电子科技大学

University of Electronic Science and Technology of China

Start from Star

Wei Han

Data Mining Lab,
Big Data Research Center, UESTC
Email：huangchen.uestc@gmail.com

Outline

1. Brain Storm
    - 1.1. Determinism
    - 1.2. Intelligence and the dominant
    - 1.3. Related neuroscience
2. Introduction to Deep Neural Network
    - 2.1. Feedforward Neural Network
        - 2.1.1. Multi Layer Perceptron
        - 2.1.1. Convolution Neural Network
    - 2.2. Feedback Neural Network
        - 2.2.1. Recurrent Neural Networks
        - 2.2.2. Long-Short Term dependencies (LSTM)
3. Illuminating Works
4. Recent Work

All story begins:

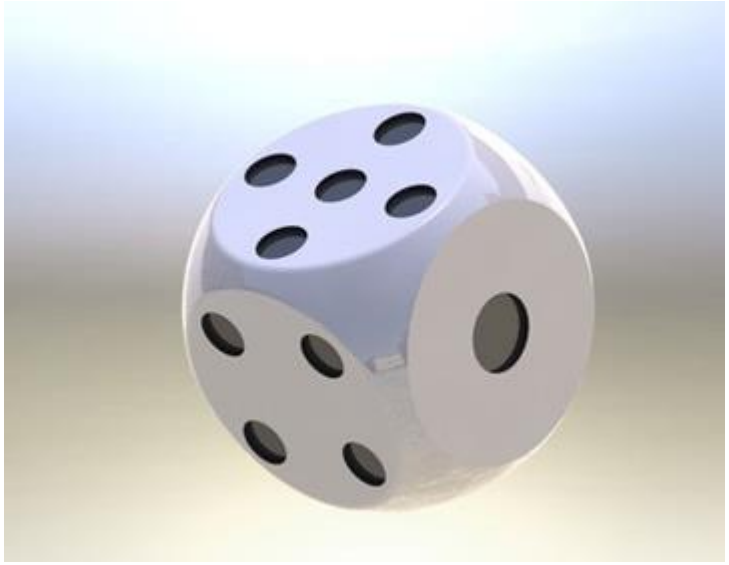


Figure 1-1. A dice

1.1. Determinism

Can we predict the result of rolling dice?

——Yes, Law of large numbers!

How about perfectly?

Furthermore, can we a hundred percent predict the world? Can we be informed our destiny and control it just as the God?

——Démon de Laplace?!

——Quantum theory!

1.2.1. What is intelligence?

What is intelligence?

——First of all, let's take a look into the surrounding environment.

Figure 1-2. Piggy bank

超智能体

1.2.1. What is intelligence?

——The uncertainty of surrounding environment limits lives.

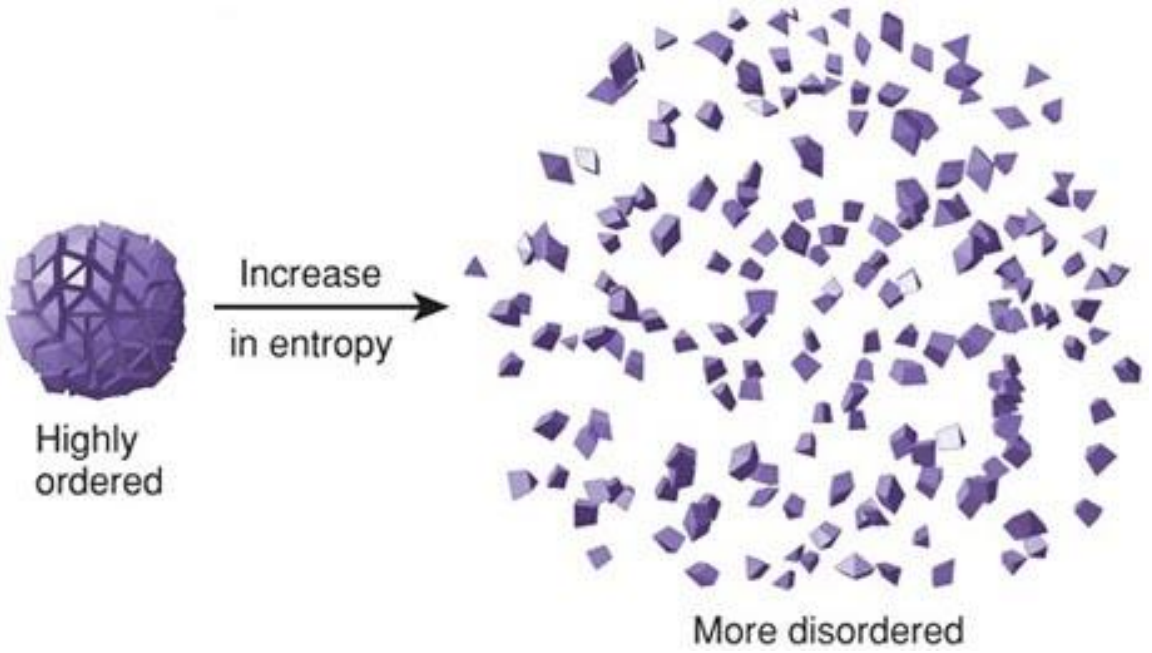


Figure 1-3. Entropy increase

1.2.1. What is intelligence?

——Schrodinger defined Intelligence as **the ability of reducing entropy** in his book, *What is Life*.

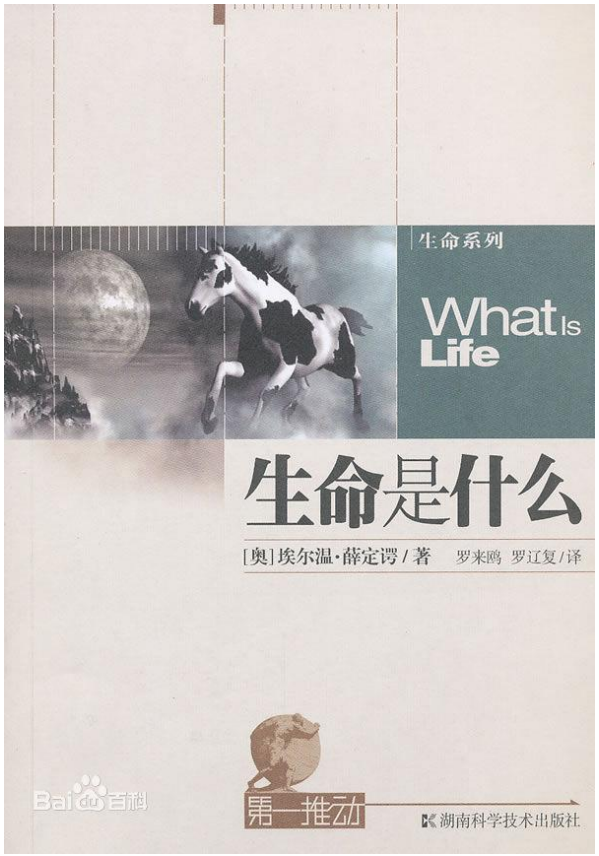


Figure 1-4.
The book *What is Life*

1.2.2. What is the dominant?

What is the dominant in intelligence?

——Speaking of that, I would sigh how pitiable the human being is!

Figure 1-5.
A screenshot

1.2.2. What is the dominant?

我，作为一个**不可分割的个体**，其经验和行动都是**统一于我的主观意识/经验**之中的。

——Thinking of organ transplant

who am I?

What is my consciousness?

Is there some different between I and my brain?

Whether my soul is just the biochemical reaction?

1.2.2. Go further through dissociation

1. Dissociation in Language——Expression and Understanding

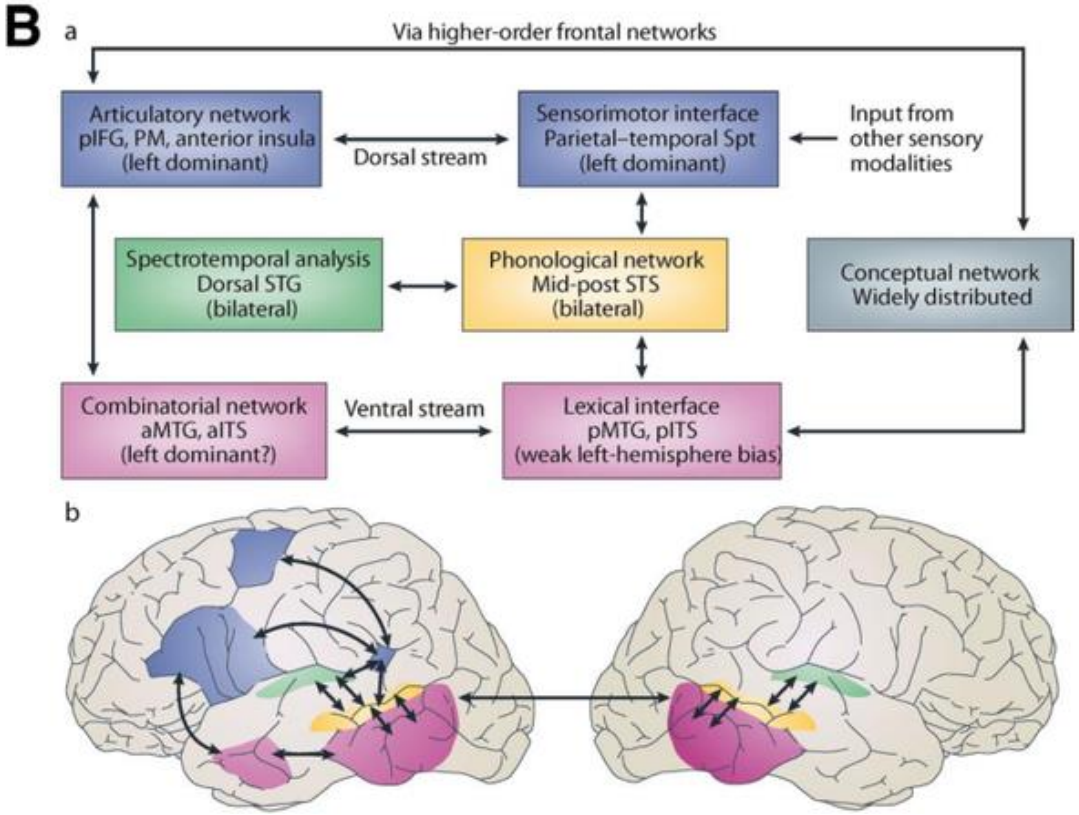


Figure 1-6. Sparse coding in the brain

Dissociation

Poeppel, D., Emmorey, K., Hickok, G., & Pylkkanen, L. (2012). *Towards a New Neurobiology of Language.* Journal of Neuroscience, 32(41), 14125–14131

1.2.2. Go further through dissociation

2. Dissociation in Vision——Objective recognition and subjective identification

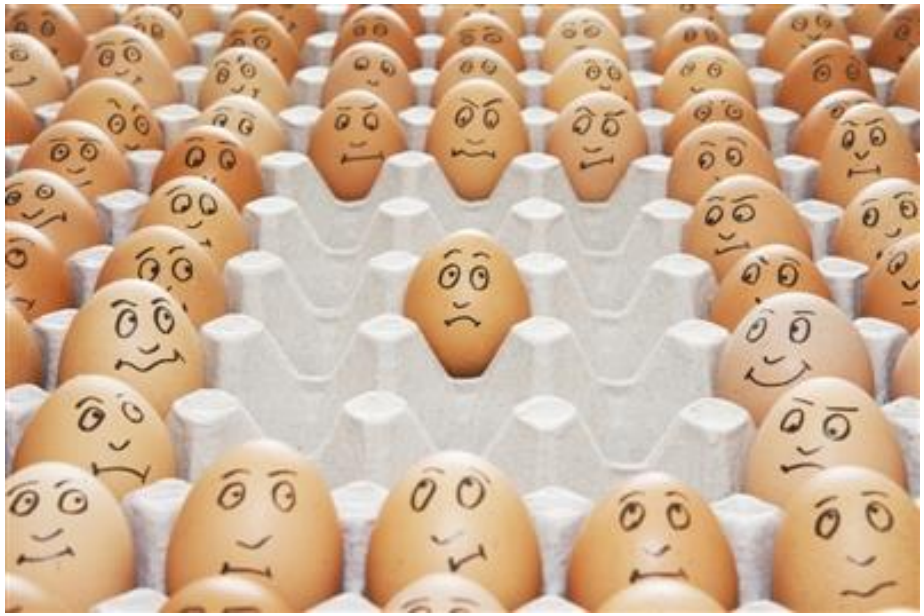


Figure 1-7.
Face blindness

Ellis, H. D., & Lewis, M. B. (2001). *Capgras delusion: a window on face recognition.* Trends in Cognitive Sciences, 5(4), 149–156.

1.2.2. Go further through dissociation

3. Dissociation in spatial perception——Left and right brain separation

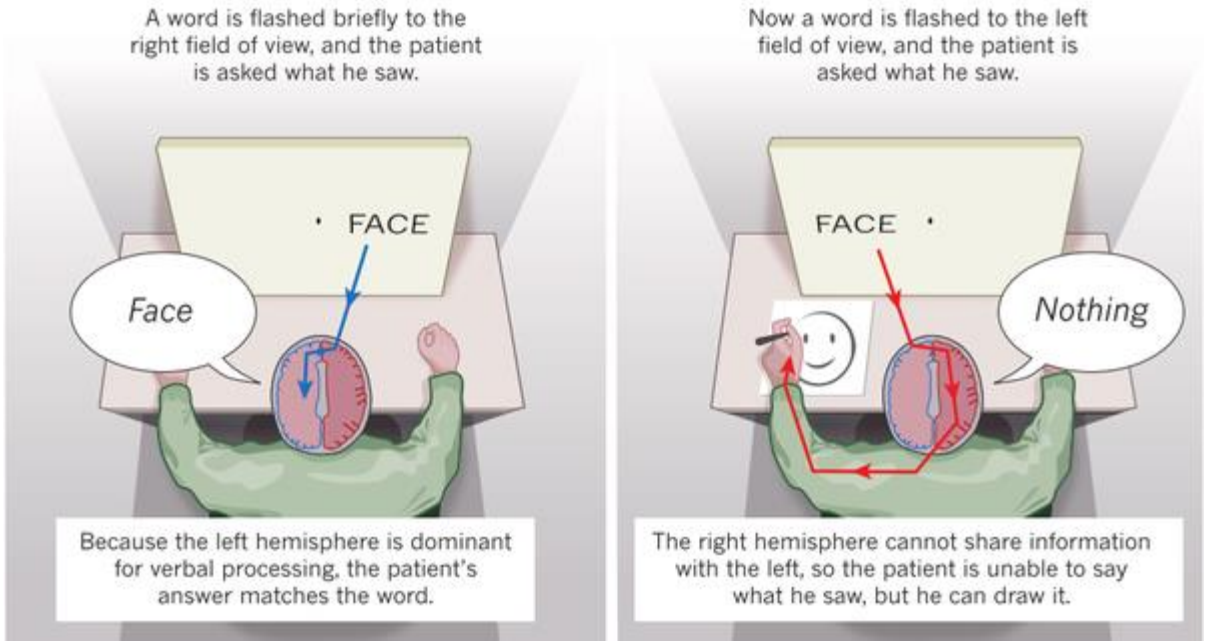


Figure 1-8.
Face blindness

Wolman, D. (2012). *The split brain: A tale of two halves.* Nature, 483(7389), 260–263.

1.2.2. Go further through dissociation

4. More extremely! Libet experiment: time separation between consciousness and decision making

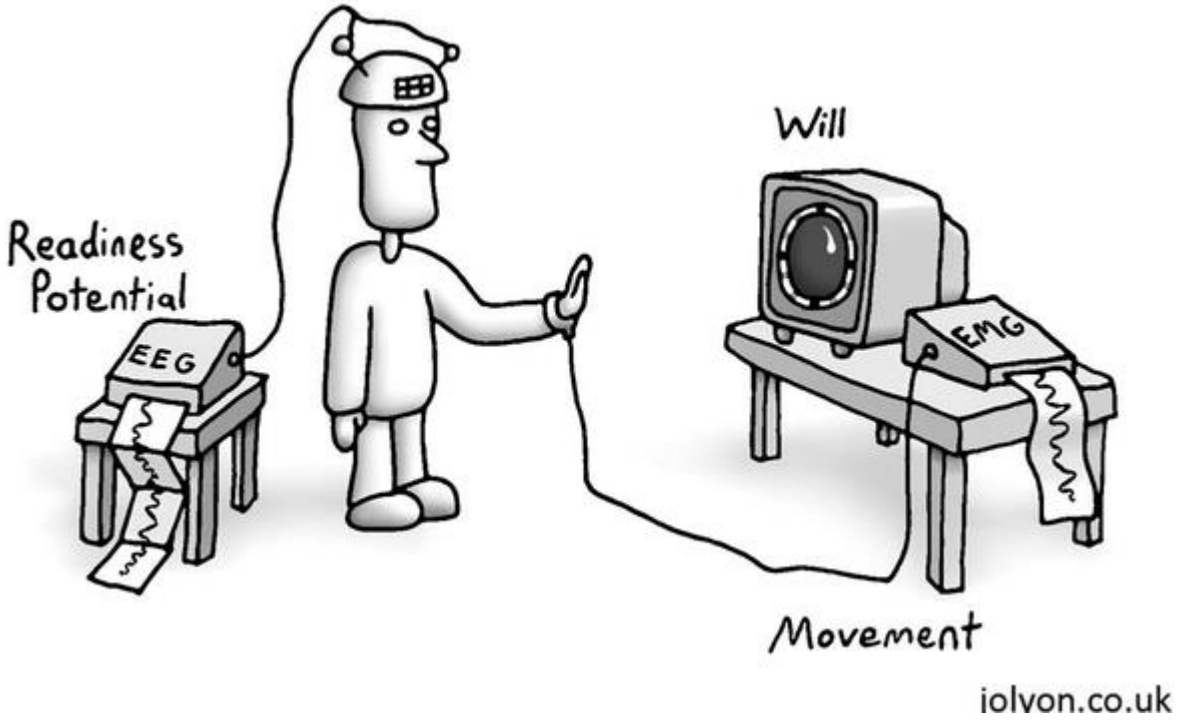


Figure 1-9. Libet experiment

Libet, B., Gleason, C. A., Wright, E. W., & Pearl, D. K. (1983). *Time of Conscious Intention To Act in Relation To Onset of Cerebral Activity (Readiness-Potential).* Brain, 106(3), 623–642.

1.2.2. Go further through dissociation

Finally, the answer (mine) is that the mind of our brain is out of our control, although we hope it controlled by ourselves.

Therefore, we name it as mind to be deeply convinced that we indeed control it. Well, out of our mind, there is a beautiful and amazing world!

1.3.1 Further——Memory Mechanism

What is the essence of memory?

——Based on neurobiological perspective, the prevalent view is **the neuron theory**, which points out that memory is encoded in the dynamic changes of the neural cell junction.

In other words, memory reflects the variability and plasticity of neural cells.

The formation of memory is not determined by one neuro, but by a lot of connection in neural cells.

1.3.1 Further——Memory Mechanism

From the molecular mechanism of learning and memory, the study suggests that the physical basis of memory is the small change in the synapse.

