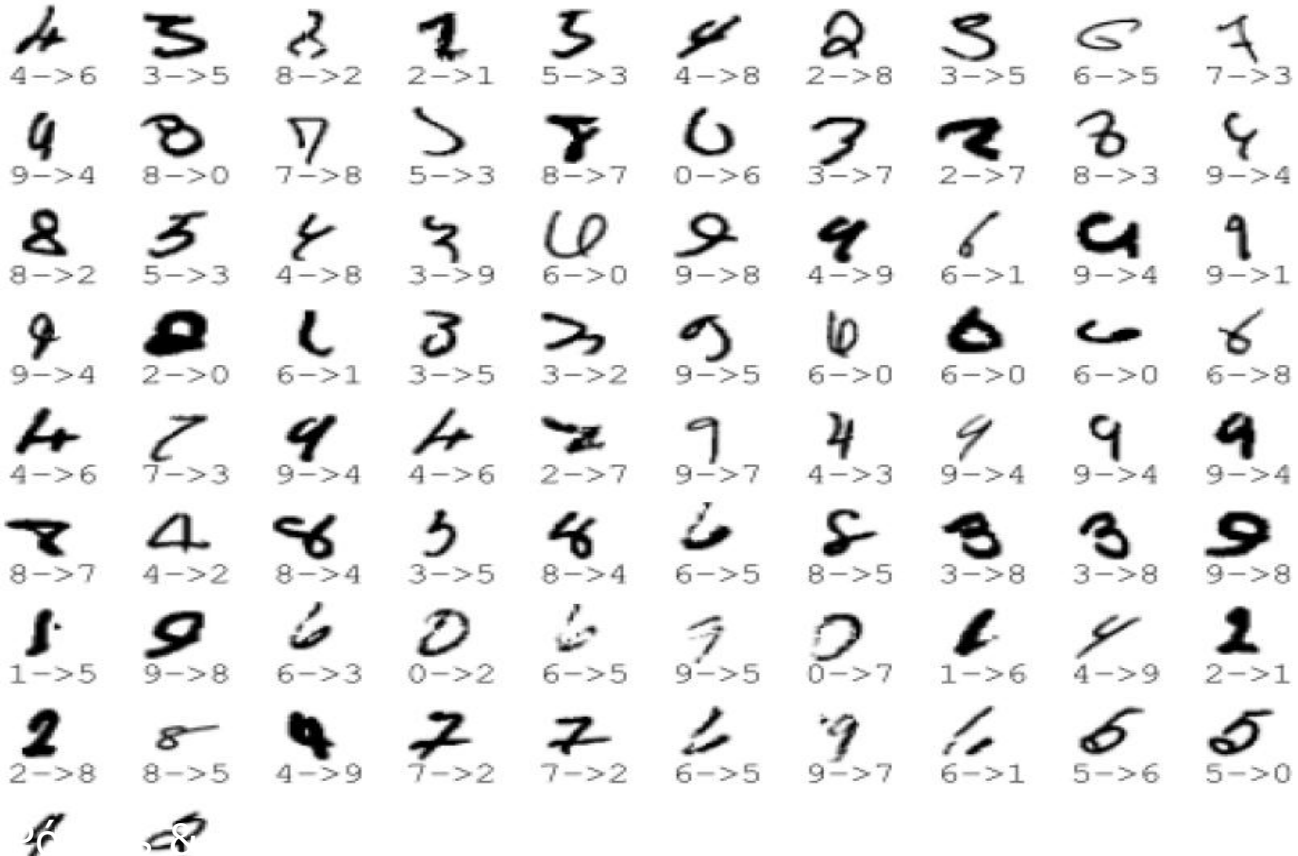
It is a good database for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on preprocessing and formatting.

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/

Misclassified examples



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:

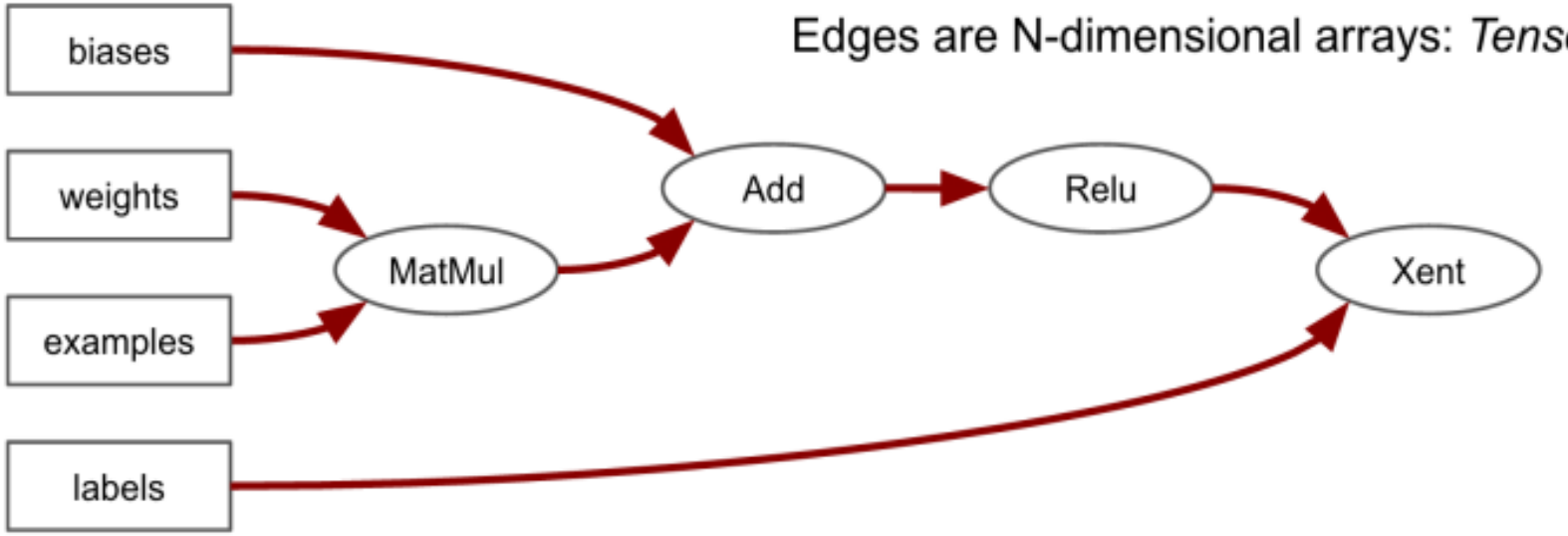
- A standard format for packaging Caffe model info.
