

MCU助您跨界机器视觉与人工智能

宋岩

ROCKY.SONG@NXP.COM

系统&应用工程师



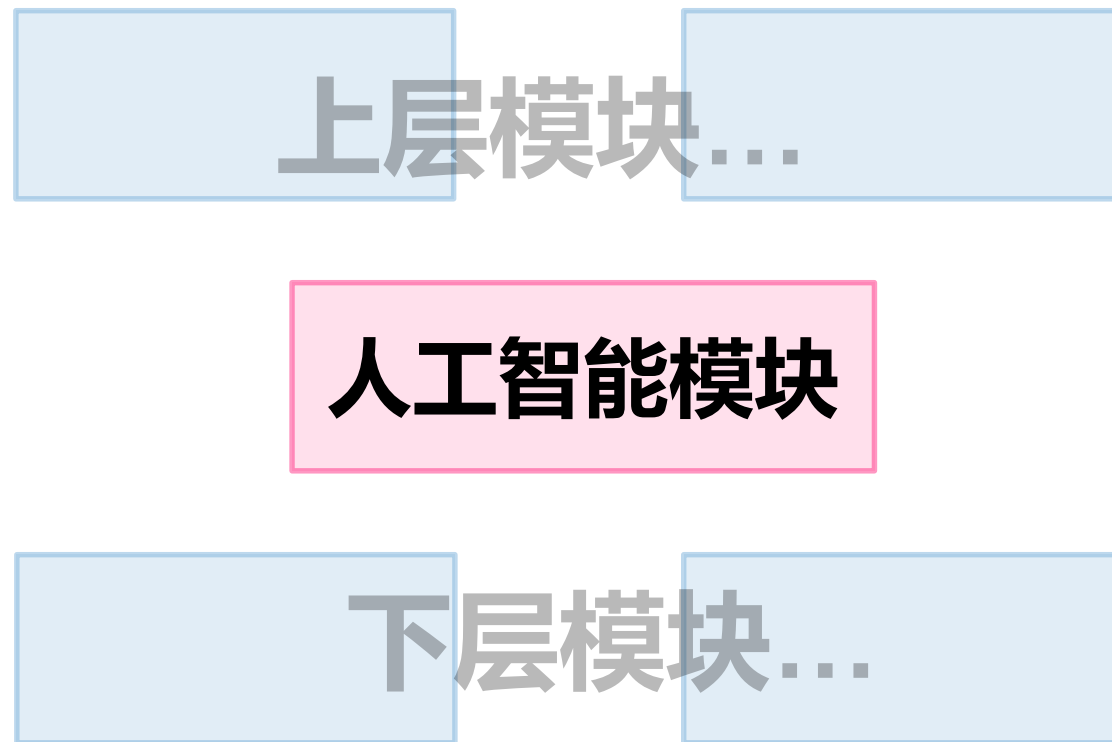
SECURE CONNECTIONS
FOR A SMARTER WORLD

在MCU上应用人工智能

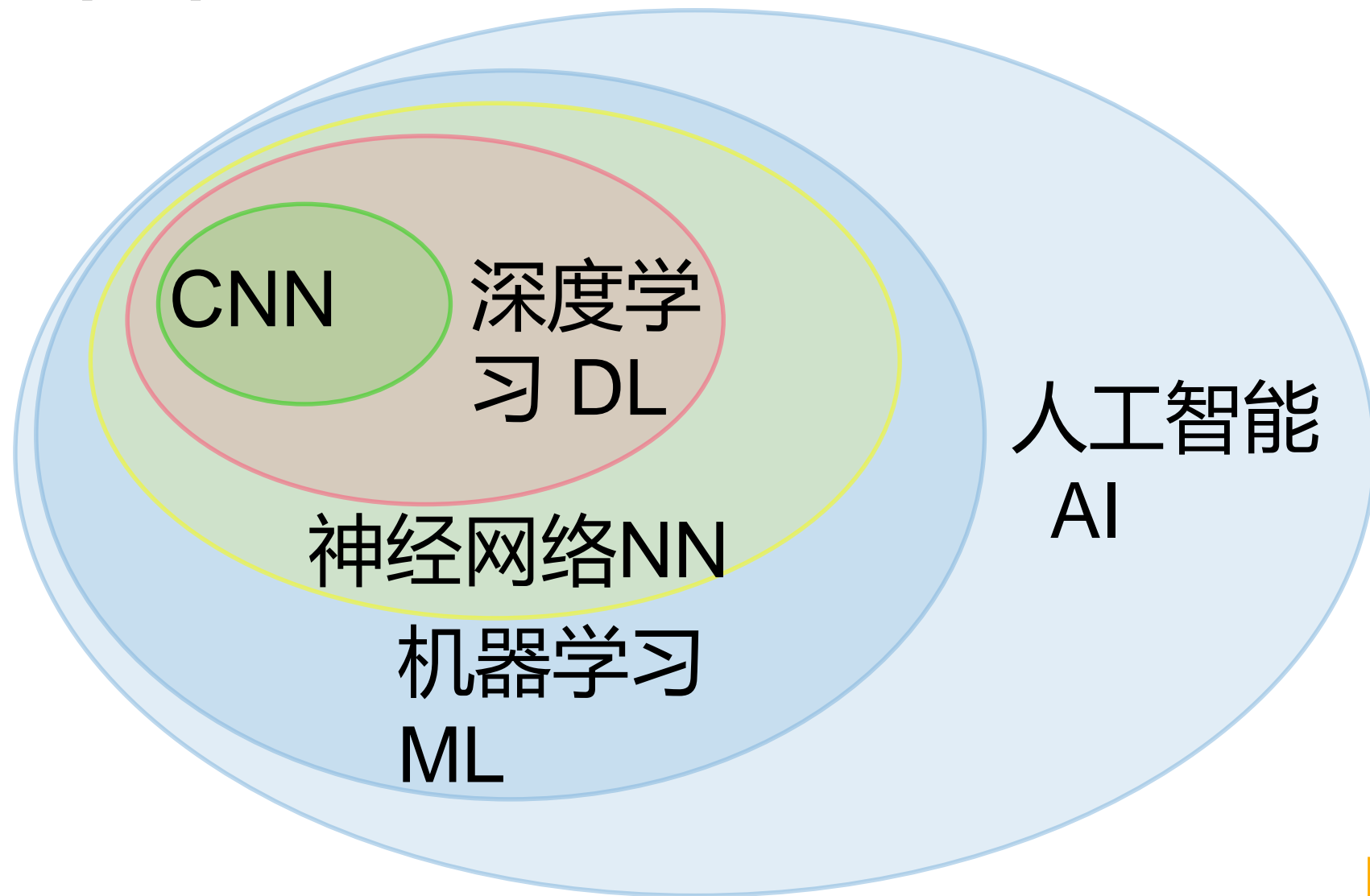


人工智能与嵌入式系统

- 系统是主体
- 人工智能是属性
- 以一个模块来呈现
- 提供新功能
- 亦可改进现有功能



人工智能 (AI)



实现人工智能的演进

xx

从经验(数据) 中
学习的组件

输出

输出

手写的算法

输入

基于规则的
专家系统

比对特征

人工提取和
提炼特征

输入

传统机器
学习(ML)