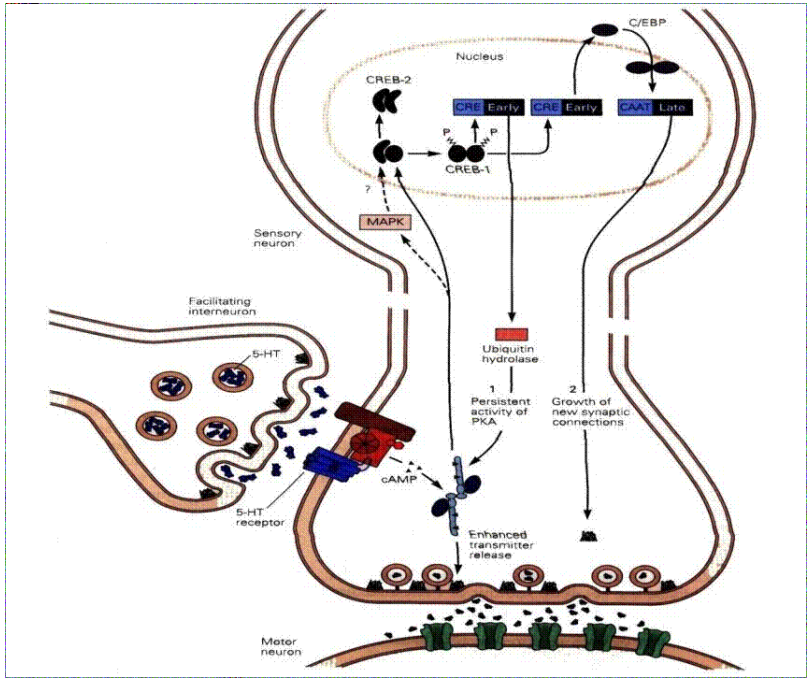


Figure 1-10.
Synapse development mechanism

Eric R. Kandel, *The Molecular Biology of Memory Storage: Adialogue between Genes and Synapse,* Science, Vol (294): 1030-1038, 2001

1.3.1 Further——Memory Mechanism

To simplify, long term activation ‘stronger’ synapse and by contrast, long term deactivation ‘weaker’ synapse.

The basis of Long term memory formation is the plasticity of neural cells. By long term potentiation (LTP) and long term depression (LTD), connections between synapses are controlled and then the learned are encoded in memory.

1.3.1 Further——Memory Mechanism

多数记忆材料都需要经过Joseph LeDoux提到的“**短时缓冲装置**”，即位于双侧听觉区、视觉区和运动知觉区的暂时存储区域后，然后才能进入工作记忆（一种暂时记忆，比短时记忆要长一些）和之后的长时记忆。

而语义记忆、情景记忆、程序记忆、自动记忆和情绪记忆的通路则用于储存和提取长时记忆的信息。其中，情景记忆和语义记忆（两者都属于陈述性记忆）主要储存在海马；程序记忆和自动记忆主要储存在小脑；情绪记忆主要储存在杏仁核。

Howard Eichenbaum著，周仁来等译. *记忆的认知神经科学——导论*[M]. 北京：北京师范大学出版社，2008.

1.3.1 Further——Memory Mechanism

语义记忆：人们对一般知识和规律的记忆，比如你阅读文献所得到到知识的记忆。

情景记忆：可以理解成关系记忆或者空间记忆，主要是指位置信息。

程序记忆：通常也成为“肌肉记忆”，比如我们学开车、骑自行车等记忆。

自动记忆：也称作为conditioned response memory，表现为特点的刺激对记忆的自动激发，例如说你唱着歌我自动的就接下一句的这种状态。

情绪记忆：就是关于情绪的记忆，值得一提的是，情绪记忆总是优先于其他任何一种记忆。

1.3.1 Hebb's hypothesis

Figure 1-11. A cattle

1.3.1 Hebb's hypothesis

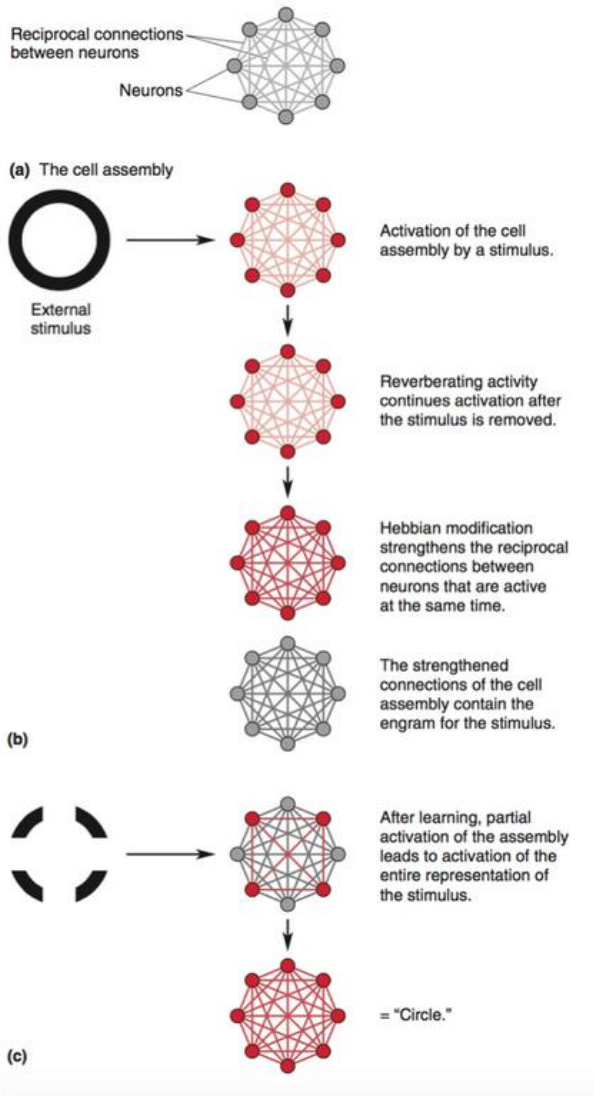


Figure 12.
Synapse development mechanism

concept set
'unsupervised' learning !
related description = label !

1.3.2 Quickly overview for others

1. 巴普洛夫的狗

早期的理论就认为多巴胺的释放让动物产生了快感。

然而，Berridge和Robinson两个人以及他们的同事，通过一系列实验否定了多巴胺导致快感。

他们发现发现，增加或者减少多巴胺并不会改变动物或者人对自然以及成瘾药物的快感体验；但是会改变动物或人对于获得奖赏刺激的动机。

Berridge KC, Robinson TE (1998) *What is the role of dopamine in reward: hedonic impact, reward learning, or incentive salience?* Brain Res Brain Res Rev 28:309-369

1.3.2 Quickly overview for others

另外一个，非常有影响力的假说，是由Schultz等人在1997年提出的“奖赏预测误差假说 (reward prediction error hypothesis)”。

这一理论是通过在体记录清醒猴子单个多巴胺神经元的放电活动的出来的。首先，训练猴子把一种视觉刺激 (CS) 和糖水的奖赏刺激 (US) 联系起来。就像当年巴浦洛夫训练他的狗是一样的。这种训练被称为巴浦洛夫条件反射。在训练的初期，多巴胺神经元对CS没有反应，但是会对US产生一个快速放电。但是到了训练后期，多巴胺神经元只对CS有反应，而对US没有反应了。另外，如果训练结束之后，在本来应该得到US的时候没有给，多巴胺神经元反而会有抑制的反应。于是他们推测，多巴胺神经元是导致动物学会CS和US建立联系的原因。他们发现，这一现象和机器学习中的reinforcement learning不谋而合。他们利用机器学习理论建立了一些模型，模拟了动物的学习过程。

Schultz W, Dayan P, Montague PR (1997) *A neural substrate of prediction and reward.* Science 275:1593-1599.

1.3.2 Quickly overview for others

2. 大脑皮层功能重塑

在现在的理论里，大脑是有很多功能分区，比如就视觉这块都有很多分区，分别在处理不同的信息上起作用。比如v1加工边缘朝向，mt加工复杂的运动模式这样的。

本来认为在成人的大脑中功能分区已经比较稳定了，不太会发生很大的变化。但是这个实验观察到了一个比较大的皮层“功能重塑”

Chen, N., Bi, T., Zhou, T., Li, S., Liu, Z., & Fang, F. (2015). *Sharpened cortical tuning and enhanced cortico-cortical communication contribute to the long-term neural mechanisms of visual motion perceptual learning. NeuroImage, 115*, 17-29. doi:10.1016/j.neuroimage.2015.04.041

Chen, N., Cai, P., Zhou, T., Thompson, B., & Fang, F. (2016*). Perceptual learning modifies the functional specializations of visual cortical areas.* Proc Natl Acad Sci U S A. doi:10.1073/pnas.1524160113

3. 语言影响知觉

包括颜色知觉（Davies&Corbett，1997；Roberson，Park，&Hanley，2008）在内的诸多领域（时间视觉、动作知觉、空间位置想象以及抽象物体的视觉搜索）中都获得了明确的实验证据。

提到语言、思维和文化，会首先想到**萨丕尔-沃尔夫假说**。假说的大概意思就是，不同文化的语言在很多方面不一样，我们的思维受到了这些方面的影响。

1.3.2 Quickly overview for others

我们汉语里面的蓝色只有一个基本颜色词对应——蓝，但是世界之大无奇不有，俄语（还嫌远的话，蒙古语也是一样）就有两个基本颜色词，分别对应着我们的“深蓝”和“浅蓝”。别看就这么一个词的差异，这就导致了我们和俄语母语者在不深不浅的蓝色（9到11）时候，颜色的分辨上巨大的不同。他们可以很快的分辨出略有区别的蓝色（9和10），但是我们就困难很多。

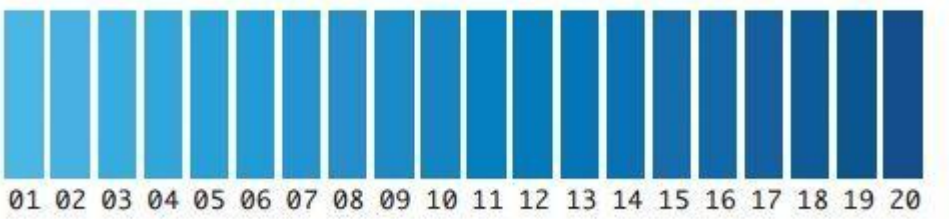


Figure 1-13.
Blue spectrum

1.3.2 Quickly overview for others

4. 时间知觉

如果我们对时间知觉多一些了解，就会知道，记忆和时间知觉是纠缠不清、相互关联的。

在时间知觉理论理论中，最经典的理论叫做**起搏器-累加器模型**（pacemaker-accumulator）。这个理论中最关键的成分有两个，这个理论认为，我们的大脑内部有一个时钟，叫做起搏器，会稳定地发出脉冲。这就像我们的时钟，滴答、滴答地不断走针，记录时间的流逝。随后，我们的大脑会收集起搏器发出的脉冲，进入累加器，然后把脉冲的数量和我们大脑中的的时间信息比较

Block, R. A., Zakay, D., & Hancock, P. A. (1998). *Human aging and duration judgments: A meta-analytic review.* Psychology and aging, 13(4), 584.

陈有国, 黄希庭, 尹天子, & 张锋. (2011). *时间知觉的理论模型与展望*. 西南大学学报: 人文社会科学版, 37(5), 26-33.

Tea Break

2.0 Be motivated in deep neural network

Overwhelming in performance!!!

Significantly broaden our available research area and imagination!

2.0 Be motivated in deep neural network

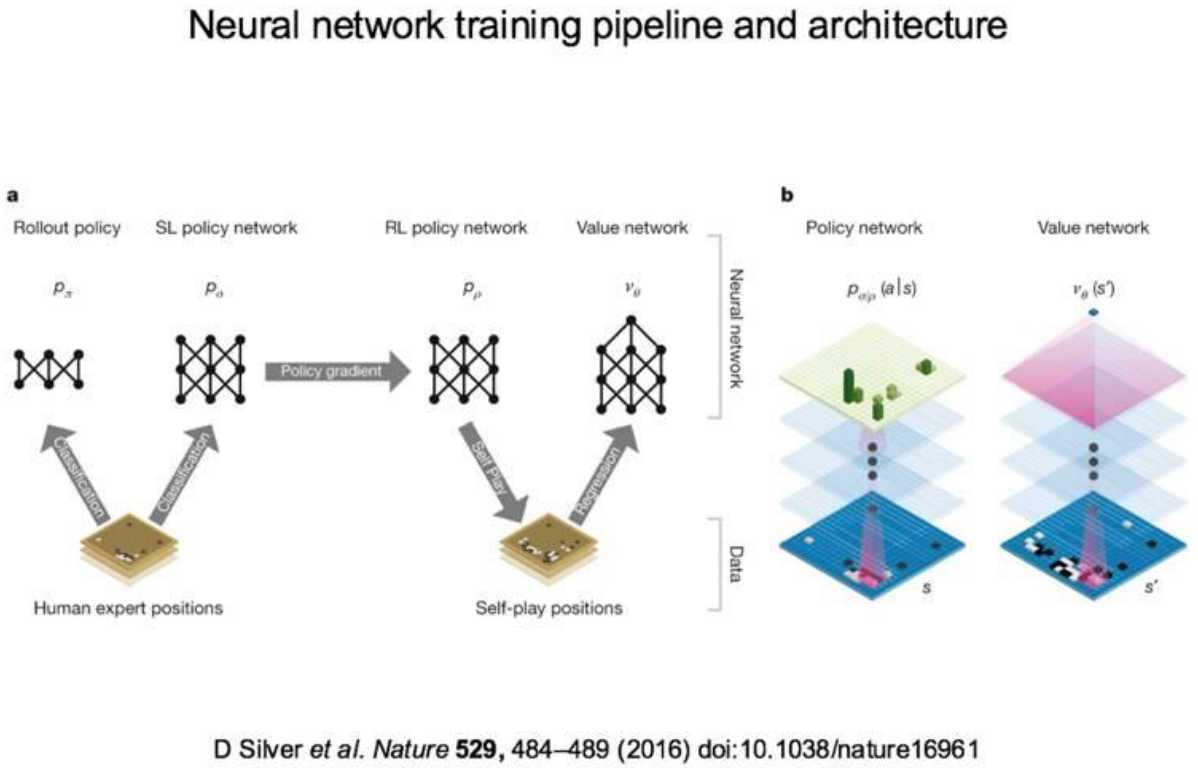


Figure 2-1. Mechanism of AlphaGo

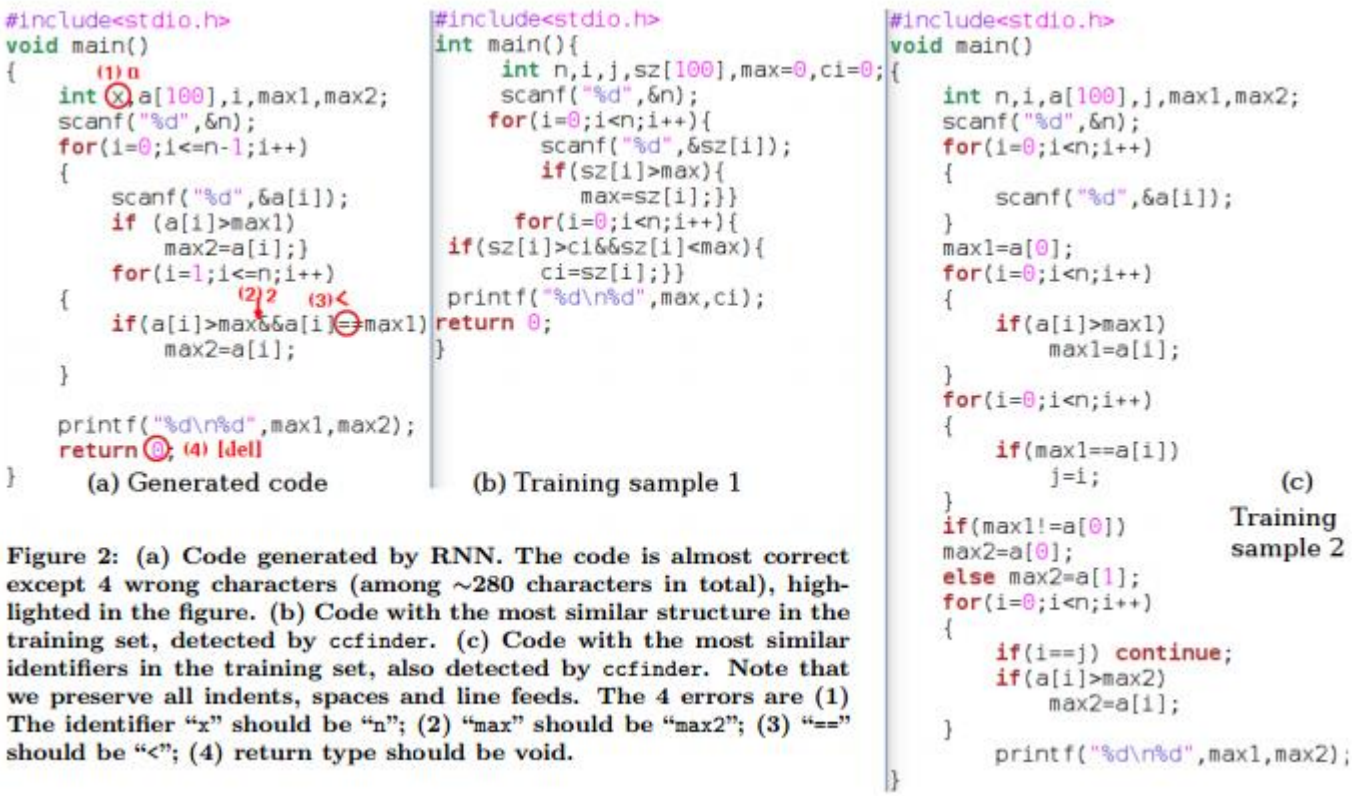
- A girl wearing glasses and a pink shirt.
- An Asian girl with a pink shirt eating at the table.

- A boy brushing his hair while looking at his reflection.
- A young male child in pajamas shaking around a hairbrush in the mirror.

- Zebra looking towards the camera.
- A zebra third from the left.

Figure 2-2. Examples of picture talking

2.0 Be motivated in deep neural network



```
#include<stdio.h>
void main()
{
    int x,a[100],i,max1,max2;
    scanf("%d",&n);
    for(i=0;i<=n-1;i++)
    {
        scanf("%d",&a[i]);
        if (a[i]>max1)
            max2=a[i];}
        for(i=1;i<=n;i++)
    {
        if(a[i]>max&&a[i]==max1)
            max2=a[i];
    }

    printf("%d\n%d",max1,max2);
    return 0;
}
```

(1) n (2) 2 (3) < (4) [del]

(a) Generated code

```
#include<stdio.h>
int main(){
     int n,i,j,sz[100],max=0,ci=0;
     scanf("%d",&n);
    for(i=0;i<n;i++){
        scanf("%d",&sz[i]);
        if(sz[i]>max){
           max=sz[i];}}
     for(i=0;i<n;i++){
 if(sz[i]>ci&&sz[i]<max){
        ci=sz[i];}}
 printf("%d\n%d",max,ci);
return 0;
}
```

(b) Training sample 1

```
#include<stdio.h>
void main()
{
    int n,i,a[100],j,max1,max2;
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    max1=a[0];
    for(i=0;i<n;i++)
    {
        if(a[i]>max1)
            max1=a[i];
    }
    for(i=0;i<n;i++)
    {
        if(max1==a[i])
            j=i;
    }
    if(max1!=a[0])
    max2=a[0];
    else max2=a[1];
    for(i=0;i<n;i++)
    {
        if(i==j) continue;
        if(a[i]>max2)
            max2=a[i];
    }
        printf("%d\n%d",max1,max2);
}
```

(c) Training sample 2

Figure 2: (a) Code generated by RNN. The code is almost correct except 4 wrong characters (among ~280 characters in total), highlighted in the figure. (b) Code with the most similar structure in the training set, detected by ccfinder. (c) Code with the most similar identifiers in the training set, also detected by ccfinder. Note that we preserve all indents, spaces and line feeds. The 4 errors are (1) The identifier "x" should be "n"; (2) "max" should be "max2"; (3) "==" should be "<"; (4) return type should be void.