- Tools to upload/download model info to/from Github Gists, and to download trained `.caffemodel` binaries.
- A central wiki page for sharing model info Gists.

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:

TensorFlow：搞清楚Tensor和Flow就行了

TensorFlow is an open source software library for numerical computation using data flow graphs.

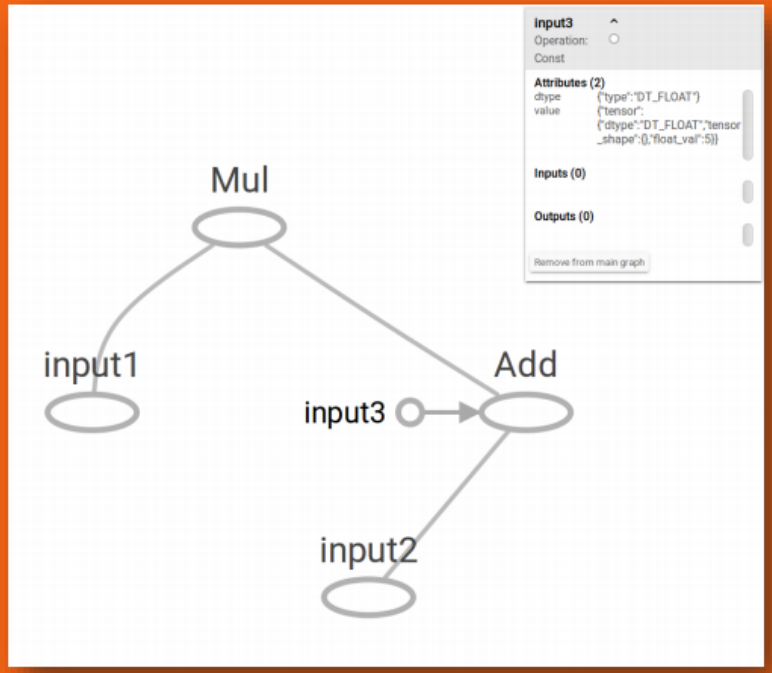


https://indico.cern.ch/event/433556/contributions/1930588/attachments/1231746/1806025/tensorflow_introduction.pdf

HELLO WORLD…

