



**Viewsitec**

**友思特**



# 一站式步入深度学习

——结合自动深度学习算法的AI视觉检测应用

汇报人：黄灿彬

# 目录 / CONTENTS

## 01. 工业AI视觉检测 应用需求&痛点

- 应用需求
- 行业痛点

## 02. AI先验知识介绍 &深度学习基础

- 先验知识
- 深度学习基础

## 03. 一站式打造卓越 的深度学习模型

- Neuro-T介绍
- 搭建自己的AI模型

## 04. 如何让AI助力工 业制造领域

- 快速部署
- 应用案例

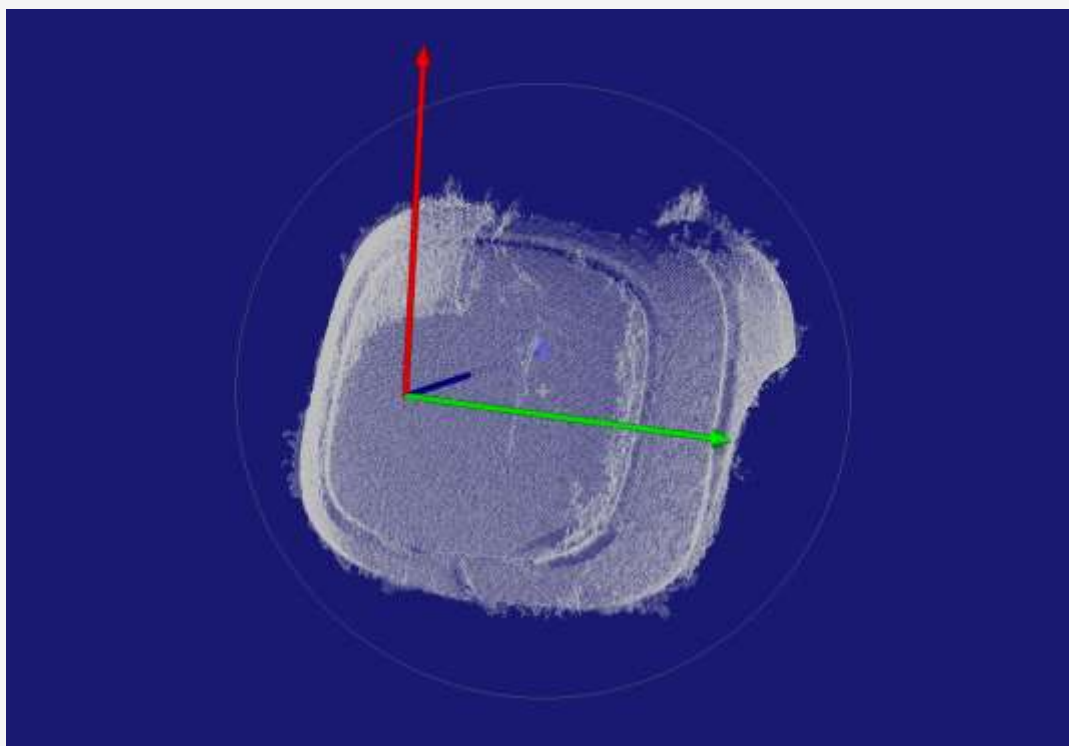
# 01

## 工业AI视觉检测 应用需求&痛点

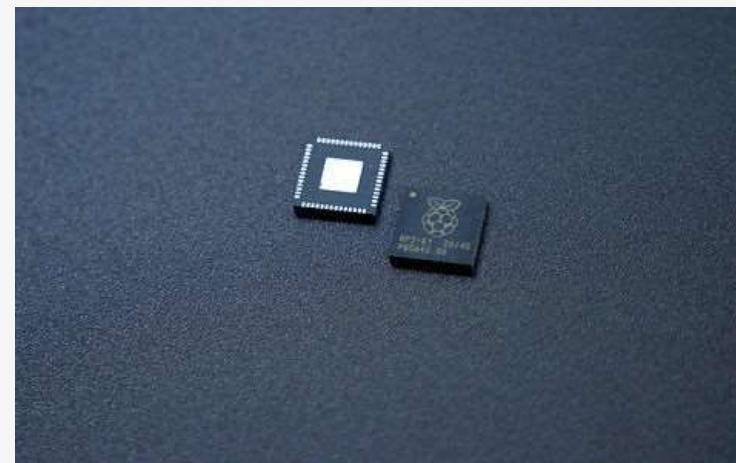


## 缺陷检测

- CAD模板匹配



- 视觉检测



## 目标检测/分类/字符条码检测

- 目标检测计数&分类



- OCR/条码检测

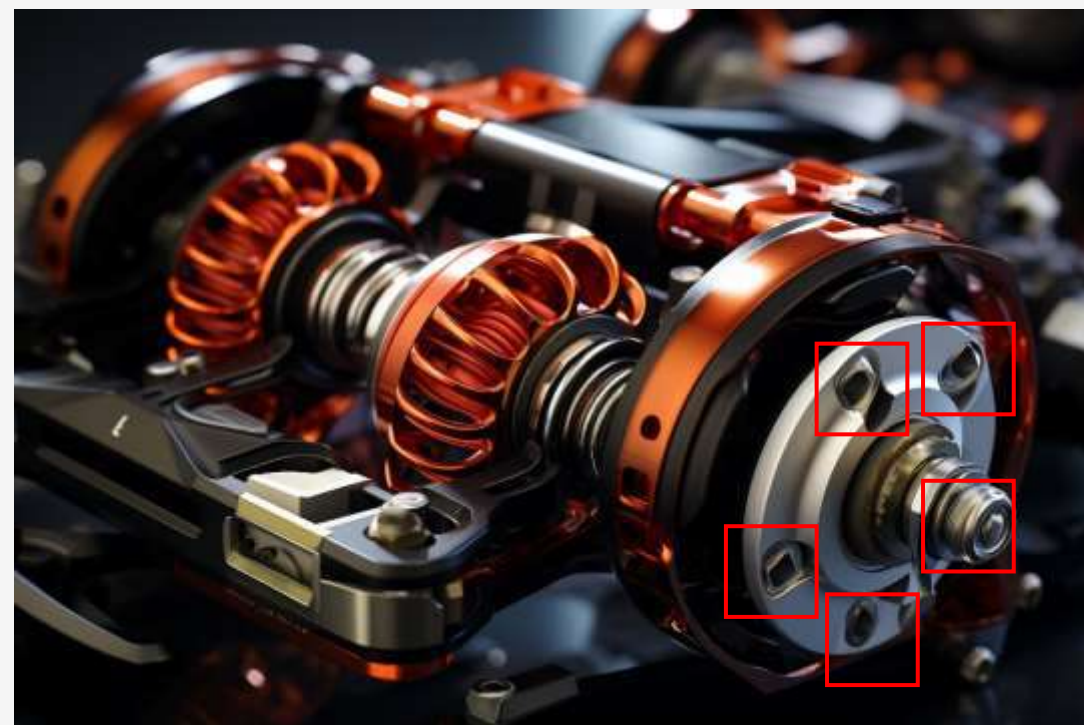




应用需求

行业痛点

## 装配检测





应用需求

行业痛点

## 工业视觉检测的重要性



### 提高生产效率

24小时工作，比人工快于5~10倍



### 减少人工误差

准确度高，误差率低，显著优于人工



### 降低生产成本

节省大量人力，降低残次品消耗



### 保障生产质量

检测产品质量，避免不合格产品流入市场





应用需求

行业痛点

## 标注数据成本高昂

模型训练需要大量数据，需要大量费用和人力对数据进行标注

## 需求大而人才短缺

AI领域对技术人员的素质要求较高，需要有足够领域知识的行业工程师和AI工程师



## 模型鲁棒性和泛化能力不高

对于不同光照、遮挡、表面材质，模型的检测精度都会受影响，且有新的需求的时候，需要重新对模型进行设计和训练

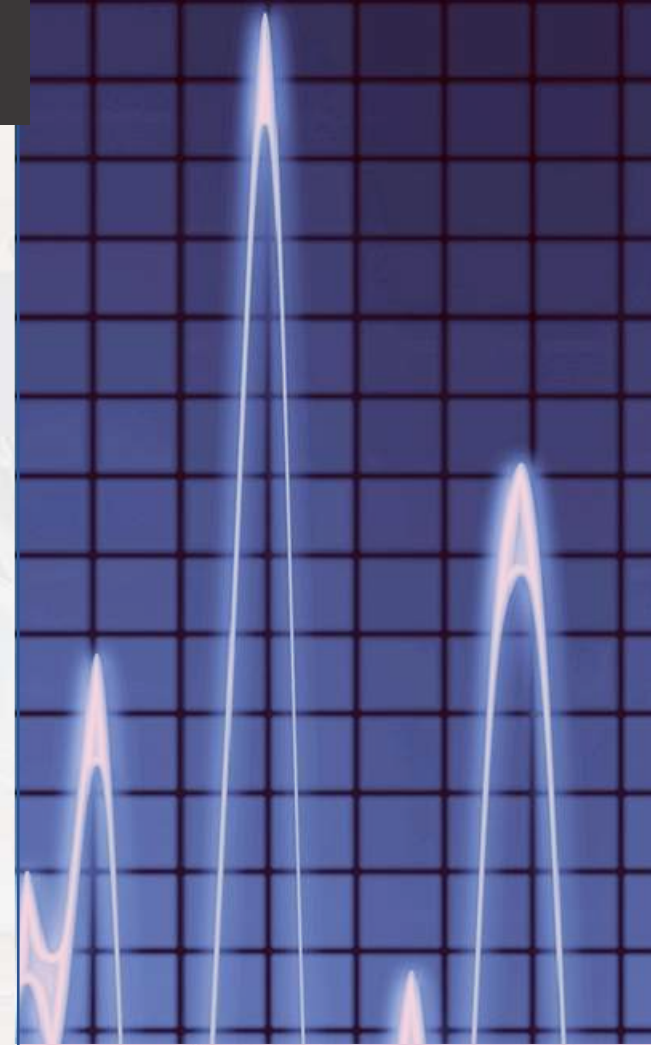
## 数据不平衡

工业应用中存在较多的缺陷检测需求，又缺乏足够的缺陷样本对模型进行训练

A collage of various images related to industry and science, including workers in hard hats, laboratory equipment, and technical diagrams. 

# 02

## AI先验知识介绍 &深度学习基础





AI(Artificial Intelligence)离不开**数学**，而**数学**，离不开**函数**



**泛函**分析——函数的函数

Eg:  $J = \int_0^1 x(t)dt$ ,  $J$ 的值随表达式 $x(t)$ 的变化而变化,  $J$ 就是一个泛函数

极值问题——工业/生活应用的**最优解**

- ① 参数优化
- ② 函数优化——泛函的极值问题

解决泛函的极值问题——**变分法**

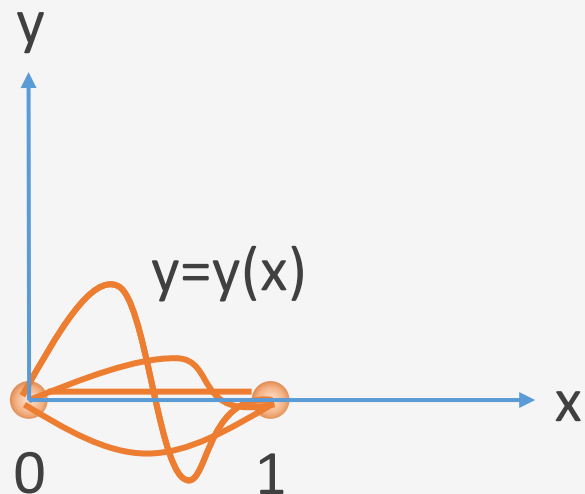
$J = \int_{t_0}^{t_f} L[x(t), \dot{x}(t), t]dt$ ,  $L$ 为泛函 $J$ 的**宗量**,  $t$ 为 $x$ 的**自变量**

$$\text{变分 } \delta J = \int_{t_0}^{t_f} \left[ \frac{\partial L}{\partial x} \delta x + \frac{\partial L}{\partial \dot{x}} \delta \dot{x} \right] dt$$

泛函取极值的必要条件——**欧拉-拉格朗日方程**:

$$\frac{\partial L}{\partial x} - \frac{d}{dt} \frac{\partial L}{\partial \dot{x}} = 0$$