Figure 2-3. Demonstration of Program Generation

2.1.1.0 Single Neuron

Single neuro:

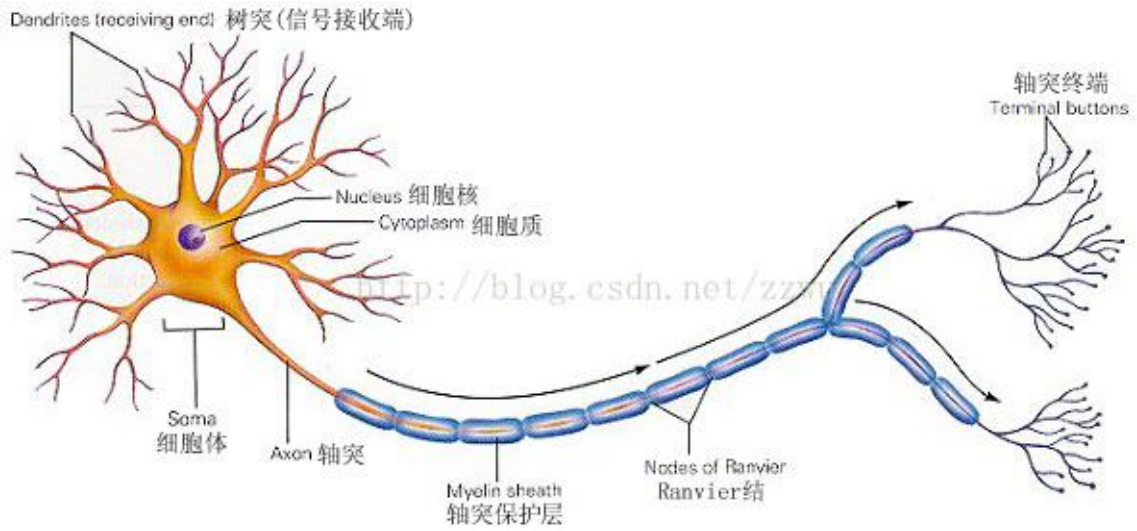


Figure 2-4. An example of neuro

2.1.1.1 Perception

The basic unit of computation in a neural network is the **neuron**, often called a **node** or **unit**. It receives input from some other nodes, or from an external source and computes an output.

The function *f* is non-linear and is called the **Activation Function**. The purpose of the activation function is to introduce non-linearity into the output of a neuron.

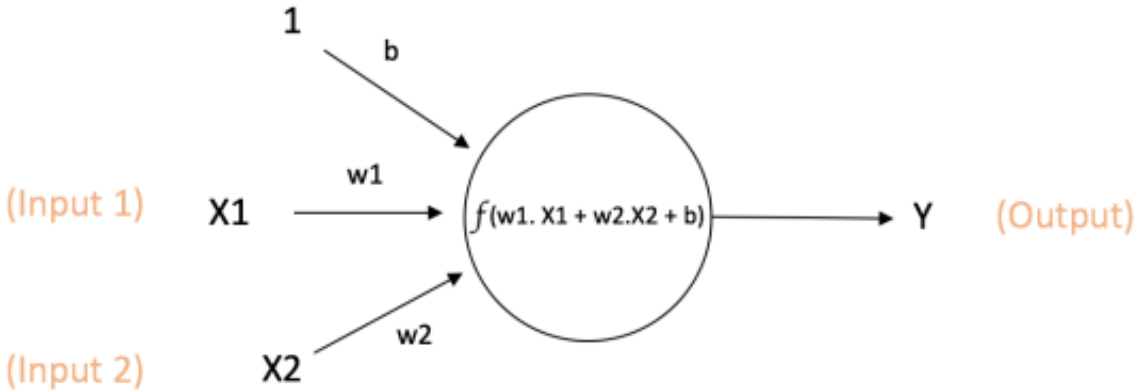


Figure 2-5. A Single Neuron

Output of neuron = Y= $f(w1.X1 + w2.X2 + b)$

2.1.1.2 Activation Function

- **Sigmoid:**

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- **tanh:**

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

- **ReLU (**Rectified Linear Unit**):**

$$f(x) = \max(0, x)$$

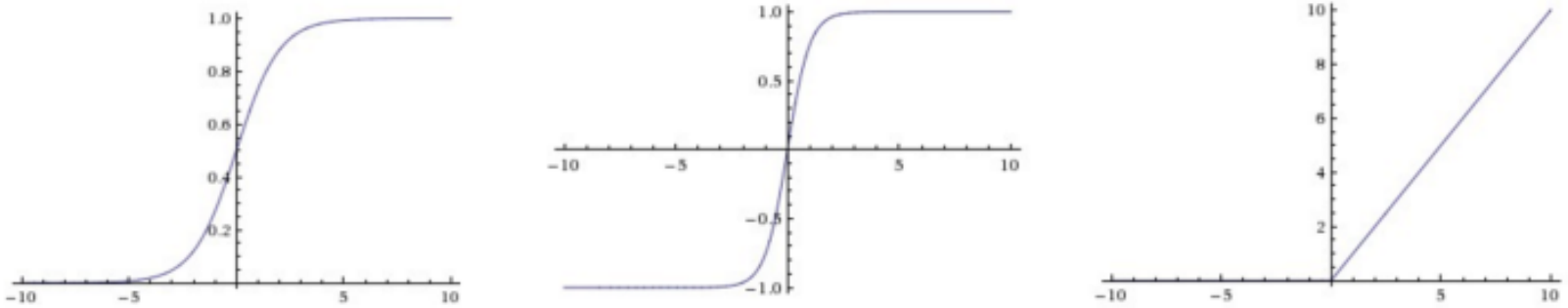


Figure 2-6. Different activation functions

2.1.1.3 Artificial Neural Network

An Artificial Neural Network (ANN)

is a computational model that is inspired by the way biological neural networks in the human brain process information.

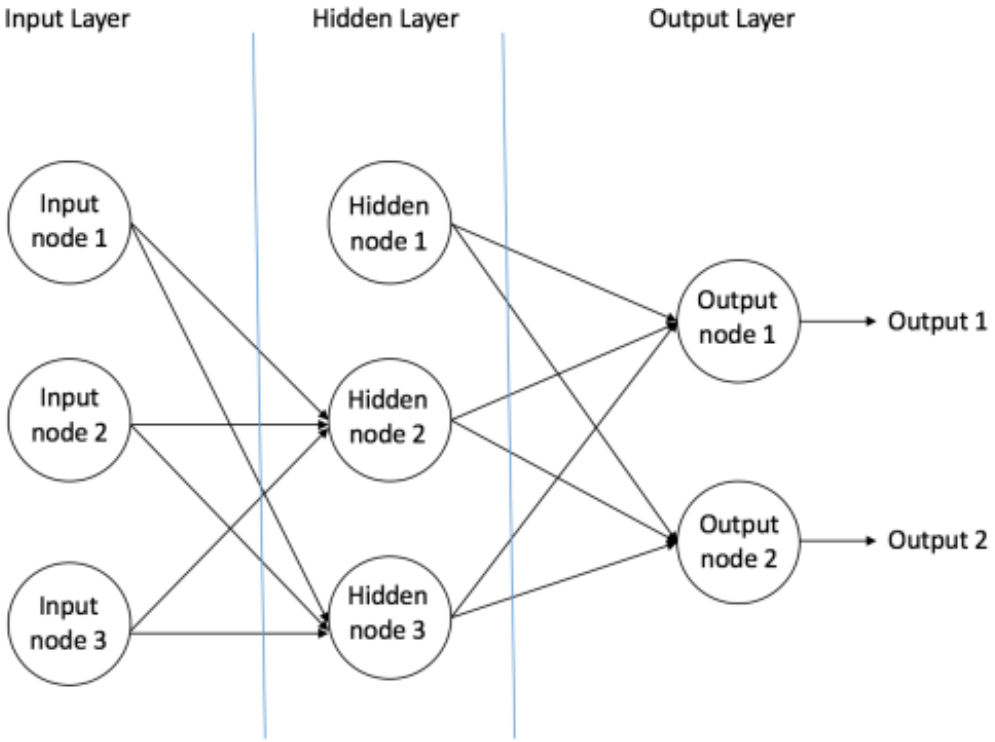


Figure 2-7. An example of feedforward neural network

A Quick Introduction to Neural Networks

2.1.1.4 Multi Layer Perceptron

A Multi Layer Perceptron (MLP) contains one or more hidden layers. While a single layer perceptron can only learn linear functions, a multi layer perceptron can also learn non – linear functions.

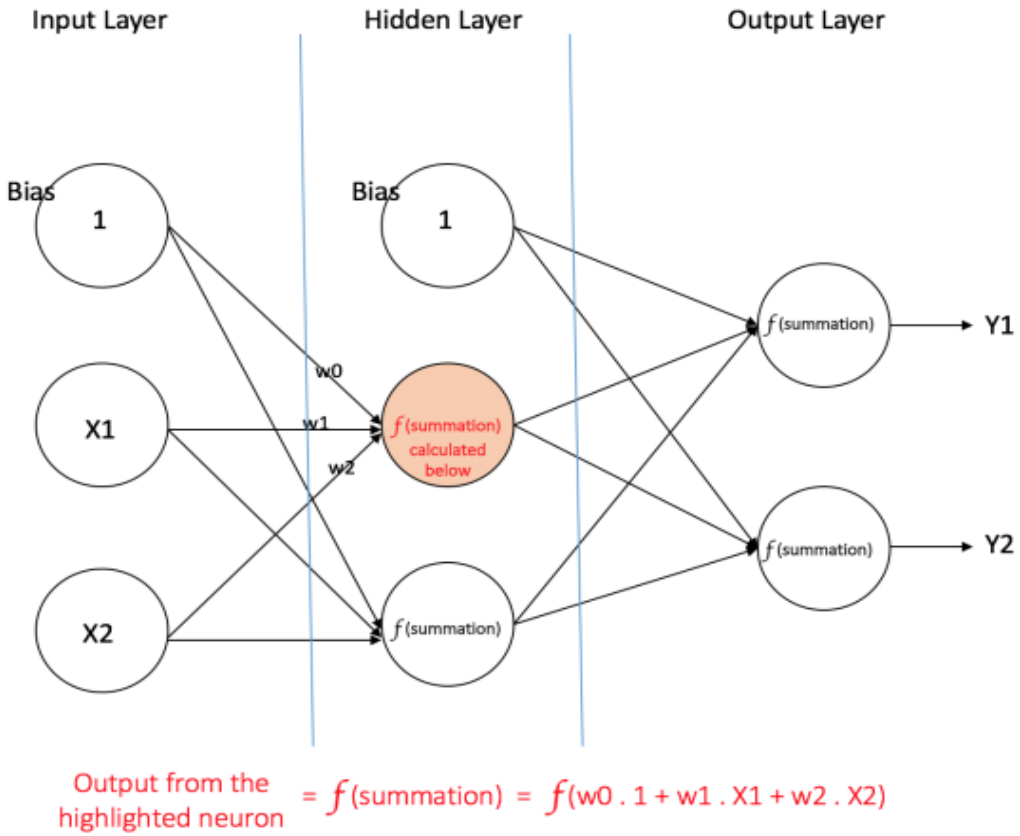


Figure 2-8.
A multi layer perceptron having one hidden layer

2.1.1.4 The Back-Propagation Algorithm

“**learning from mistakes**“

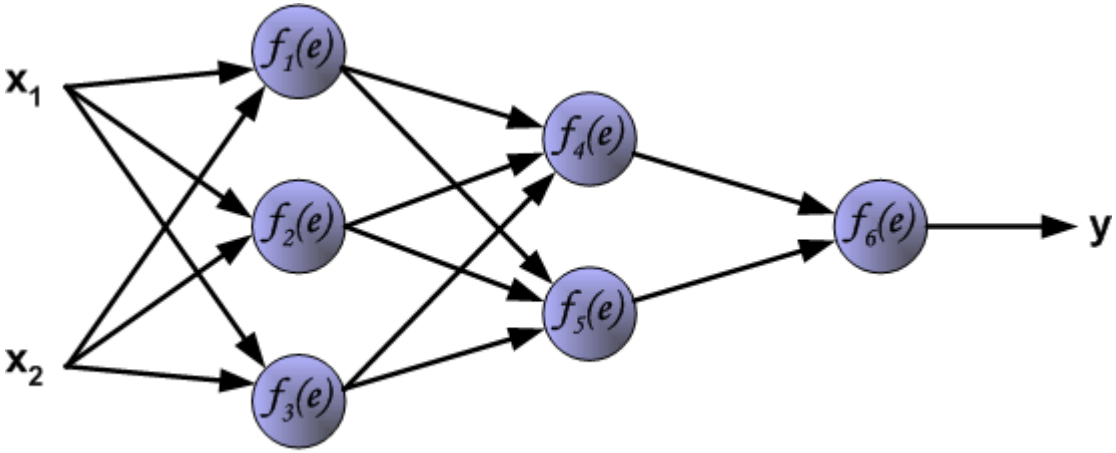
This output is compared with the desired output that we already know, and the error is “propagated” back to the previous layer.

This error is noted and the weights are “adjusted” accordingly.

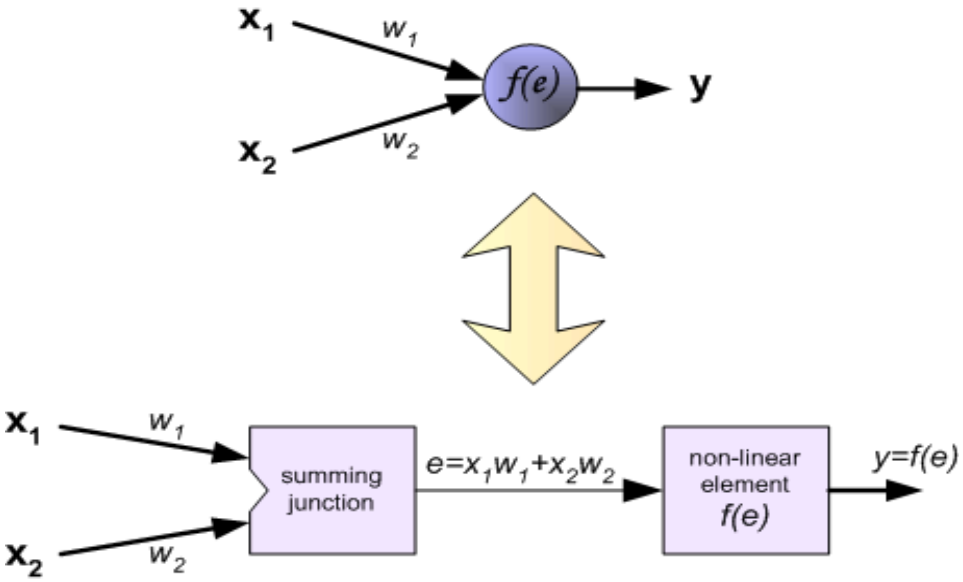
This process is repeated until the output error is below a predetermined threshold.

From Quora

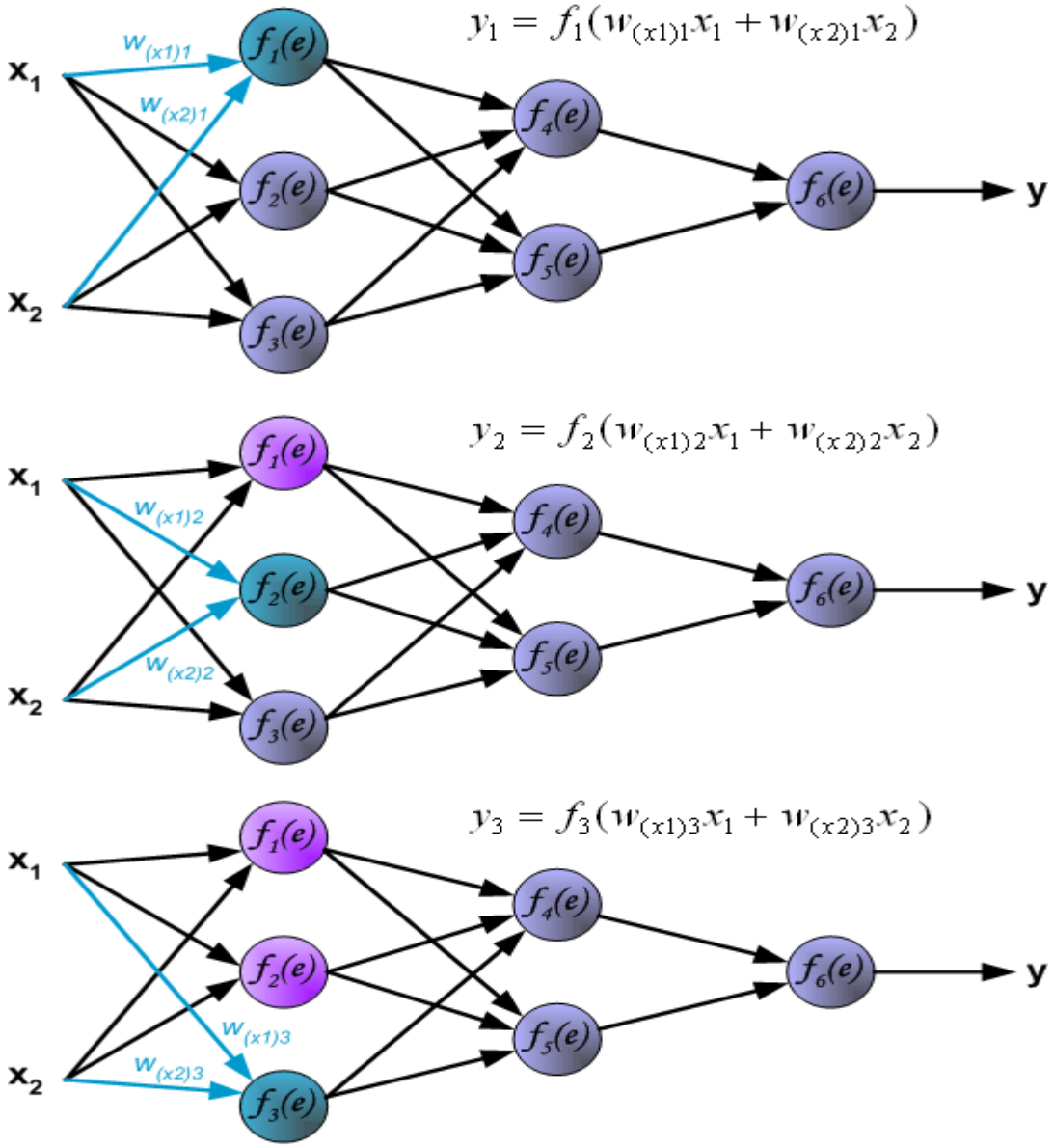
2.1.1.4 Demonstration of BP Algorithm



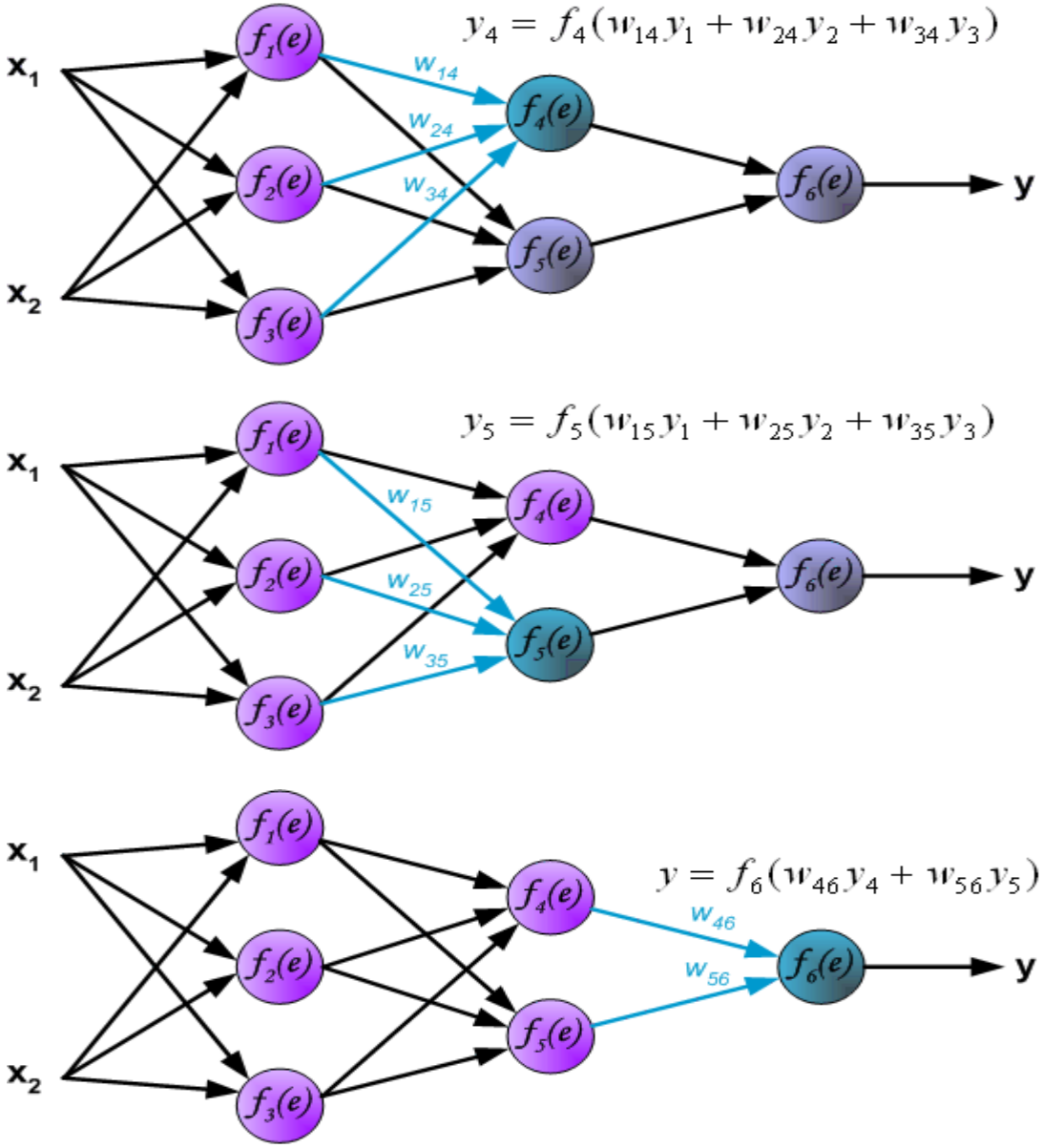
[Principles of training multi-layer neural network using backpropagation]



