A “Quick” Introduction to Machine Learning

Outline

- **Introduction & Methodology**
- MLP: Feed Forward & Back Propagation
- Further Discussion
  - Model Architecture: Convolutional Neural Network
  - Loss Function: Regression or Classification
  - Optimization: Stochastic Gradient Descent

Find an algorithm to this Question!

Input

Output: Mountain / Sky / Water

NO, YES, YES; **YES, YES, NO;**

YES, YES, YES; **NO, NO, YES.**

Machines could tackle problems with deterministic solution

程序设计基础_最长上升子序列

问题描述

给定一个长为 n 的序列，求它的最长上升子序列的长度。

输入格式

输入第一行包含一个整数 n。第二行包含 n 个整数，为给定的序列。

输出格式

输出一个非负整数，表示最长上升子序列的长度。

样例输入

```
5
1 3 2 5 4
```

样例输出

```
3
```

数据规模和约定

- $0 < n \leq 1000$
- 每个数不超过 10^6

CST2021F 1-1 A+B problem

描述

抱歉，这题实际是 A*B problem。

邓俊辉老师的作业常常过于简单，数据类型只需使用 int。助教们一致认为，向同学们介绍 Python 中自带的长整型是十分有必要的。例如，它可以计算几百位的整数乘法。 但是，在介绍长整型之前，助教决定让你自己实现一遍长整型乘法，以加深对它的理解。

输入

输入共包含 n + 1 行，第 1 行包含一个整数 n，表示你需要计算 n 组乘法。

接下来 n 行，每行包含两个非负整数 a 和 b。

输出

输出共包含 n 行，请对于每一组输入的 a、b，输出他们的乘积。

输入样例

```
3
1 1
2 2
123123 789789
```

*此样例是第 1 个测试点。

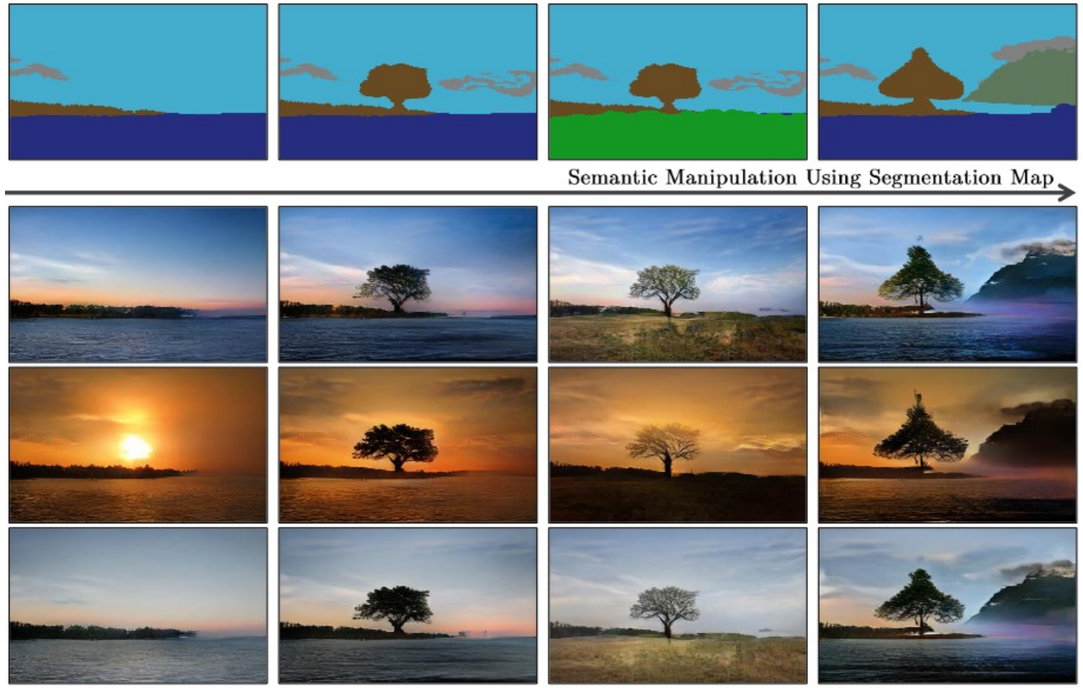
*本课堂的编程作业中，对于非全 int 输入的题目，有的会把一个样例作为第 1 个测试点方便调试。

That means, given <u>a particular input</u>, the algorithm will always produce <u>the same output</u>.

But what about those problems?

1. Problems with deterministic solution but not empirically solvable

- Classification problems



- Generation problems

2. Problems even without a deterministic solution …

- What shall we eat tonight?

- It's up to you.

> It may depend on the context…?

We learn to know the world, so do machines!

But, how can we teach machines to learn?

Role of AI Scientists: teach AI how to learn.

1. Hand-crafted features

E.g. You want to build a Chat-bot …
- If there is "turn off" in the input, then "turn off the music" (hand-crafted rules)
  - You can say "Please turn off the music" or "Can you turn off the music?". Smart?
  - What if someone says "Please don't turn off the music" ……

Weaknesses:
- They need domain specific knowledge of human
- They can never surpass their human teachers

But, how can we teach machines to learn?

2. Numerical Solution

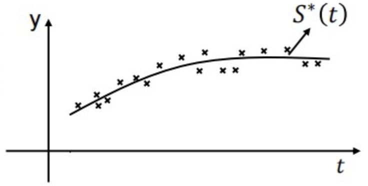
➢ We fit拟合 on data !

What is a fit problem?

曲线拟合问题

实例演示Matlab, Excel

- **Motivation**
  - 发现数据的规律, "回归分析"
  - 由于数据可能有误差, 逼近曲线不必通过所有点



- 问题描述
  - 数据点(t_i, f_i), $(i=1,\cdots,m)$, 用含参数的函数$S(t)$来拟合
  - 拟合要求: $\sum_{i=1}^{m}[S(t_i)-f_i]^2$最小, "最小二乘"　几何意义?
  - 通常 $m>n$ 若$S(t)\in\Phi$, $S(t)=\sum_{j=1}^{n}x_j\varphi_j(t)$, 求系数$x_j$, 线性最小二乘
  - 定义在离散点$\{t_i\}$上的表格函数构成线性空间 是一种最佳平方逼近!
    (空间元素可用离散点函数值构成的向量表示)

Wenjian Yu　20

【例 1】有一位同学家开了一个小卖部，他为了研究气温对热饮销售的影响，经过统计，得到一个卖出热饮杯数与当天气温的对比表：

摄氏温度(℃)	−5	0	4	7	12	15	19	23	27	31	36
热饮杯数	156	150	132	128	130	116	104	89	93	76	54

（1）画出散点图；

（2）你能从散点图中发现气温与热饮销售杯数之间关系的一般规律吗？

（3）求回归方程；

（4）如果某天的气温是 2℃，预测这天卖出的热饮杯数.

解：（1）散点图如图 1−9−8 所示.

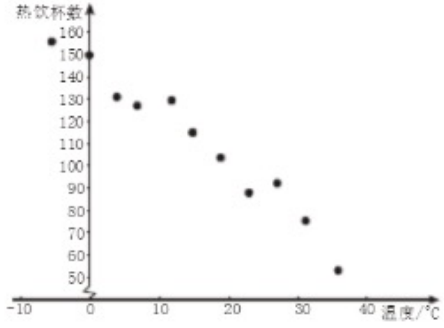


图 1−9−8

（2）从图 1−9−8 中看到，各点散布在从左上角到右下角的区域里，因此，气温与热饮销售杯数之间成负相关，即气温越高，卖出去的热饮杯数越少.

（3）从散点图可以看出，这些点大致分布在一条直线的附近，因此，可用公式①求出回归方程的系数.

利用计算器容易求得回归方程为 $\hat{y}=-2.352x+147.767$.

（4）当 $x=2$ 时，$\hat{y}=143.063$. 因此，某天的气温为 2℃时，这天大约可以卖出 143 杯热饮.

We teach machines to learn **numerically**!

(1) Define a mathematical **model**.

e.g. y=ax+b

(2) Determine an approach to evaluate which model is **better**

Namely we define a ***loss function*** *for each model*

(3) Propose an optimization method to find the “best” model

Go from the current model parameters to ***better*** *ones:* ***LEARNING****!!!*

What is learning?

We teach **models** to learn numerically!

【例 1】有一位同学家开了一个小卖部，他为了研究气温对热饮销售的影响，经过统计，得到一个卖出热饮杯数与当天气温的对比表：

摄氏温度(℃)	−5	0	4	7	12	15	19	23	27	31	36
热饮杯数	156	150	132	128	130	116	104	89	93	76	54

（1）画出散点图；

（2）你能从散点图中发现气温与热饮销售杯数之间关系的一般规律吗？

（3）求回归方程；

（4）如果某天的气温是 2℃，预测这天卖出的热饮杯数.

解：（1）散点图如图 1−9−8 所示.

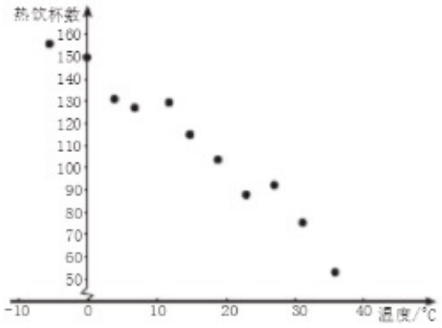


图 1−9−8

（2）从图 1−9−8 中看到，各点散布在从左上角到右下角的区域里，因此，气温与热饮销售杯数之间成负相关，即气温越高，卖出去的热饮杯数越少.

（3）从散点图可以看出，这些点大致分布在一条直线的附近，因此，可用公式①求出回归方程的系数.

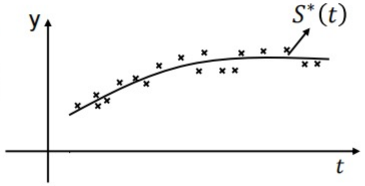
利用计算器容易求得回归方程为 $\hat{y}=-2.352x+147.767$.

（4）当 $x=2$ 时，$\hat{y}=143.063$. 因此，某天的气温为 2℃时，这天大约可以卖出 143 杯热饮.