$$mul = i_1 * (i_2 + i_3)$$

```
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

Version: r0.11

Basic Neural Networks

MNIST For ML Beginners

About this tutorial

The MNIST Data

Softmax Regressions

Implementing the Regression

Training

Evaluating Our Model

Deep MNIST for Experts

About this tutorial

Setup

Load MNIST Data

Start TensorFlow InteractiveSession

Build a Softmax Regression Model

Placeholders

Variables

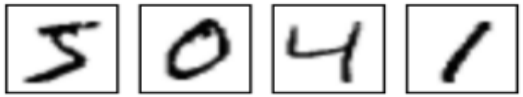
Predicted Class and Loss Function

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.

深度学习怎么玩，取决于你多有钱

GPU Acceleration

`-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

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

基础 | 前沿 | 应用 | 评价

前沿 | NPU | GAN | 自编程

语言处理 | 语音识别 | 图像识别 | 视频理解

自然语言处理 | TL;DR

词性标注 | 组块识别 | 命名实体识别 | 语法解析 | 语义角色标注

乔姆斯基 | n-gram

自然语言处理 | RNN

自然语言处理 | LSTM, Bi-LSTM

自然语言处理 | Memory Network

自然语言处理 | DNC(?)

自然语言处理 | 接近并超越传统方法

Task		Benchmark	SENNA
Part of Speech (POS)	(Accuracy)	97.24 %	97.29 %