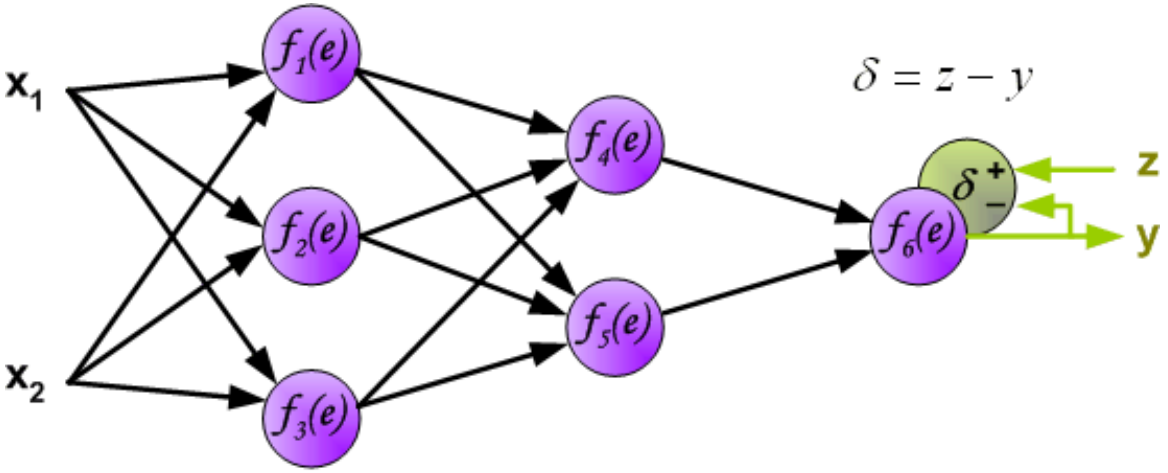
2.1.1.4 Demonstration of BP Algorithm



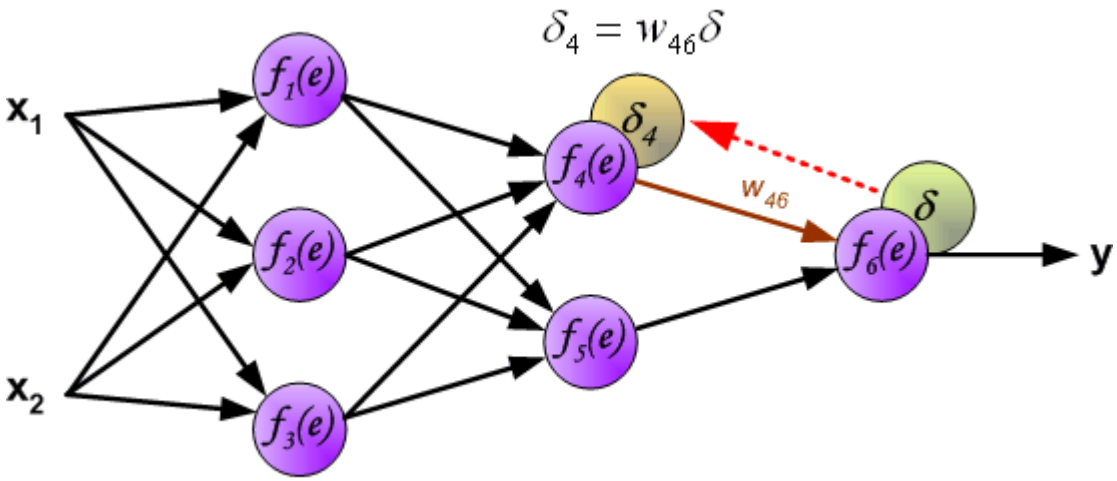
2.1.1.4 Demonstration of BP Algorithm

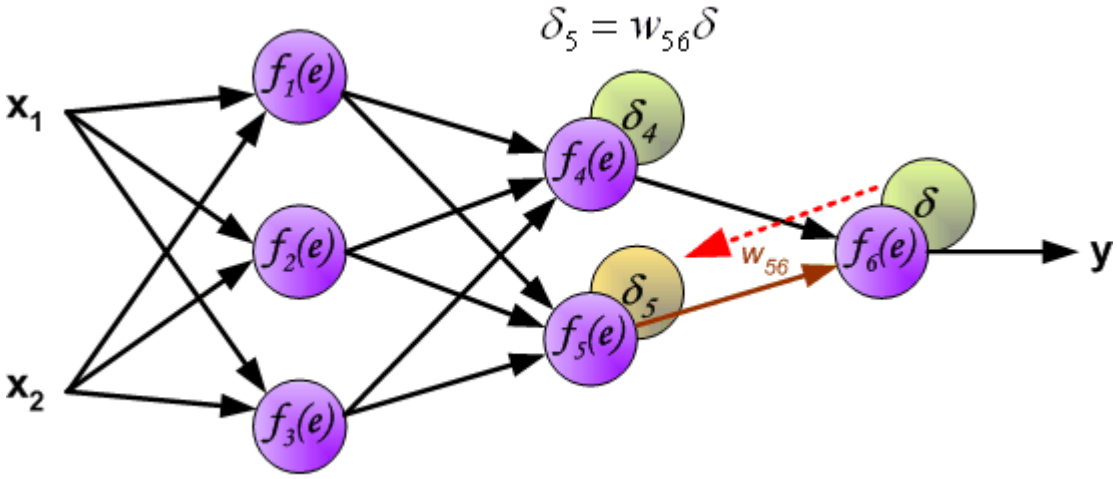


2.1.1.4 Demonstration of BP Algorithm

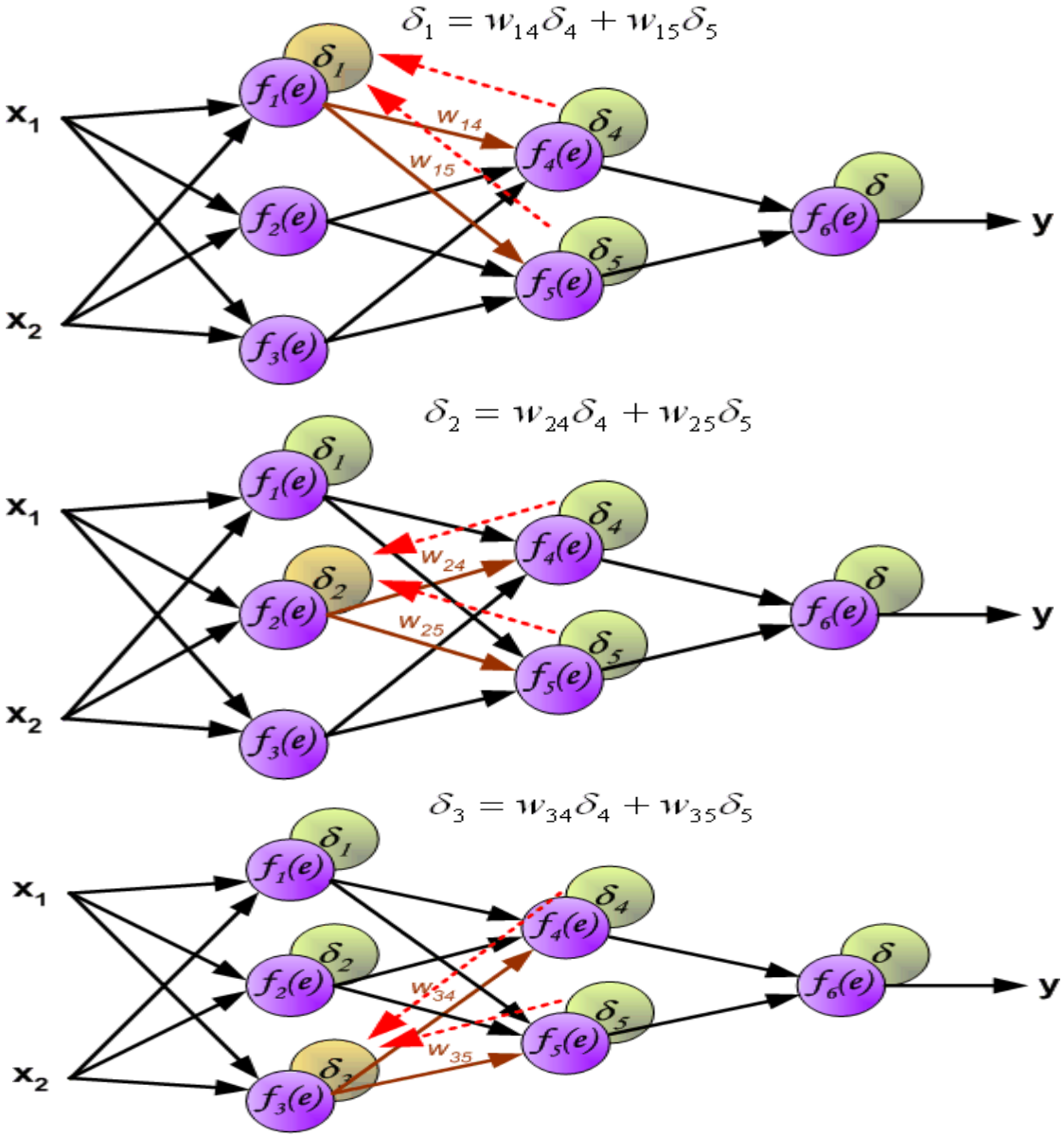


2.1.1.4 Demonstration of BP Algorithm





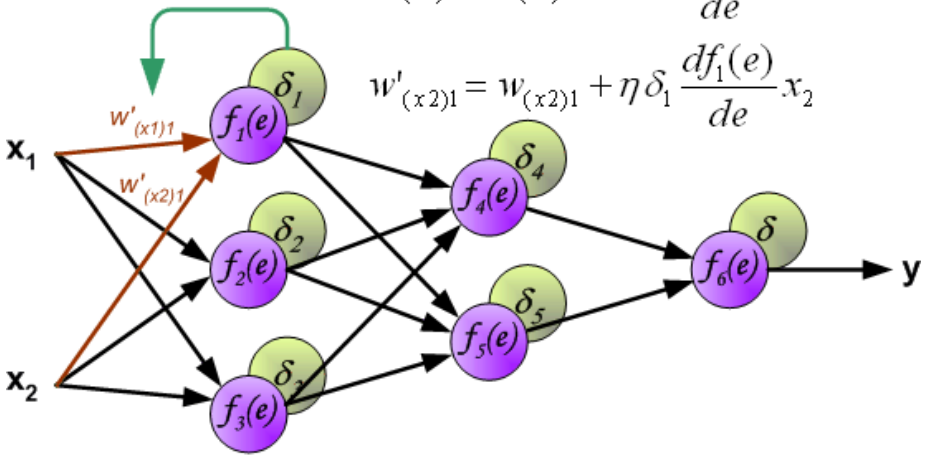
2.1.1.4 Demonstration of BP Algorithm



2.1.1.4 Demonstration of BP Algorithm

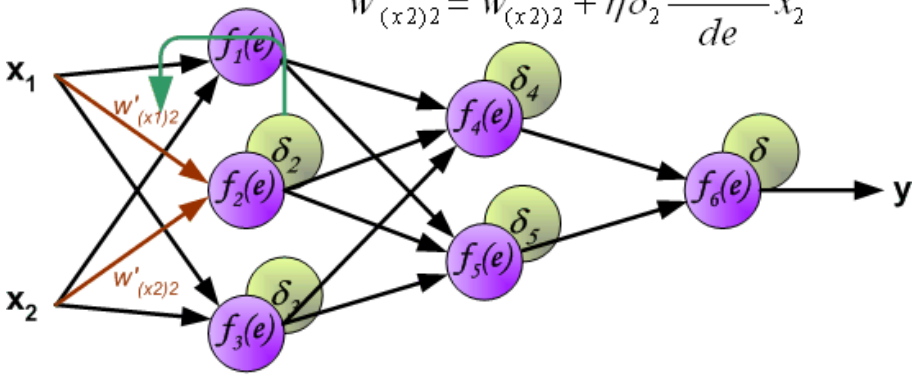
$$w'_{(x1)1} = w_{(x1)1} + \eta\delta_1 \frac{df_1(e)}{de} x_1$$

$$w'_{(x2)1} = w_{(x2)1} + \eta\delta_1 \frac{df_1(e)}{de} x_2$$



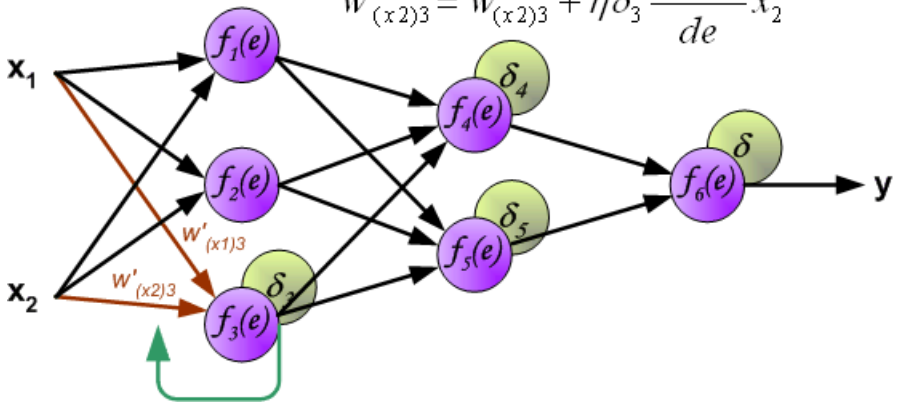
$$w'_{(x1)2} = w_{(x1)2} + \eta\delta_2 \frac{df_2(e)}{de} x_1$$

$$w'_{(x2)2} = w_{(x2)2} + \eta\delta_2 \frac{df_2(e)}{de} x_2$$



$$w'_{(x1)3} = w_{(x1)3} + \eta\delta_3 \frac{df_3(e)}{de} x_1$$

$$w'_{(x2)3} = w_{(x2)3} + \eta\delta_3 \frac{df_3(e)}{de} x_2$$

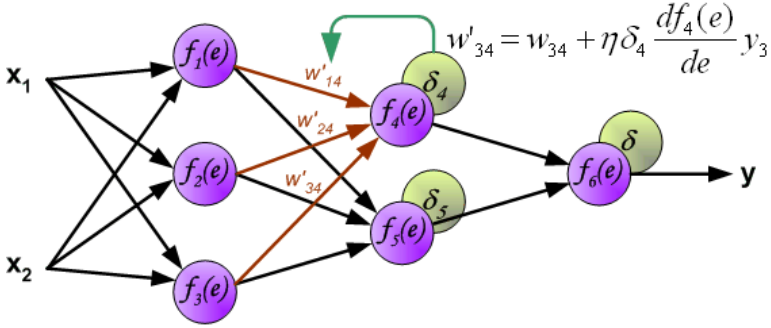


2.1.1.4 Demonstration of BP Algorithm

$$w'_{14} = w_{14} + \eta\delta_4 \frac{df_4(e)}{de} y_1$$

$$w'_{24} = w_{24} + \eta\delta_4 \frac{df_4(e)}{de} y_2$$

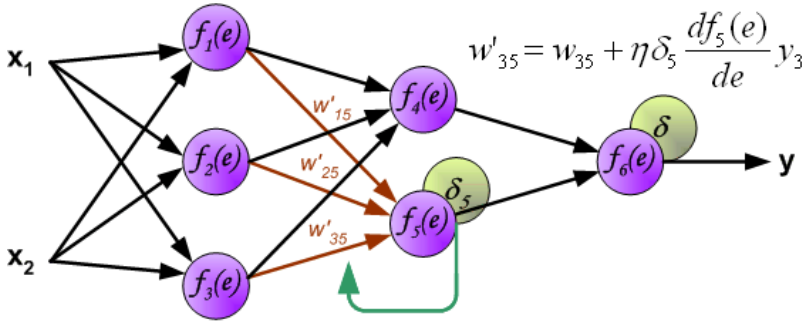
$$w'_{34} = w_{34} + \eta\delta_4 \frac{df_4(e)}{de} y_3$$



$$w'_{15} = w_{15} + \eta\delta_5 \frac{df_5(e)}{de} y_1$$

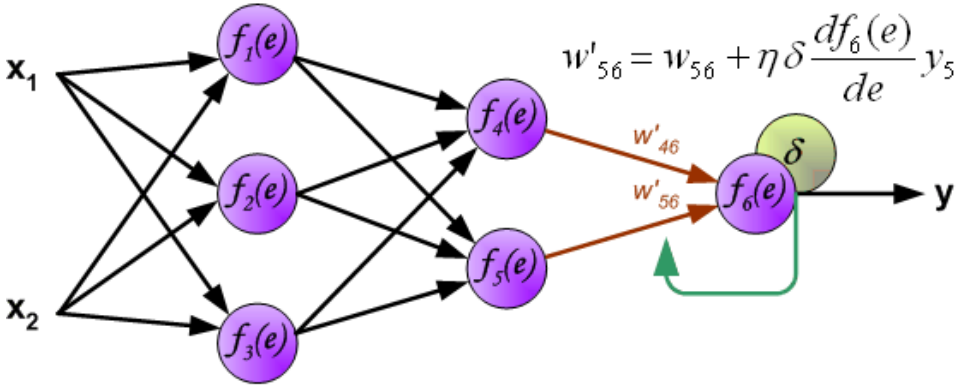
$$w'_{25} = w_{25} + \eta\delta_5 \frac{df_5(e)}{de} y_2$$

$$w'_{35} = w_{35} + \eta\delta_5 \frac{df_5(e)}{de} y_3$$



$$w'_{46} = w_{46} + \eta\delta \frac{df_6(e)}{de} y_4$$

$$w'_{56} = w_{56} + \eta\delta \frac{df_6(e)}{de} y_5$$



2.1.1.4 Mathematics

For given $x^{(i)}$,$y^{(i)}$, the output of feedforward neural network is $f(x|w,b)$, and then the objective function is:

$$J(W,b) = \sum_{i=1}^{N} L(y^{(i)}, f(x^{(i)} \mid W,b)) + \frac{1}{2}\lambda \|W\|_F^2$$

$$= \sum_{i=1}^{N} J(W,b;x^{(i)},y^{(i)}) + \frac{1}{2}\lambda \|W\|_F^2$$

If we adopt gradient descent method, then

$$W^{(l)} = W^{(l)} - \alpha \frac{\partial J(W,b)}{\partial W^{(l)}}$$

$$= W^{(l)} - \alpha \sum_{i=1}^{N} \left(\frac{\partial J(W,b;x^{(i)},y^{(i)})}{\partial W^{(l)}}\right) - \lambda W$$

$$b^{(l)} = b^{(l)} - \alpha \frac{\partial J(W,b)}{\partial b^{(l)}}$$

$$= b^{(l)} - \alpha \sum_{i=1}^{N} \left(\frac{\partial J(W,b;x^{(i)},y^{(i)})}{\partial b^{(l)}}\right)$$

2.1.1.4 Mathematics

By chain rule,

$$\frac{\partial J(W, b; x, y)}{\partial W_{ij}^{(l)}} = tr\left(\left(\frac{\partial J(W, b; x, y)}{\partial z^{(l)}}\right)^T \frac{\partial z^{(l)}}{\partial W_{ij}^{(l)}}\right)$$

Define the first term as a error term $\delta^{(l)}$,

$$\delta^{(l)} = \frac{\partial J(W, b; x, y)}{\partial z^{(l)}} \in \mathbb{R}^{n^{(l)}}$$

Indicating the influence of l^{th} level neuros to final error.