曲线拟合问题

实例演示Matlab, Excel

- **Motivation**
  - 发现数据的规律, “回归分析”
  - 由于数据可能有误差, 逼近曲线不必通过所有点



- **问题描述**
  - 数据点(t_i, f_i), $(i=1,\cdots,m)$, 用含参数的函数$S(t)$来拟合
  - 拟合要求: $\sum_{i=1}^{m}[S(t_i)-f_i]^2$最小, “最小二乘” 几何意义?
  - 若$S(t)\in\Phi$, $S(t)=\sum_{j=1}^{n} x_j\varphi_j(t)$, 求系数$x_j$, 线性最小二乘
  - 定义在离散点$\{t_i\}$上的表格函数构成线性空间 是一种最佳平方逼近!
    (空间元素可用离散点函数值构成的向量表示)

通常 $m > n$

$y = a + bx$，其中

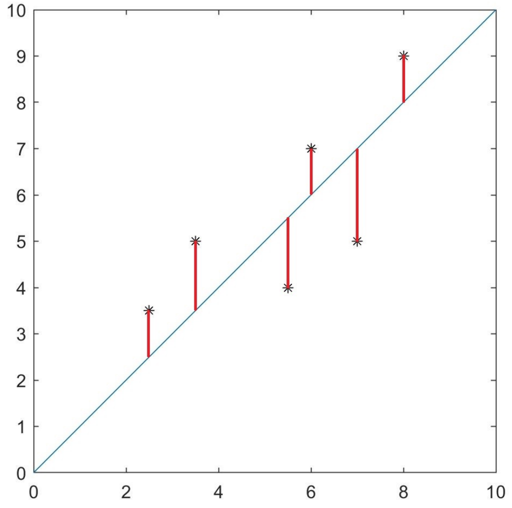
$$b=\frac{(x_1-\bar{x})(y_1-\bar{y})+(x_2-\bar{x})(y_2-\bar{y})+\ldots\ldots+(x_n-\bar{x})(y_n-\bar{y})}{(x_1-\bar{x})^2+(x_2-\bar{x})^2+\ldots\ldots+(x_n-\bar{x})^2}$$

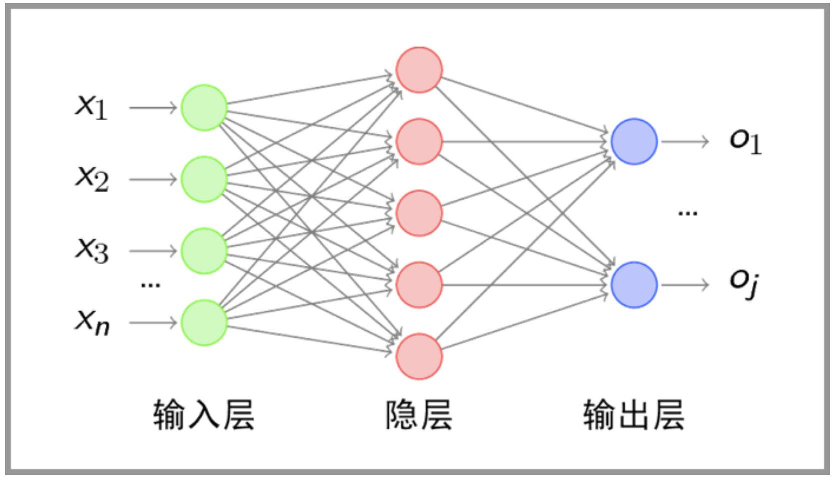
$$=\frac{x_1y_1+x_2y_2+\ldots\ldots+x_ny_n-n\bar{x}\bar{y}}{x_1^2+x_2^2+\ldots\ldots+x_n^2-n\bar{x}^2}$$

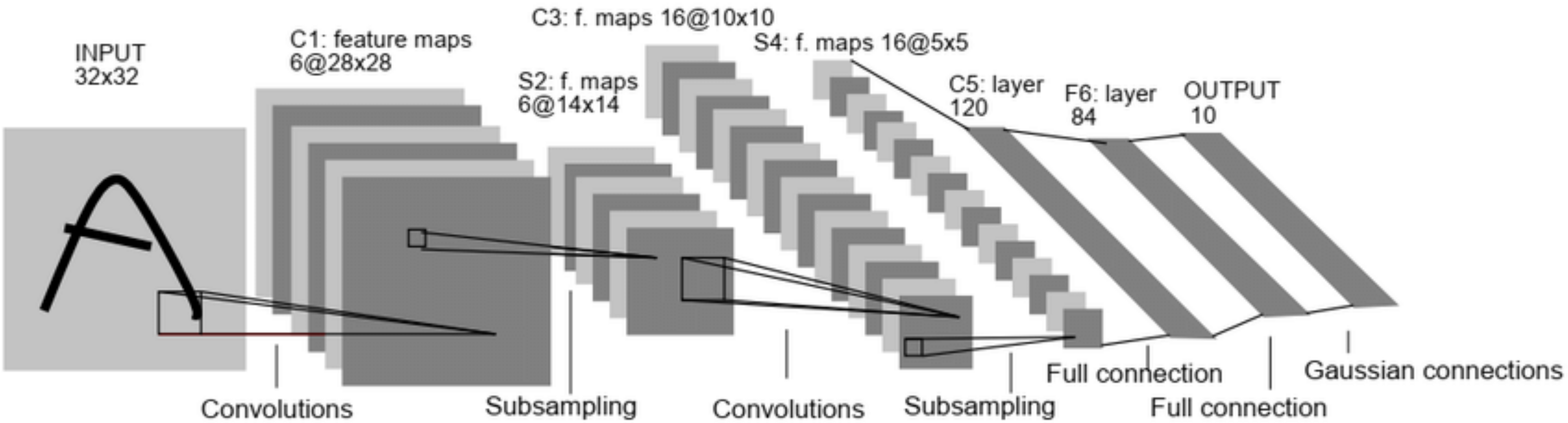
$$=\frac{\sum_{i=1}^{n}(x_iy_i-\bar{x}\bar{y})}{\sum_{i=1}^{n}(x_i^2-\bar{x}^2)}$$

$$a=\bar{y}-b\bar{x}$$

1. We define model: ability to express







2. We define loss function: which model is better

- Loss function: the “difference” of current model to the ideal one

- L1 Loss: **| current_result – ideal_result |**
- L2 Loss: **(current_result – ideal_result)^2**
- Cross Entropy Loss
  - To measure the difference between two possibility distributions

$$H(p,q)=\sum_{x} p(x) \cdot log\left(\frac{1}{q(x)}\right)$$

3. We propose optimization methods

线性最小二乘 – 法方程法

- 法方程方法: 求解$\boldsymbol{Gx}=\boldsymbol{b}$

$$G=\begin{bmatrix}\langle\varphi_1,\varphi_1\rangle & \langle\varphi_2,\varphi_1\rangle & \cdots & \langle\varphi_n,\varphi_1\rangle\\ \langle\varphi_1,\varphi_2\rangle & \langle\varphi_2,\varphi_2\rangle & \cdots & \langle\varphi_n,\varphi_2\rangle\\ \vdots & \vdots & \ddots & \vdots\\ \langle\varphi_1,\varphi_n\rangle & \langle\varphi_2,\varphi_n\rangle & \cdots & \langle\varphi_n,\varphi_n\rangle\end{bmatrix},\quad b=\begin{bmatrix}\langle f,\varphi_1\rangle\\ \langle f,\varphi_2\rangle\\ \vdots\\ \langle f,\varphi_n\rangle\end{bmatrix}$$

$$A=\begin{bmatrix}\varphi_1(t_1) & \varphi_2(t_1) & \cdots & \varphi_n(t_1)\\ \varphi_1(t_2) & \varphi_2(t_2) & \cdots & \varphi_n(t_2)\\ \vdots & \vdots & \vdots & \vdots\\ \varphi_1(t_m) & \varphi_2(t_m) & \cdots & \varphi_n(t_m)\end{bmatrix}\quad A=[\boldsymbol{\varphi}_1\ \ \boldsymbol{\varphi}_2\ \ \cdots\ \ \boldsymbol{\varphi}_n]\quad A^T=\begin{bmatrix}\boldsymbol{\varphi}_1^T\\ \boldsymbol{\varphi}_2^T\\ \vdots\\ \boldsymbol{\varphi}_n^T\end{bmatrix}\Rightarrow G=A^TA,\ b=A^Tf$$

算法6.2

- 形成矩阵A; 得到$A^TAx=A^Tf$; 解之 → A列满秩
- 若表格函数$\varphi_1(t),\ldots,\varphi_n(t)$线性无关, 法方程存在唯一解

线性最小二乘

Excel例子

- 例6.5: 一组数据如下表, 用适当的函数对它们进行拟合

t_i	1.00	1.25	1.50	1.75	2.00
y_i	5.10	5.79	6.53	7.45	8.46
$\tilde{y}_i$	1.6292	1.7561	1.8764	2.0082	2.1353

(非线性拟合问题怎么用线性最小二乘?)

- 解: 在直角坐标系里绘出这些数据点, 根据其分布趋势, 采用指数函数来描述: $y\approx x_1e^{x_2t}$ ⇔ $\tilde{y}=\tilde{x}_1+x_2t$
不能直接用线性最小二乘, 需做变换$\ln y\approx\ln x_1+x_2t$
线性拟合基函数$\varphi_1(t)=1,\ \varphi_1(t)=t$

$$A=\begin{bmatrix}1 & 1.00\\ 1 & 1.25\\ 1 & 1.5\\ 1 & 1.75\\ 1 & 2.00\end{bmatrix},\ f=\begin{bmatrix}1.6292\\ 1.7561\\ 1.8764\\ 2.0082\\ 2.1353\end{bmatrix}$$

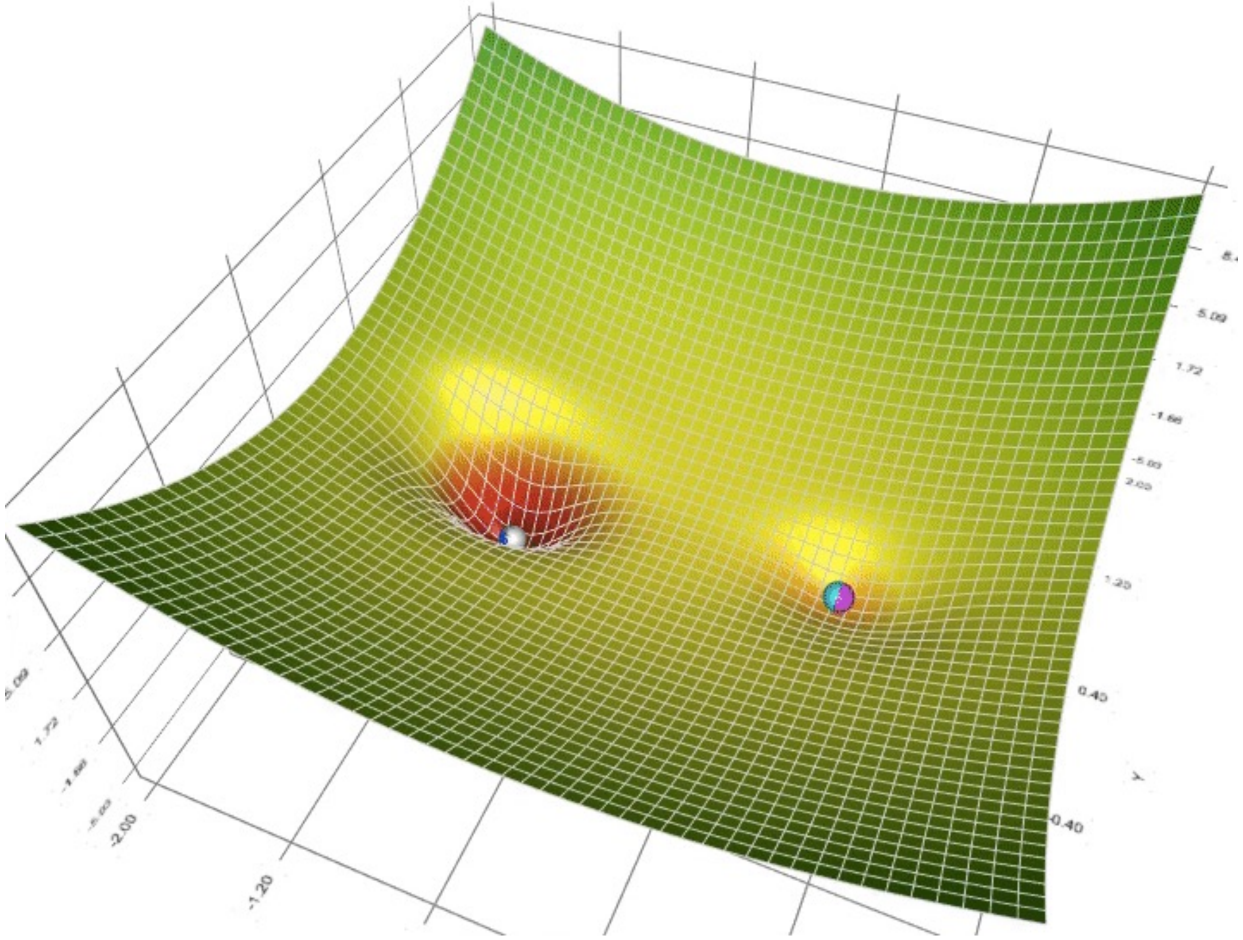
要解$A^TAx=A^Tf$, 即

$$\begin{bmatrix}5 & 7.5\\ 7.5 & 11.875\end{bmatrix}\begin{bmatrix}\tilde{x}_1\\ x_2\end{bmatrix}=\begin{bmatrix}9.4052\\ 14.4239\end{bmatrix}$$