Chunking (CHUNK)	(F1)	94.29 %	94.32 %
Named Entity Recognition (NER)	(F1)	89.31 %	89.59 %
Parse Tree level 0 (PT0)	(F1)	91.94 %	92.25 %
Semantic Role Labeling (SRL)	(F1)	77.92 %	75.49 %

机器自然语言翻译 | 大幅度改进

语音识别 | 热火朝天

语音识别 | 百度,微软,讯飞,谷歌,苹果

语音识别 | GMM-HMM (李开复)

语音识别 | Warp-CTC (百度)

Connectionist Temporal Classification

语音识别 | FSMN, DFCNN (讯飞)

前馈型序列记忆网络 | 深度全序列卷积神经网络

语音识别 | ResNet+Σ（微软）

NIST 2000 Switchboard WER 5.9%

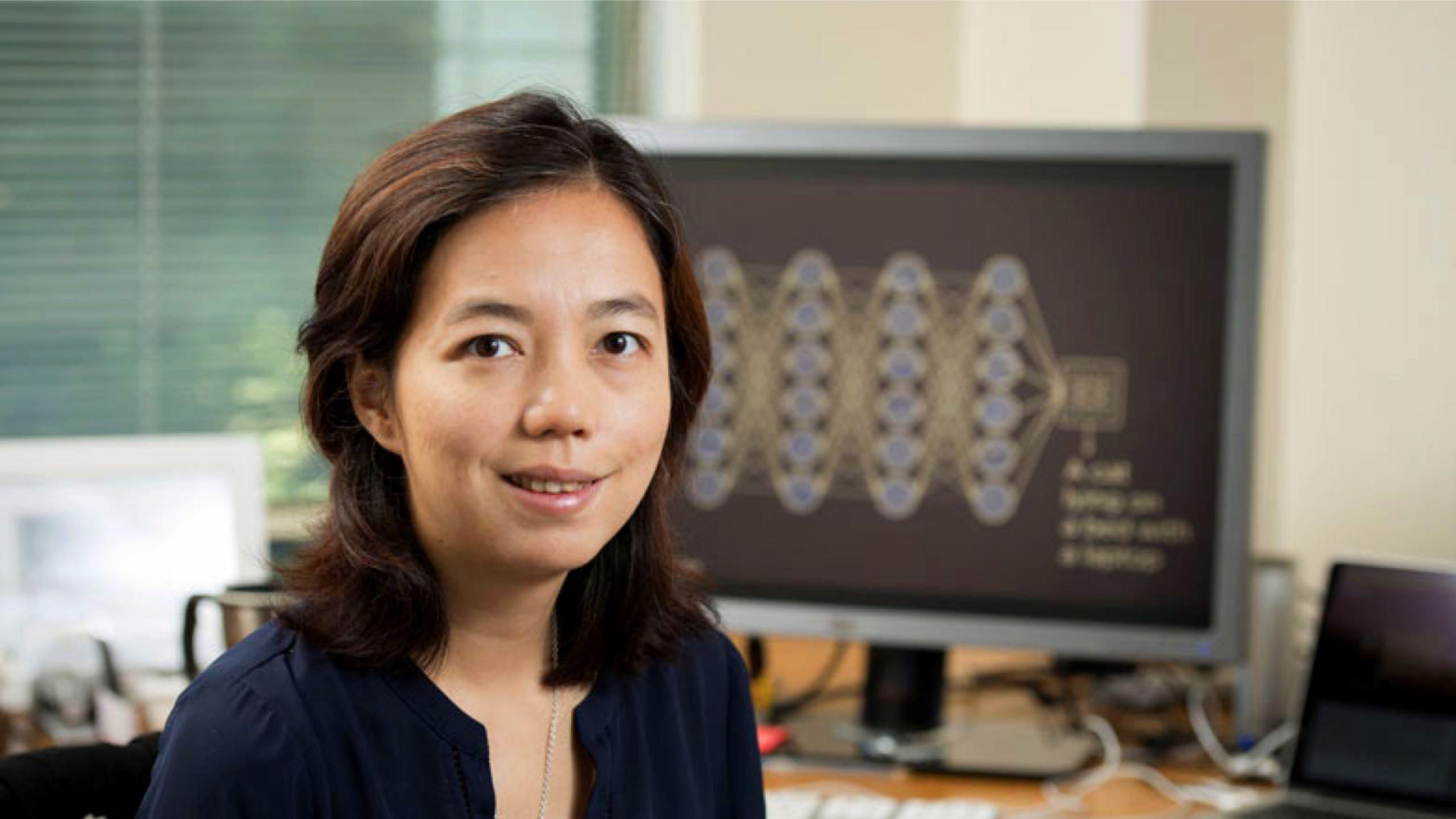
TTS | WaveNet (谷歌)

图像识别 | 成果最突出显著

图像识别 | MNIST

图像识别 | IMAGENET