## 泛函分析——证明两点之间直线段最短



套欧拉-拉格朗日方程： $\frac{\partial L}{\partial y} - \frac{d}{dx} \frac{\partial L}{\partial \dot{y}} = 0$

$$\frac{\partial L}{\partial y} = 0$$

$$\frac{\partial L}{\partial \dot{y}} = \frac{\dot{y}}{\sqrt{1 + \dot{y}^2}}$$

$$\frac{d}{dx} \frac{\partial L}{\partial \dot{y}} = 0$$

$$\begin{cases} \dot{y} = a \\ y = ax + b \end{cases}$$

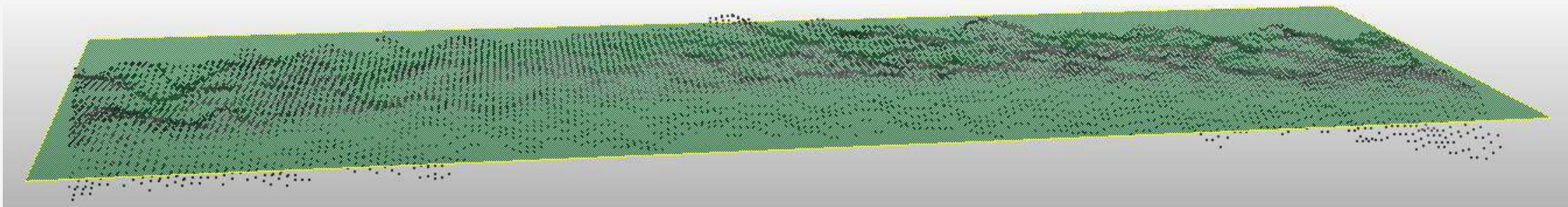
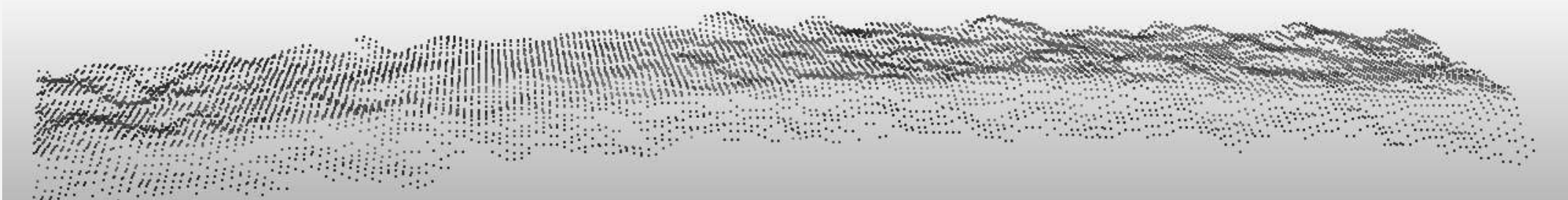
$$\frac{\dot{y}}{\sqrt{1 + \dot{y}^2}} = C$$

最优曲线为直线

$$J[y(x)] = \int ds = \int_0^1 \sqrt{1 + [\dot{y}(x)]^2} dx$$

$$\text{宗量 } L = \sqrt{1 + [\dot{y}(x)]^2}$$

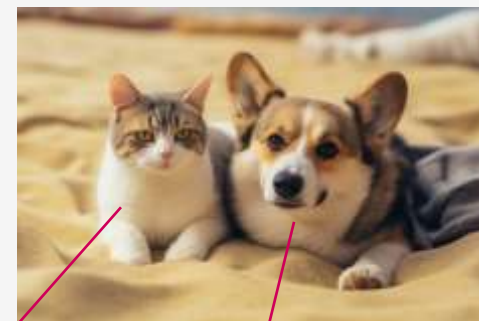
## 点云空间平面拟合



$$Ax + By + Cz + D = 0,$$

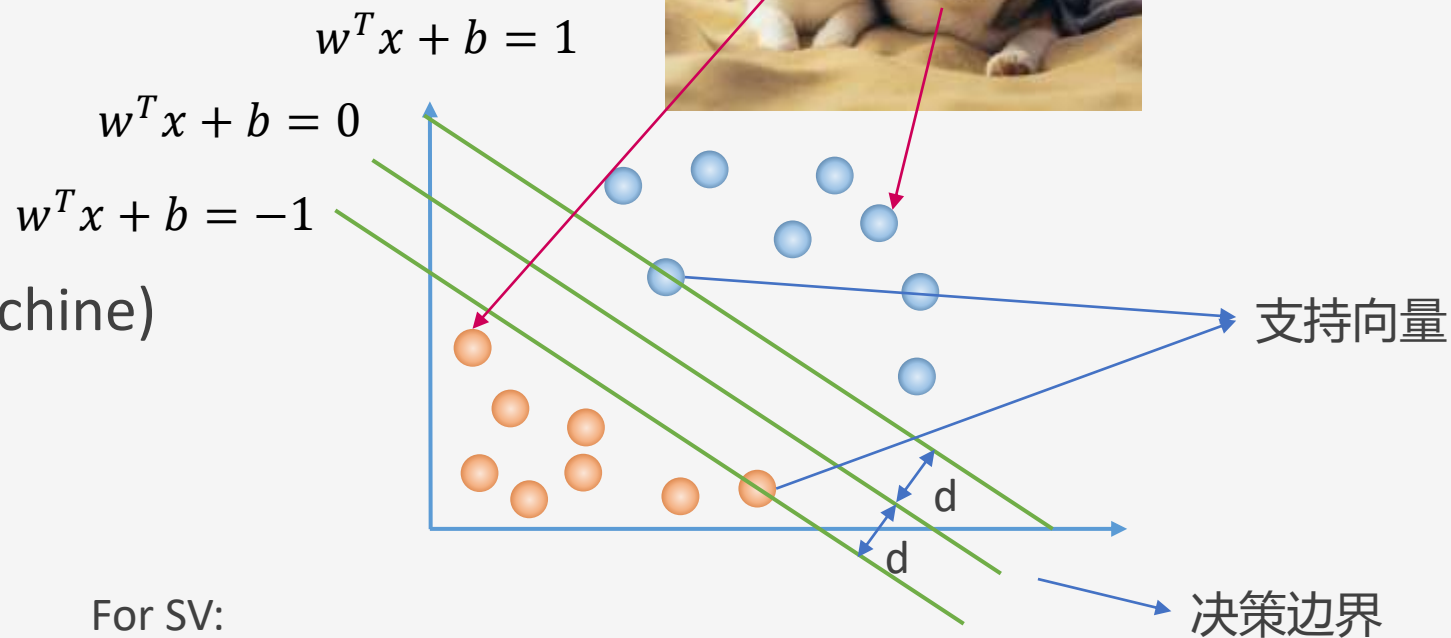
$$s. t. \min \sum_{i=1}^n d_i^2$$

$$d_i = \frac{|Ax_i + By_i + Cz_i + D|}{\sqrt{A^2 + B^2 + C^2}}$$



## SVM与二分类

### SVM(Support Vector Machine) 支持向量机



$$\begin{cases} \min \frac{1}{2} \|w\|^2, \\ \text{s.t. } y^i (w^T x^i + b) \geq 1 \end{cases}$$

For SV:  
 $\|w^T x + b\| = 1$

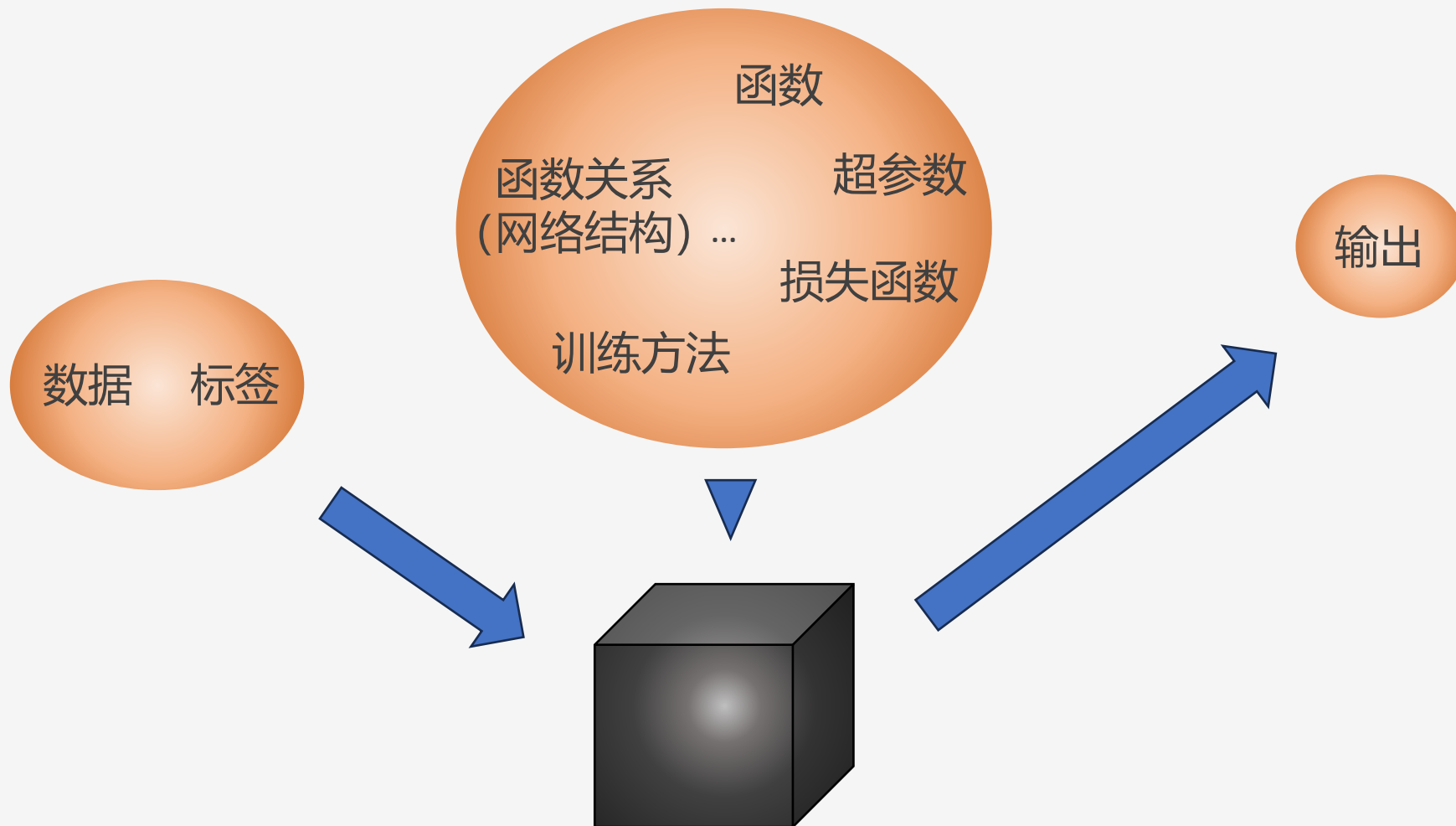
$$\max(d) = \max \frac{1}{\|w\|} = \min \|w\|$$

$$y^i (w^T x^i + b) \geq 1$$

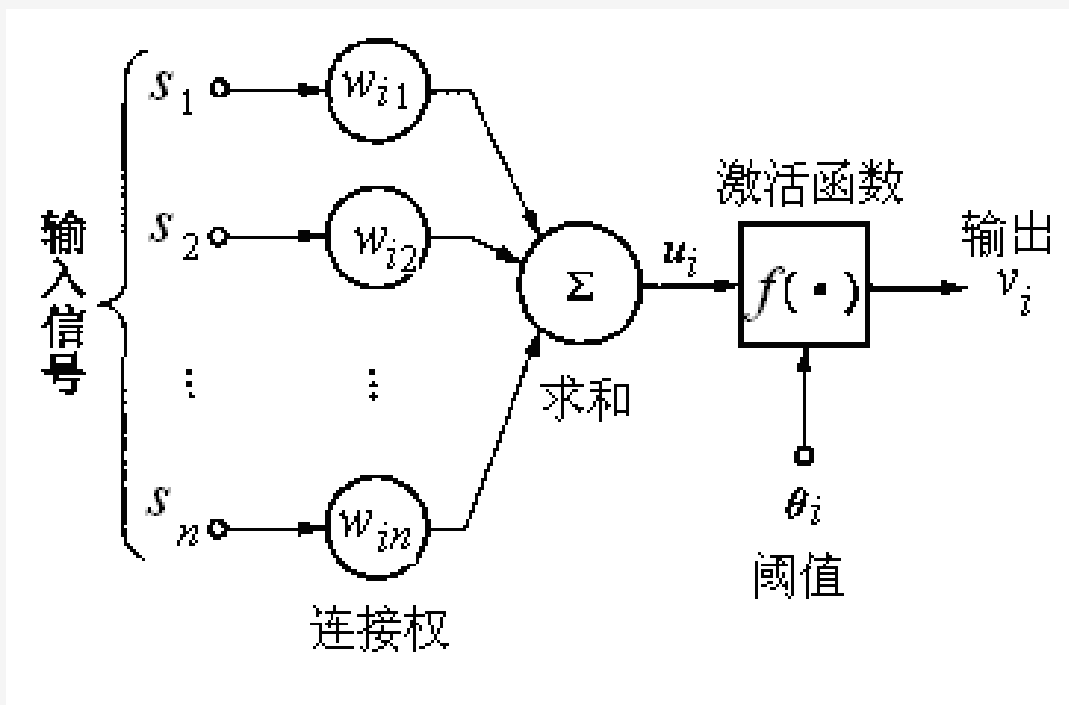
$$d = \frac{\|w^T x + b\|}{\|w\|}$$

$$\begin{cases} w^T x^i + b \geq 1, & \forall y^i = 1 \\ w^T x^i + b \leq -1, & \forall y^i = -1 \end{cases}$$