解得: $\begin{bmatrix}\tilde{x}_1\\ x_2\end{bmatrix}=\begin{bmatrix}1.1225\\ 0.5057\end{bmatrix}$

⇒ $x_1=e^{\tilde{x}_1}=3.0725$

最后的解为$S(t)=3.0725e^{0.5057t}$



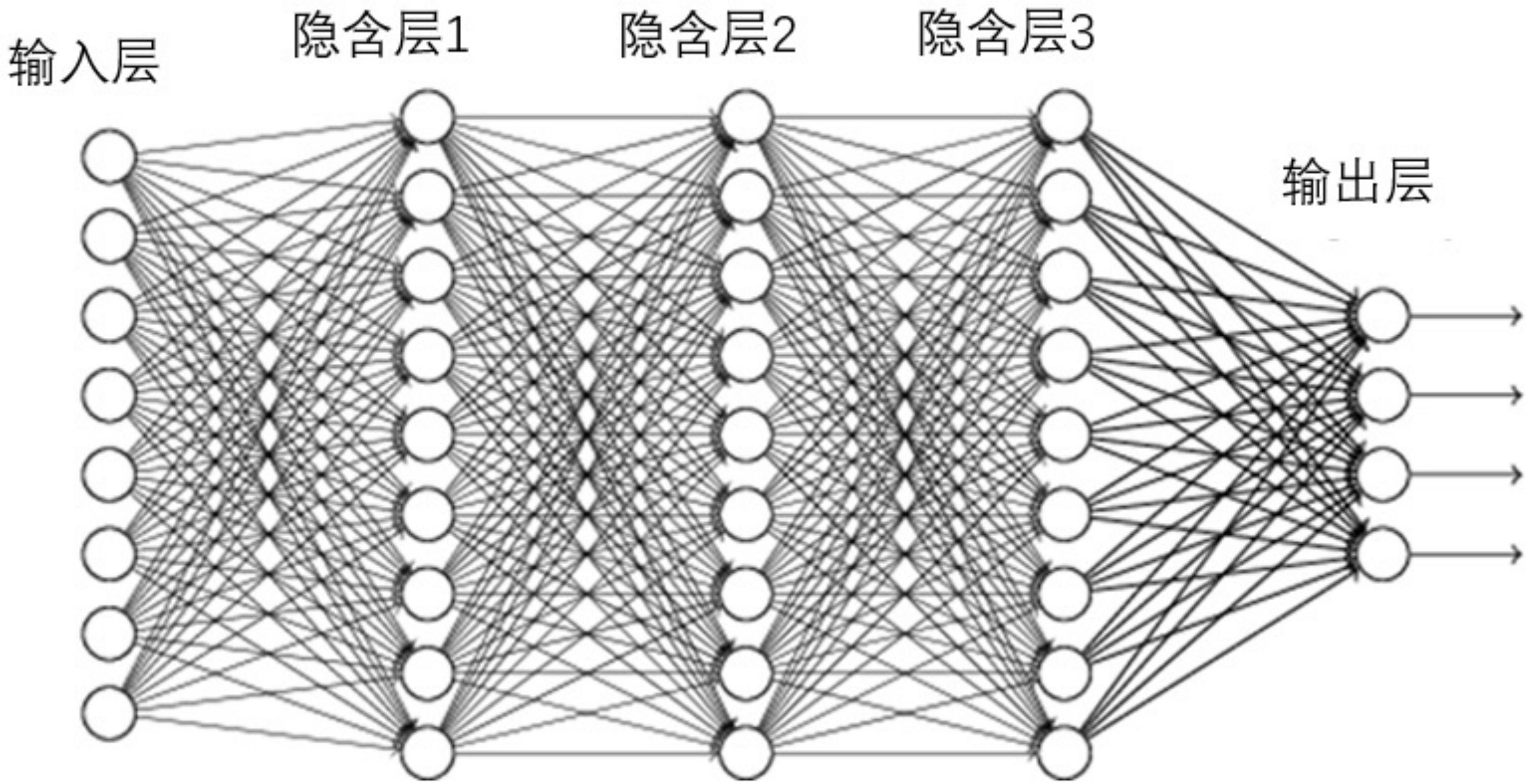
Outline

- Introduction & Methodology
- **MLP: Feed Forward & Back Propagation**
- Further Discussion
  - Model Architecture: Convolutional Neural Network
  - Loss Function: Regression or Classification
  - Optimization: Stochastic Gradient Descent

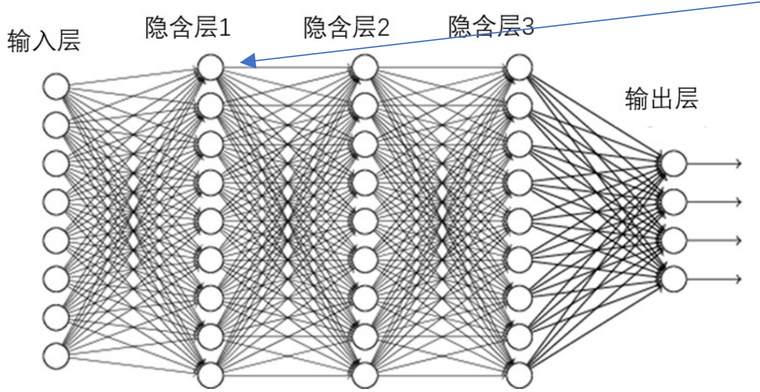
Task: MNIST Classification

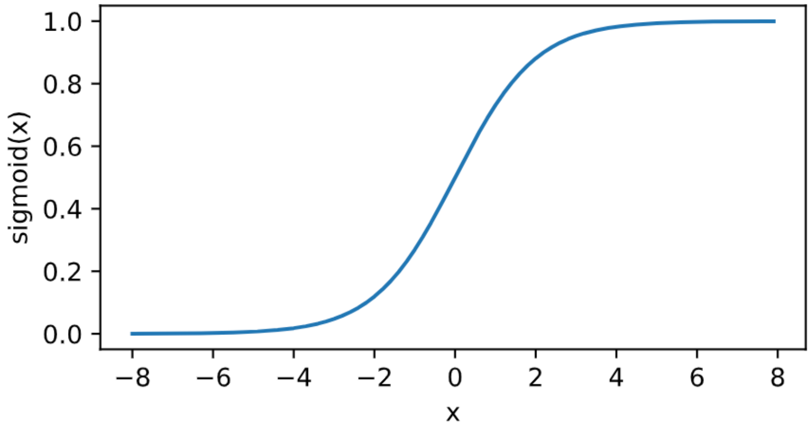
Input: 28x28 Image
Output: 1 of 10 class

MLP: Architecture



MLP: A Single "Neural"

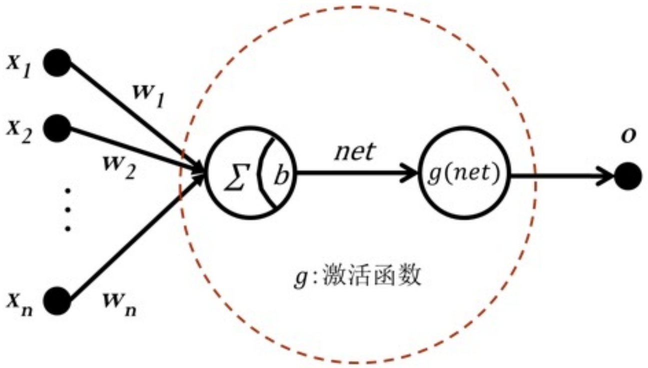




$$y = o(w \cdot x + b)$$

$$net = w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n + b$$
$$= \sum_{i=1}^{n} w_i \cdot x_i + b$$

$$o = g(net)$$



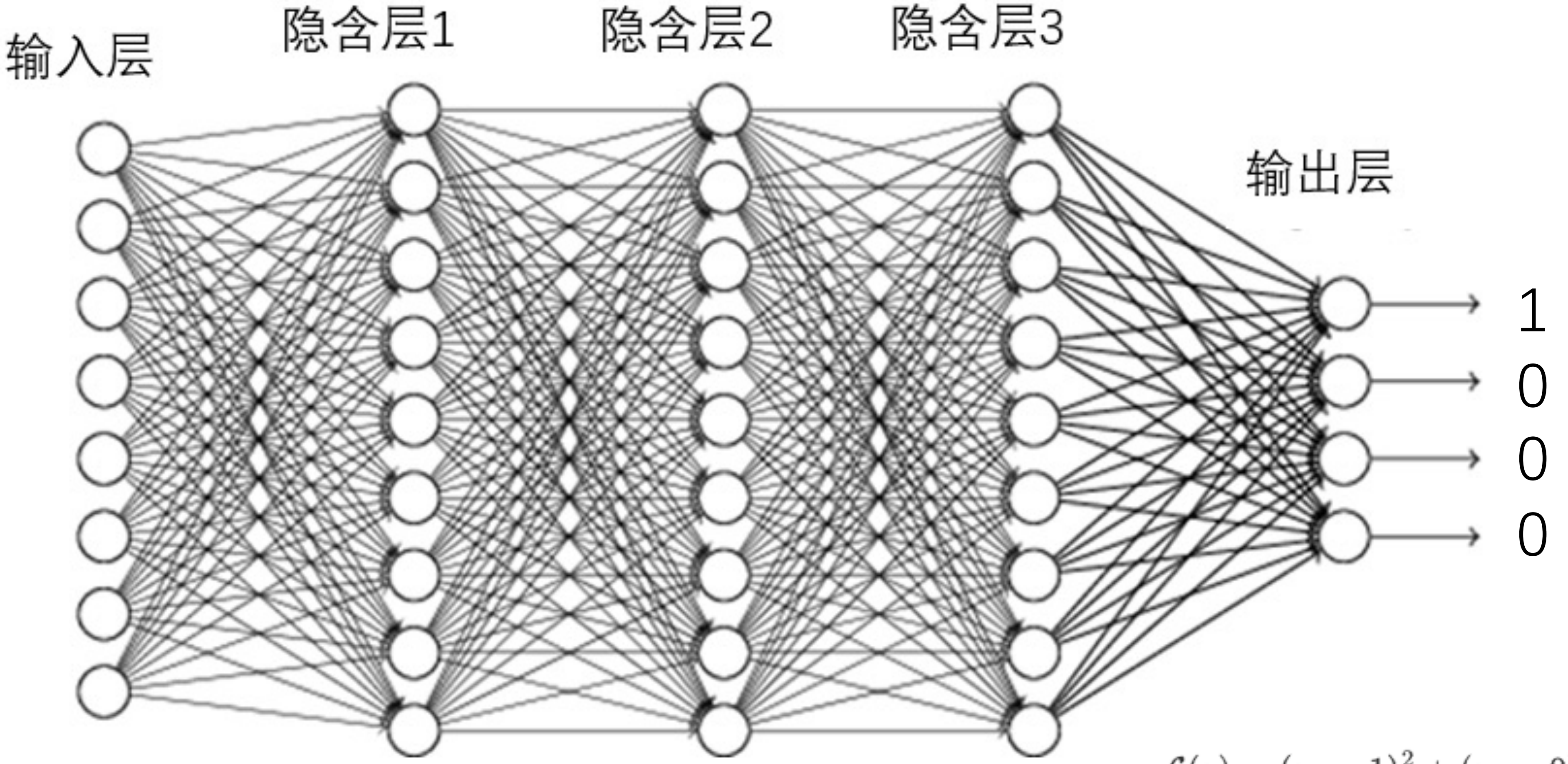
令：$x_0 = 1, \quad w_0 = b$

则：$net = w_0 \cdot x_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n$
$$= \sum_{i=0}^{n} w_i \cdot x_i = w \cdot x$$