输出

比对特征

层层相扣，
提炼出更高
级的特征

基本的特征

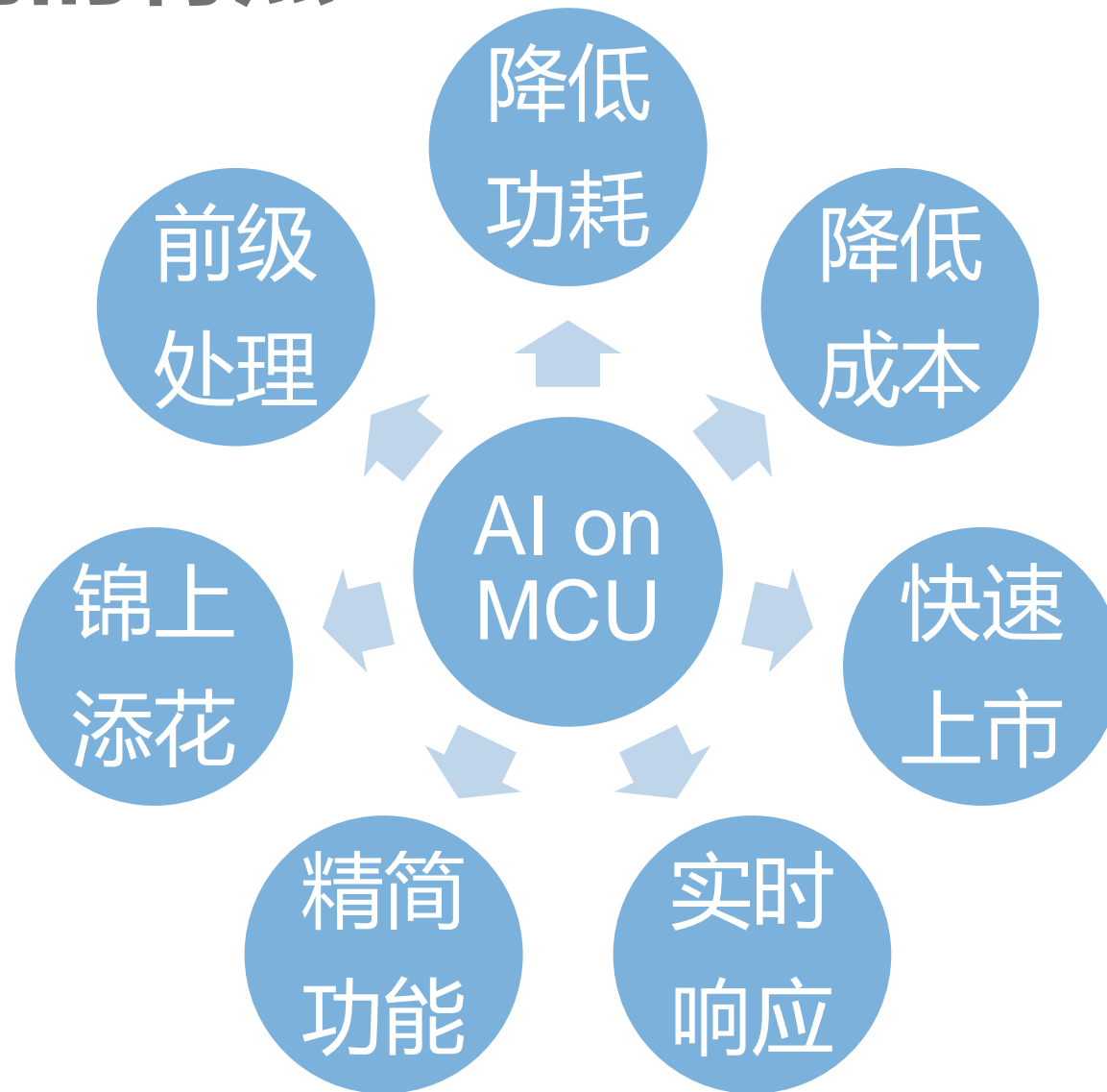
输入

深度学习(DL)

DL模型**自动**找
出特征，多达
成百上千上万!

DL模型基于多
层**神经网络**

MCU上AI应用的特点



AI为MCU应用“属性加成”