## 数学与人工智能的联系



# 人工神经网络



人工神经元模型

## 激活函数

提高网络学习复杂数据的能力，起非线性映射作用

## 数学模型

$$u_i = \sum_{j=1}^n w_{ij} s_j$$

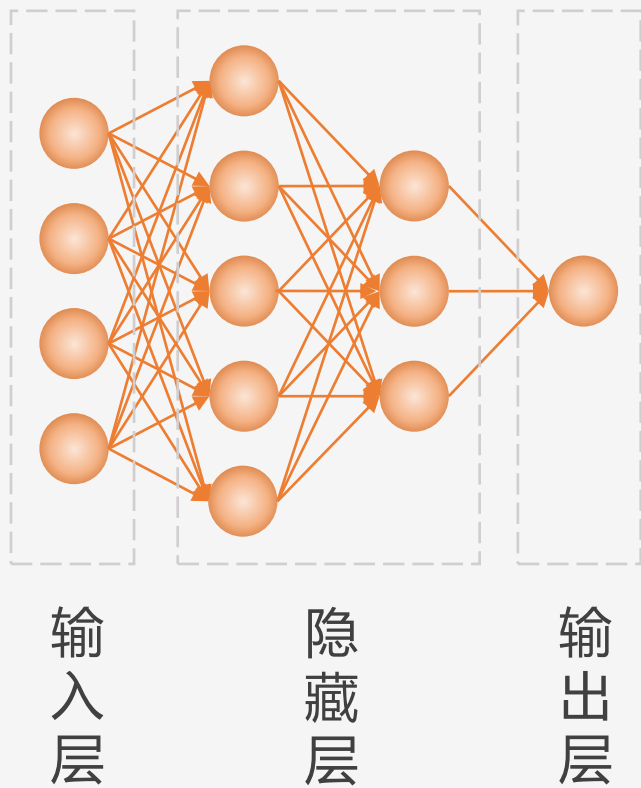
$$x_i = u_i - \theta_i$$

$$v_i = f(x_i)$$

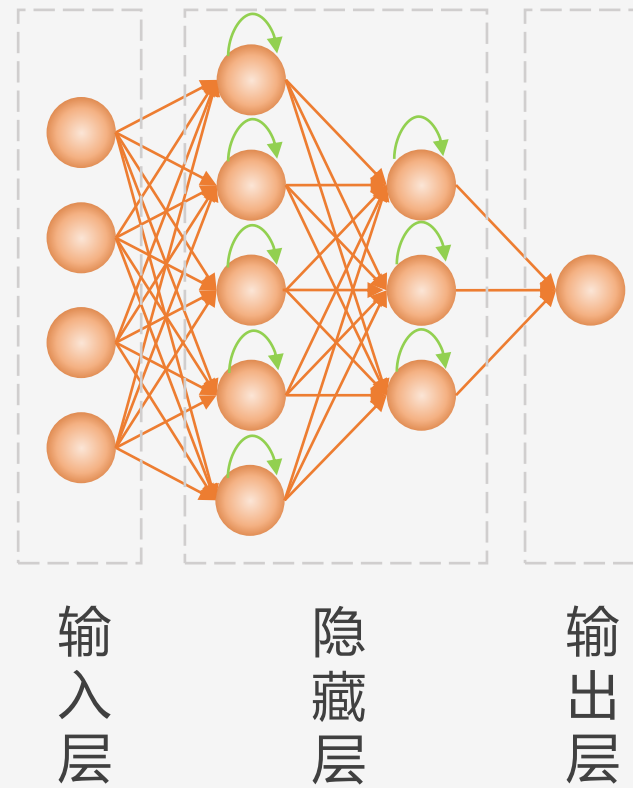


# 神经网络连接方式

## 前馈型



## 反馈型





## 常见的激活函数

### Sigmoid

常用于二分类输出层

### Tanh

常用于二分类隐藏层

### ReLU

常用于深度卷积网络（存在死区）

### Leaky ReLU

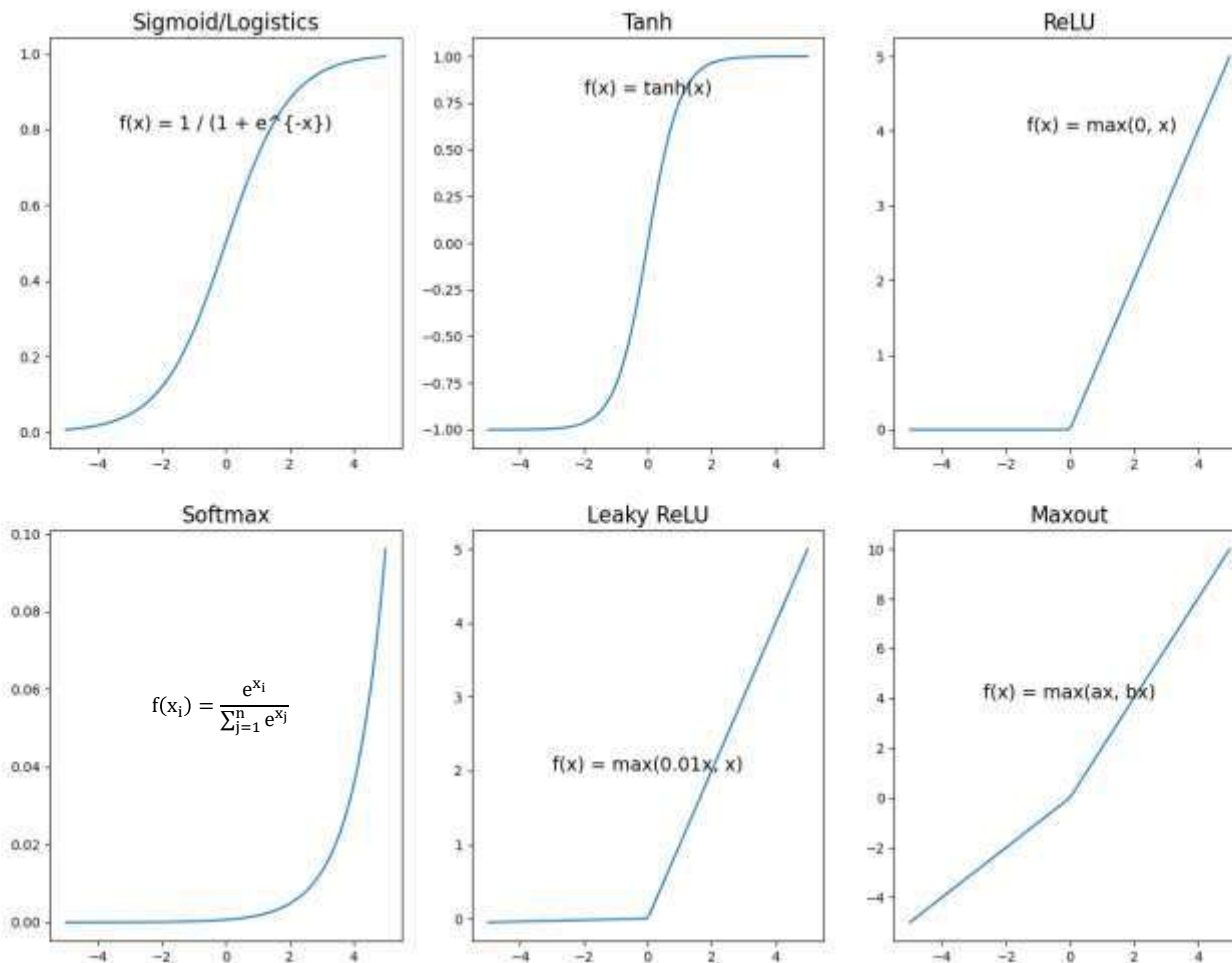
ReLU的改进版本（针对死区问题）

### Maxout

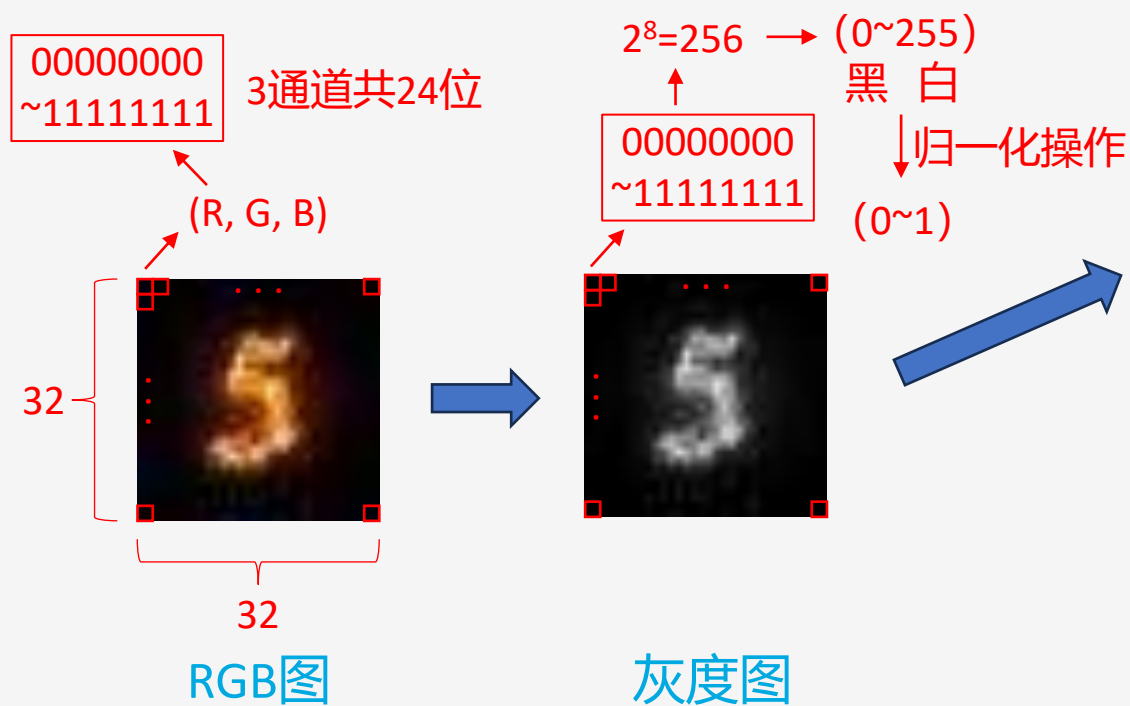
拟合能力强但参数量大

### Softmax

多分类问题，归一化操作



# 神经网络如何识别图像



```
[0.00392157 0.00392157 0.00392157 ... 0.00392157 0.00392157 0.00392157]  
[0.00392157 0.00392157 0.00392157 ... 0.00392157 0.00392157 0.00392157]  
[0.00392157 0.00392157 0.00392157 ... 0.00392157 0.00392157 0.00392157]  
...  
[0.00392157 0.00392157 0.00392157 ... 0.00784314 0.00392157 0.00392157]  
[0.00392157 0.00392157 0.00392157 ... 0.00392157 0.00392157 0.00392157]  
[0.00392157 0.00392157 0.00392157 ... 0.00392157 0.00392157 0.00392157]
```

图像数值矩阵 (32\*32)

展平

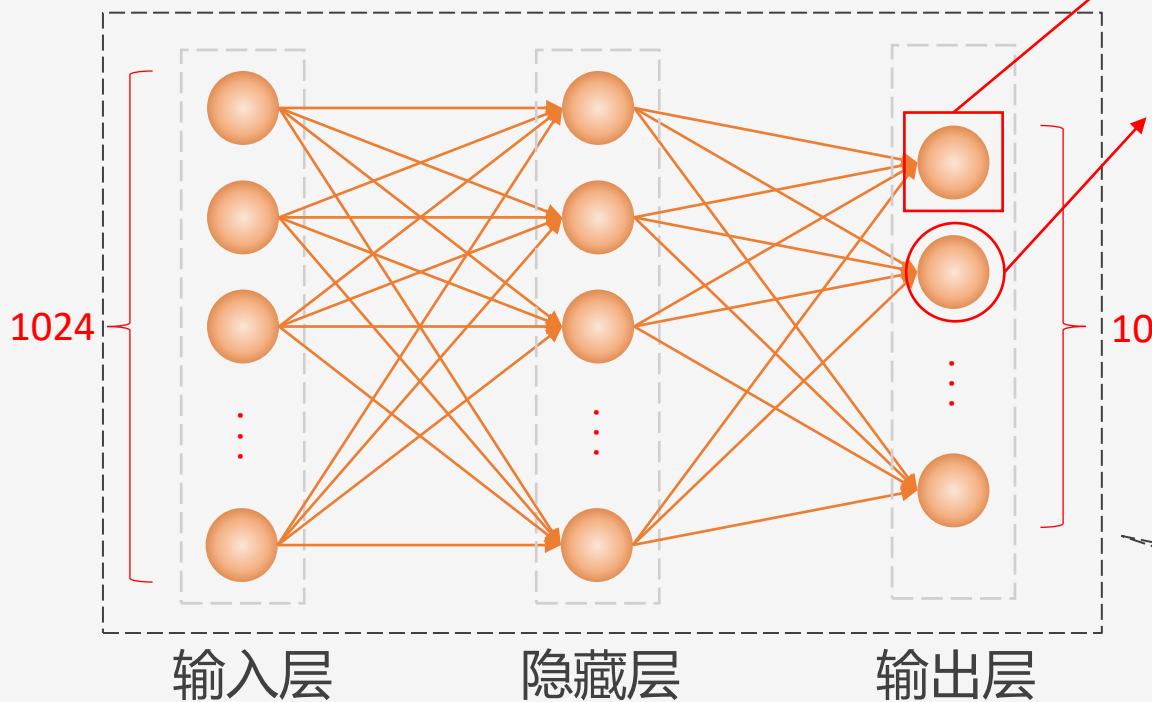
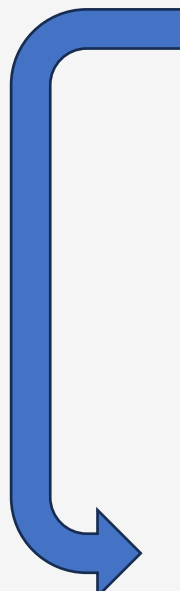
```
[0.00392157 0.00392157 0.00392157 ... 0.00392157 0.00392157 0.00392157]
```