Here:

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

MLP: Calculate Loss



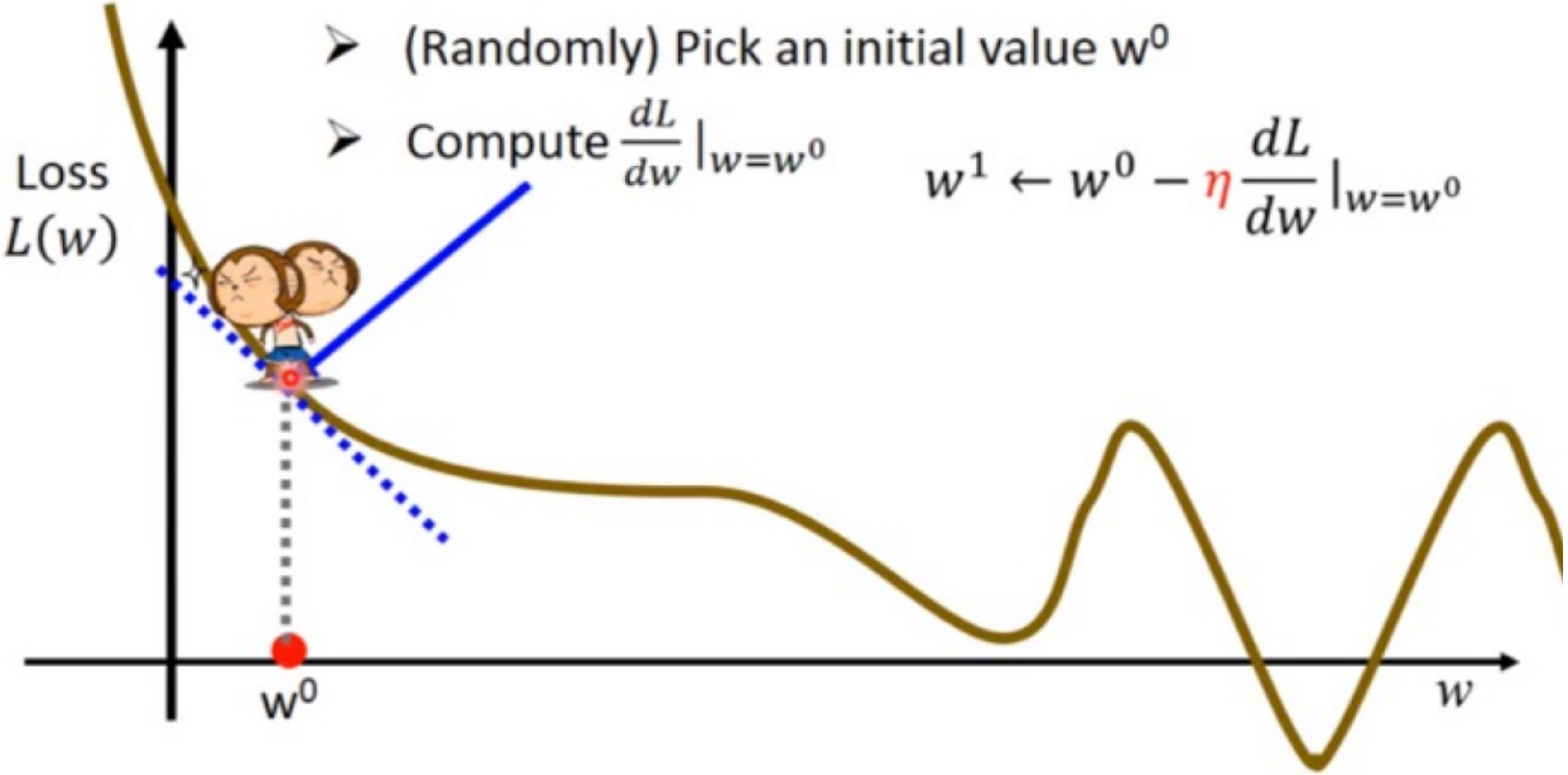
$$\mathcal{L}(y) = (y_1 - 1)^2 + (y_2 - 0)^2 + (y_3 - 0)^2 + (y_4 - 0)^2$$

MLP: How to optimize

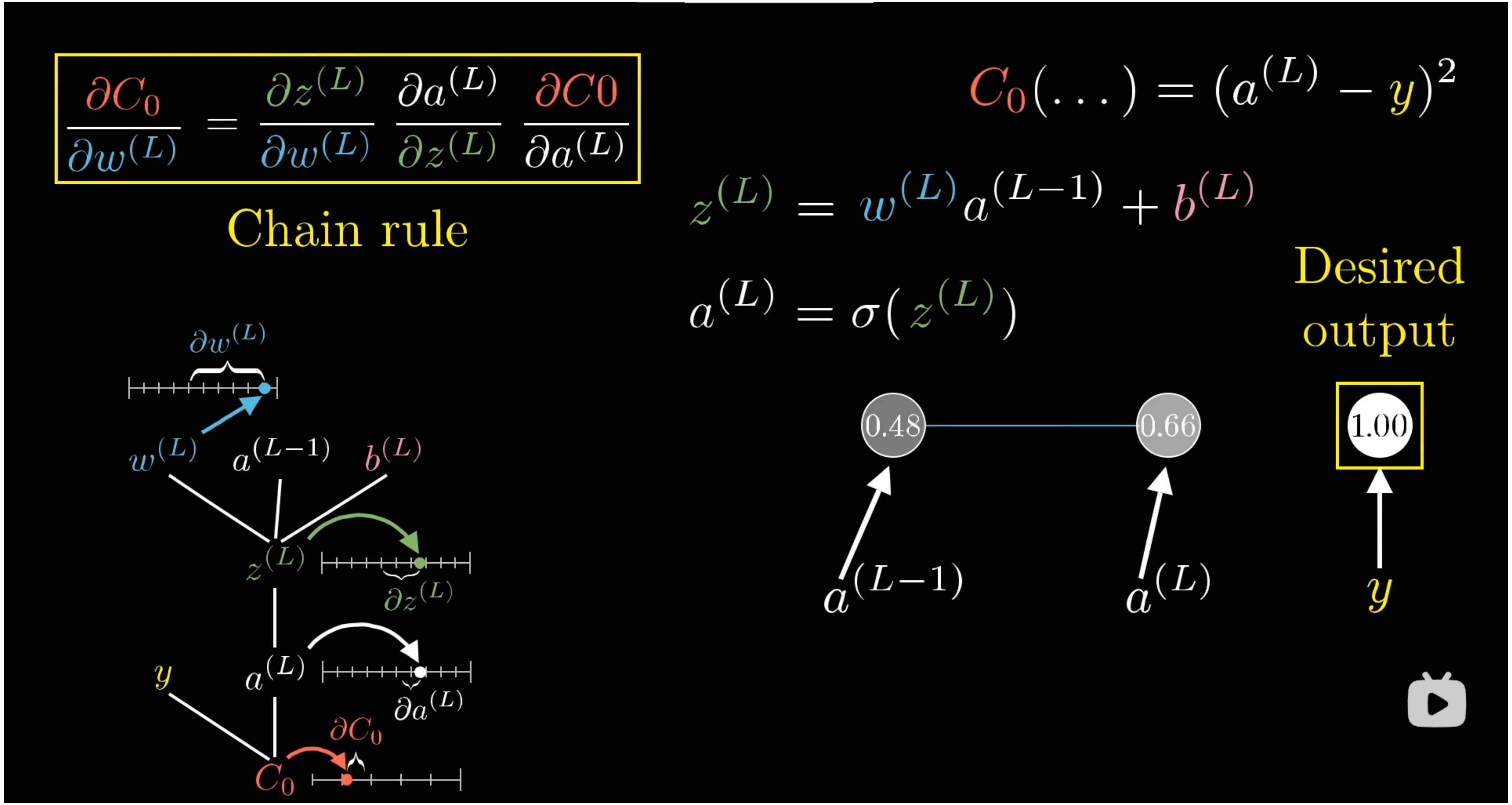
$$w^* = arg \min_w L(w)$$

- Consider loss function $L(w)$ with one parameter w:
  - ➢ (Randomly) Pick an initial value w^0
  - ➢ Compute $\frac{dL}{dw}|_{w=w^0}$

$$w^1 \leftarrow w^0 - \eta \frac{dL}{dw}|_{w=w^0}$$



MLP: Back Propagation (Recap)



MLP: Back Propagation (Recap)

$$\frac{\partial C_0}{\partial w^{(L)}} = \frac{\partial z^{(L)}}{\partial w^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial C0}{\partial a^{(L)}}$$

$$C_0 = (a^{(L)} - y)^2$$

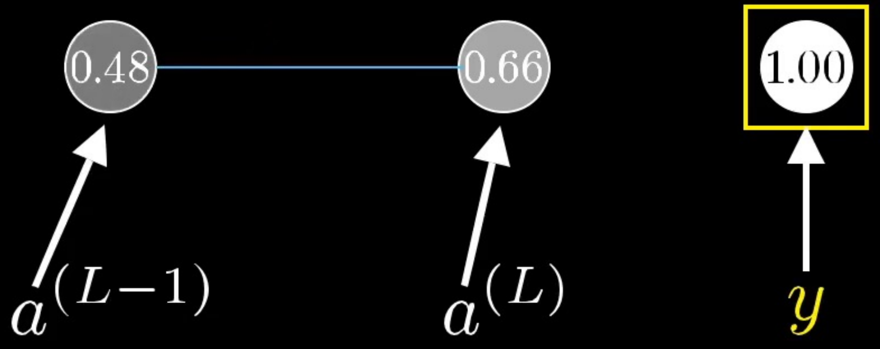
$$z^{(L)} = w^{(L)} a^{(L-1)} + b^{(L)}$$