- 强化图像分类性能

- 智能电器
- 工业次品检测
- 智能家居

- 强化音频分类性能

- 口令识别

- 改进控制算法的性能

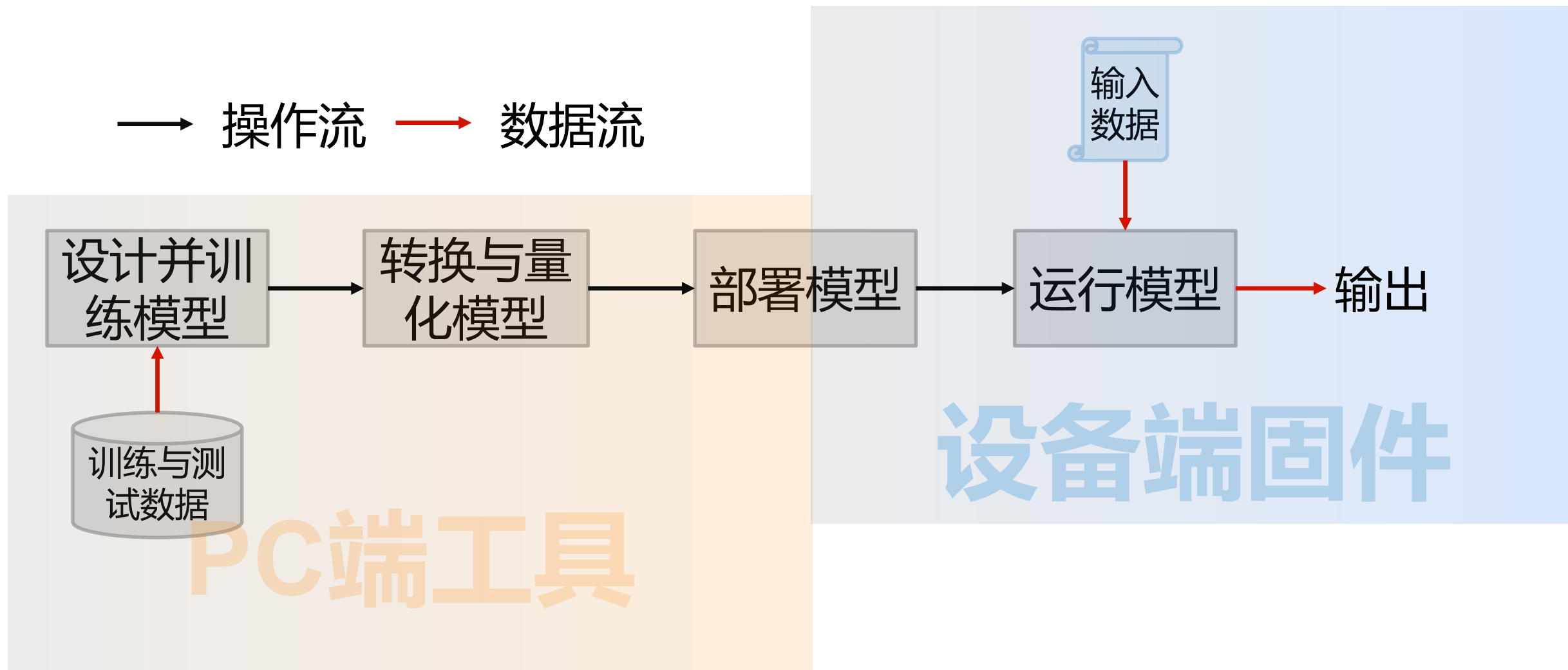
- 电机与传动控制
- 电源转换