一维向量 (1\*1024)

# 神经网络如何识别图像

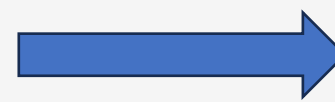
[0.00392157 0.00392157 0.00392157 ... 0.00392157 0.00392157 0.00392157]

一维向量 (1\*1024)



Activation: Softmax  
将结果映射到 (0, 1)

概率最大的即为预测结果



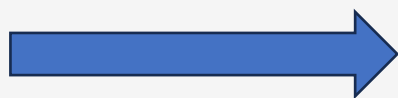
识别数字的神经网络模型

训练 (喂数据, 确定模型参数)

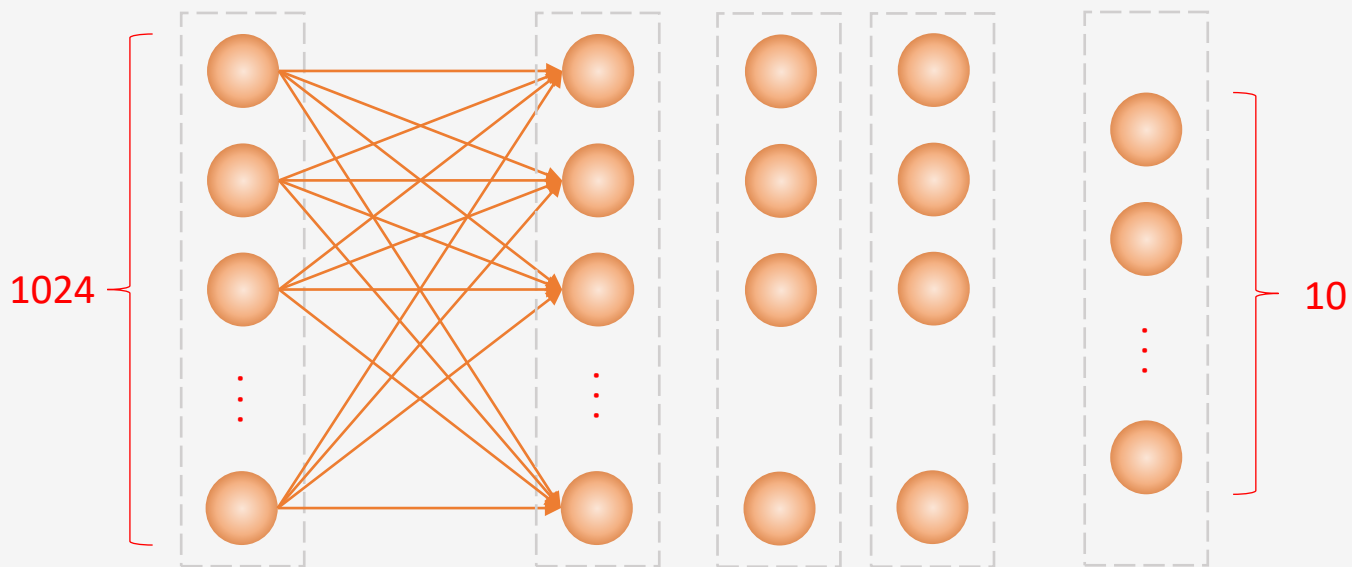
三层全连接神经网络

## 神经网络的参数量

隐藏层的每个神经元，  
输入  $\times w_{ij} + bias_i$ ，再进入  
activation, 1025个参数



1025个参数  $\times$  隐藏层神经元数量



多个隐藏层?

参数量指数增长

全连接带来的

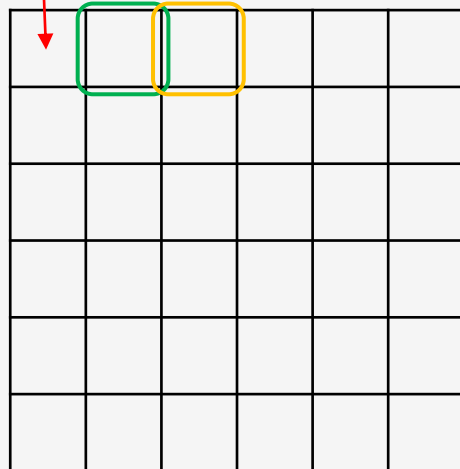
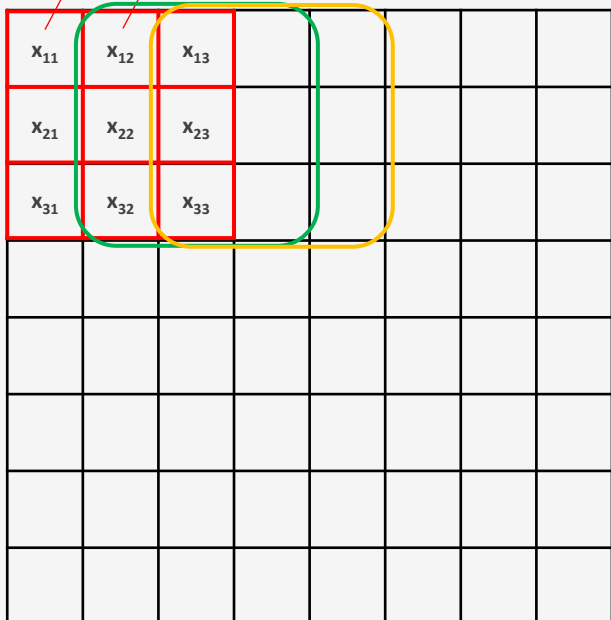
能否不全连接?

能

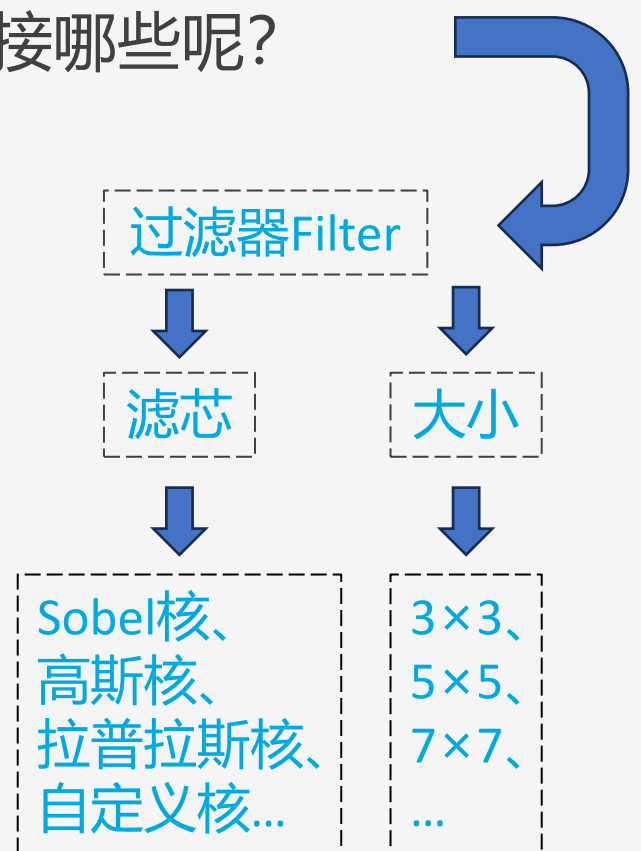
如果不全连接, 应该连接哪些呢?

$w_{11}$	$w_{12}$	$w_{13}$
$w_{21}$	$w_{22}$	$w_{23}$
$w_{31}$	$w_{32}$	$w_{33}$

$$\sum_{i=1, j=1}^{3,3} w_{ij}x_{ij}$$



大大减少参数量



卷积神经网络CNN (convolutional neural network)



## 不全连接会不会影响识别结果?



提取主要特征，过滤冗余信息，  
达到参数量和识别结果的平衡



## 深度学习(DL)

机器学习的子集，通过神经网络自动提取数据的特征并学习



## 迁移学习(TL)

将从一个领域学习到的知识迁移到另一个领域



# 人工智能

## 机器学习(ML)

基于先前数据的特征，使用统计方法自动执行给定任务



## 强化学习(RL)

机器学习的子集，根据环境的反馈学习最优的行为策略

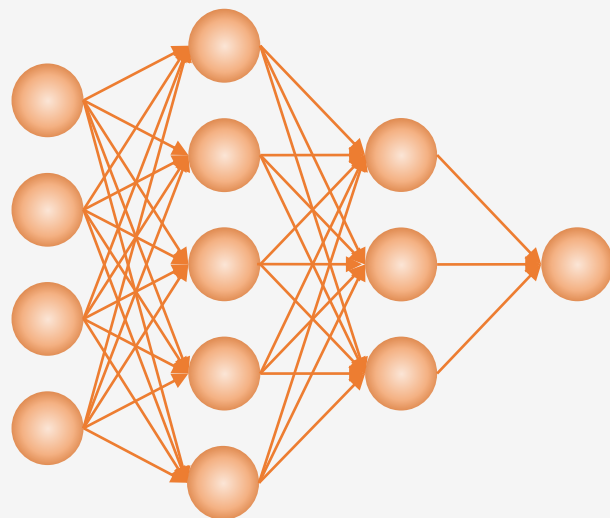




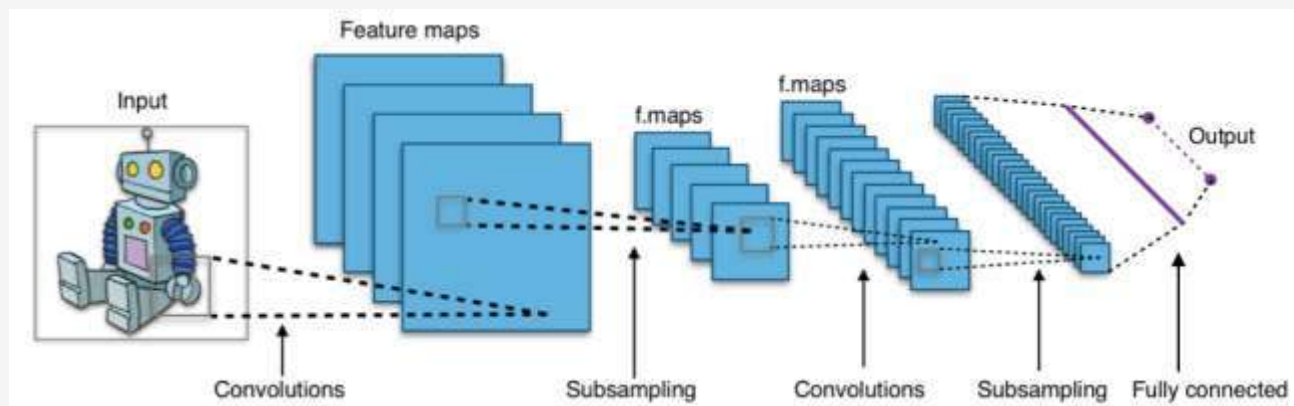


# 常见的神经网络

多层感知器MLP  
(Multi-Layer Perceptron)

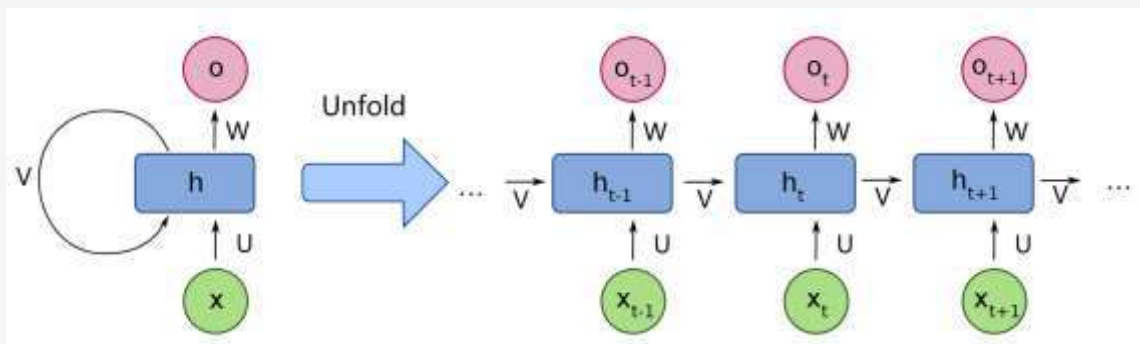


卷积神经网络CNN  
(convolutional neural network)