$$a^{(L)} = \sigma(z^{(L)})$$

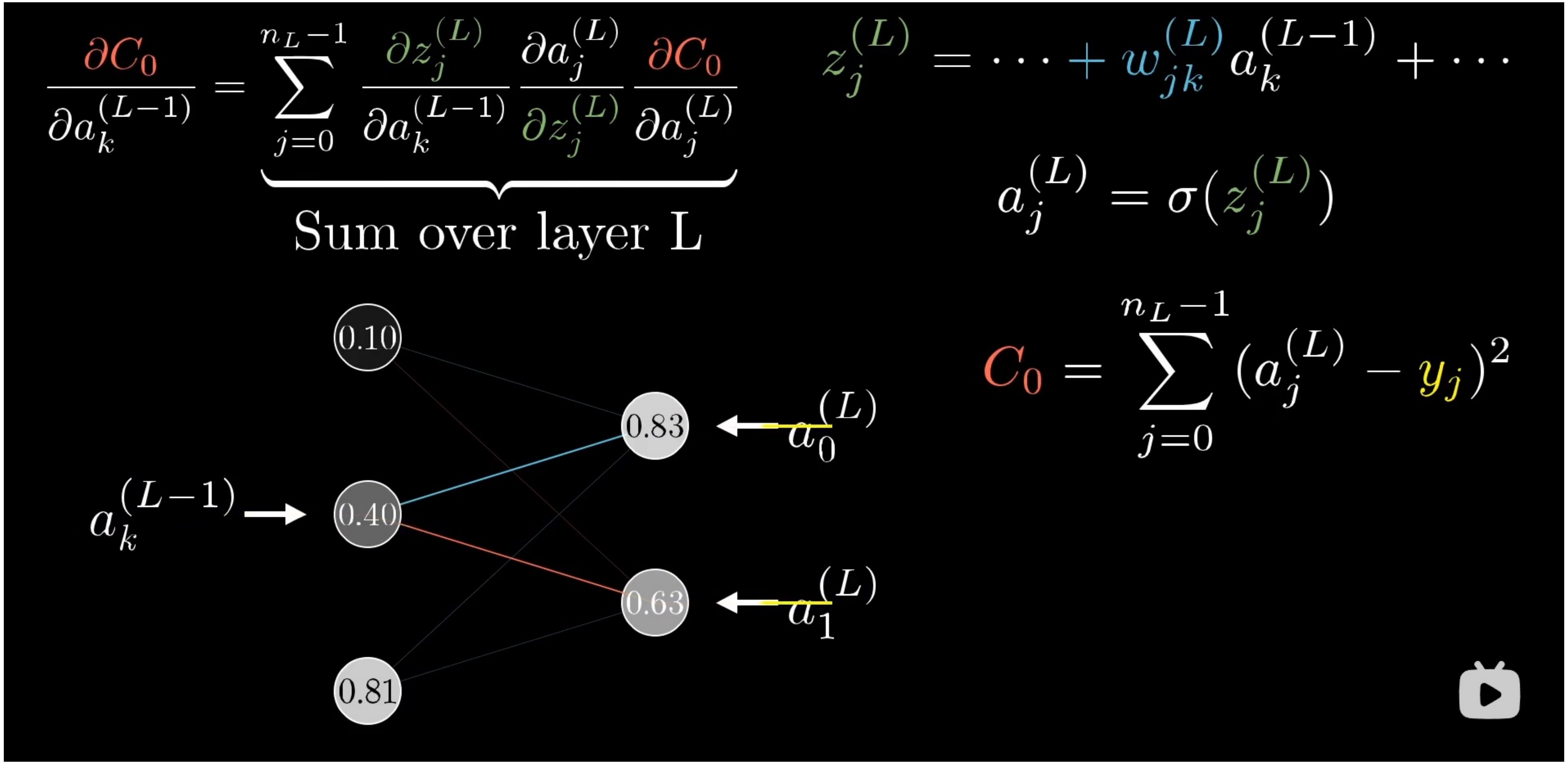
$$\frac{\partial C0}{\partial a^{(L)}} = 2(a^{(L)} - y)$$

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = \sigma'(z^{(L)})$$

$$\frac{\partial z^{(L)}}{\partial w^{(L)}} = a^{(L-1)}$$



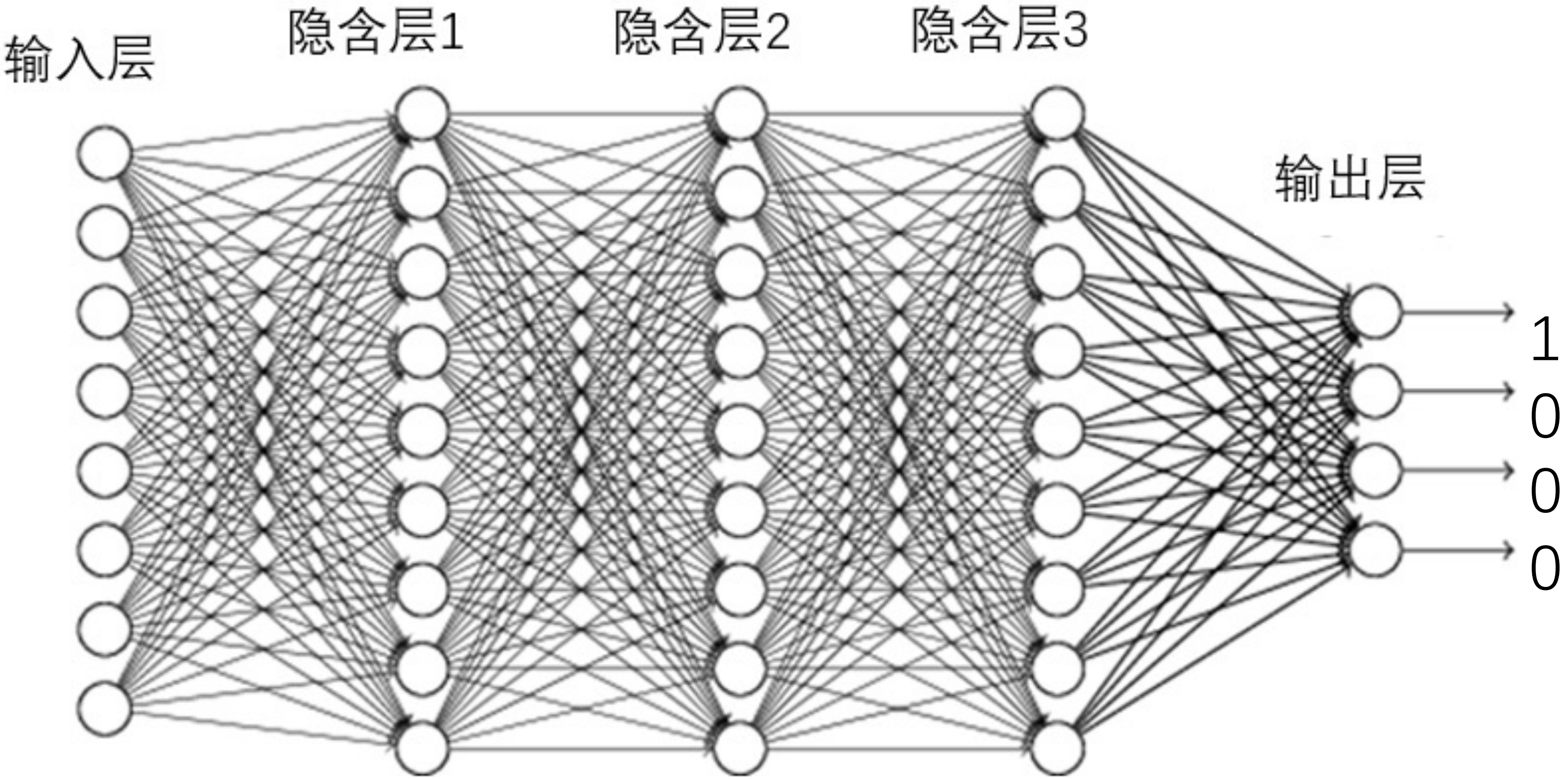
MLP: Chain Rule in Computational Graph



MLP: Chain Rule in Computational Graph

- We perform **topological sorting** on the computational graph obtained by the forward propagation process
  1. Find a node with zero out-degree
  2. Back propagate its gradient to its parent nodes in an accumulative manner
  3. Remove the node from computational graph

MLP: Chain Rule in Computational Graph



$$\mathcal{L}(y) = (y_1 - 1)^2 + (y_2 - 0)^2 + (y_3 - 0)^2 + (y_4 - 0)^2$$

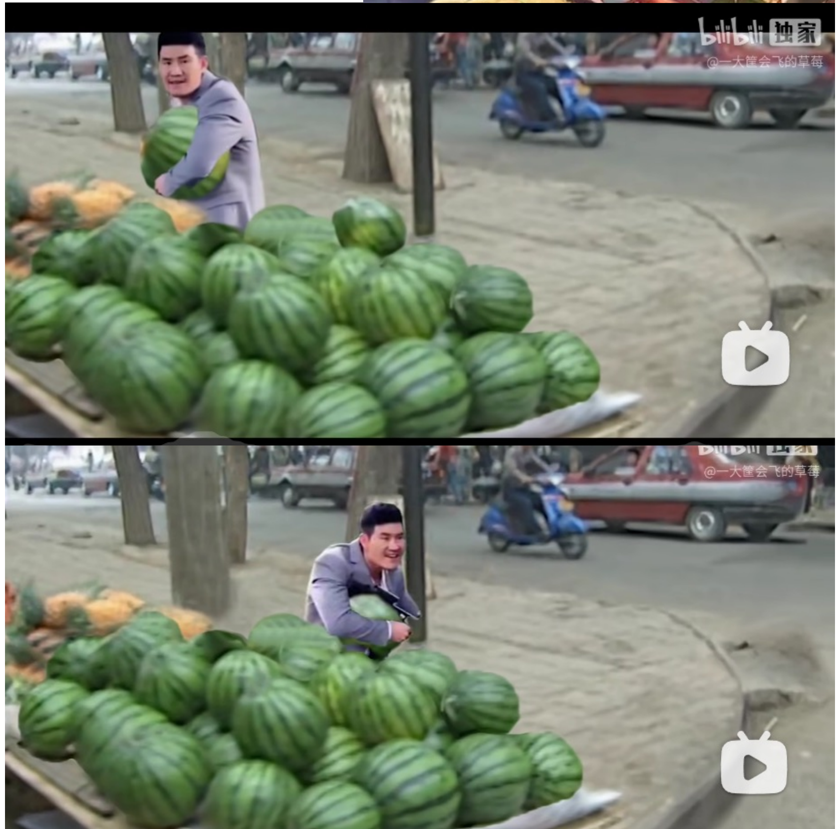
Luckily, we don't need to calculate gradients ourselves.
Deep learning frameworks like PyTorch have implemented this for us, you can look up to Autograd ☺

Outline

- Introduction & Methodology
- MLP: Feed Forward & Back Propagation
- **Further Discussion**
  - **Model Architecture: Convolutional Neural Network**
  - Loss Function: Regression or Classification
  - Optimization: Stochastic Gradient Descent

Convolutional Neural Network

- Intuitions
  - **Translational** or **flip equivariance** characteristics are held by objects.
  - Sometimes we don't care where it appears in the image.
  - We only care about the fact that whether an object appears or not.
  - It is too costly to use a MLP for this …