For the second term $z^{(l)} = W^{(l)} \bullet a^{(l-1)} + b^{(l)}$,

$$\frac{\partial z^{(l)}}{\partial W_{ij}^{(l)}} = \frac{\partial (W^{(l)} \cdot a^{(l-1)} + b^{(l)})}{\partial W_{ij}^{(l)}} = \begin{bmatrix} 0 \\ \vdots \\ a_j^{(l-1)} \\ \vdots \\ 0 \end{bmatrix}$$

2.1.1.4 Mathematics

Therefore,

$$\frac{\partial J(W, b; x, y)}{\partial W_{ij}^{(l)}} = \delta_i^{(l)} a_j^{(l-1)}$$

and

$$\frac{\partial J(W, b; x, y)}{\partial W^{(l)}} = \delta^{(l)} (a^{(l-1)})^T$$

Similarly, the gradient of b:

$$\frac{\partial J(W, b; x, y)}{\partial b^{(l)}} = \delta^{(l)}$$

2.1.1.4 Mathematics

Finally, the error term of l^{th} level, $\delta^{(l)}$

$$\delta^{(l)} \triangleq \frac{\partial J(W, b; x, y)}{\partial z^{(l)}}$$

$$= \frac{\partial a^{(l)}}{\partial z^{(l)}} \cdot \frac{\partial z^{(l+1)}}{\partial a^{(l)}} \cdot \frac{\partial J(W, b; x, y)}{\partial z^{(l+1)}}$$

$$= diag(f_l'(z^{(l)})) \cdot (W^{(l+1)})^T \cdot \delta^{(l+1)}$$

$$= f_l'(z^{(l)}) \odot ((W^{(l+1)})^T \cdot \delta^{(l+1)})$$

As is shown, the error term of l^{th} level can be calculated from the one of $(l+1)^{th}$ level as back propagation.

2.1.2.0 Convolutional Neural Networks

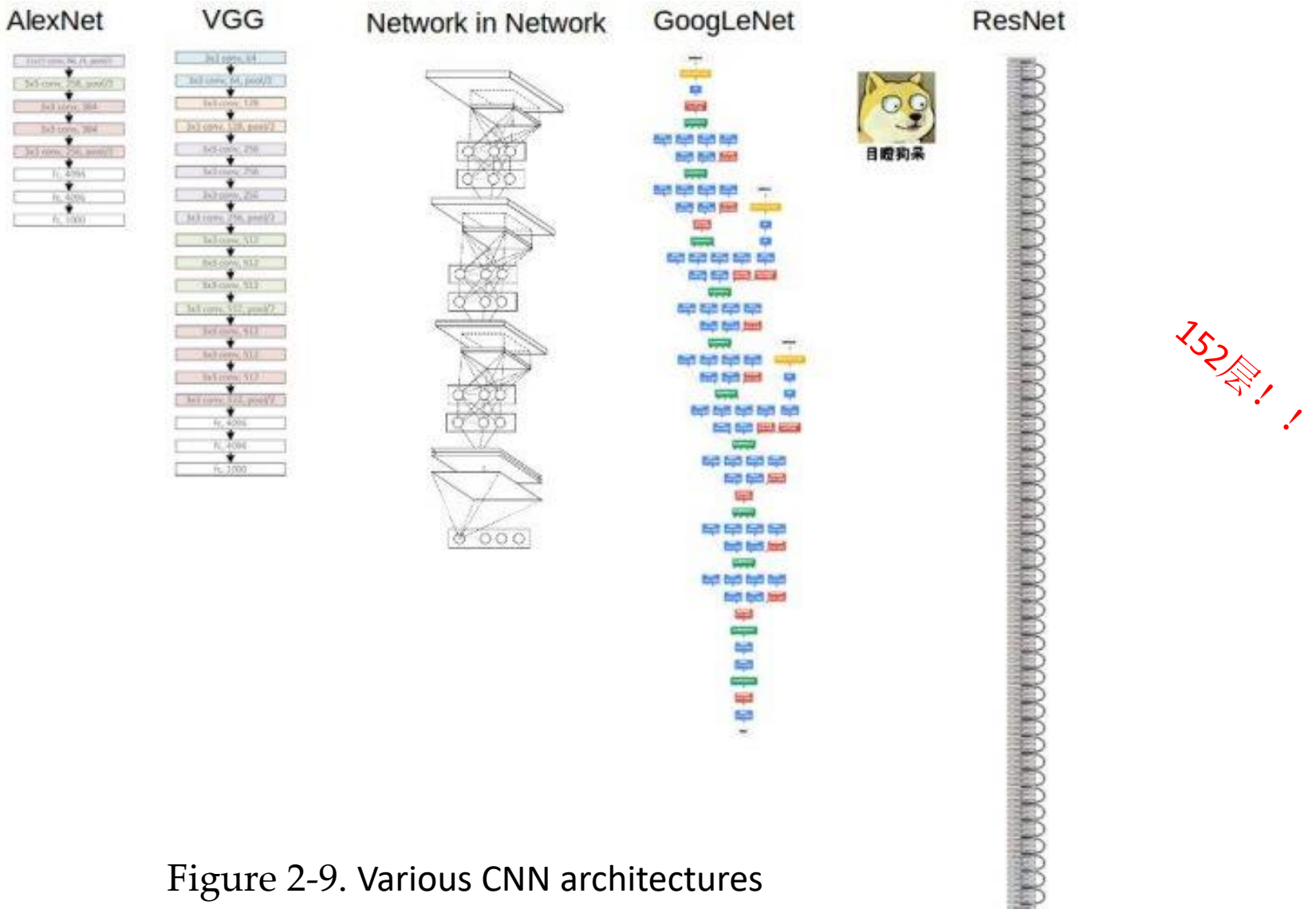


Figure 2-9. Various CNN architectures

2.1.2.1 LeNet intuitive introduction

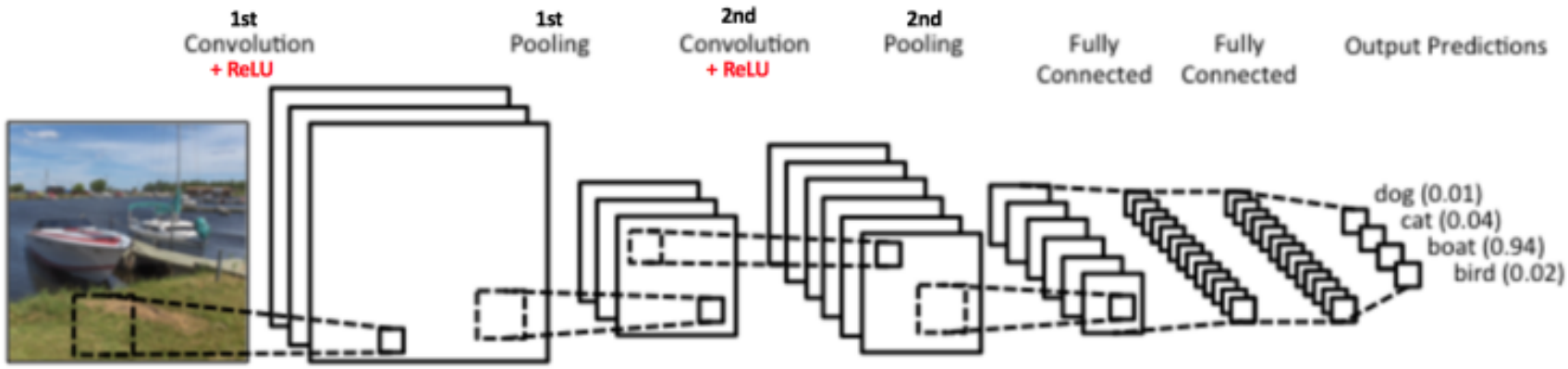


Figure 2-10. The architecture of LeNet

[An Intuitive Explanation of Convolutional Neural Networks]

2.1.2.1 Start from raw data

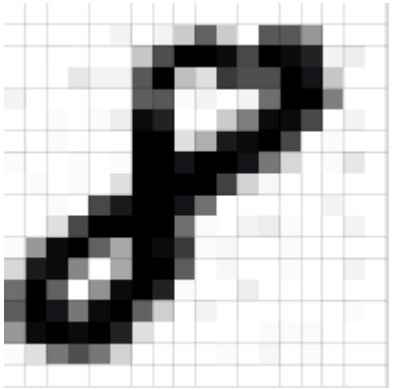


Figure 2-11. The raw data of handwriting

2.1.2.2 Convolution layer

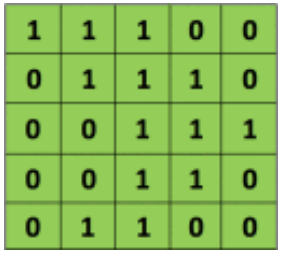


Figure 2-12. The raw data

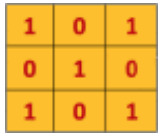
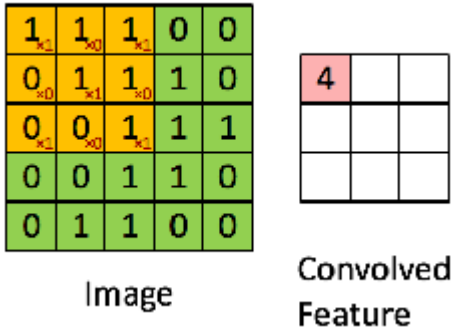


Figure 2-13. The convolution kernel



Key words:

- Convolution kernel
- Stride
- Zero-padding

Figure 2-14. The demonstration of convolution

2.1.2.2 Convolution layer

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

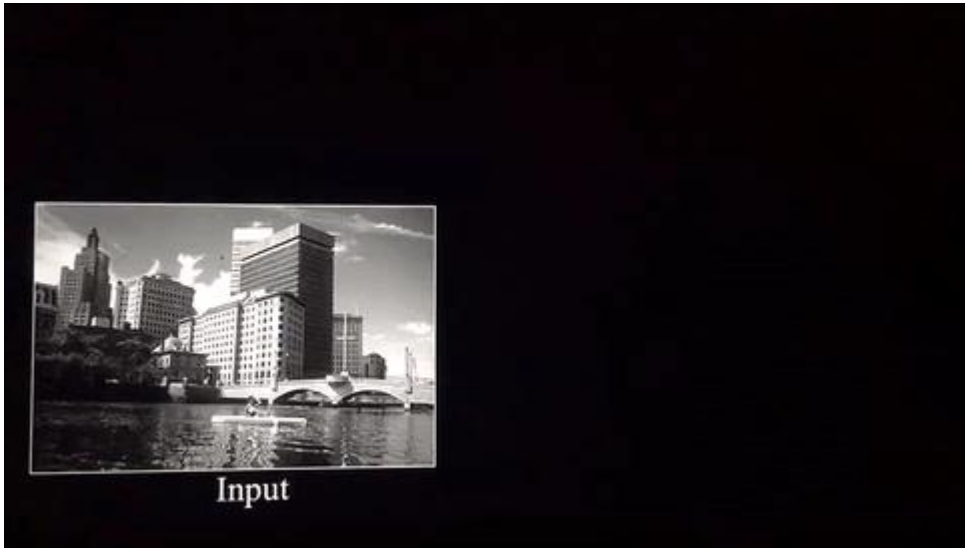


Figure 2-16. The demonstration of future extract

Figure 2-15. different feature filter (kernel)

2.1.2.3 The Pooling Step

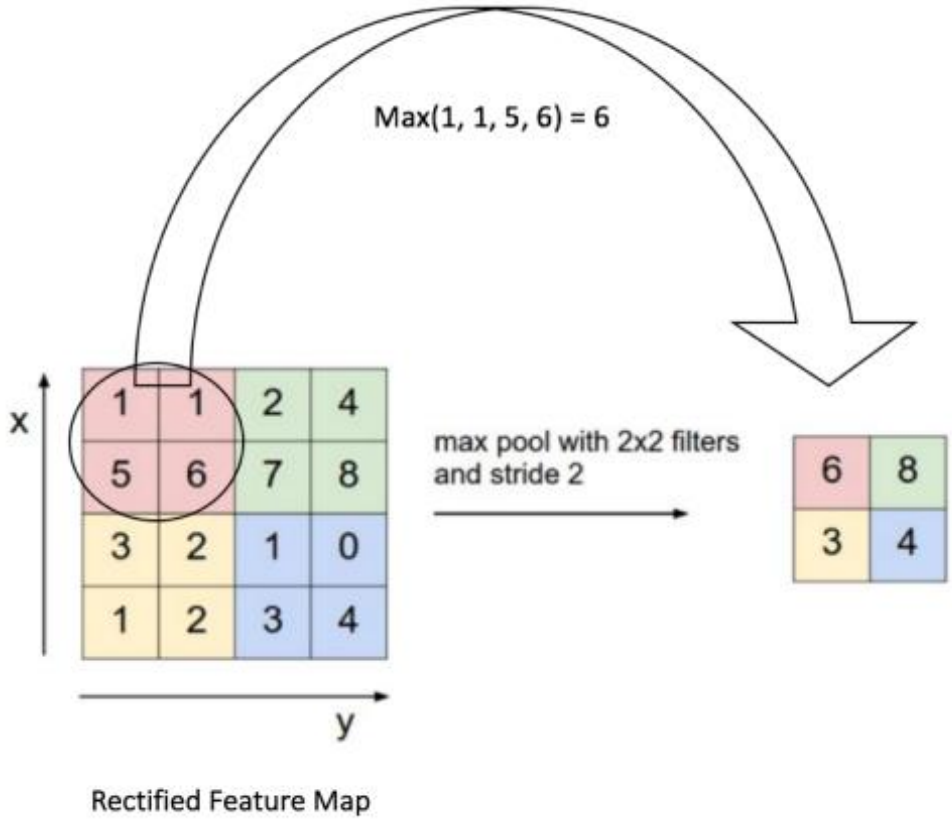


Figure 2-17. Max Pooling Source

2.1.2.3 Story so far

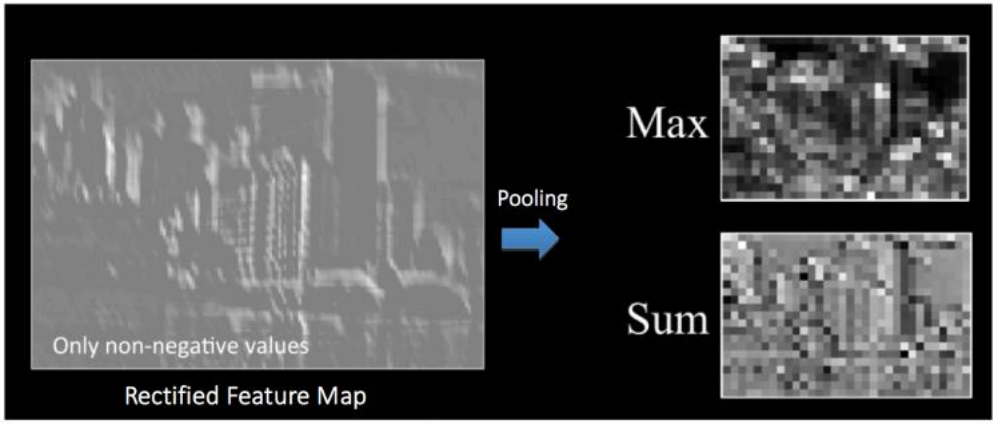


Figure 2-18. Pooling Source

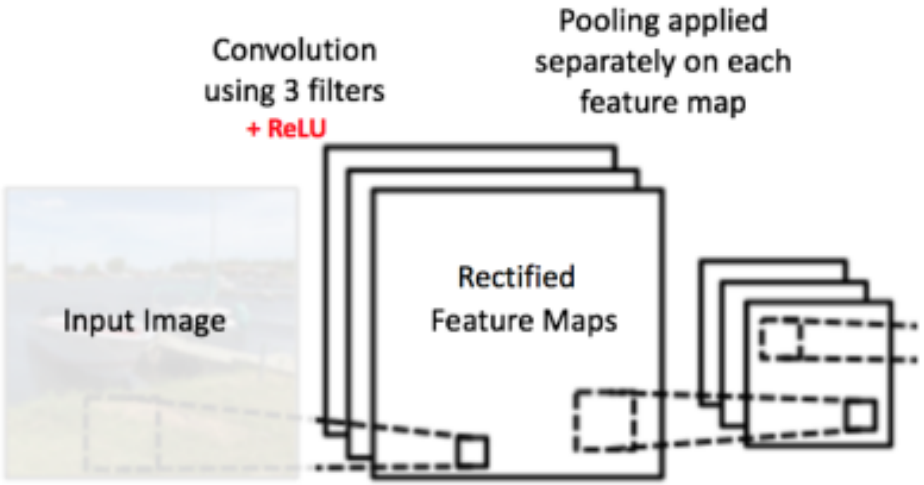


Figure 2-19. Pooling applied to Rectified Feature Maps

2.1.2.3 Convolution layer mathematics

Replace fully connected mode by convolution layer:

$$a^{(l)} = f(w^{(l)} \otimes a^{(l-1)} + b^{(l)})$$

where $w^{(l)} \in \mathbb{R}^m$ is m-dimensional feature filter and share the same value for all neuros in the l^{th} convolution layer. So, we need only m+1 parameters. Usually, the number of neuro in the $(l+1)^{th}$ layer is designed as $n^{(l+1)} = n^{(l)} - m + 1$.

In l^{th} layer and k^{th} set feature map:

$$X^{(l,k)} = f(\sum_{p=1}^{n^{l-1}} (w^{(l,k,p)} \otimes X^{(l-1,p)}) + b^{(l,k)})$$

where $w^{(l,k,p)}$ is the map parameter of p^{th} set feature vector in $(l-1)^{th}$ layer to of l^{th} set feature vector in l^{th} layer.

2.1.2.3 Convolution layer mathematics

For gradient computation, we assume that the l^{th} layer is the convolution layer and the down sample layer is placed at the $(l+1)^{th}$ layer.