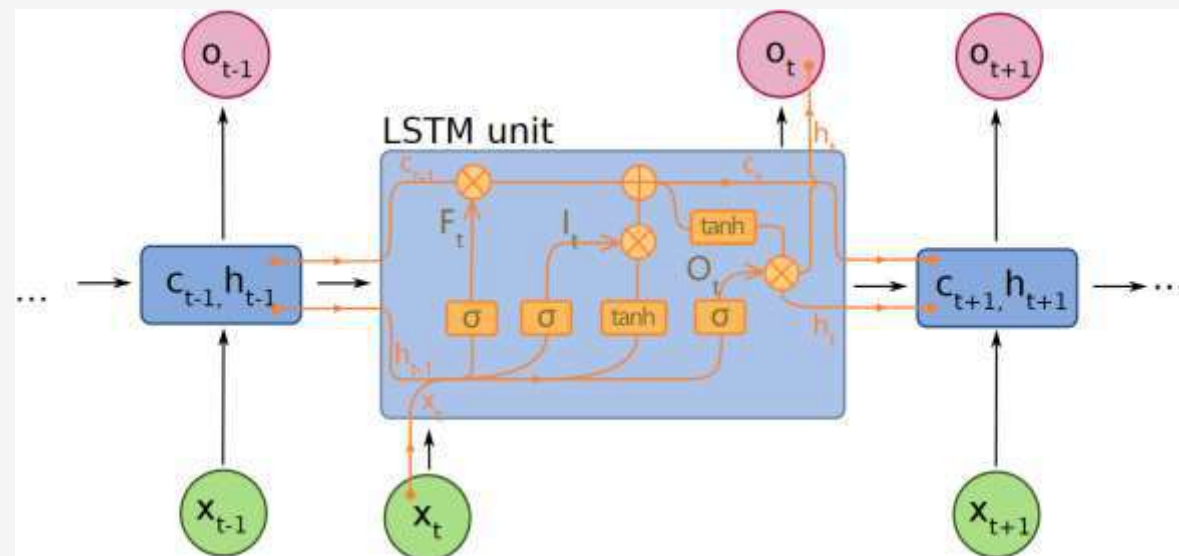


# 常见的神经网络

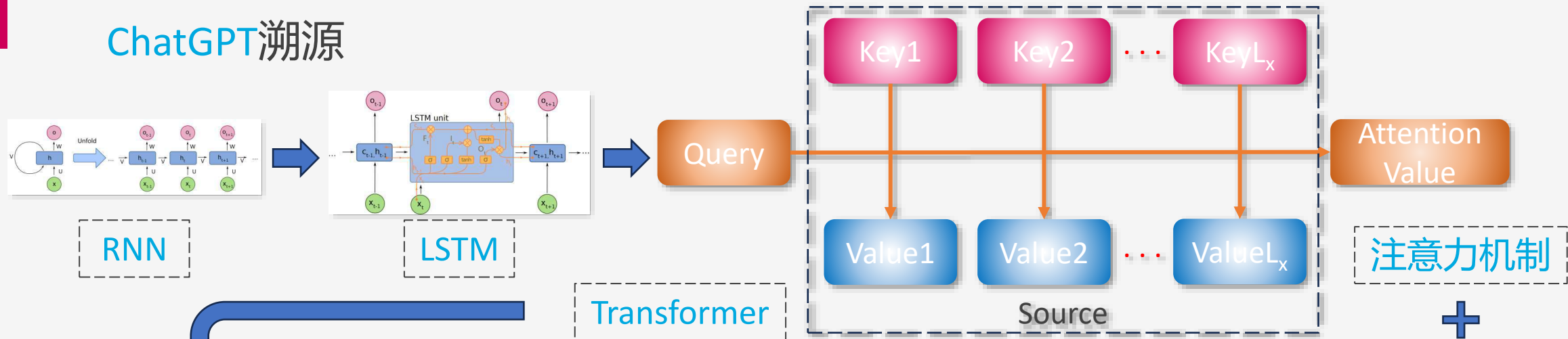
## 循环神经网络RNN (Recurrent Neural Network)



## 长短时记忆网络LSTM (Long Short Term Memory)

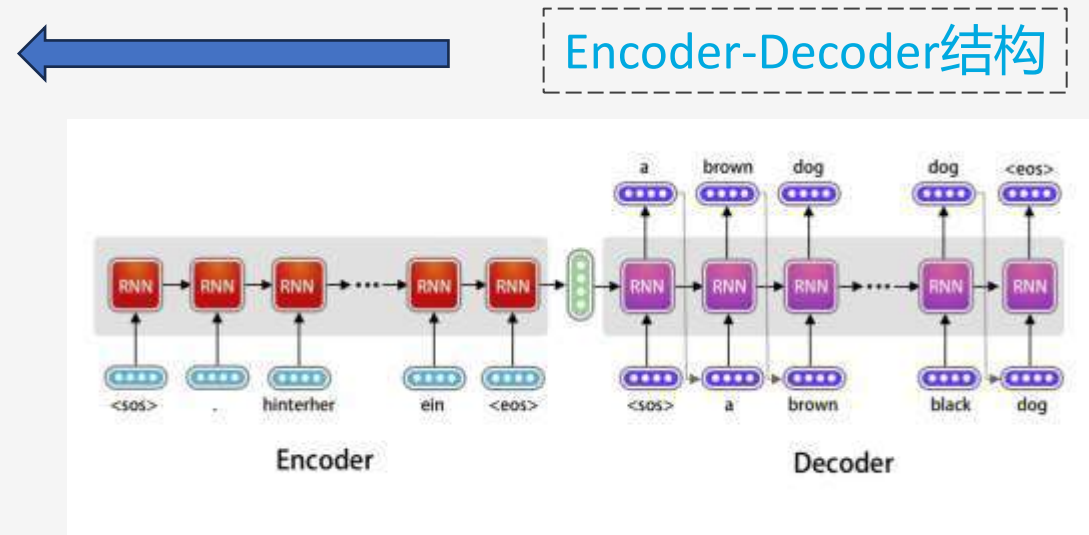
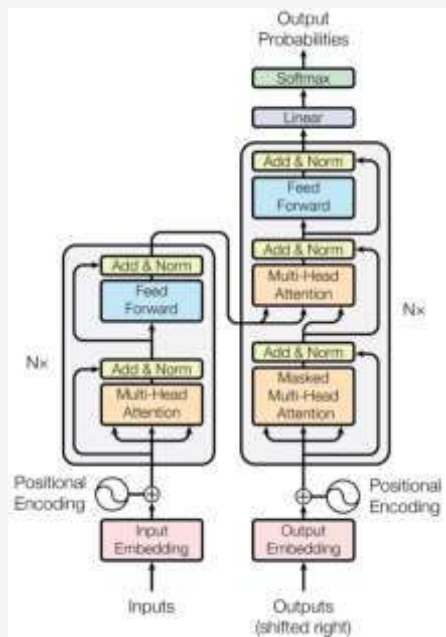