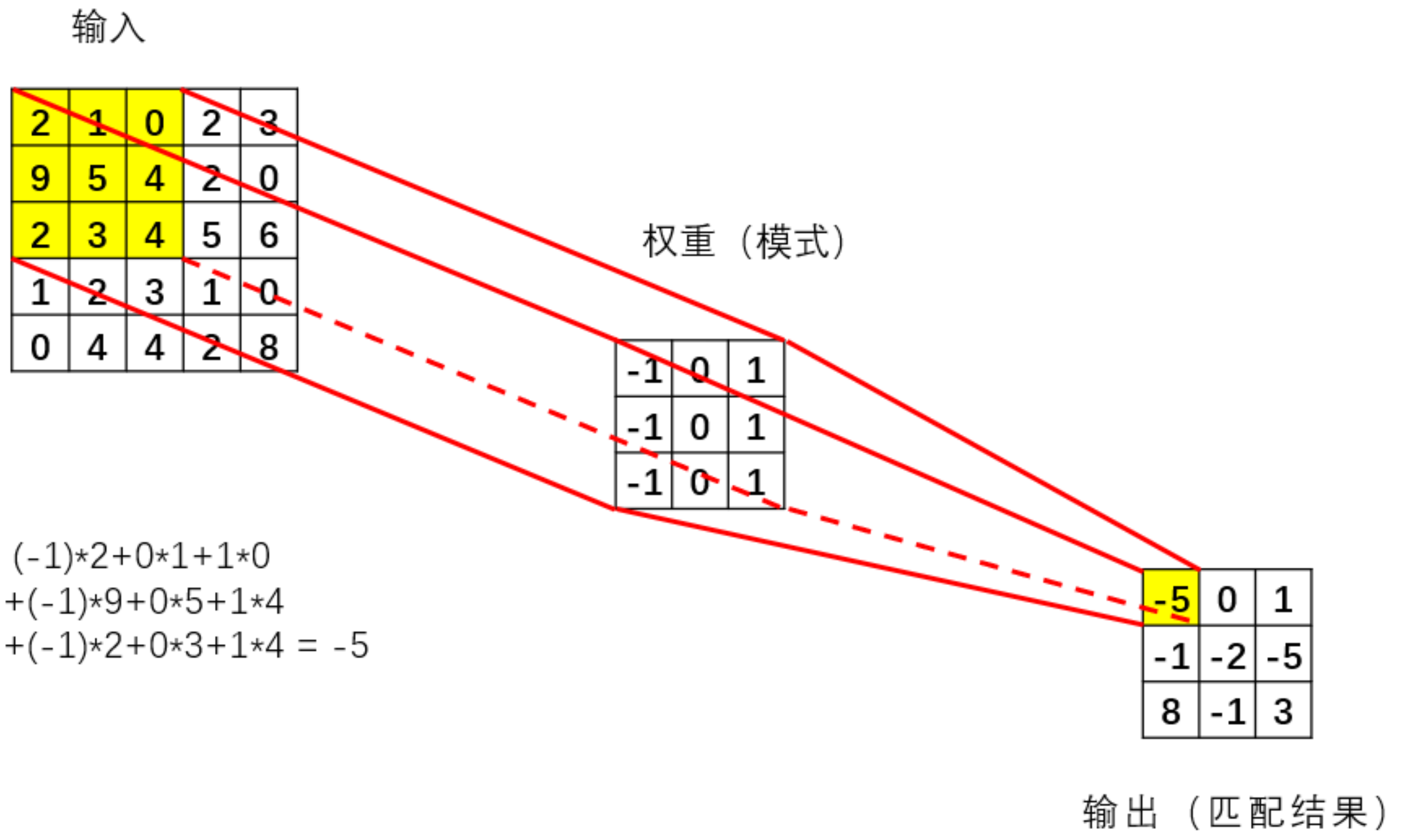


Convolutional Neural Network

- **1. Conv Layer**
  - To detect the existence of Karyl [1]
- **2. Max Pooling Layer**
  - To aggregate the information whether Karyl [1] exists or not in the local reception

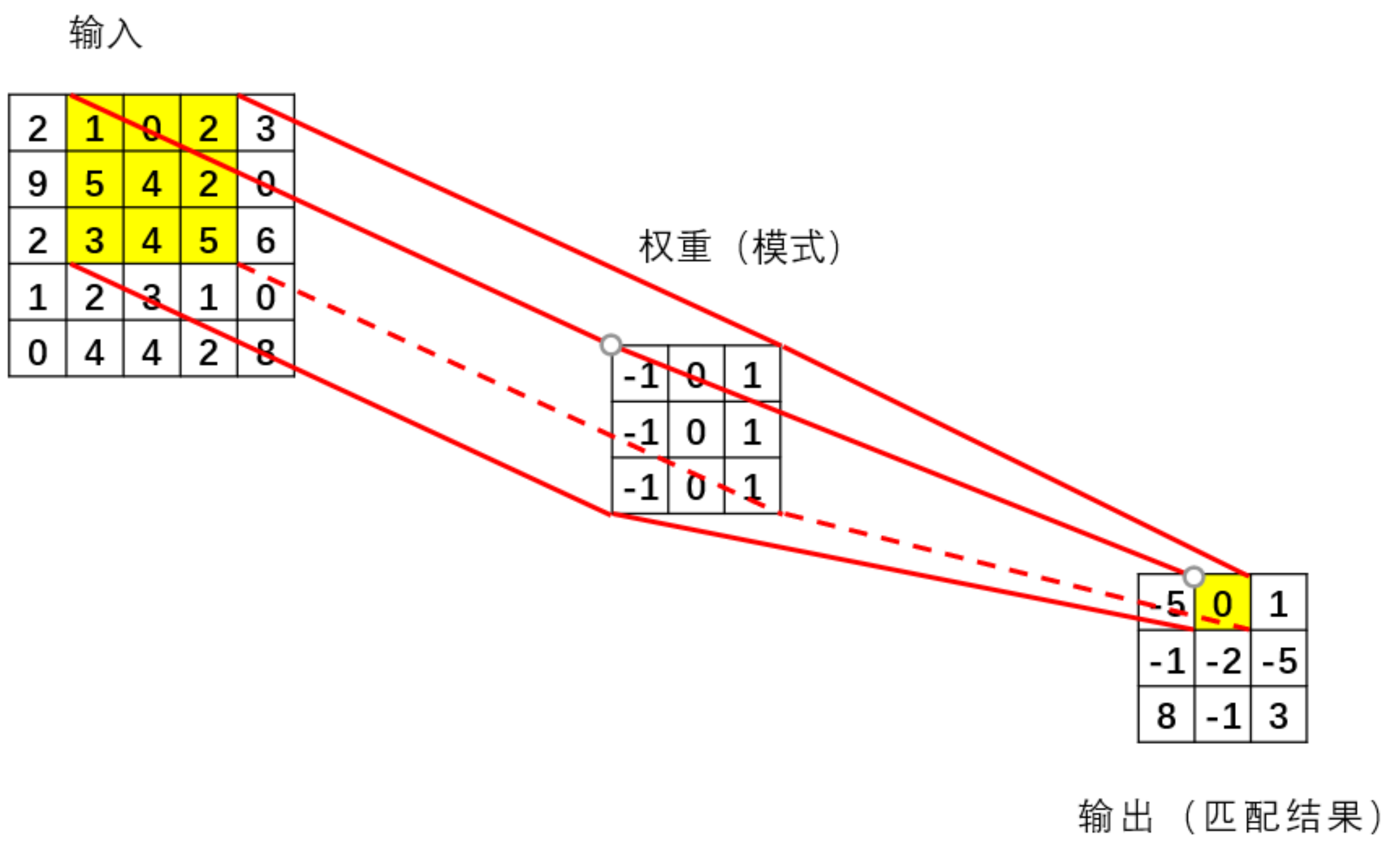
[1] Cygames et.al. Princess Connect! Re: Dive. Google Play 2018.

Convolutional Neural Network: Conv Layer

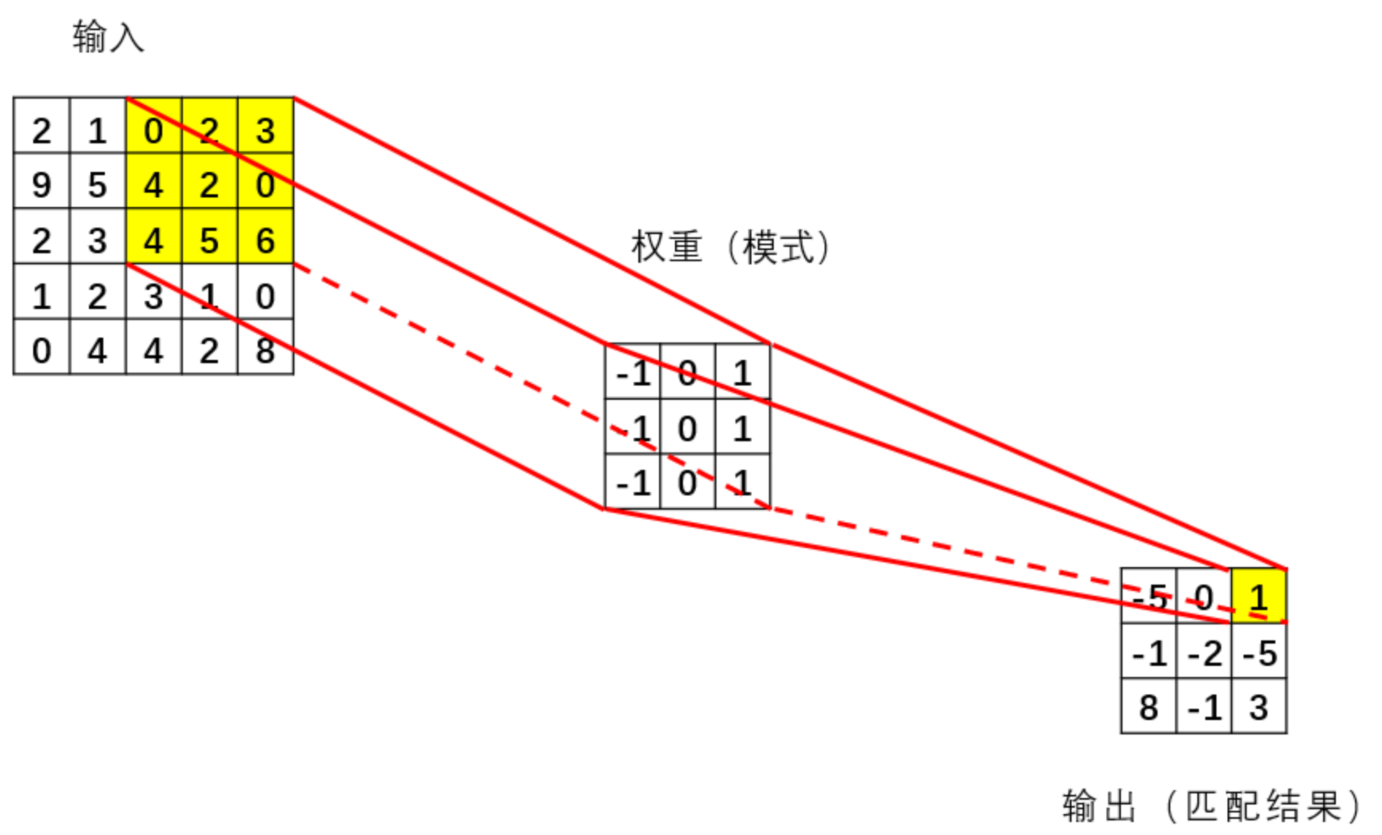


(-1)*2+0*1+1*0
+(-1)*9+0*5+1*4
+(-1)*2+0*3+1*4 = -5

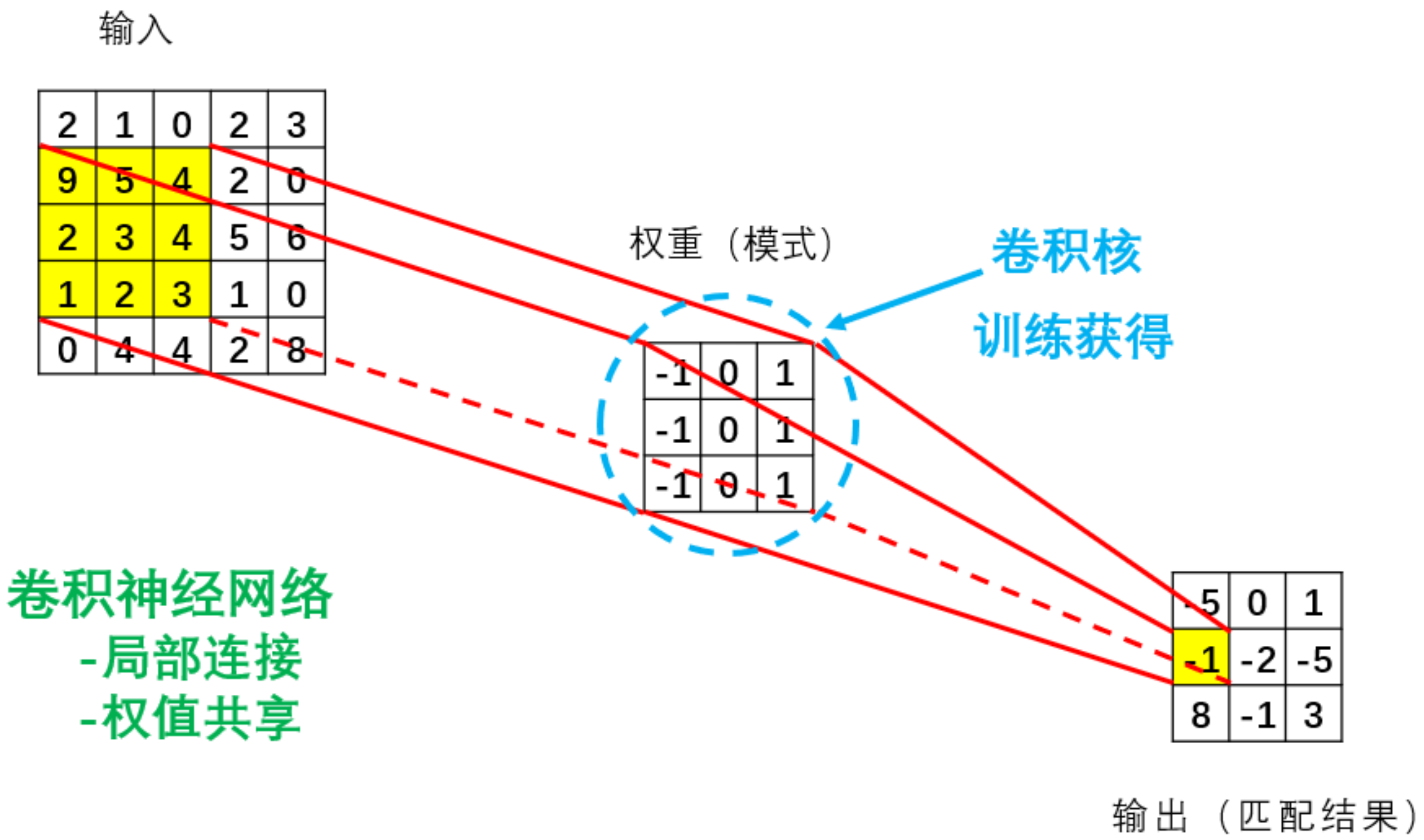
Convolutional Neural Network: Conv Layer