To derivate the k^{th} error term in the l^{th} layer $\delta^{(l,k)}$:

$$\delta^{(l,k)} \triangleq \frac{\partial J(W,b;X,y)}{\partial Z^{(l,k)}}$$

$$= \frac{\partial X^{(l,k)}}{\partial Z^{(l,k)}} \cdot \frac{\partial Z^{(l+1,k)}}{\partial X^{(l,k)}} \cdot \frac{\partial J(W,b;X,y)}{\partial Z^{(l+1,k)}}$$

$$= f_l'(Z^{(l)}) \odot (up(w^{(l+1,k)}\delta^{(l+1)}))$$

$$= w^{(l+1,k)}(f_l'(Z^{(l)}) \odot up(\delta^{(l+1)}))$$

2.1.2.3 Pooling layer mathematics

After the convolution layer, we get a feature map $X^{(l)}$. Divide $X^{(l)}$ into a series of areas R_k. k=1,…,K

$$X_k^{l+1} = f(Z_k^{l+1})$$

$$= f(w^{l+1} \cdot down(R_k) + b^{l+1})$$

therefore,

$$X^{l+1} = f(w^{l+1} \cdot down(X^l) + b^{l+1})$$

Usually, down sample function down(·) is Maximum Pooling or Average Pooling.

$$pool_{\max}(R_k) = \max_{i \in R_k} a_i$$

$$pool_{avg}(R_k) = \frac{1}{|R_k|} \sum_{i \in R_k} a_i$$

2.1.2.3 Pooling layer mathematics

For gradient computation, we assume that the l^{th} layer is the down sample layer and the convolution layer is placed at the $(l+1)^{th}$ layer.

$$Z^{(l+1,k)} = \sum_{p,T_{p,k}=1} (W^{(l+1,k,p)} \otimes X^{(l,p)}) + b^{(l+1,k)}$$

To derivate the k^{th} error term in the l^{th} layer $\delta^{(l,k)}$:

$$\delta^{(l,k)} \triangleq \frac{\partial J(W,b;X,y)}{\partial Z^{(l,k)}}$$

$$= \frac{\partial X^{(l,k)}}{\partial Z^{(l,k)}} \cdot \frac{\partial Z^{(l+1,k)}}{\partial X^{(l,k)}} \cdot \frac{\partial J(W,b;X,y)}{\partial Z^{(l+1,k)}}$$

$$= f_l'(Z^{(l)}) \odot (\sum_{p,T_{p,k}=1} (\delta^{(l+1,p)}) \otimes rot180(W^{(l,k,p)}))$$

2.1.2.4 Classifier

- Full Connected Layer

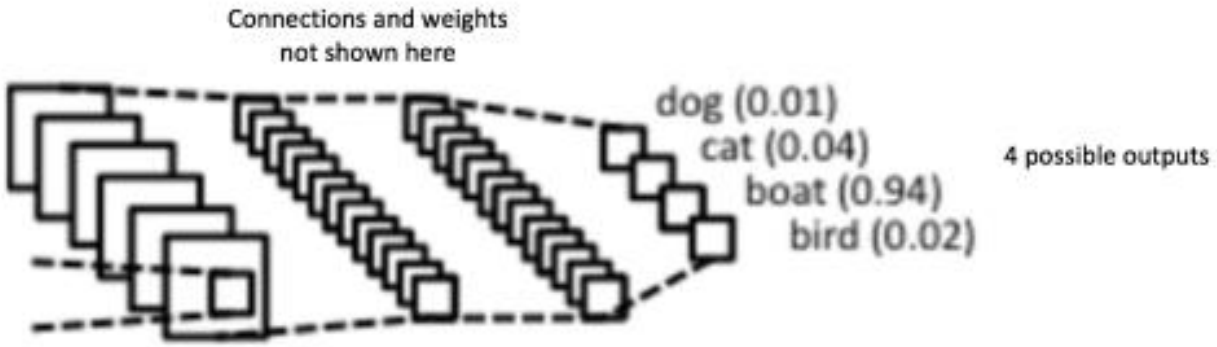


Figure 2-20. Full Connected Layer

- Global Average Pooling + Softmax

2.1.2.5 Think twice

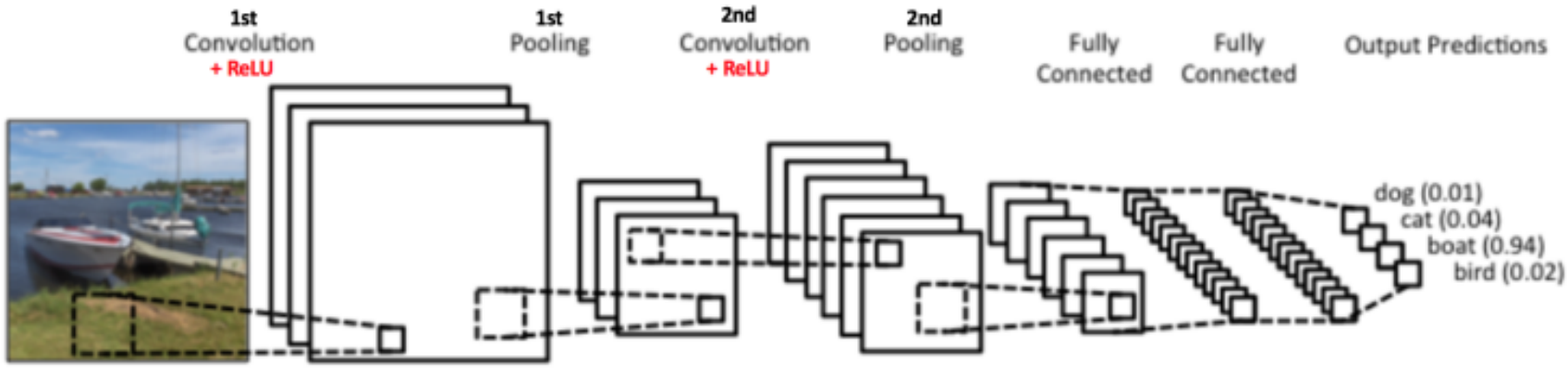


Figure 2-21. The architecture of LeNet

- Why is it successful?
- Why should it be deeper?
- Is that OK?

2.1.2.5 Think twice

- Why is it successful?
  - Neuroscience
    - ✓ Receptive field in vision area
    - ✓ Top-down processing
    - ✓ Hierarchical structure
  - Five space transformation operations
    - ✓ Increasing/Reducing dimensionality
    - ✓ Zooming
    - ✓ Rotating
    - ✓ Translation
    - ✓ Bending
  - Dropout

2.1.2.5 Think twice

- Why is it successful?

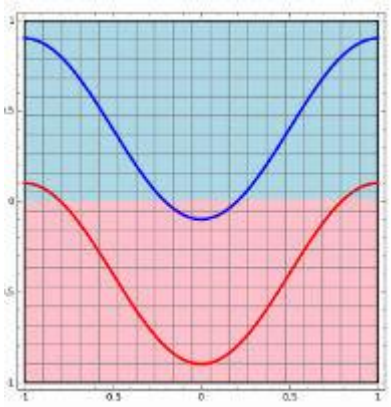


Figure 2-22.
Raw data space

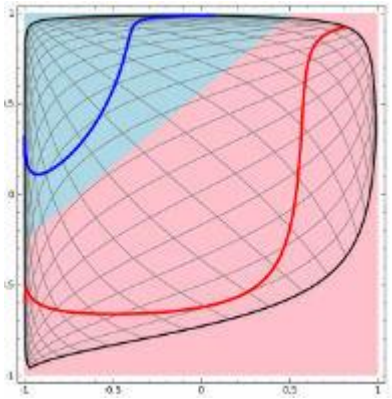


Figure 2-23.
Transformed data space

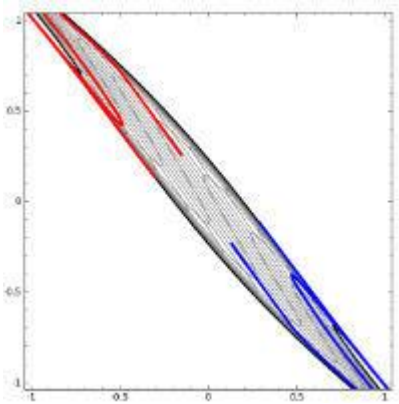


Figure 2-24.
Stronger transformation

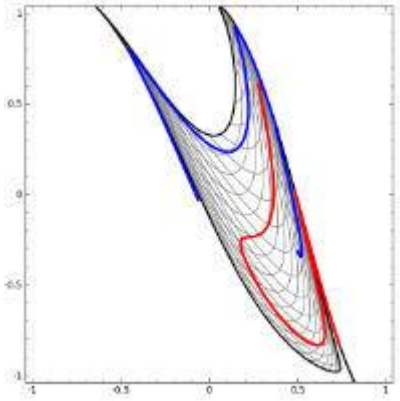


Figure 2-25.
Failed transformation

2.1.2.5 Think twice

- Why should it be deeper?

  The most direct explanation is that the number of critical point varies with the size of network which is proportional to

$$\sqrt{width} \times (depth)^{width/2}$$

  Therefore, the increase of width leads to explosive critical points while the increase of depth outcomes a slower increasing.

2.1.2.5 Think twice

- Is that OK?