# ChatGPT溯源



**GPT**  
(Generative Pre-Trained Transformer)

**ChatGPT**



## 深度学习框架

 TensorFlow

 PyTorch

 Keras

 飞桨  
PaddlePaddle

 mxnet

需要一定的代码能力和深度学习领域知识

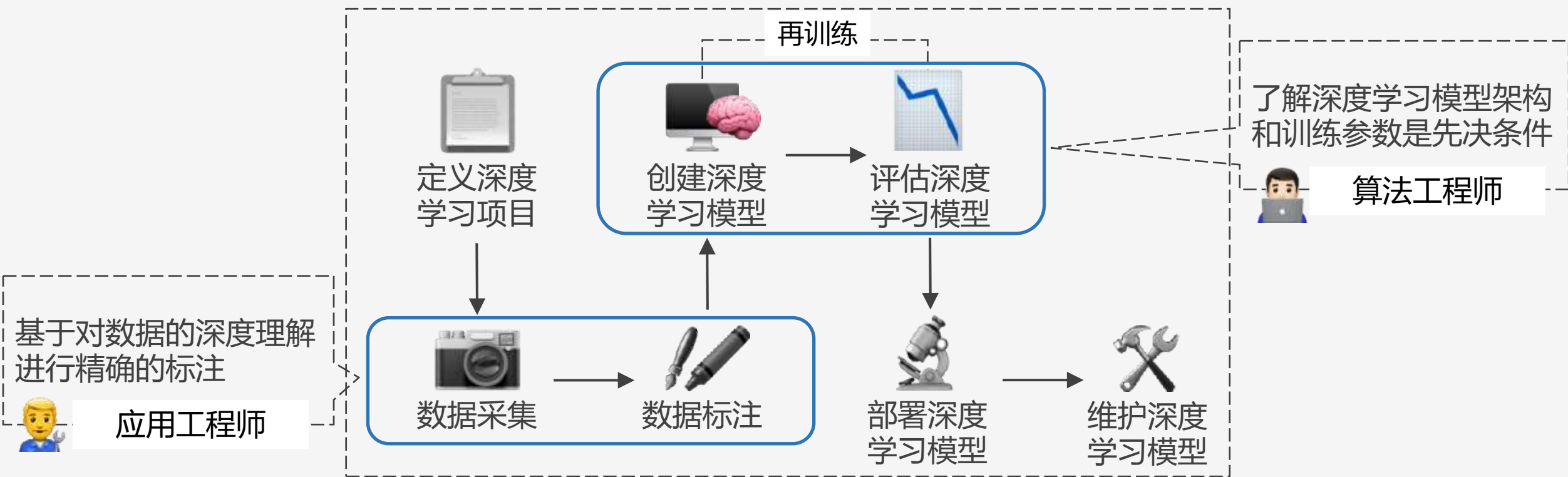


# 03

## 一站式打造卓越的深度学习模型



# 深度学习项目流程



## 深度学习项目流程



应用工程师

高质量的训练数据对于创建高性能的深度学习模型至关重要

创建高性能的深度学习模型需要丰富的专业知识



算法工程师

完成一个深度学习项目，需要丰富经验的应用工程师和算法工程师

## 友思特Neuro-T——深度学习视觉检测项目的一体化平台

自动深度学习算法

自动标注功能



无需AI领域专业知识

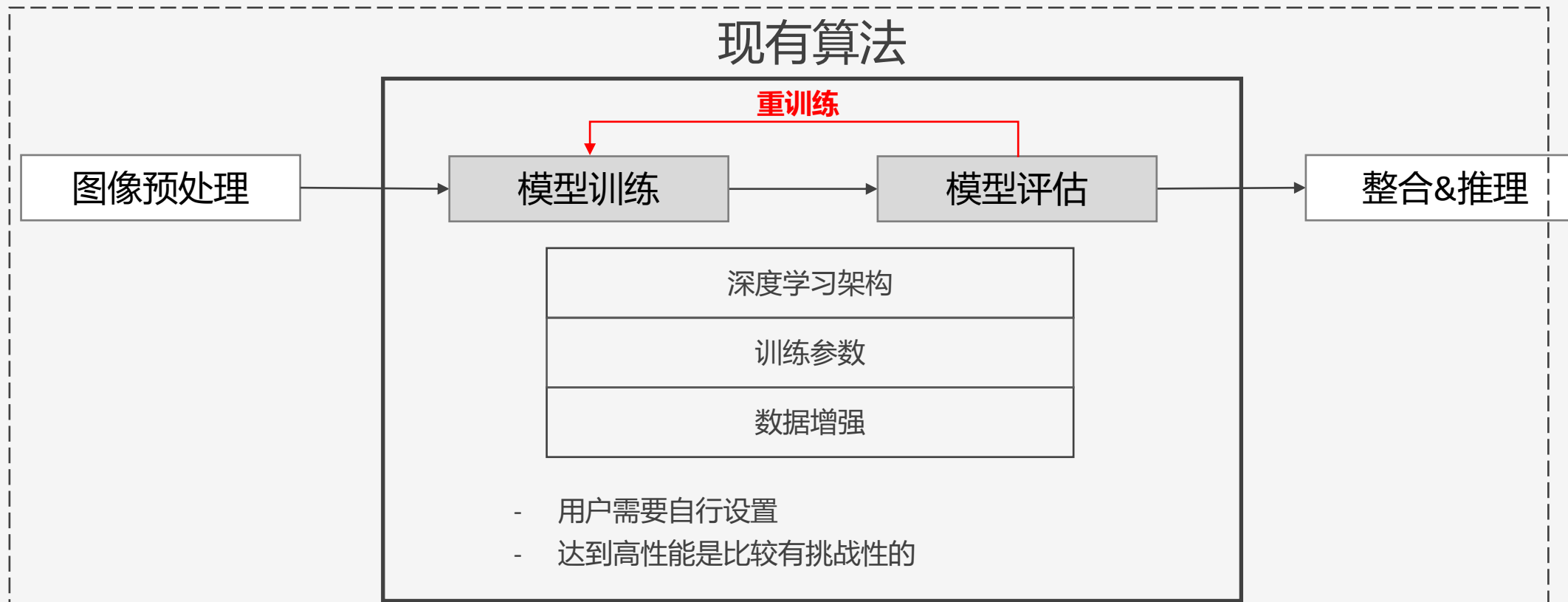
快速搭建深度学习模型





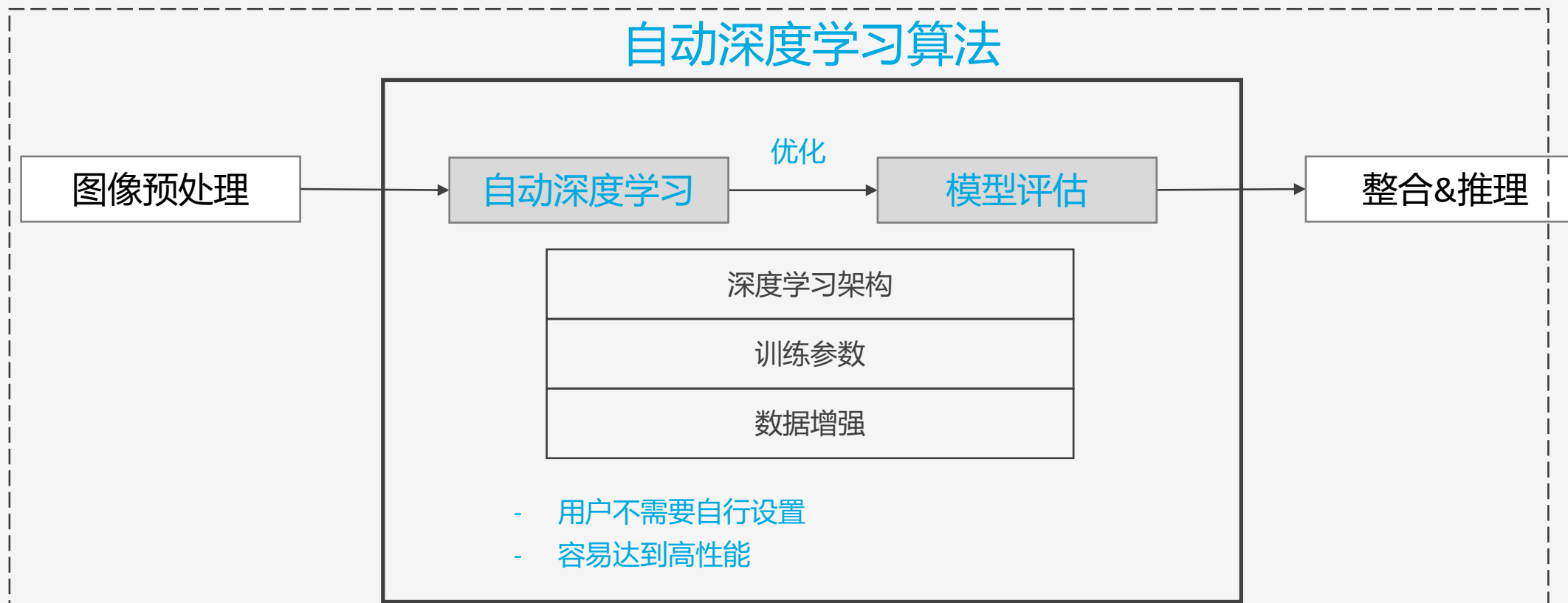
## 友思特Neuro-T 特性一：自动深度学习算法

- 深度学习算法分为：自动深度学习算法和现有算法
- 自动深度学习算法使得每个人都可以轻松地创建高性能的深度学习模型



## 友思特Neuro-T 特性一：自动深度学习算法

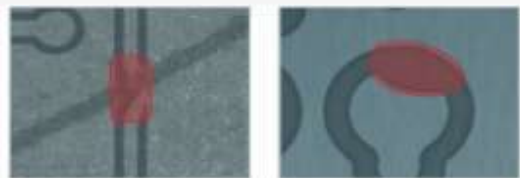
- 深度学习算法分为：自动深度学习算法和现有算法
- 自动深度学习算法使得每个人都可以轻松地创建高性能的深度学习模型



## 友思特Neuro-T 特性二：自动标注

- 在大数据量深度学习任务中，标注任务需要耗费大量时间
- Neuro-T通过自动标注显著缩短项目周期时间，基于用户已标注的数据来保证后续标注的一致性

### Step 01



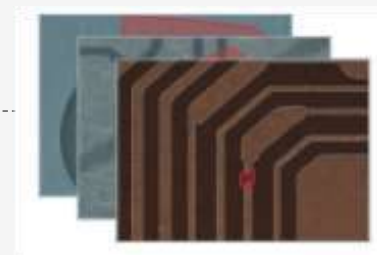
标注整张图像的部分内容

### Step 02



运行自动标注

### Step 03



查看自动标注区域

## 友思特Neuro-T 特性三：本地云环境

- 用户可以在安全的环境中与团队成员协作
- Neuro-T的服务端-客户端架构只允许团队成员共享工作区



## 友思特Neuro-T 特性四：流程图和推理中心

- 流程图可以链接多个不同类型的模型来简化项目设计，如分类+检测模型组合
- 推理中心可以评估项目流程图的推理时间和准确率，从而以更少的尝试和错误创建最佳模型





Start Inference

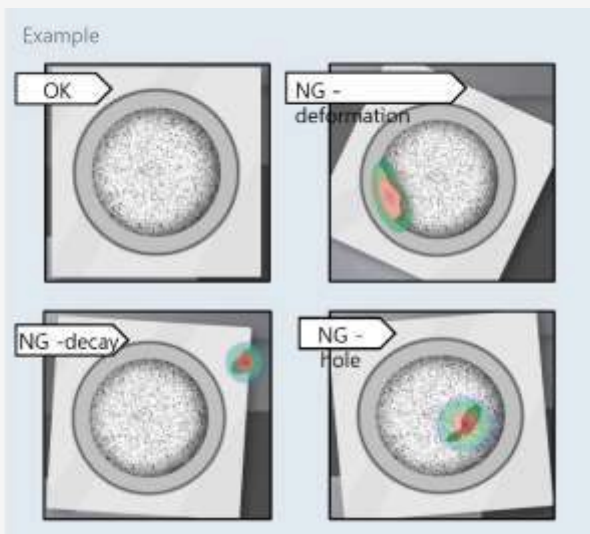


## 友思特Neuro-T 特性五：快速重训练

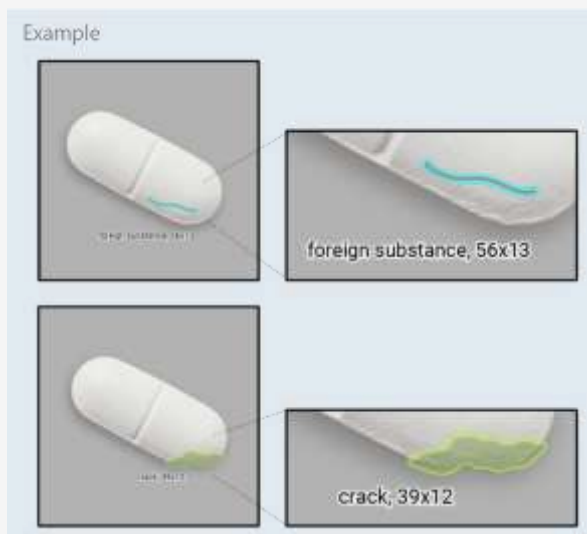
- 如果出现新的缺陷类型或设计修改，需要重新进行训练，且存在时间延迟和效果下降的问题
- Neuro-T通过自动深度学习和平衡数据，以较短的训练时间实现较高的模型精度

	数据量		训练时间	
常规重训练	所有数据 	+	长时间	需要多次尝试 比较长的时间
微调训练 (迁移学习)	附加数据	+	短时间 	不平衡的数据 可能会降低性能
<b>NEURO-T</b>	所有数据	+	短时间	在短训练时间内 实现高性能

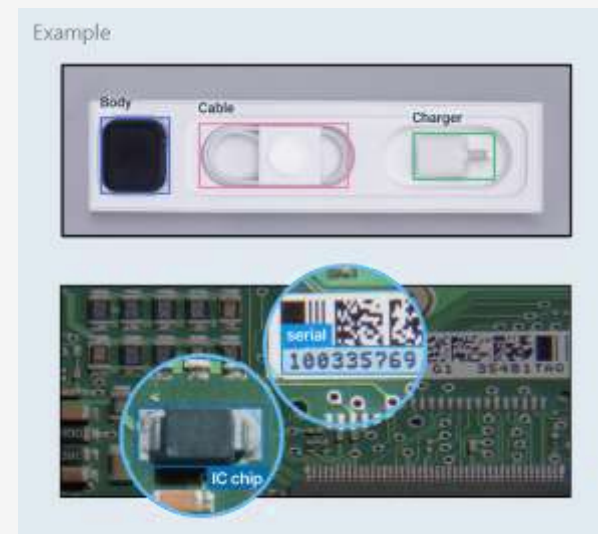
## 友思特Neuro-T 支持模型类型



分类



分割



目标检测

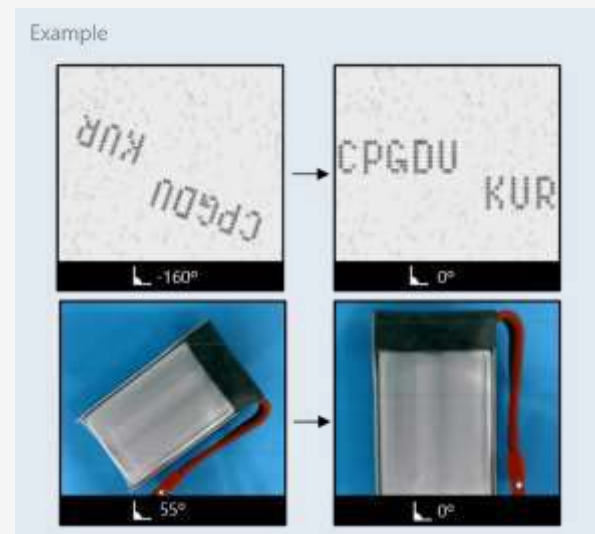
# 友思特Neuro-T 支持模型类型



异常检测



OCR



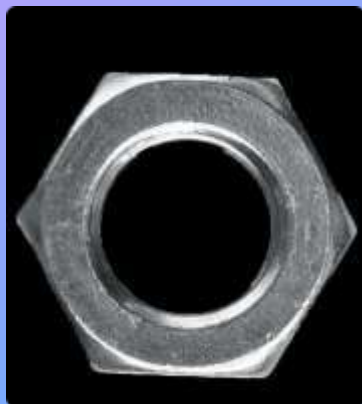
旋转



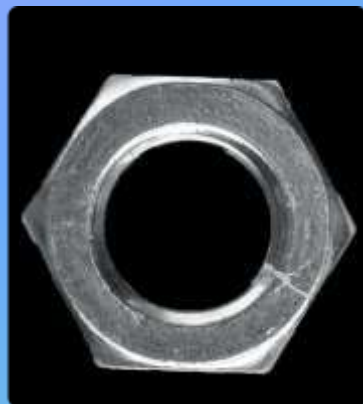


## 友思特Neuro-T 支持模型类型

真图



假图



GAN

有监督学习模型:

- 模型训练数据包括输入图像的特征和标签/目标值
- 训练过程中, 模型尝试通过特征和标签之间的联系来学习如何预测/分类

无监督学习模型:

- 模型训练数据只包含输入图像的特征, 无标签/目标值
- 模型尝试发现数据中的模型或特征关系, 而非预测特定的输出

热图

提取

无监督模型

## 友思特Neuro-T 项目搭建流程

### 新建项目

- 导入数据
- 图像预处理

图像切片(ROI)  
图像增强

### 训练模

- 划分数
- 启动训

### 标注数据

- 选择模型类型
- 标注数据

(1)标注部分数据  
(2)训练标注模型  
(3)自动标注剩下数据

### 结果评估

- 模型效果数据
- 导出模型

准确率(Accuracy)  
精确率(Precision)  
召回率(Recall)  
F1-Score

TP——正确预测为正例 TN——正确预测为负例  
FP——错误预测为正例 FN——错误预测为负例

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

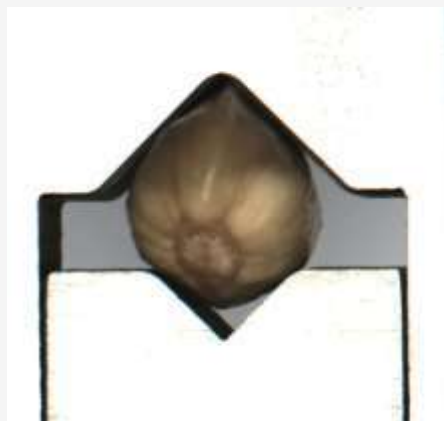
$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$