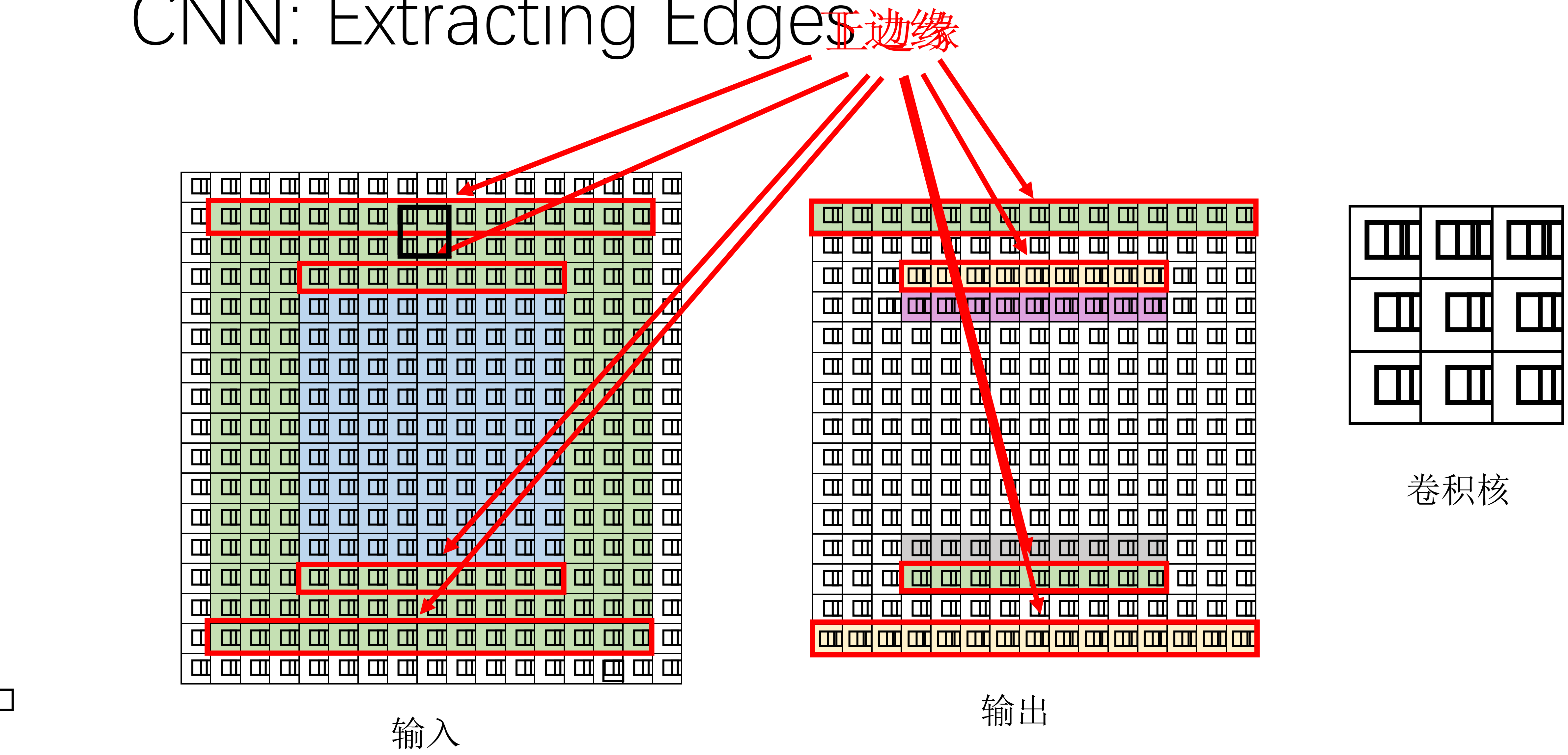
Convolutional Neural Network: Conv Layer



Convolutional Neural Network: Conv Layer

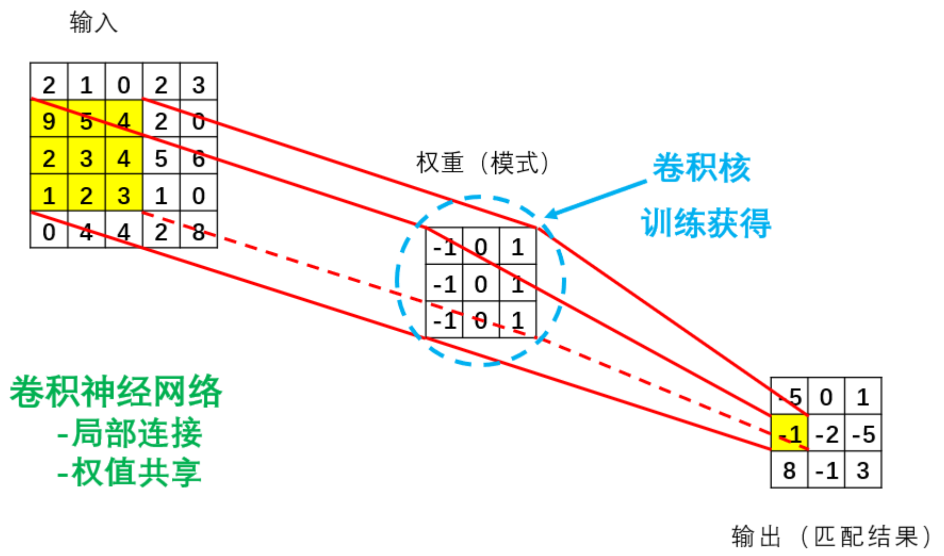


CNN: Extracting Edges



Convolutional Neural Network: **Hyper**parameters

- Kernel Size (3 x 3 in the illustration)
- Padding
- Stride
- #in_channels
- #out_channels

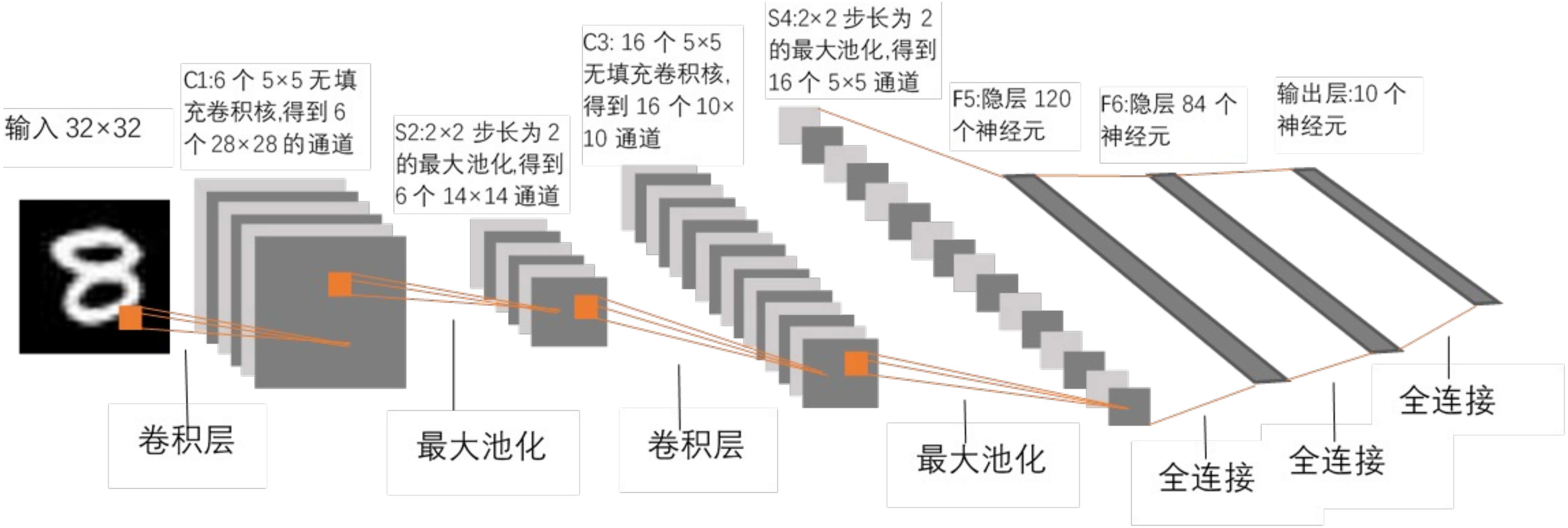


Convolutional Neural Network: Max Pooling Layer

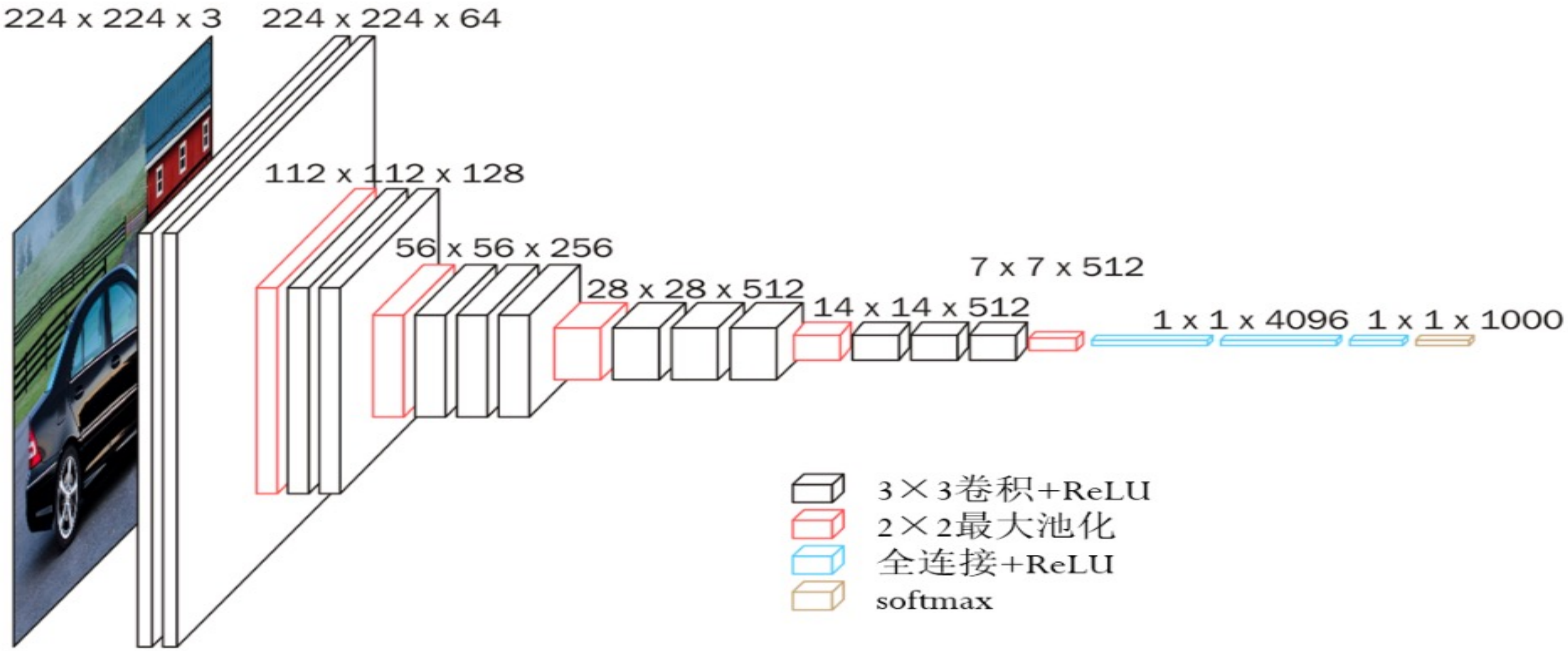
- Pooling Window Size (3 x 3 in the illustration)
- Padding
- Stride