2.2.1 Recurrent Neural Networks

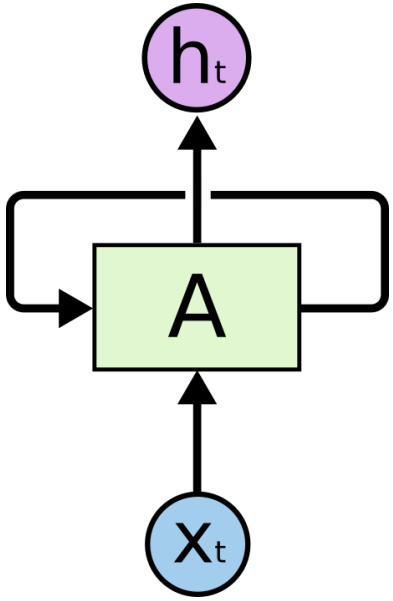


Figure 2-26. Recurrent Neural Networks

Understanding LSTM Networks

2.2.1 Recurrent Neural Networks

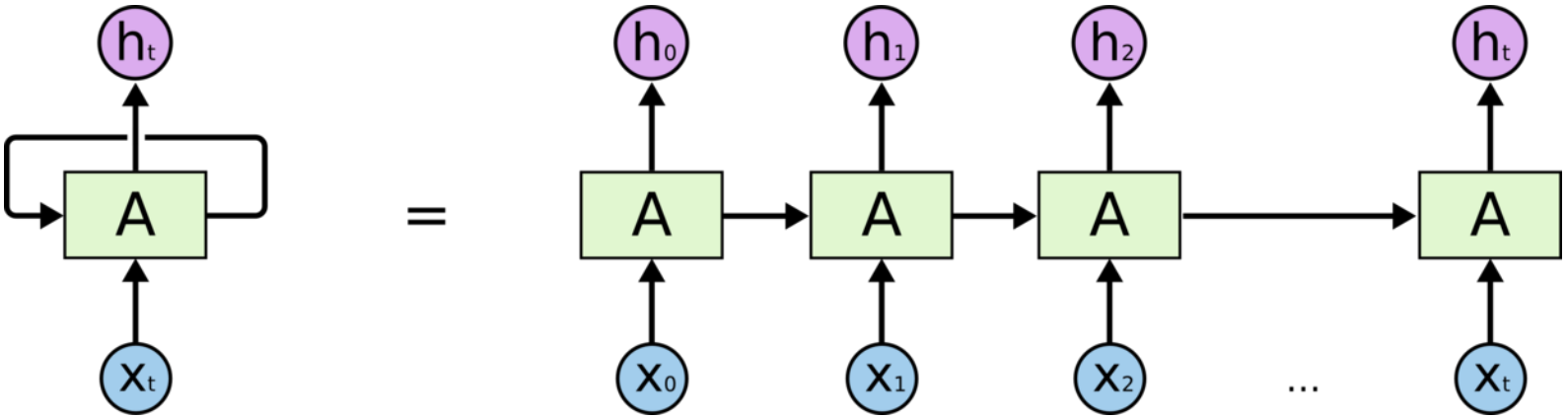


Figure 2-27. An unrolled recurrent neural network

2.2.1 Long-Term Dependency Problem

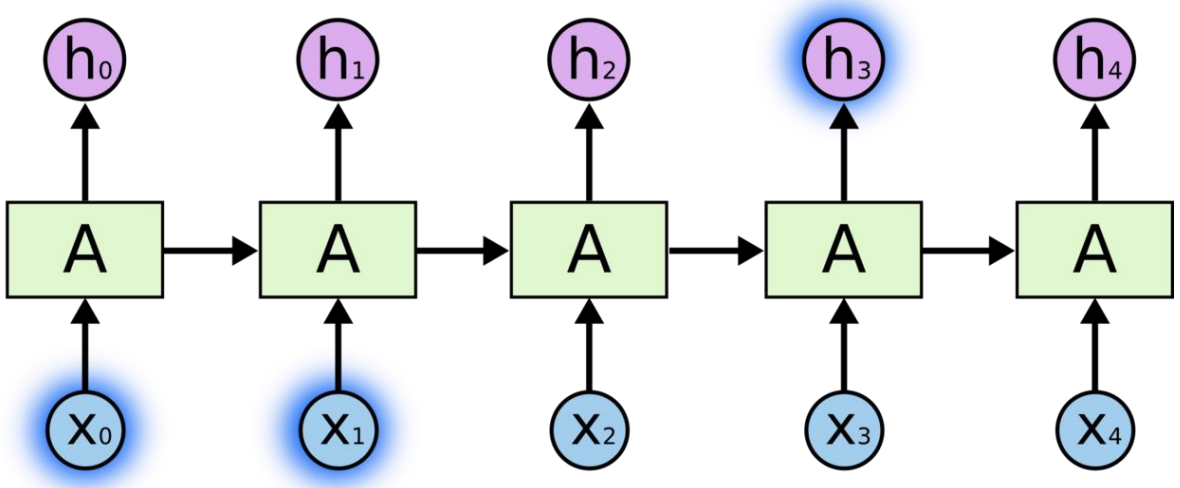


Figure 2-28. Short-term dependency

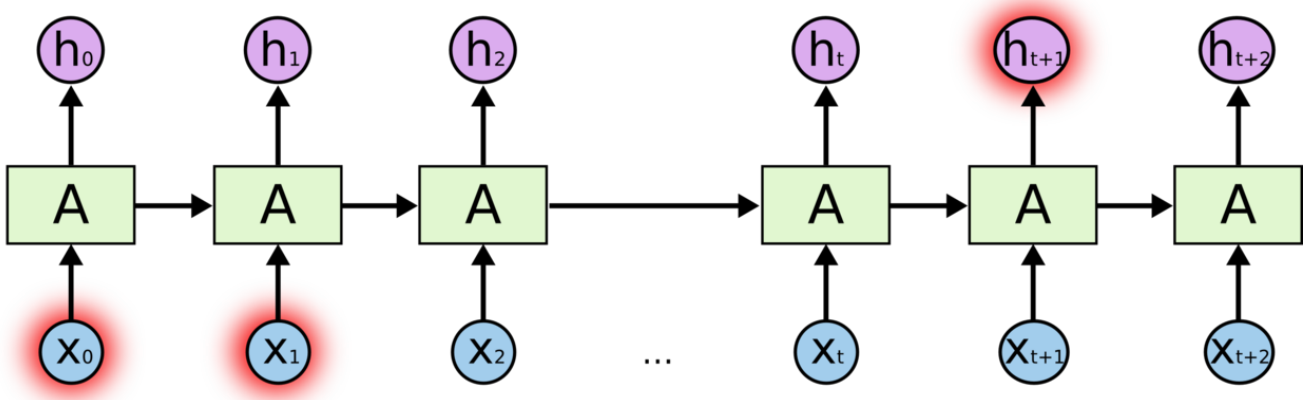


Figure 2-29. Long-term dependency

2.2.2 Long Short Term (LSTM)

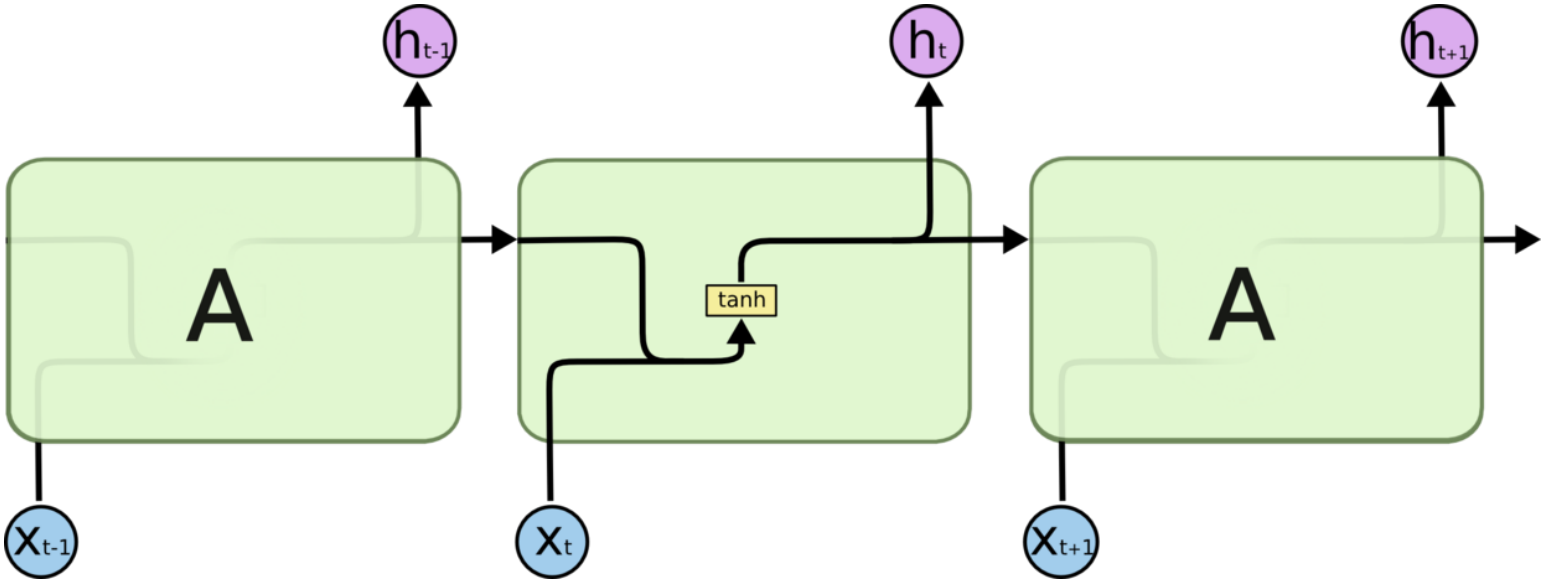


Figure 2-30. Standard RNN with single layer

2.2.2 Long Short Term (LSTM)

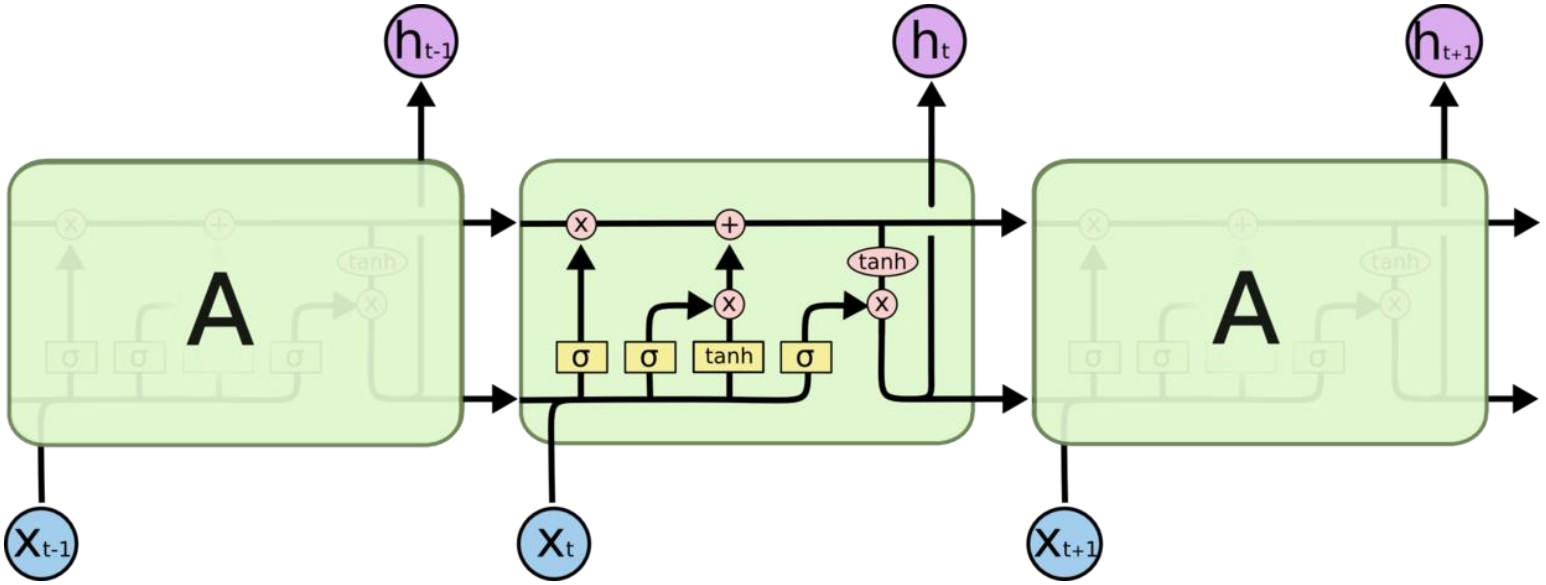


Figure 2-31. Well designed Long Short Term

2.2.2 Illustrate LSTM Structure

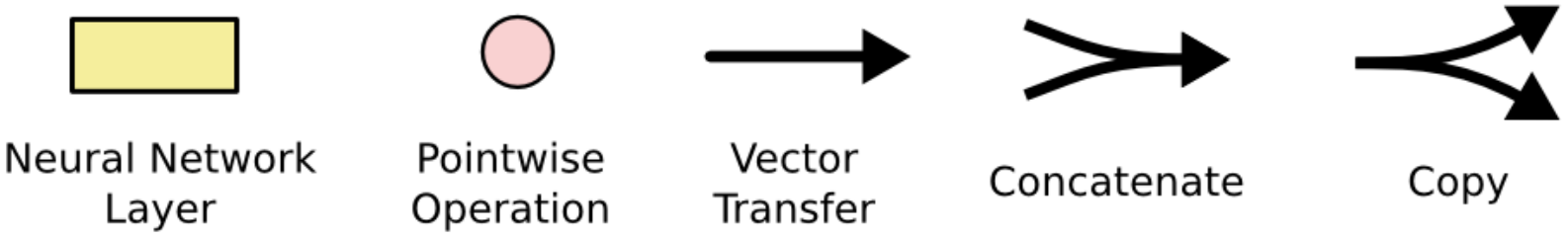


Figure 2-32. Corresponding meaning of diagram

2.2.2 Core Idea behind LSTM

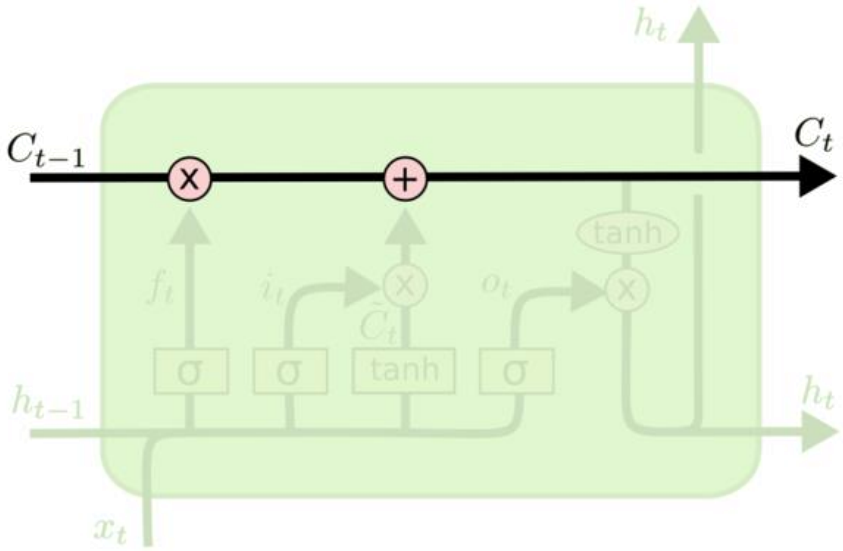


Figure 2-33. the cell state horizontally running through the top of the diagram

2.2.2 Illustrate LSTM Structure

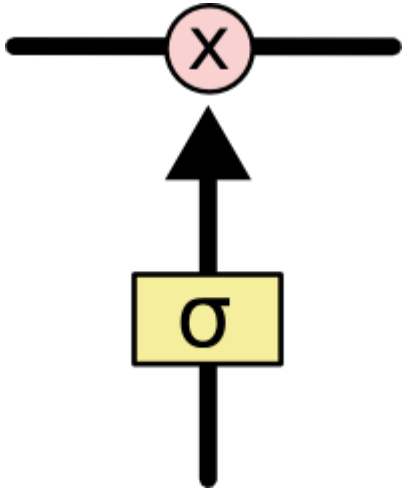
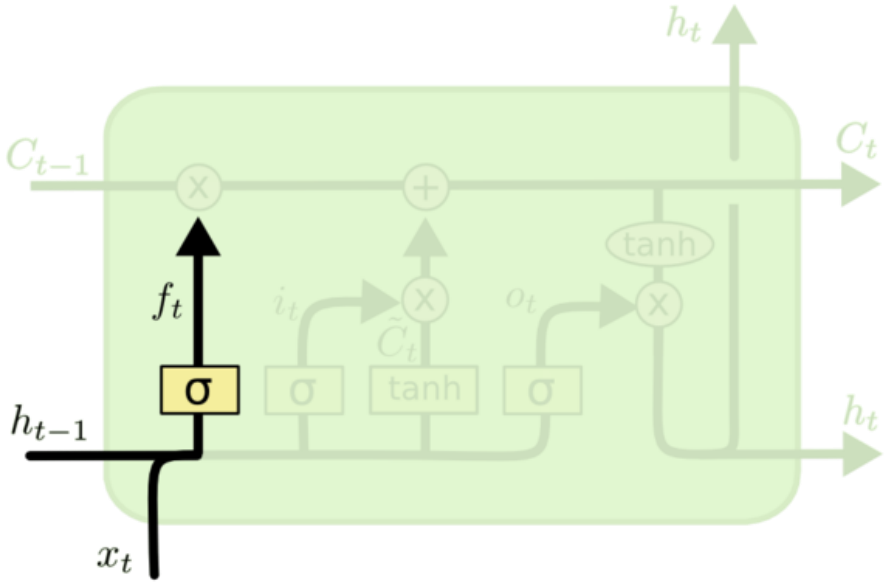


Figure 2-34. The gates structures

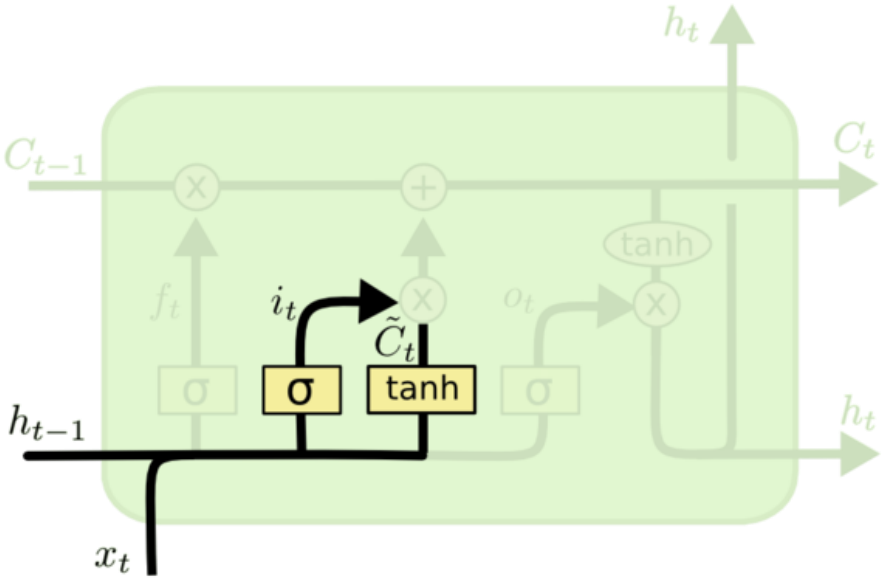
2.2.2 Illustrate LSTM Structure



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure 2-35. The forget gate layer of LSTM

2.2.2 Illustrate LSTM Structure

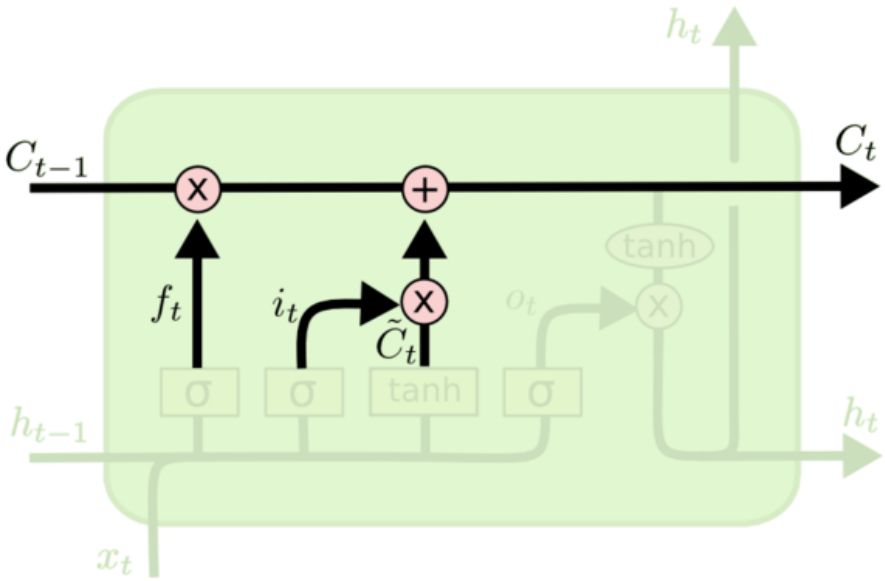


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure 2-36. The input gate layer of LSTM

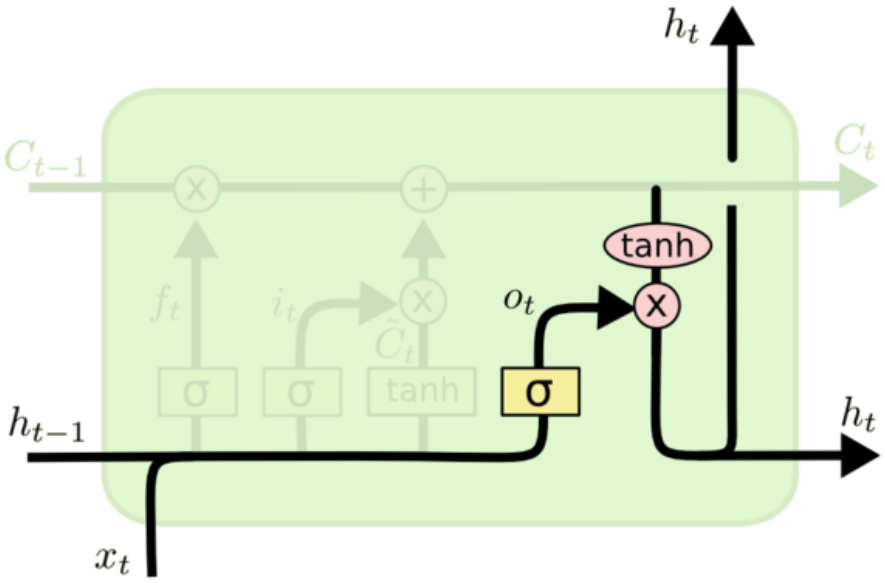
2.2.2 Illustrate LSTM Structure



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 2-37. The implement gate layer of LSTM

2.2.2 Illustrate LSTM Structure



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 2-38. The output gate layer of LSTM

2.2.2 Long Short Term (LSTM)

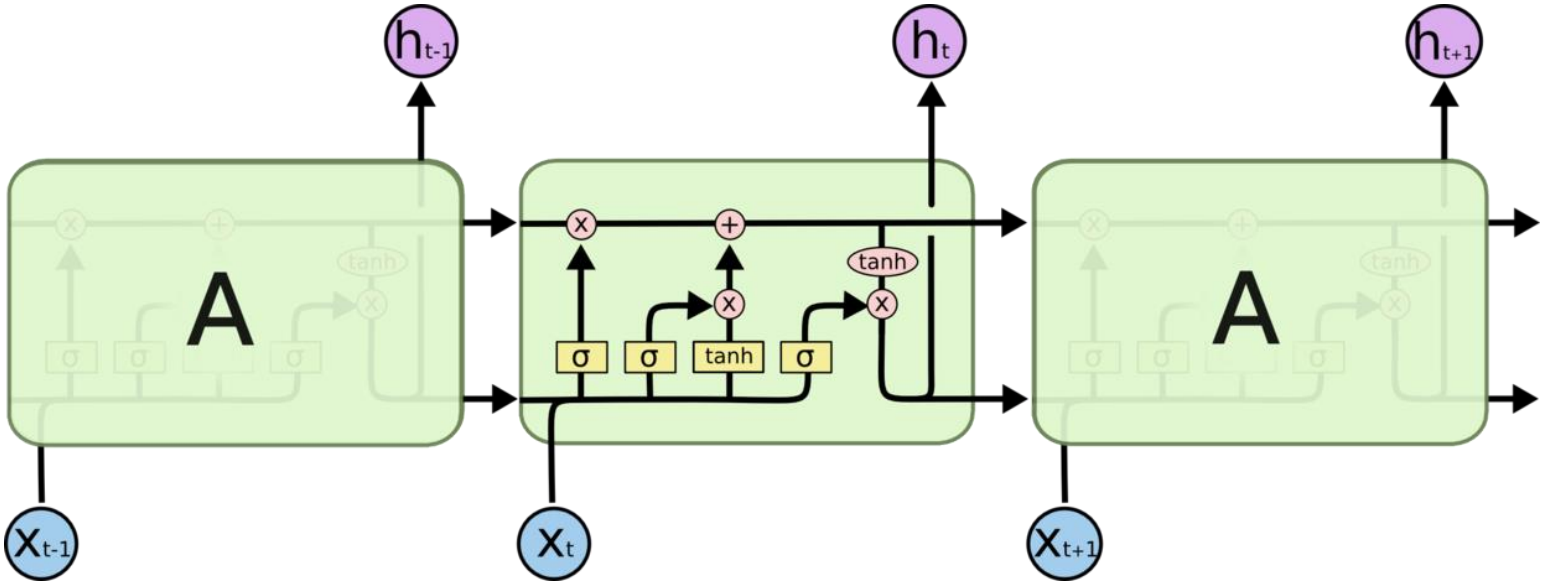


Figure 2-39. Well designed Long Short Term

3.0 Illuminating Works (挖坑专题)

1. Understanding CNN
2. Neural Turing Machines (NTM)
3. Learning to learn
4. Center Loss
5. Matrix Completion under Self-Expressive Models
6. Generalized Similarity Measure

3.1 Understanding CNN

Figure 1. A simple modification of the global average pooling layer combined with our class activation mapping (CAM) technique allows the classification-trained CNN to both classify the image and localize class-specific image regions in a single forward-pass e.g., the toothbrush for *brushing teeth* and the chainsaw for *cutting trees*.

Figure 3-1. The result of CAM

Zhou B, Khosla A, Lapedriza A, et al. *Learning Deep Features for Discriminative Localization*[J]. arXiv preprint arXiv:1512.04150, 2015.

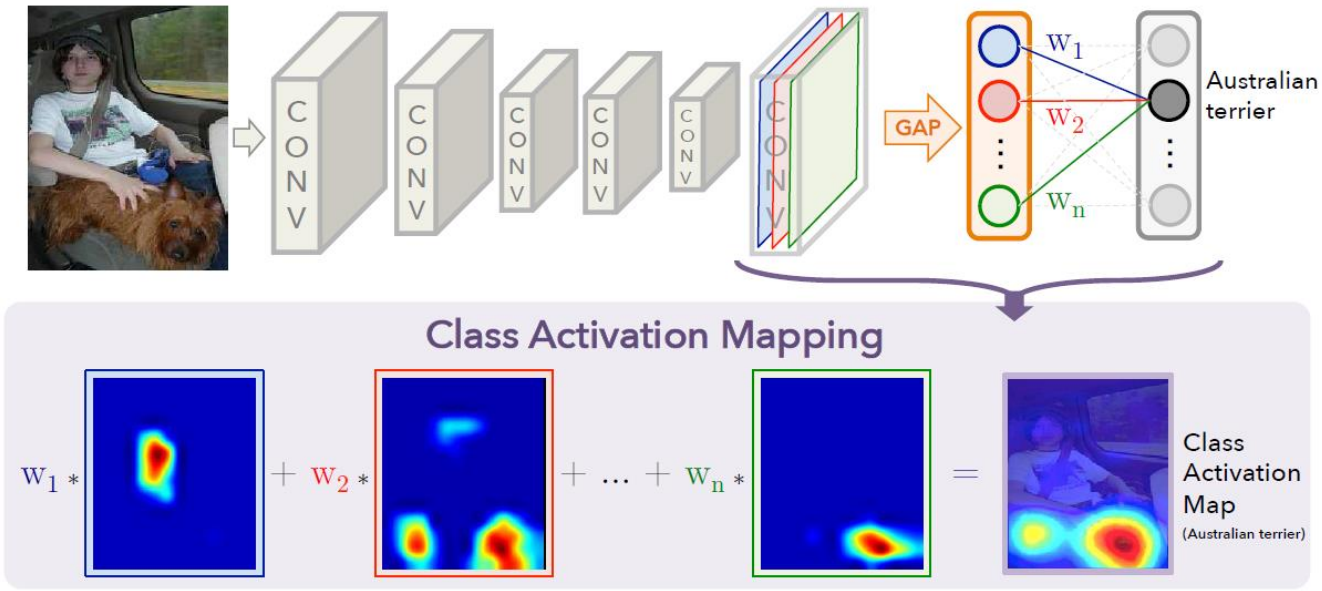


Figure 3-2. The constitute of CAM

The input to softmax: $$S_c = \sum_k w_k^c \sum_{x,y} f_k(x, y) = \sum_{x,y} \sum_k w_k^c f_k(x, y)$$

The CAM definition: $$M_c(x, y) = \sum_k w_k^c f_k(x, y)$$

3.2 Neural Turing Machines

Recurrent neural networks (RNNs) stand out from other machine learning methods for their ability to learn and carry out complicated transformations of data over extended periods of time.

Moreover, it is known that RNNs are Turing-Complete (Siegelmann and Sontag, 1995), and therefore have the capacity to simulate arbitrary procedures, if properly wired. Yet what is possible in principle is not always what is simple in practice.

Graves A, Wayne G, Danihelka I. *Neural turing machines*[J]. arXiv preprint arXiv:1410.5401, 2014.

3.2 Neural Turing Machines

We extend the capabilities of neural networks by coupling them to external memory resources, which they can interact with by attentional processes.

The combined system is analogous to a Turing Machine or Von Neumann architecture but is differentiable end-toend, allowing it to be efficiently trained with gradient descent. Preliminary results demonstrate that Neural Turing Machines can infer simple algorithms such as copying, sorting, and associative recall from input and output examples.

3.2 Neural Turing Machines

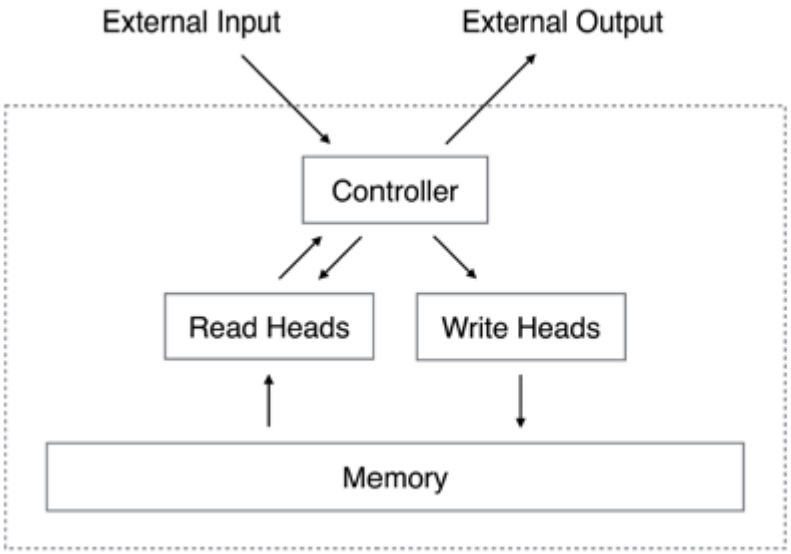


Figure 3-3. The structure of NTM

The copy task tests whether NTM can store and recall a long sequence of arbitrary information.