- 运动控制

- 异常状态与故障检测

- 部件损坏
- 过程稳定性
- 意外：跌落、碰撞
- 生理信号

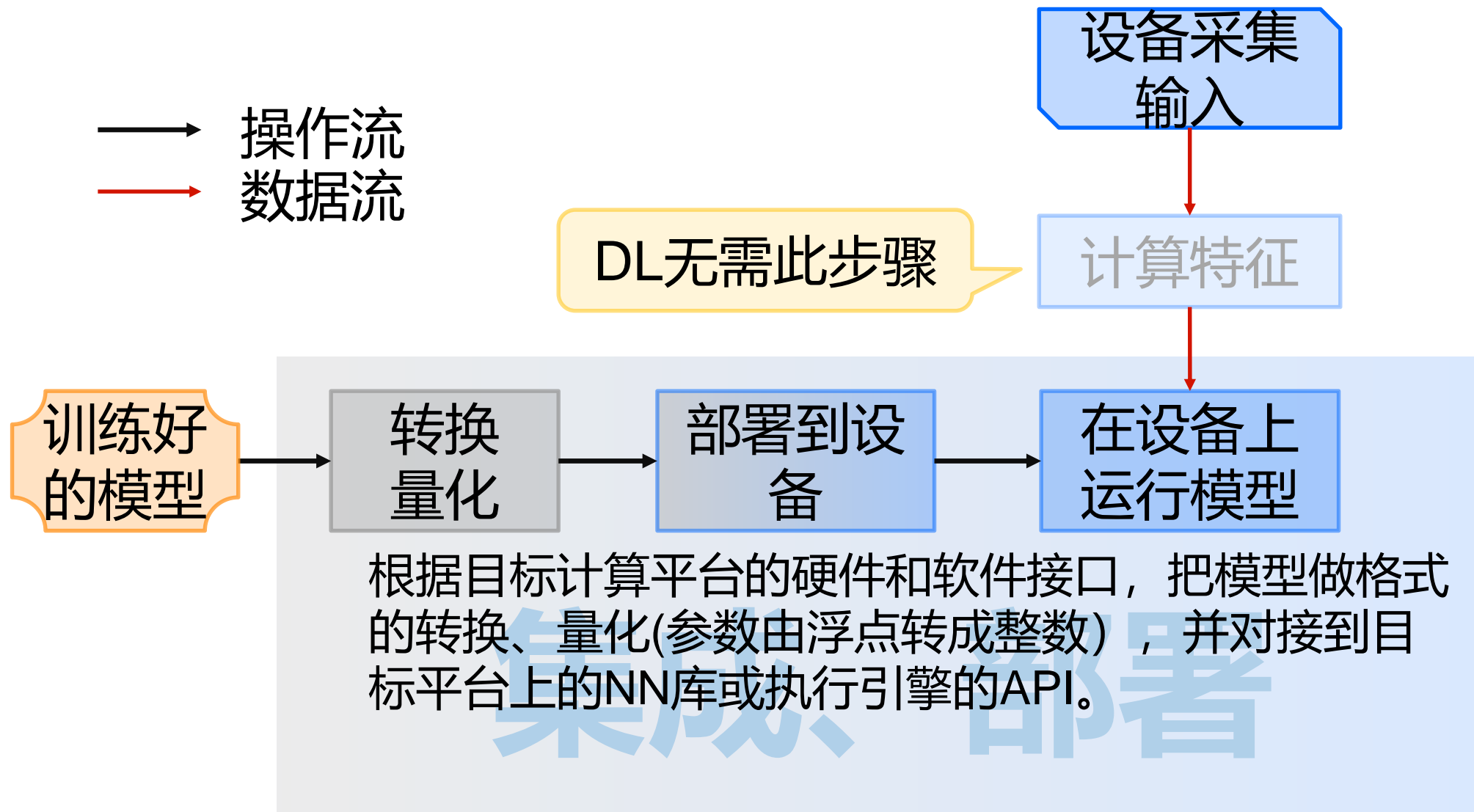
在MCU上集成AI的整体流程

→ 操作流 → 数据流