## 友思特Neuro-T 效果测试——硬件配置

Checklist	My PC
GPU	NVIDIA GeForce RTX 3060Ti / 8GB
O/S	Windows 10 / 64bit
CPU	Intel(R) Core(TM) i7-9700 CPU @ 3.00GHz
RAM	16GB
Disk storage	Free space: 245.9GB
Nvidia GPU driver version	536.67

## 友思特Neuro-T 效果测试——分类模型案例



大头



小头

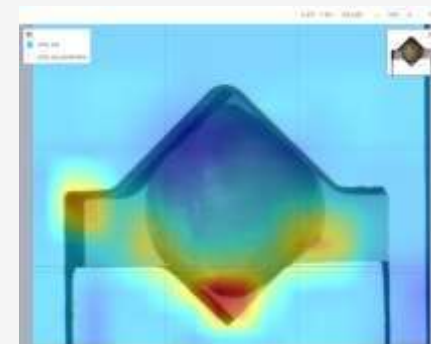
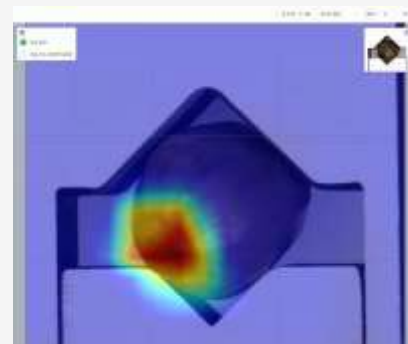
### Metrics



### Labeling information

Label set name	size
Label set type	Classification
ROI/Mask	None

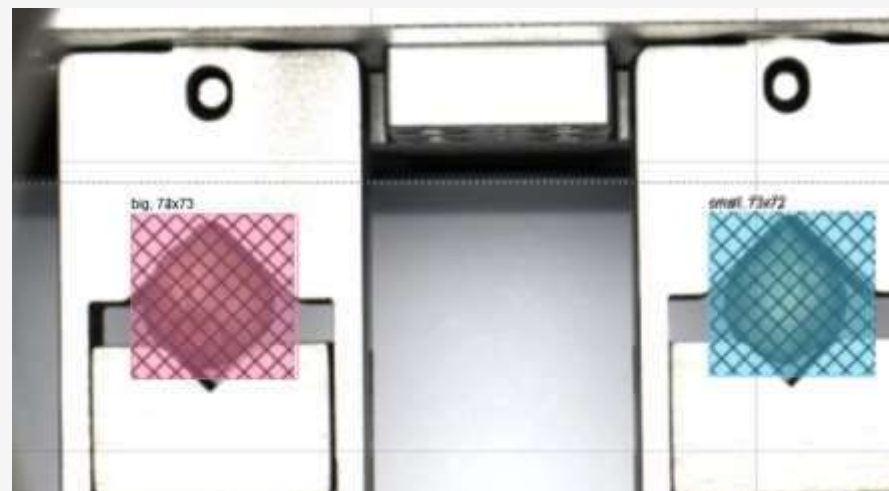
- 按85：15划分训练集和测试集
- 数据量：二分类，375/295张
- 图片尺寸：672\*630
- 训练时长：11min



## 友思特Neuro-T 效果测试——目标检测模型案例1



- 按85：15划分训练集和测试集
- 数据量：141张,手动标注25张,调整6张
- 图片尺寸：512\*384
- 训练时长：11min



## 友思特Neuro-T 效果测试——目标检测模型案例2



## 友思特Neuro-T 效果测试——目标检测模型案例2

- 按85：15划分训练集和测试集
- 数据量：147张,手动标注40张,调整1张
- 图片尺寸：1280\*720
- 训练时长：14min

### Metrics



### Labeling information

Label set name	multi_objs
Label set type	Object Detection
ROI/Mask	None



## 友思特Neuro-T 效果测试——实例分割案例







# 04

## 如何让AI助力工业制造领域



# 如何部署友思特Neuro-T训练出来的深度学习模型





# 友思特Neuro-R——快速部署实时推理API

兼容性强

支持语言



C++



Python

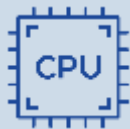


C#

支持平台



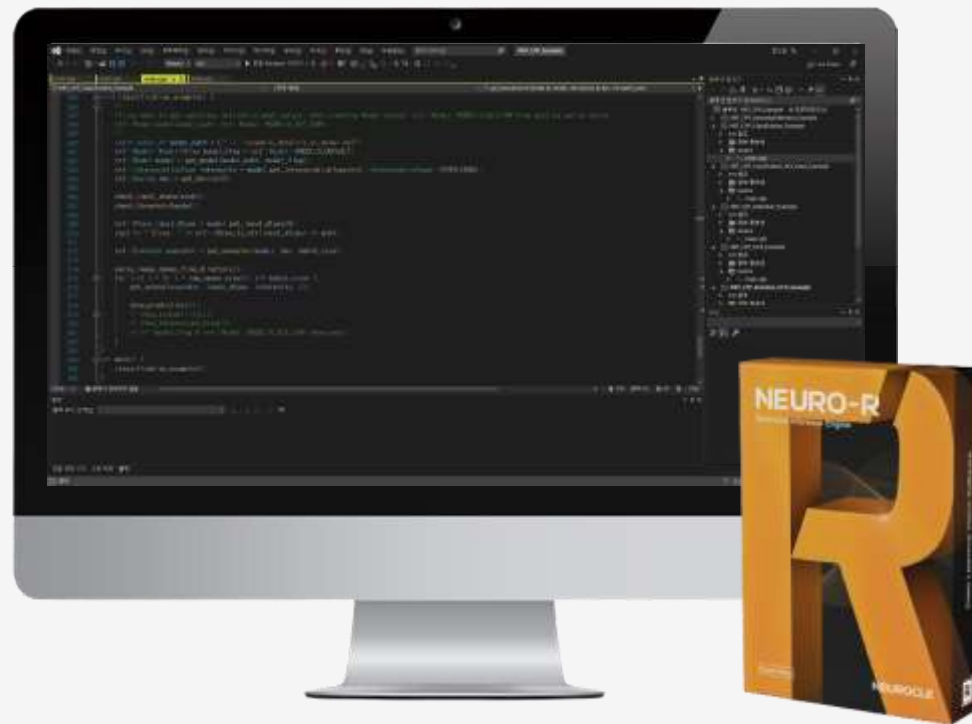
Jetson



CPU



GPU





## 友思特Neuro-R——快速部署实时推理API

### 推理速度快

类型	分类	分割	检测	异常	OCR	旋转
默认	2.2~4.7	3.1~7.2	31.2~34.7	11.8~19.9	65.6~72.9	2.3~5.1
提高推理速度	2.2~3.9	3.1~4.3	31.4~33.7	11.8~15.5	70.3~72.9	2.3~3.8
对于嵌入式设备	2.2~3.2	3.1~4.3	32.1~33.0	11.8~13.8	70.3~70.8	2.3~2.8

输入图像尺寸 (Image Size) : 512 × 512

GPU: 3090

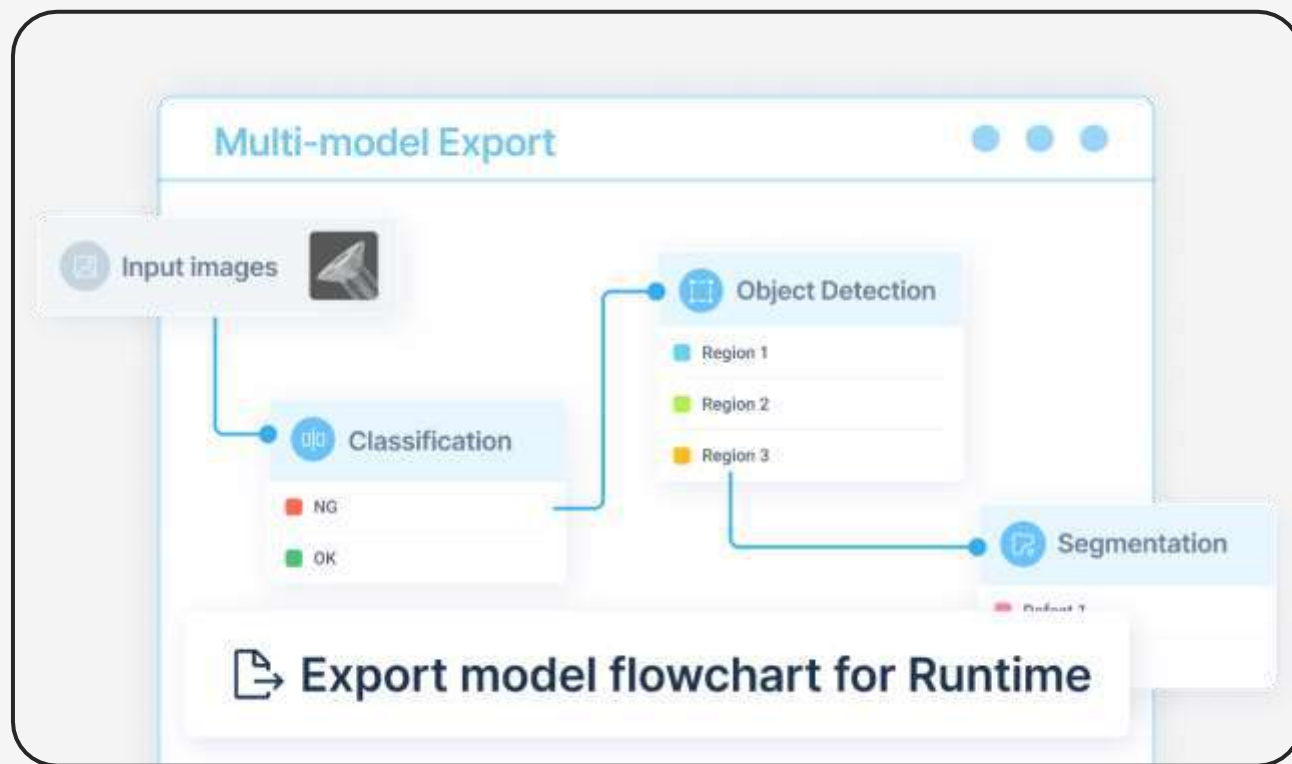
单位: 毫秒 (ms)

- 分类和分割模型相对较小因此推理速度更快，检测和OCR模型相对较大因此推理速度稍慢
- 选项 “Enhance Inference Speed” (提高推理速度) 和 “For Embedded Device” (对于嵌入式设备) 可以提高推理速度。