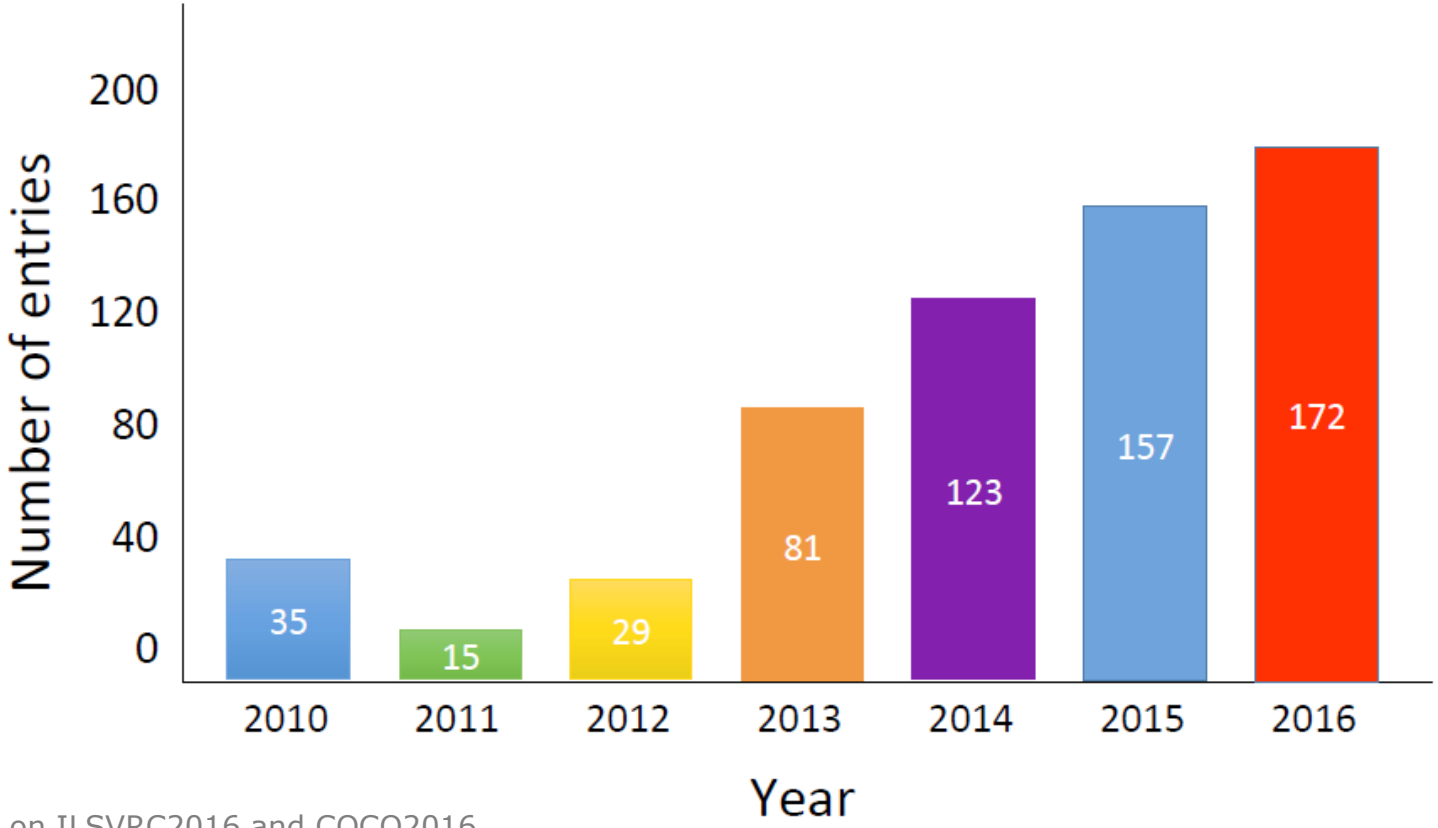
WordNet | 21K | 14M | 1M



图像识别 | ILSVRC

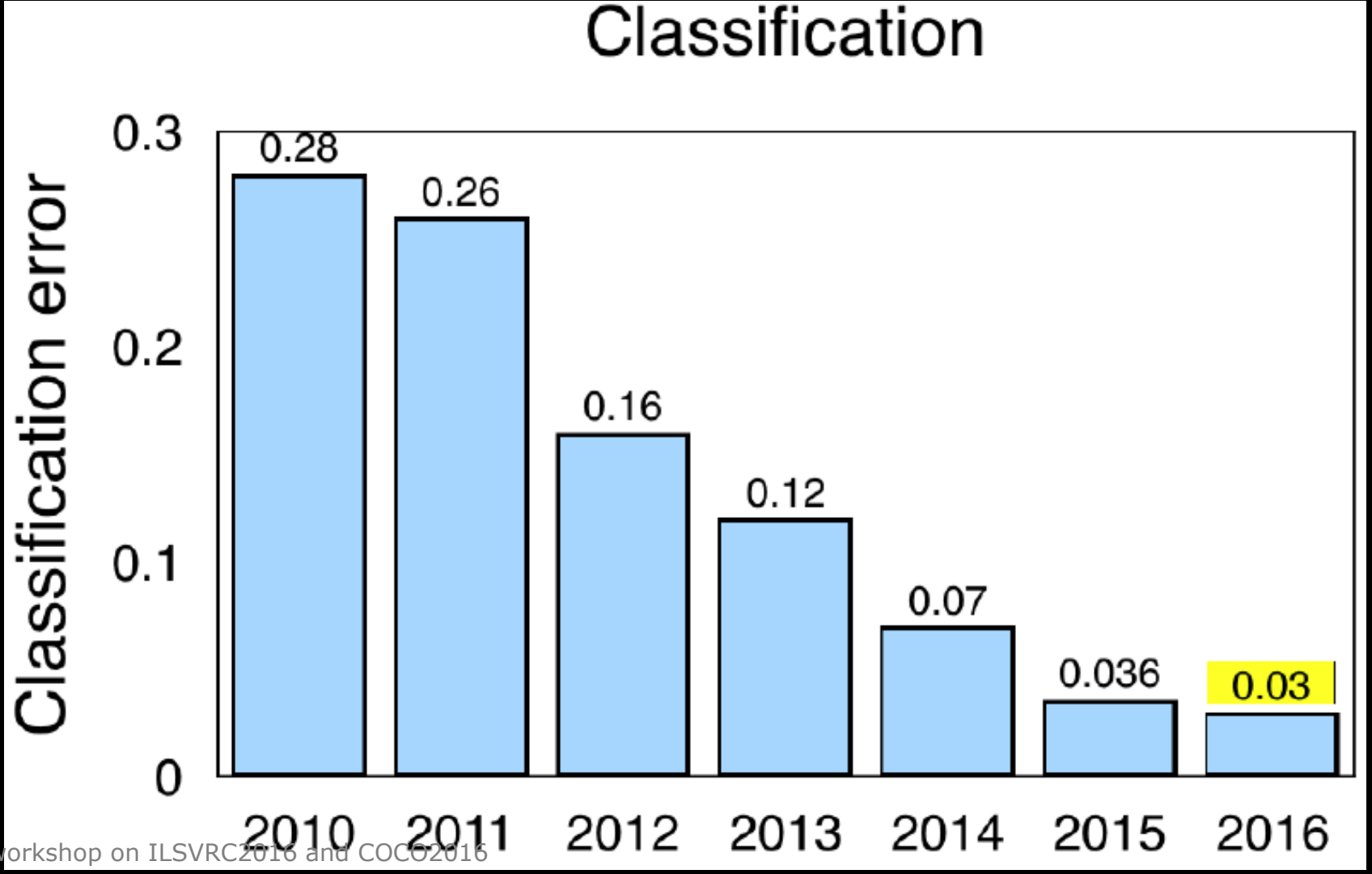
始于2010, 里程碑

Participation in ILSVRC over the years



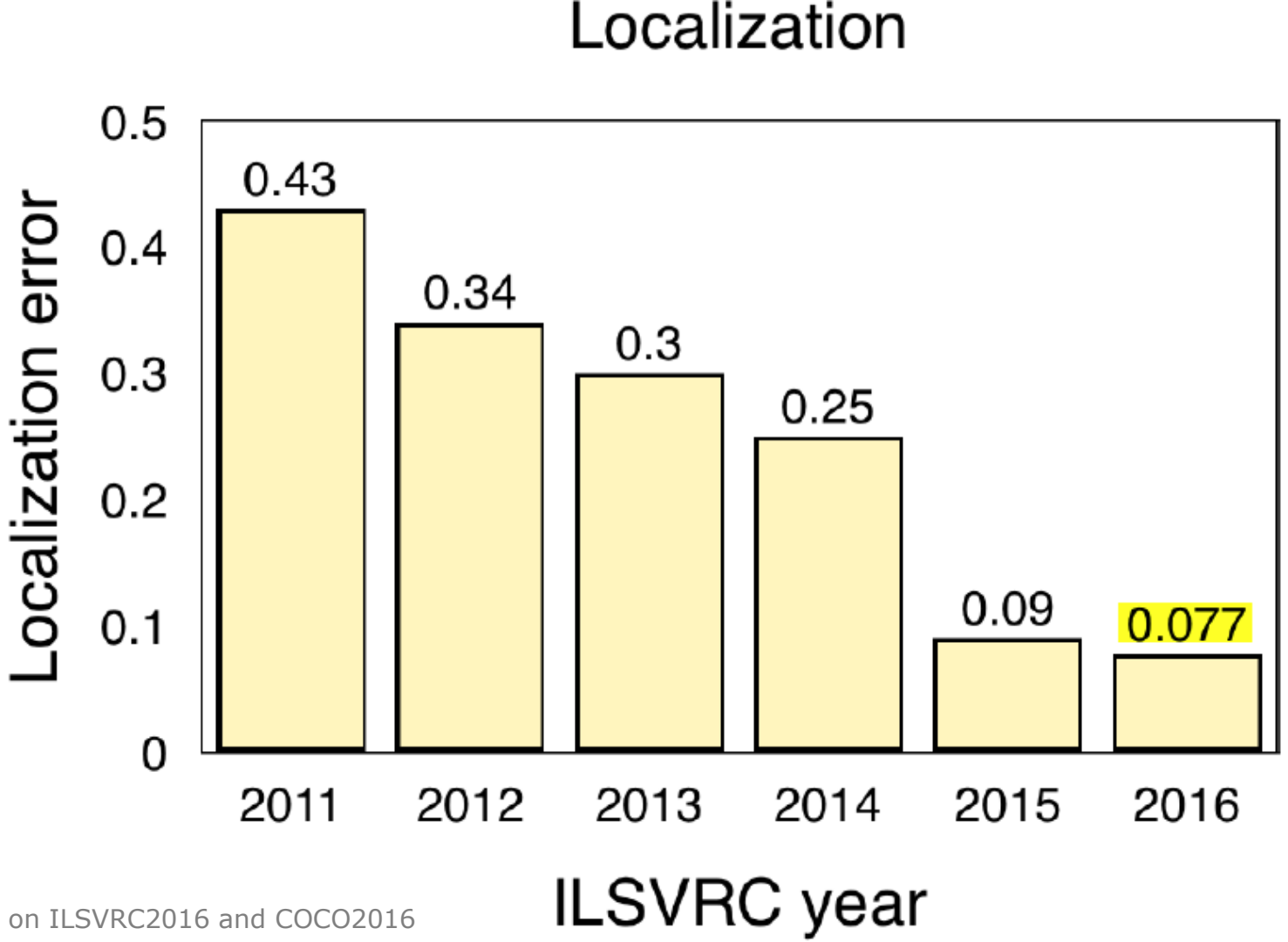
Credit: workshop on ILSVRC2016 and COCO2016

redit: workshop on ILSVRC2016 and COCO2016

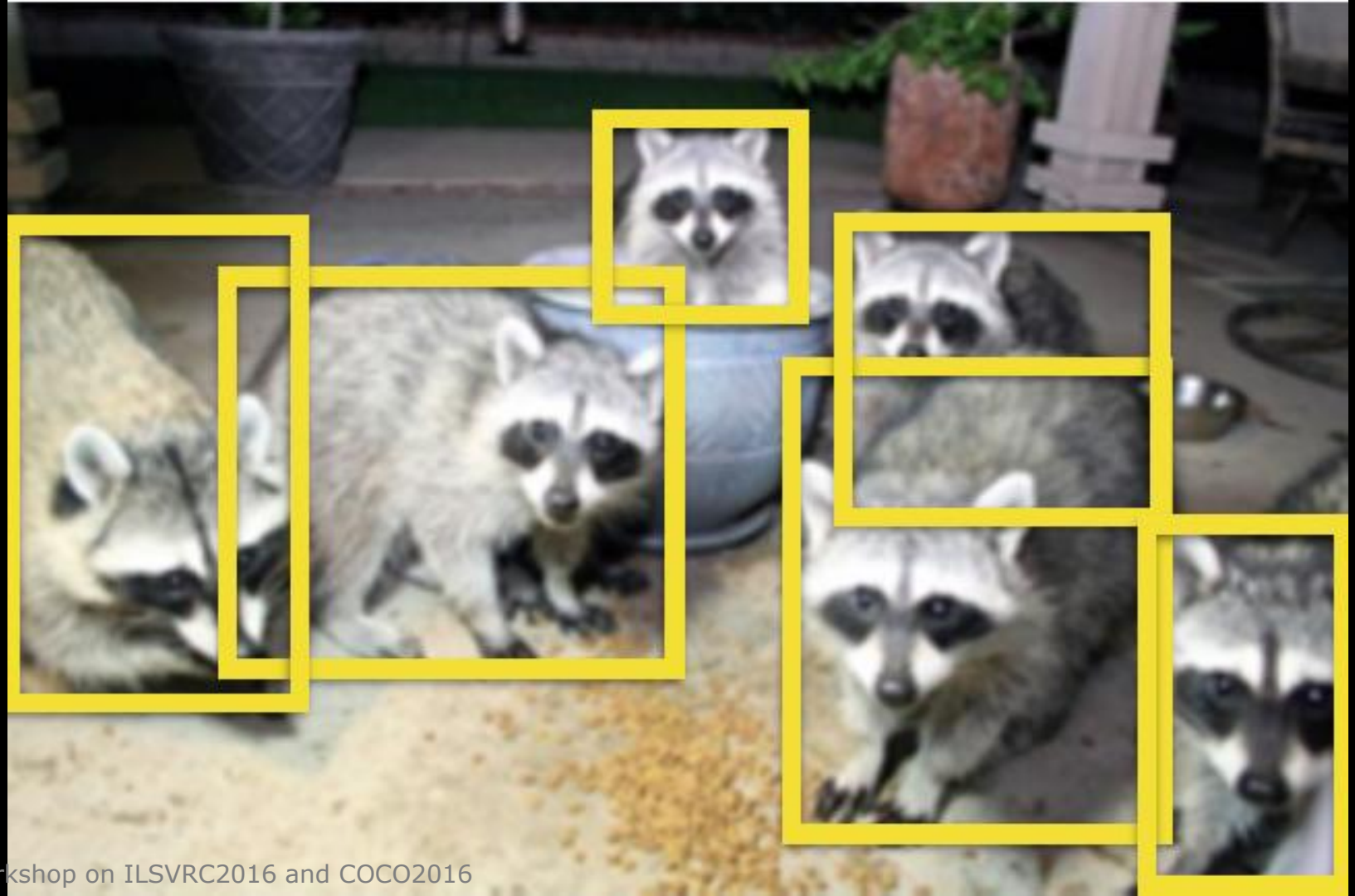


redit: workshop on ILSVRC2016 and COCO2016

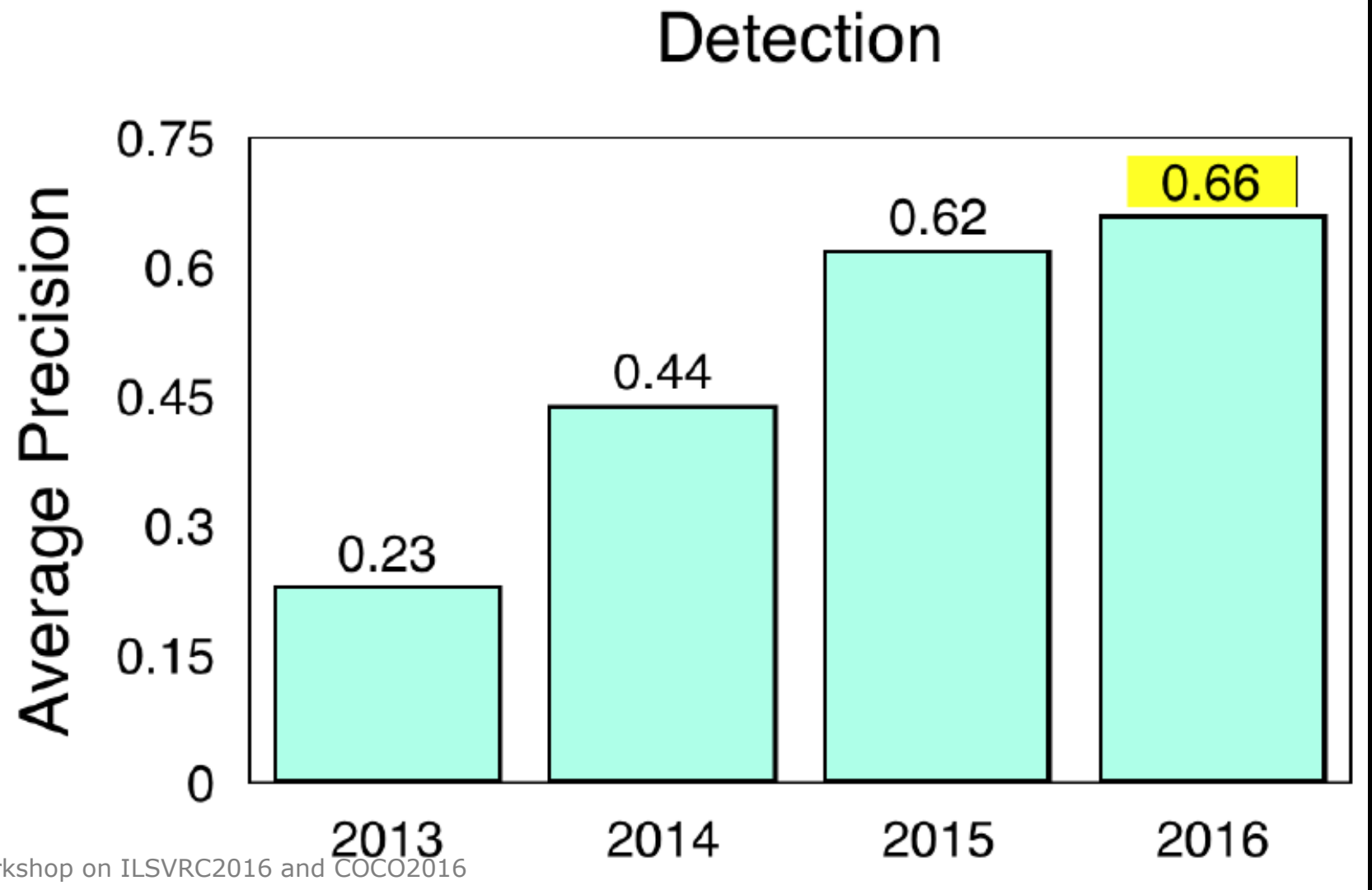
redit: workshop on ILSVRC2016 and COCO2016



redit: workshop on ILSVRC2016 and COCO2016