输入为一个通道

Convolutional Neural Network: Example LeNet



Convolutional Neural Network: Example VGG-16



Outline

- Introduction & Methodology
- MLP: Feed Forward & Back Propagation
- **Further Discussion**
  - Model Architecture: Convolutional Neural Network
  - **Loss Function: Regression or Classification**
  - Optimization: Stochastic Gradient Descent

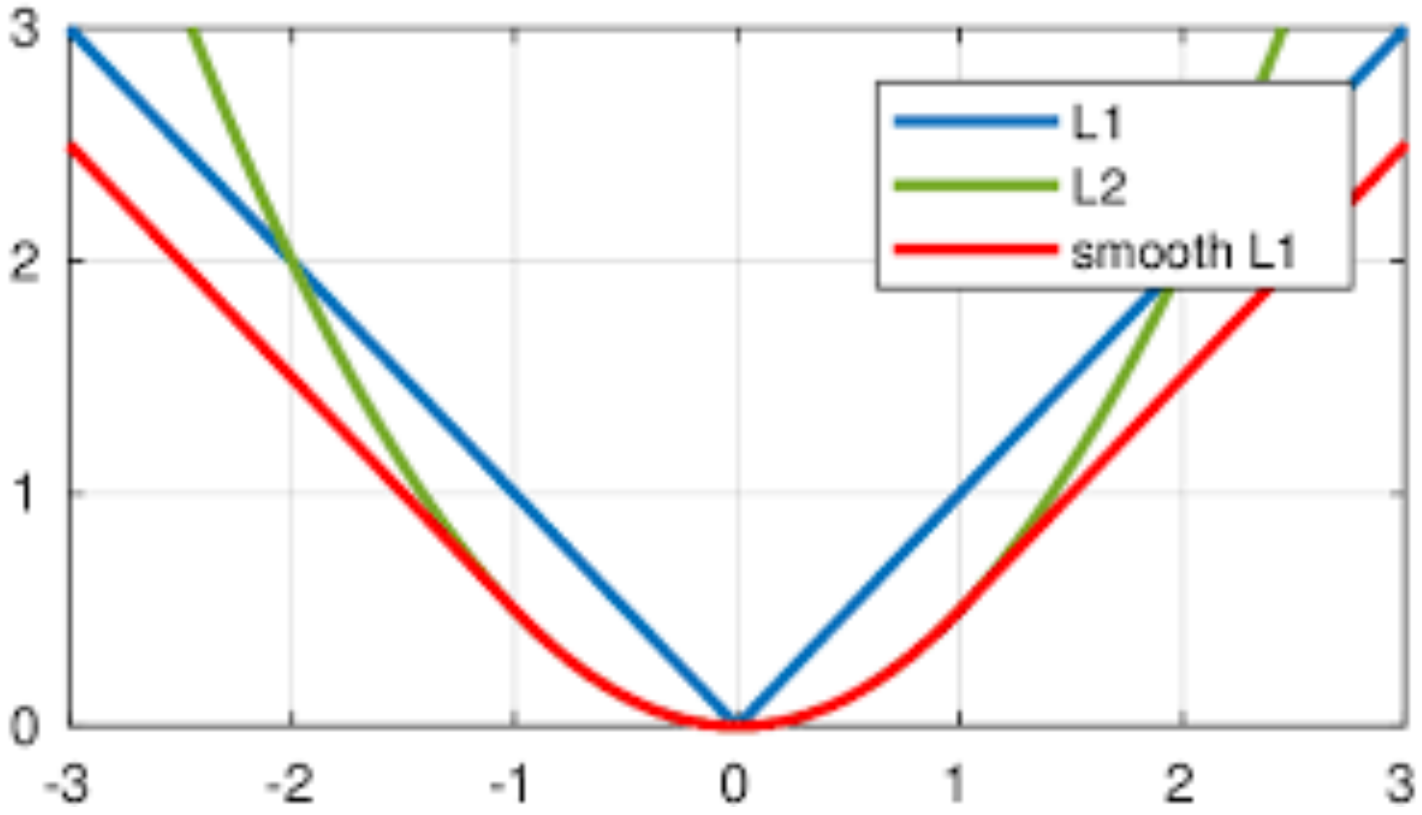
Loss Function: Regression

- To predict a certain value from inputs.

- L1 Loss

- L2 Loss

$$Smooth \quad L_1 = \begin{cases} 0.5x^2, & |x| < 1 \\ |x| - 0.5, & x < -1 \, or \, x > 1 \end{cases}$$



Loss Function: Classification

- Cross Entropy Loss

$$\text{H(p,q)} = \sum_x p(x) \cdot log\left(\frac{1}{q(x)}\right)$$

- (**B**inary) **C**ross **E**ntropy Loss

Question: If q is predicted distribution and p is the distribution of label / ground truth, then when does $H(p,q) = -\sum_x p(x) \ln q(x)$ achieves its minimum?

- Make the assumption that both p and q are defined on finite sets
  - Fix p as it's ground truth, use *Lagrange multiplier* to find which vector q minimize the function.

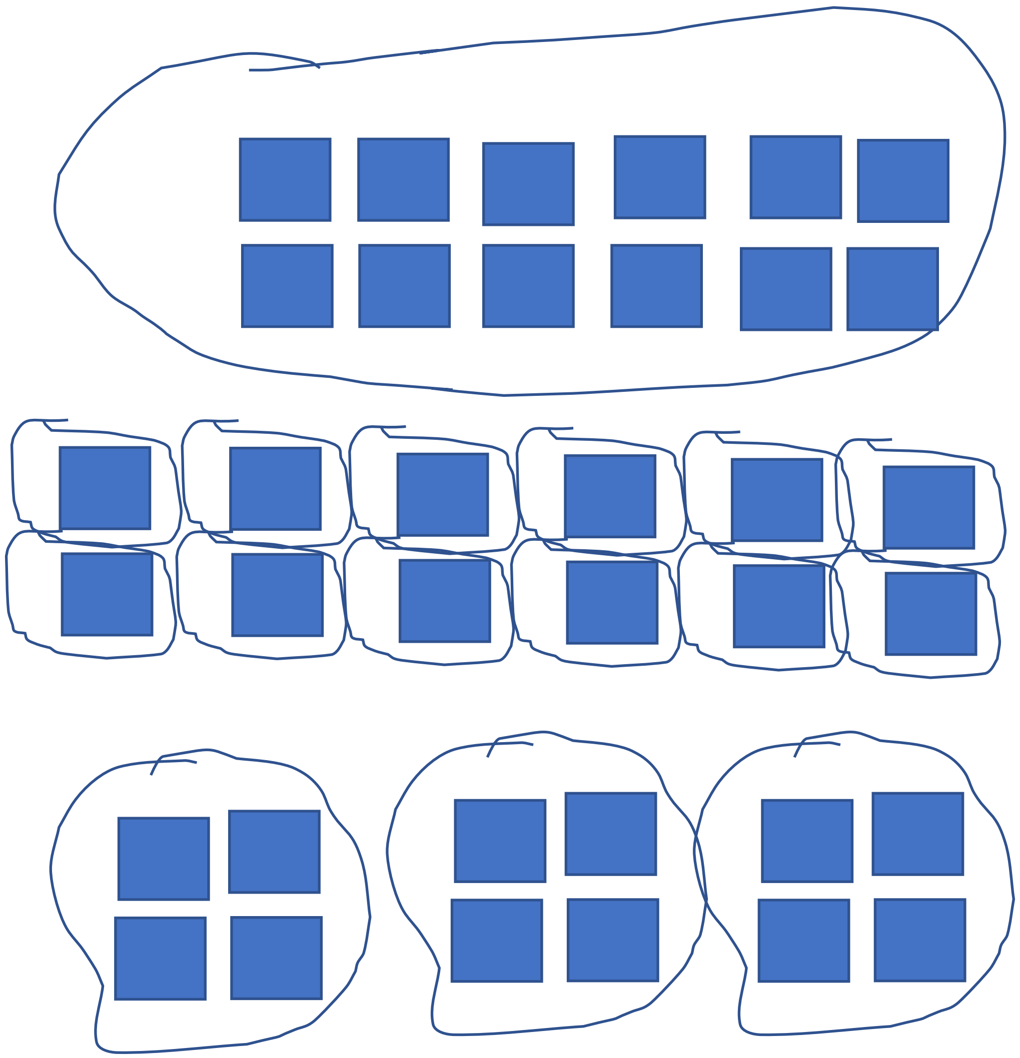
$$\min H(q) = -(p_1 \ln q_1 + p_2 \ln q_2 + \cdots + p_n \ln q_n)$$

$$\sum_{i=1}^{n} q_i = 1,\ 0 \leq q_i \leq 1.$$

Outline

- Introduction & Methodology
- MLP: Feed Forward & Back Propagation
- **Further Discussion**
  - Model Architecture: Convolutional Neural Network
  - Loss Function: Regression or Classification
  - **Optimization: Stochastic Gradient Descent**

Mini-batch Stochastic Gradient Descent



- Advantage of (3)
  - Compared to (1)
    - **computationally more efficient**
  - Compared to (2)
    - **Denoising with multiple items in batch**

Outline

- Introduction & Methodology
- MLP: Feed Forward & Back Propagation
- Further Discussion
  - Model Architecture: Convolutional Neural Network
  - Loss Function: Regression or Classification
  - Optimization: Stochastic Gradient Descent

PyTorch: Dive into Deep Learning

- Strong recommendation:
- https://tangshusen.me/Dive-into-DL-PyTorch/#/

- We'll learn about…
  - Dataset & DataLoader
  - Model declaration
  - Loss functions
  - Optimizer

Further study or further research

- Material Recommendations:
  - Machine Learning online courses led by Hung-yi Lee
    - 2017: https://www.bilibili.com/video/BV13x411v7US/
    - 2022: https://www.bilibili.com/video/BV1Wv411h7kN/
  - CS231n: Deep Learning for Computer Vision
  - CS224n: Natural Language Processing with Deep Learning
  - D2DL: https://tangshusen.me/Dive-into-DL-PyTorch/

- Also, laboratories are the best place for your research ☺
  - Meet the **s**tate-**o**f-**t**he-**a**rt technologies!

Q & A

Questions?