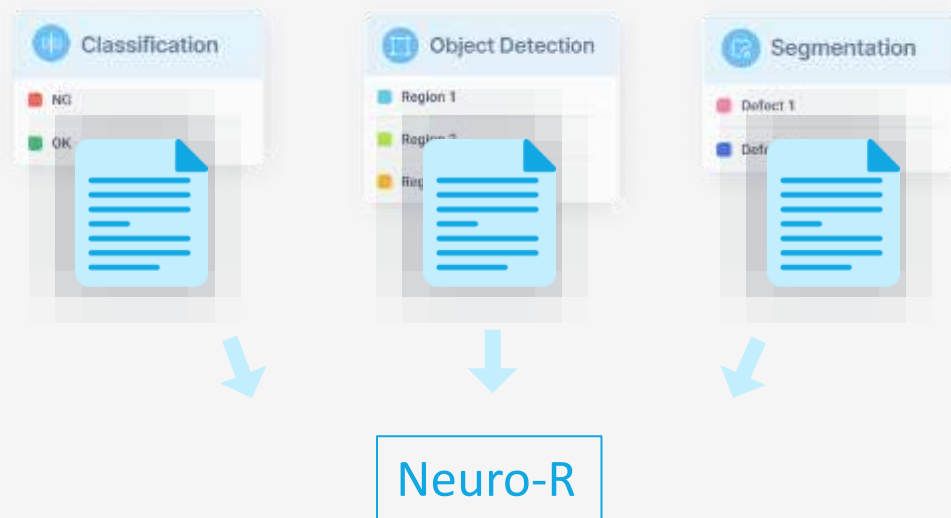
## 部署具体操作（导出模型）

大多数工业应用场景，单个模型具有较大的局限性





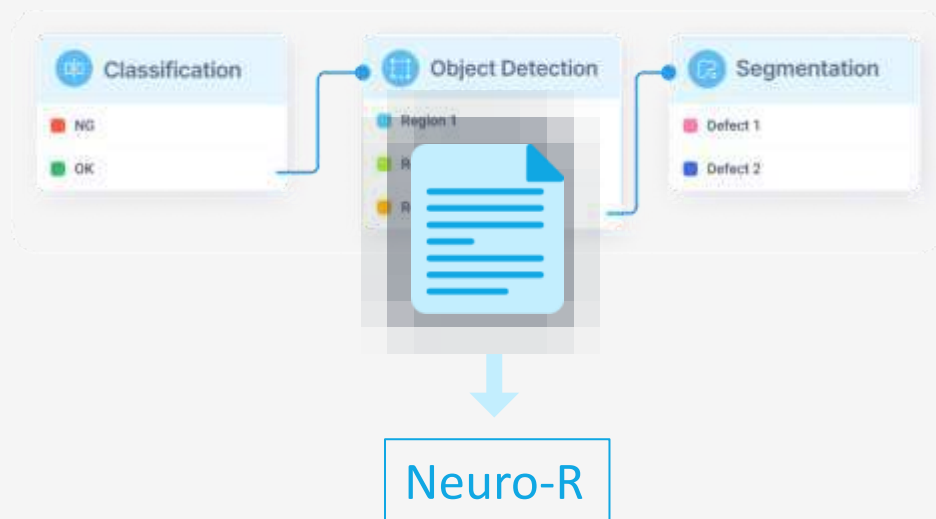
## 部署具体操作（导出模型）



在Neuro-T可以分别导出单个模型，再通过Neuro-R部署，自行设计多个模型的交互逻辑



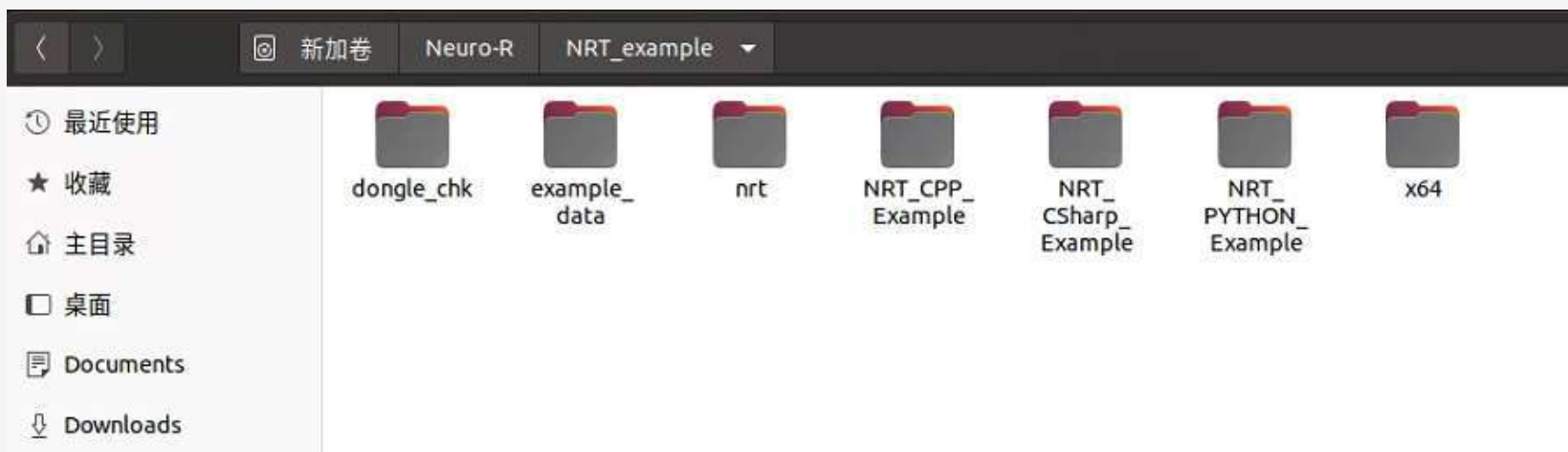
## 部署具体操作（导出模型）



在Neuro-T可以基于flowchart功能链接多个不同的模型，并导出一个单一的文件，通过Neuro-R直接部署



## 部署具体操作（代码示例）



友思特Neuro-R对每一种深度学习模型均提供了C++/C#/Python语言的样例代码，代码位于路径../Neuro-R/NRT\_example中





## 部署具体操作（三步走）

修改深度学习模型路径为自己训练的模型



修改图像数据路径为相机的实时图像流



设备选择和相关参数配置  
(综合推理速度和检测精度进行设置)

## VS code 预览

```
media > hongke > 新窗口1 > Neuro-R > NRT_example > NRT_CPP_Example > NRT_CPP_Detection_Example > main.cpp > ...
422
423     std::cout << "Batch index for probability: " << prob_ptr[0] << "\n";
424     for (int k = 0; k < num_classes; ++k)
425         std::cout << "Probability for class " << k << ": " << prob_ptr[k + 1] << "\n";
426
427     cv::rectangle(cur_imgs[bbox.batch_index],
428                 cv::Point(bbox.box_center_X - bbox.box_width / 2, bbox.box_center_Y - bbox.box_height / 2), // (Left top X point, Left top Y point)
429                 cv::Point(bbox.box_center_X + bbox.box_width / 2 + (bbox.box_width % 2), bbox.box_center_Y + bbox.box_height / 2 + (bbox.box_height % 2)), // (Right bottom X point, Right bottom
430                 // % operations for odd numbers.
431                 color_vector[bbox.class_number],
432                 5
433             );
434
435     for (auto cur_img : cur_imgs) {
436         cv::imshow("w", cur_img);
437         cv::waitKey(0);
438     }
439 }
440
441
442 void detection_example() {
443     string image_dir = "../example_data/det_ex_imgs/";
444     /*
445     Trained Model File Path
446     */
447     const wchar_t* model_path = L"../example_data/det_ex_model.net";
448
449     nrt::Model model = get_model(model_path);
450     nrt::InterpolationType interpoly = model.get_InterpolationType(0);
451     nrt::Device dev = get_device(0);
452
453     check_input_shape(model);
454     check_threshold(model);
455
456     nrt::DType input_dtype = model.get_input_dtype(0);
457     nrt::Executor executor = get_executor(model, dev, batch_size);
458
459     parse_image_names_from_directory();
460     for (int i = 0; i < img_names.size(); i += batch_size) {
461         get_output(executor, input_dtype, interpoly, i);
462         show_prediction(i);
463     }
464 }
465 int main() {
466     detection_example();
467 }
```

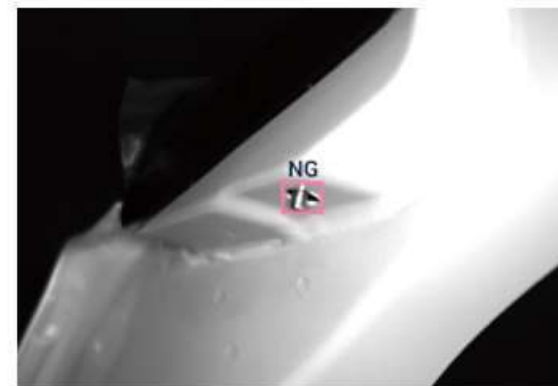
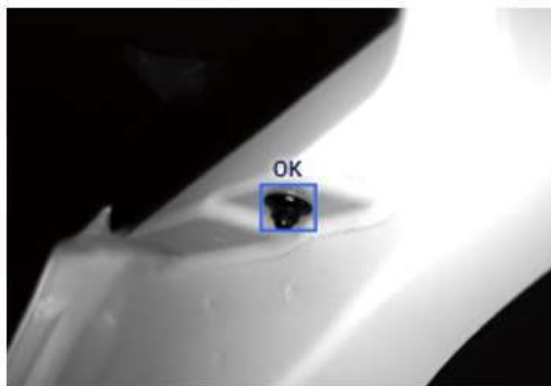


# 汽车用钢材

## 检测要点:

- 汽车表面缺陷检测和装配完成检测
- VIN编号识别
- 材料表面涂层区域的识别
- 无损检测、焊接/卷材/板材检测

螺栓/螺母组件检测



OK

NG

目标	组装完成检测		
现有方法	人工检测	使用的模型	目标检测
训练图像	1400 张	项目工期	2 周
关键成果	——检测时间减少到 1/6 ——实现了检测自动化和无人操作		

项目工期：从图像采集完成到项目结束的时间



# 汽车用钢材

## 检测要点:

- 汽车表面缺陷检测和装配完成检测
- VIN编号识别
- 材料表面涂层区域的识别
- 无损检测、焊接/卷材/板材检测

### VIN 编号识别



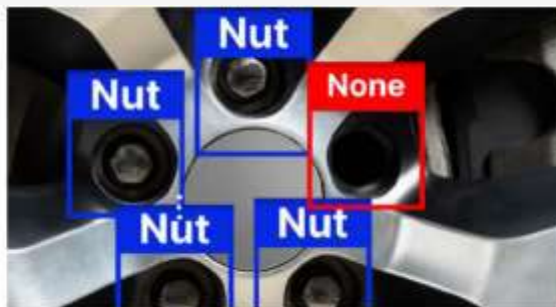
目标	VIN 编码识别和进出口用途分类		
现有方法	人工检测	使用的模型	OCR
训练图像	5000 张	项目工期	3 周
关键成果	——将检测过程由两个步骤简化为一个步骤 ——平均检测准确率达 96.9%		



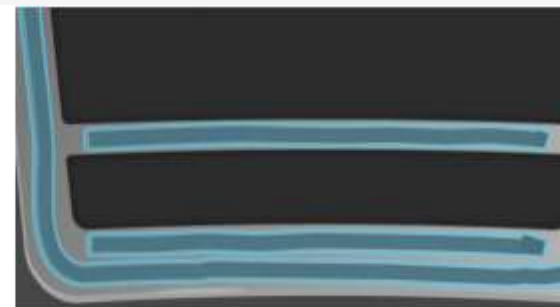
## 其他汽车制造业应用领域



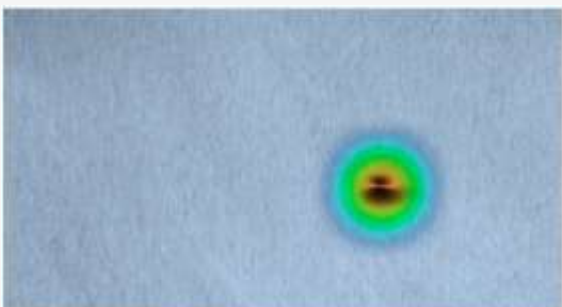
组件检测



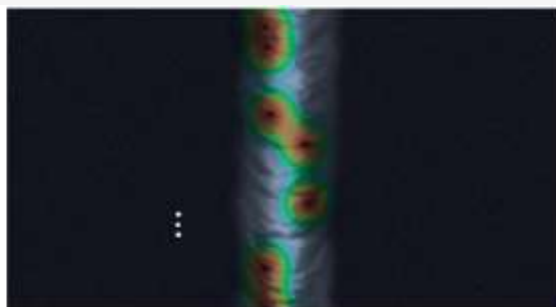
装配检测



天窗粘合剂检测



表面裂纹检测



焊缝缺陷检测



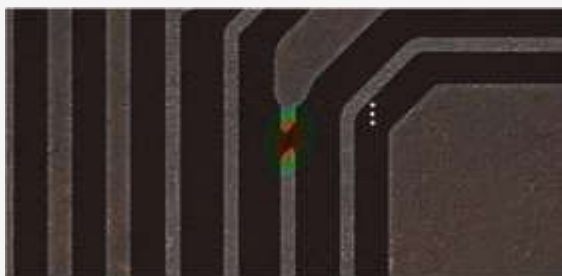
无损检测



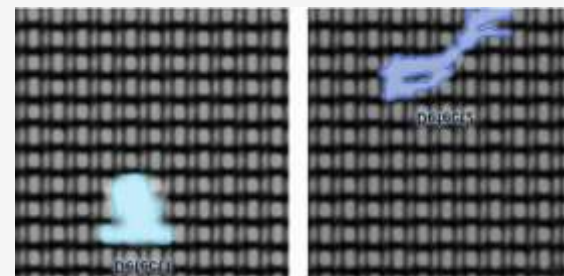
## 其他电子产品行业应用领域



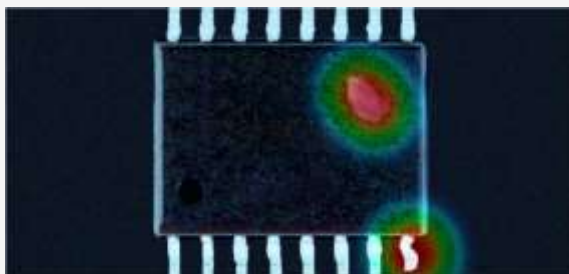
晶圆表面检测



PCB缺陷检测



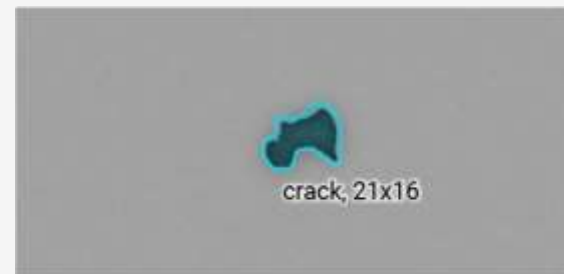
OLED检测



IC芯片检测



太阳能电池检测



玻璃表面检测

# 联系我们

如果您对上述AI软件平台感兴趣，欢迎联系我们！



## 工作时间

周一到周五  
09.00 – 18.00

## 全国免费热线

400-999-3848  
分公司: 广州 | 上海 | 苏州 | 北京 | 西安 | 成都 | 台湾 | 香港 | 日本 | 韩国

## 关注我们

viewsittec.com

## 邮箱

sales@hkaco.com



进一步交流



关注我们



# 提问环节

欢迎大家积极参与~

# THANKS



友思特 机器视觉与光电



viewsitec.com

## 友思特 AI视觉检测应用解决方案



联系讲师 即刻交流

友思特高级技术工程师 黄灿彬