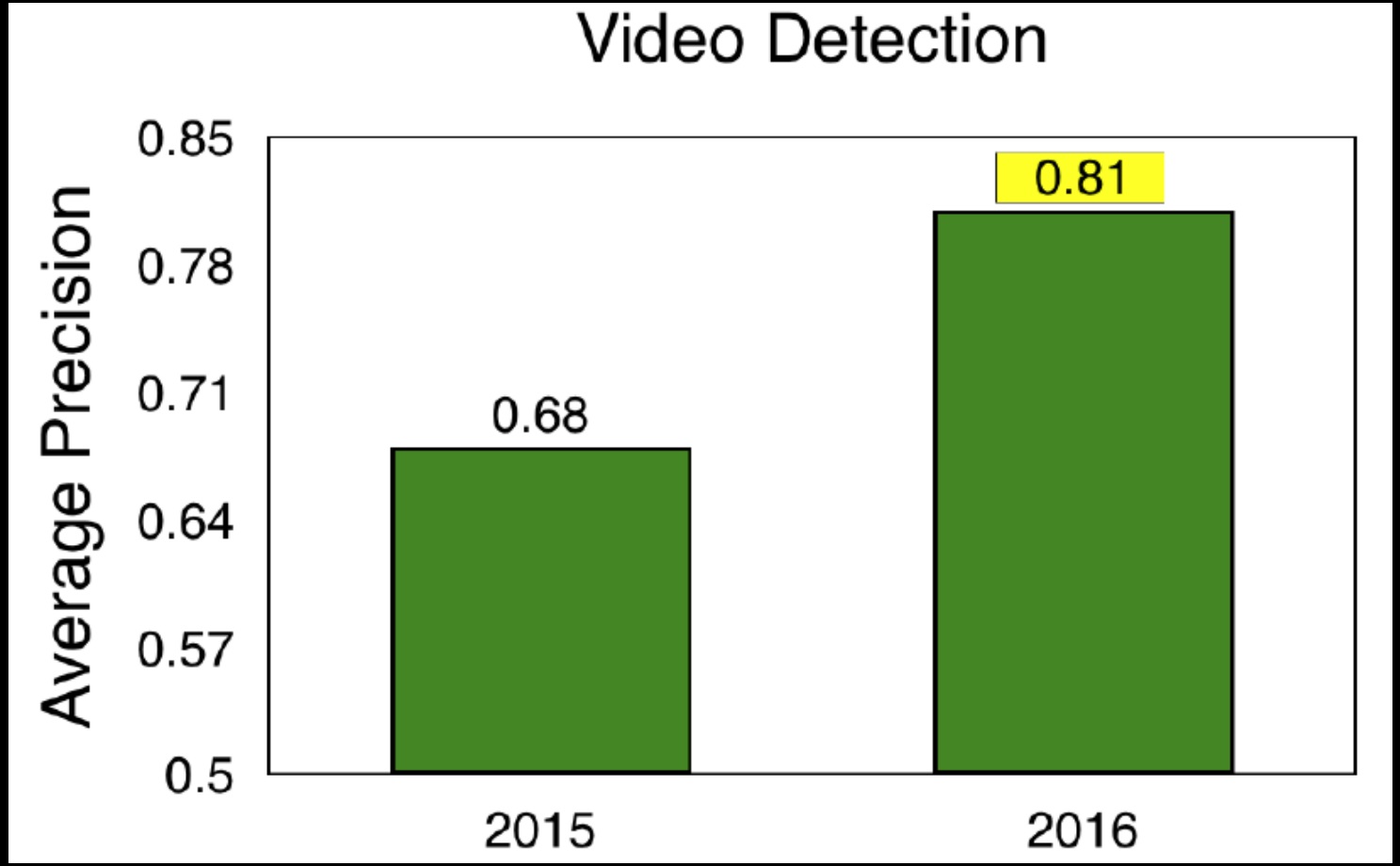
redit: workshop on ILSVRC2016 and COCO2016



redit: workshop on ILSVRC2016 and COCO2016

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

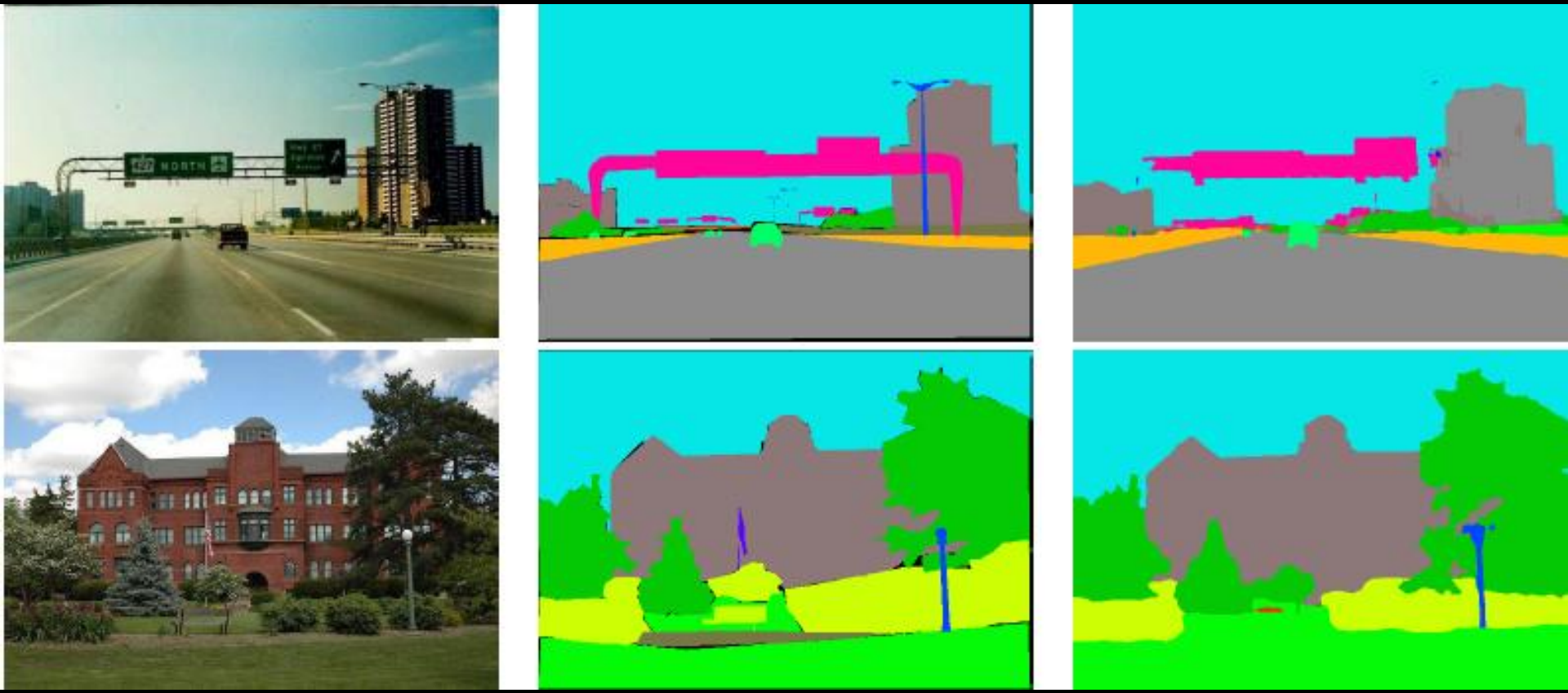
redit: workshop on ILSVRC2016 and COCO2016



redit: workshop on ILSVRC2016 and COCO2016

图像识别 | Beyond ImageNet

Sense Classification & Parsing



Credit: HikVision on ILSVRC2016

Image Captioning

A person riding a motorcycle on a dirt road.

Two dogs play in the grass.