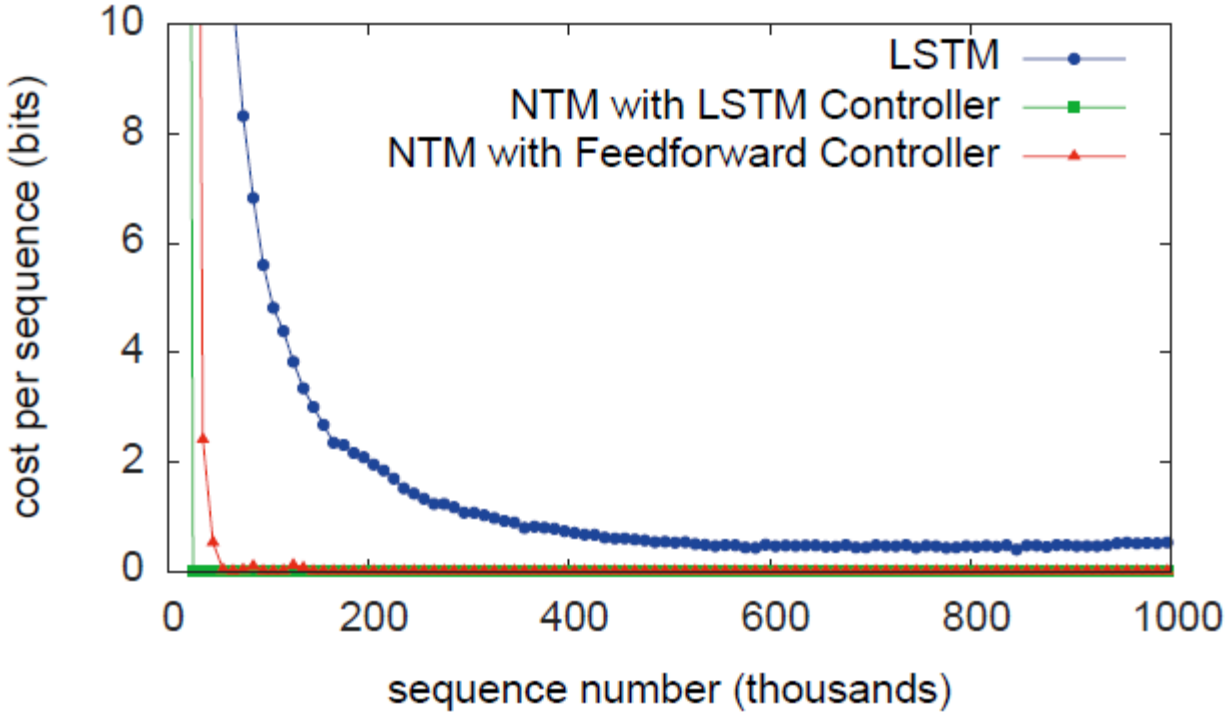
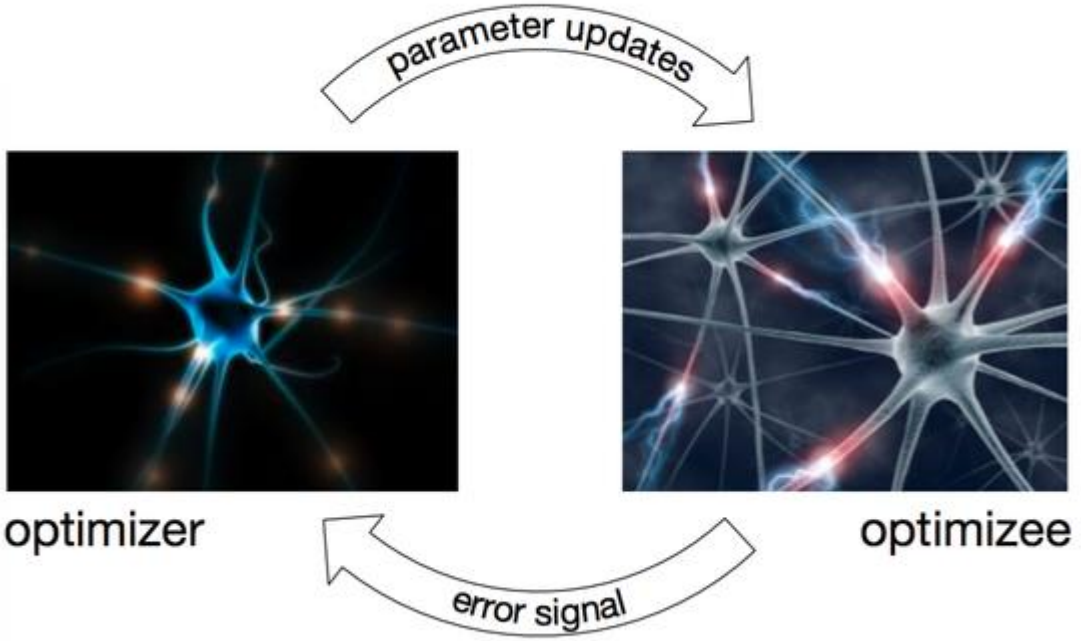


Figure 3-4. Copy learning curve

3.3 Learning to learn



Replace

$$\theta_{t+1} = \theta_t - \alpha \nabla f(\theta_t)$$

by

$$\theta_{t+1} = \theta_t + g_t(\nabla f(\theta_t), \phi)$$

Figure 3-5.
The idea of learning to learn

Andrychowicz M, Denil M, Gomez S, et al. *Learning to learn by gradient descent by gradient descent*[C]. Advances In Neural Information Processing Systems. 2016: 3981-3989.

3.3 Learning to learn

How is it trained?

——Train an optimizer on a simple class of synthetic 10-dimensional quadratic functions.

$$f(\theta) = \|W\theta - y\|_2^2$$

What is it used to do?

—— Train other neural networks, such as neural network for MNIST, neural network for CIFAR-10 and neural network for Neural Art and perform wonderfully.

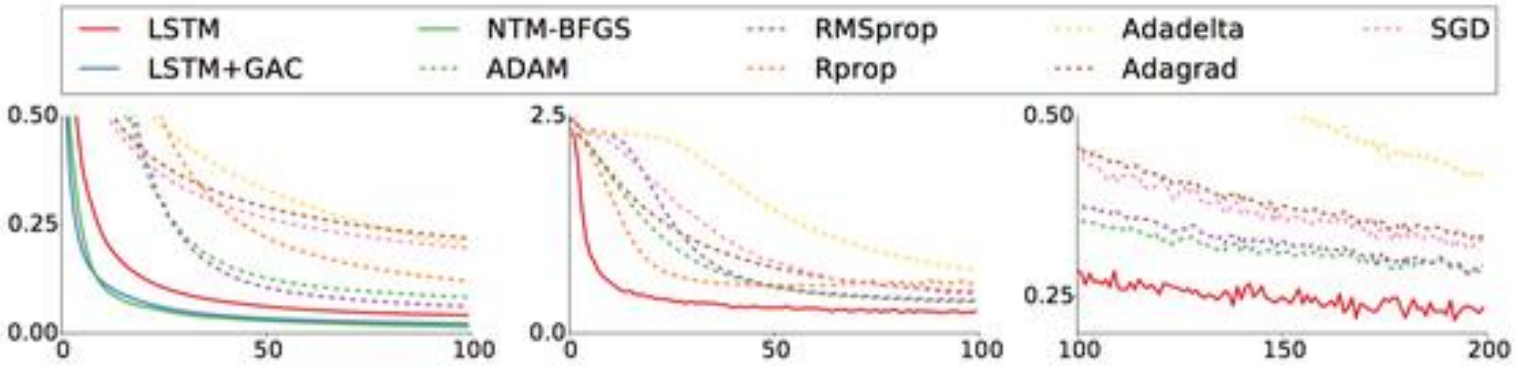


Figure 3-6.
The learning curve of learning to learn

NAODONG！！！

$$\arg\min_{A,S} \|X - AS\|_2^2$$
$$s.t. \quad A, S \in \mathbb{R}_+$$

Now that the updating has been the result of learning, can the objective function/assumption + metric criterion design also be the propose of learning to learn?

Idea is cheap…

Marblestone A H, Wayne G, Kording K P. *Toward an integration of deep learning and neuroscience*[J]. Frontiers in Computational Neuroscience, 2016, 10.

3.4 Center Loss

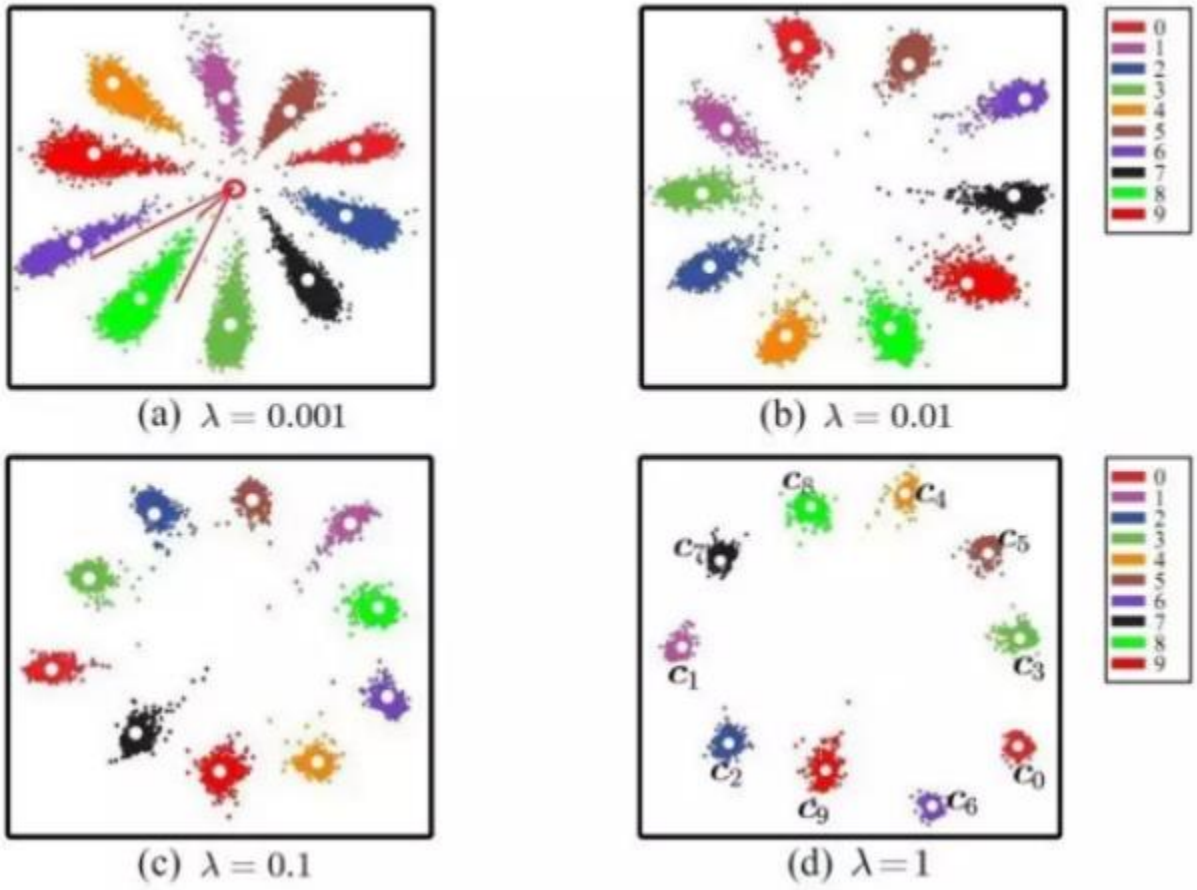


Figure 3-7.
The result with center loss

$$L = L_s + \lambda L_C$$

$$= -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_{y_j}}} + \frac{\lambda}{2} \sum_{i=1}^{m} \left\| x_i - c_{y_i} \right\|_2^2$$

Wen Y, Zhang K, Li Z, et al. *A Discriminative Feature Learning Approach for Deep Face Recognition*[C]. European Conference on Computer Vision. Springer International Publishing, 2016: 499-515.

3.5 MC under Self-Expressive Models

The solution of our algorithms recover missing entries and provides a similarity matrix for clustering.

Our algorithms can deal with both low-rank and high-rank matrices, does not suffer from initialization, does not need to know dimensions of subspaces and can work with a small number of data points.

Elhamifar E. *High-Rank Matrix Completion and Clustering under Self-Expressive Models*[C]. Advances In Neural Information Processing Systems. 2016: 73-81.

3.6 Generalized Similarity Measure

Resulting in a novel generalized similarity measure, defined as:

$$S(x, y) = [x^T \; y^T \; 1] \begin{bmatrix} A & C & d \\ C^T & B & e \\ d^T & e^T & f \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Our similarity model [1] can be viewed as a generalization of several recent metric learning models [2] [3].

[1] Lin, L., Wang, G., Zuo, W., Xiangchu, F., & Zhang, L. (2016). *Cross-Domain Visual Matching via Generalized Similarity Measure and Feature Learning*.
[2] Z. Li, S. Chang, F. Liang, T. S. Huang, L. Cao, and J. R. Smith, "Learning locally-adaptive decision functions for person verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, 2013, pp.3610–3617.
[3] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, "Bayesian face revisited: A joint formulation," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2012, pp. 566–579.

From linear to affine transformation:

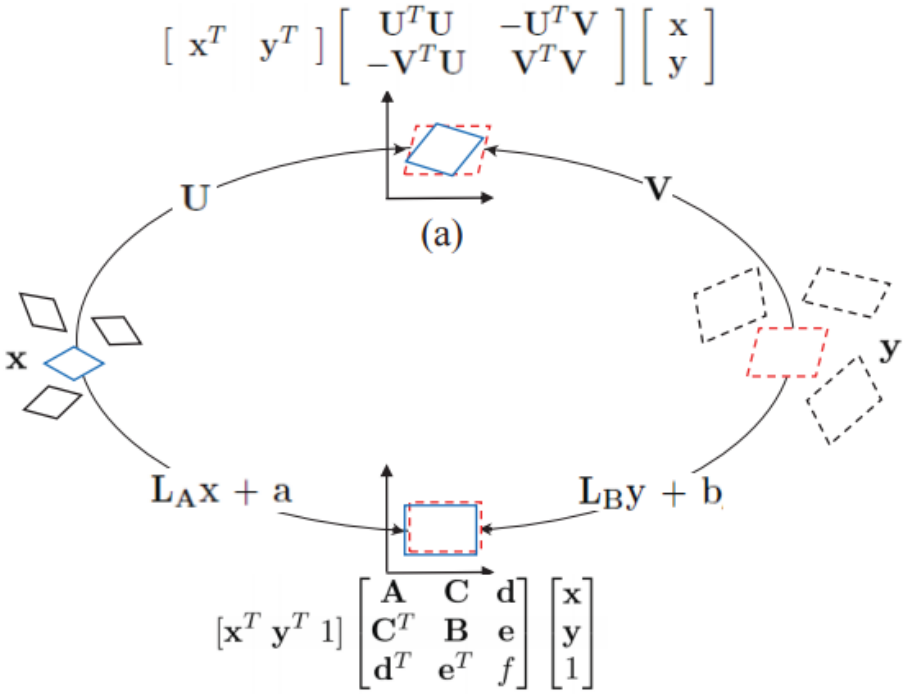


Figure 3-8.
The generalized similarity measure

3.6 Generalized Similarity Measure

To carry on deep learning framework:

$$\tilde{S}(x, y) = S(f_1(x), f_2(y))$$

$$= [f_1(x)^T \; f_2(y)^T \; 1] \begin{bmatrix} A & C & d \\ C^T & B & e \\ d^T & e^T & f \end{bmatrix} \begin{bmatrix} f_1(x) \\ f_2(y) \\ 1 \end{bmatrix}$$

$$= \left\| L_A f_1(x) \right\|^2 + \left\| L_B f_2(y) \right\|^2 + 2d^T f_1(x) - 2(L_C^x f_1(x))^T (L_C^x f_2(x)) + 2e^T f_2(y) + f$$

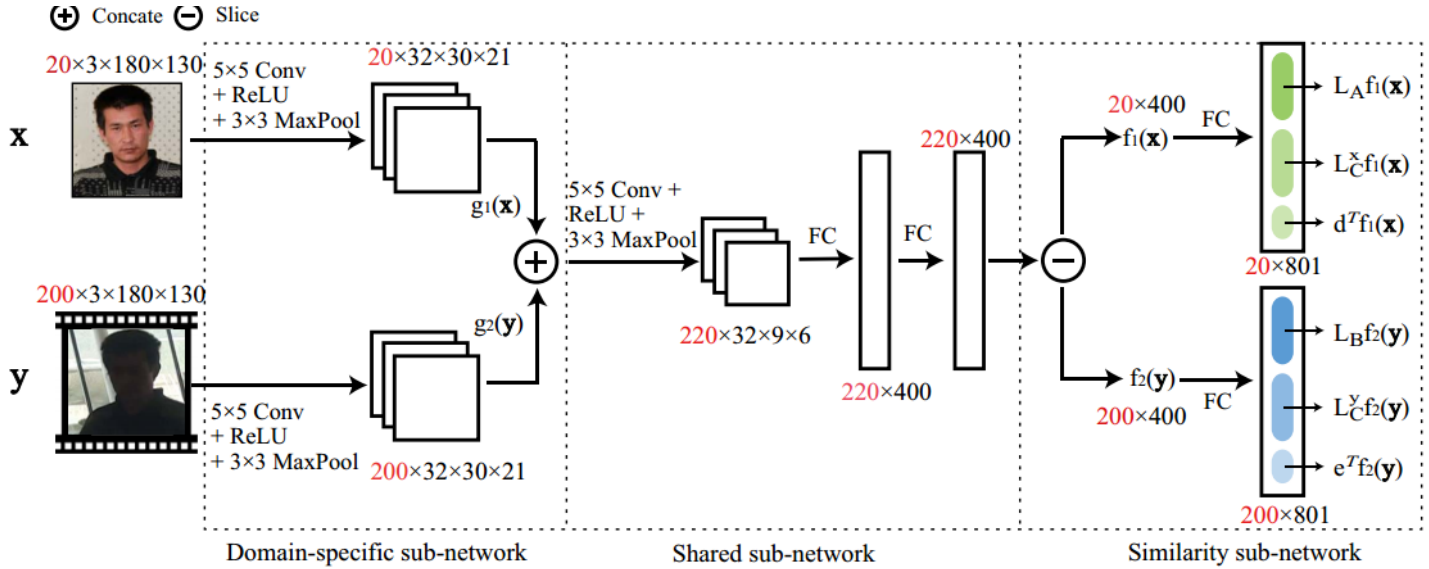
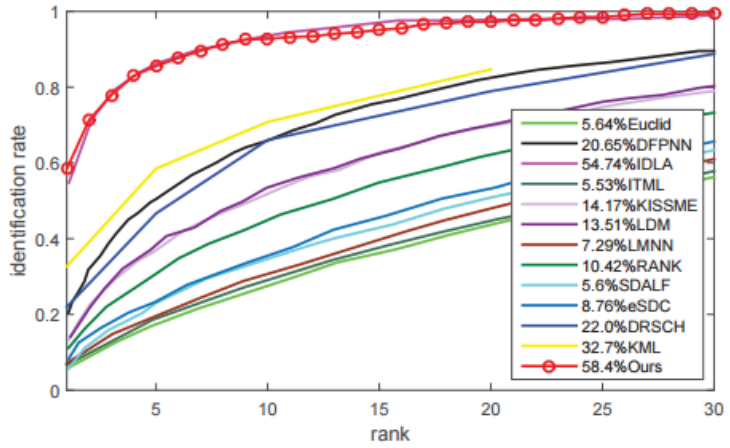


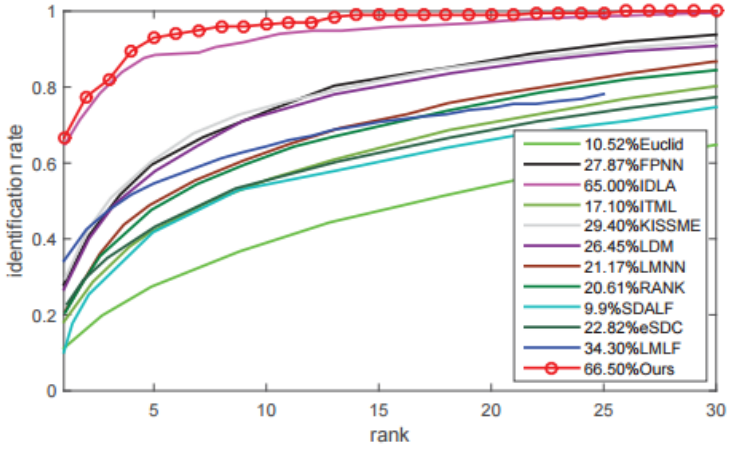
Figure 3-9. The generalized similarity measure

3.6 Generalized Similarity Measure

Result in person re-identification:



(a) CUHK03



(b) CUHK01

Figure 3-10. The evaluation in person re-identification

Thanks