Credit: workshop on ILSVRC2016 and COCO2016

人脸检测 | 进入混战模式

视频分析丨门槛在慢慢降低

视频预测 | 效果还比较诡异

基础 | 前沿 | 应用 | 评价

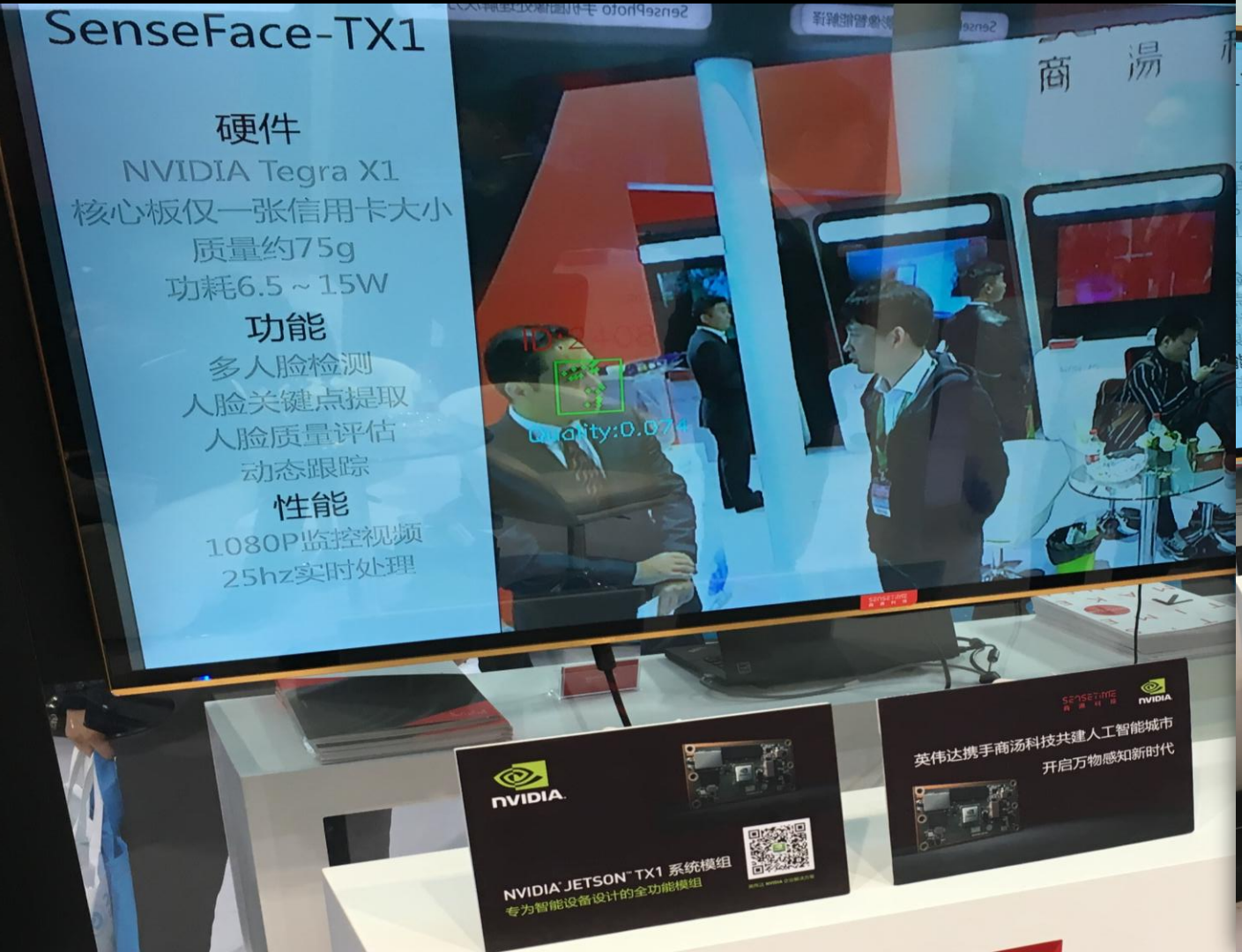

中文版 | English | Русский
OK
SECURITY CHINA
2016 Security China
中国国际社会公共安全产品博览会
2016 China International Exhibition on Public Safety and Security
服务经济民生
展示安防技术
构建智慧城市
推进平安中国
网站首页
关于展会
展会活动
展会新闻
参展服务
观众服务
交通及住宿
下载中心
联系我们
主办单位：中国安全防范产品行业协会
展览日期：2016年10月25-28日
展览地点：中国国际展览中心(新馆


旷视智能安防解决方案
Face++ 旷视


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
HUMAN SEGMENTATION


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
展会2号相机 2016-10
姓名 王晶
性别 男
安博展员工库
梁博
男
安博展员工库


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


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


E1025 11:21:40.573937 16821 senseface_app.cpp:164]
E1025 11:21:40.582726 16821 senseface_app.cpp:164]
E1025 11:21:40.590593 16821 senseface_app.cpp:164]
E1025 11:21:40.590679 16821 senseface_app.cpp:164]
E1025 11:21:40.699939 16821 senseface_app.cpp:164]
E1025 11:21:40.707291 16821 senseface_app.cpp:164]
E1025 11:21:40.720096 16821 senseface_app.cpp:164]
E1025 11:21:40.728065 16821 senseface_app.cpp:164]
E1025 11:21:40.736455 16821 senseface_app.cpp:164]
E1025 11:21:40.743432 16821 senseface_app.cpp:164]
E1025 11:21:41.093533 16821 senseface_app.cpp:164]
E1025 11:21:41.093899 16821 senseface_app.cpp:164]
E1025 11:21:41.101660 16821 senseface_app.cpp:164]
E1025 11:21:41.108548 16821 senseface_app.cpp:164]
E1025 11:21:41.162178 16821 senseface_app.cpp:164]
E1025 11:21:41.171079 16821 senseface_app.cpp:164]
E1025 11:21:41.177114 16821 senseface_app.cpp:164]
E1025 11:21:41.188246 16821 senseface_app.cpp:164]
E1025 11:21:41.195377 16821 senseface_app.cpp:164]
E1025 11:21:41.201452 16821 senseface_app.cpp:164]
E1025 11:21:41.208427 16821 senseface_app.cpp:164]
OpenCV Error: Assertion failed (0 <= roi.x && 0 <= roi.width && roi.x + roi.width <= m.cols && 0 <= roi.y && 0 <= roi.height && roi.y + roi.height <= m.rows) in
les/core/src/matrix.cpp, line 323
terminate called after throwing an instance of 'cv::Exception'
what(): /home/
m.rows in function Mat
./cs_senseface.sh: line 2: 16755 Aborted (core dumped) ./SenseFace
sensetime@sensetime-Super-Server:~/Desktop$ ./cs_senseface.sh
release version
------query db task: i=0, taskName=人脸分析系统1, vsN.ame=展会1号相机, type=1, path=rtsp://admin:admin12345@192.10.0.200, stat
------query db task: i=1, taskName=人脸分析系统2, vsN.ame=展会2号相机, type=1, path=rtsp://admin:admin12345@192.10.0.201, stat
------query db task: i=2, taskName=人脸分析系统3, vsN.ame=展会3号相机, type=1, path=rtsp://admin:admin12345@192.10.0.202, stat
------query db task: i=3, taskName=人脸分析系统4, vsN.ame=展会4号相机, type=1, path=rtsp://admin:admin12345@192.10.0.203, stat
libpng warning: iCCP: known incorrect sRGB profile
libpng warning: iCCP: known incorrect sRGB profile
libpng warning: iCCP: known incorrect sRGB profile
libpng warning: iCCP: known incorrect sRGB profile

基础 | 前沿 | 应用 | 评价

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 | 自动编程，呵呵哒

评价丨警惕伪科学和各类歧视

评价 | 新交互开启新攻击面

谢谢各位抽出时间来一起讨论

有问题可以提问交流

深度学习：基础，前沿，应用，评价

吴 伟
lazyparser@gmail.com
中国科学院软件研究所