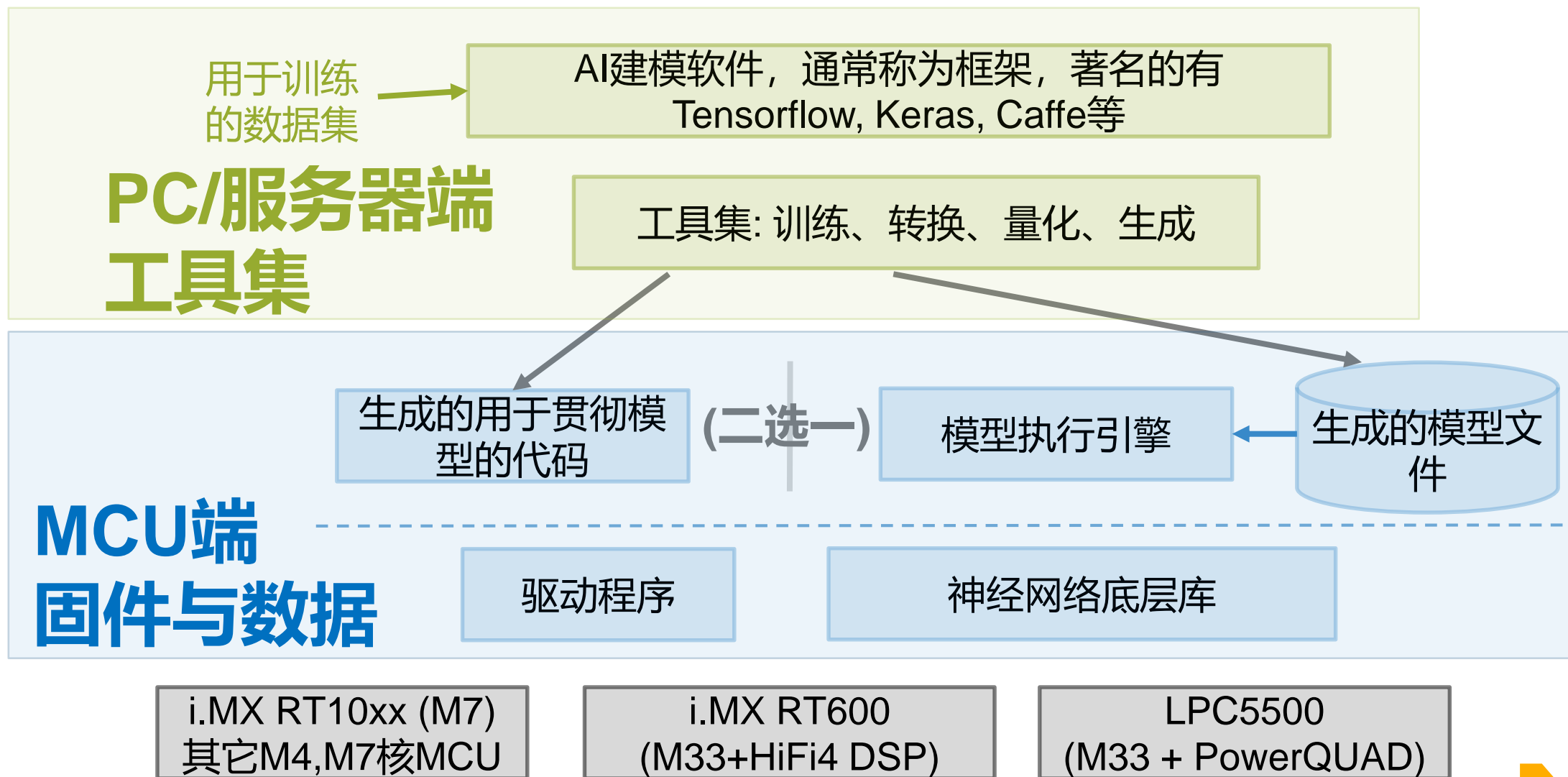


部署过程

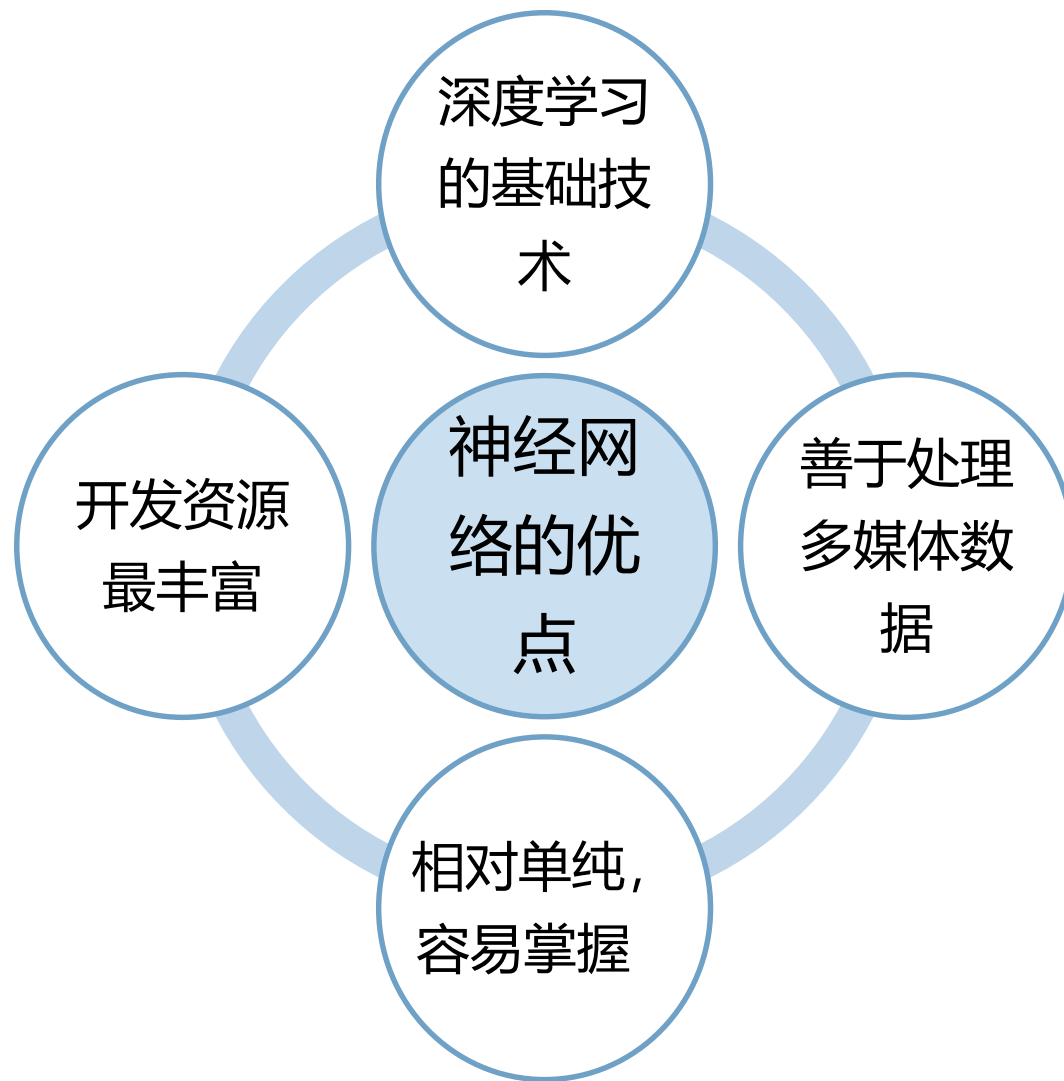
—→ 操作流
—→ 数据流



MCU上部署AI的配套工具



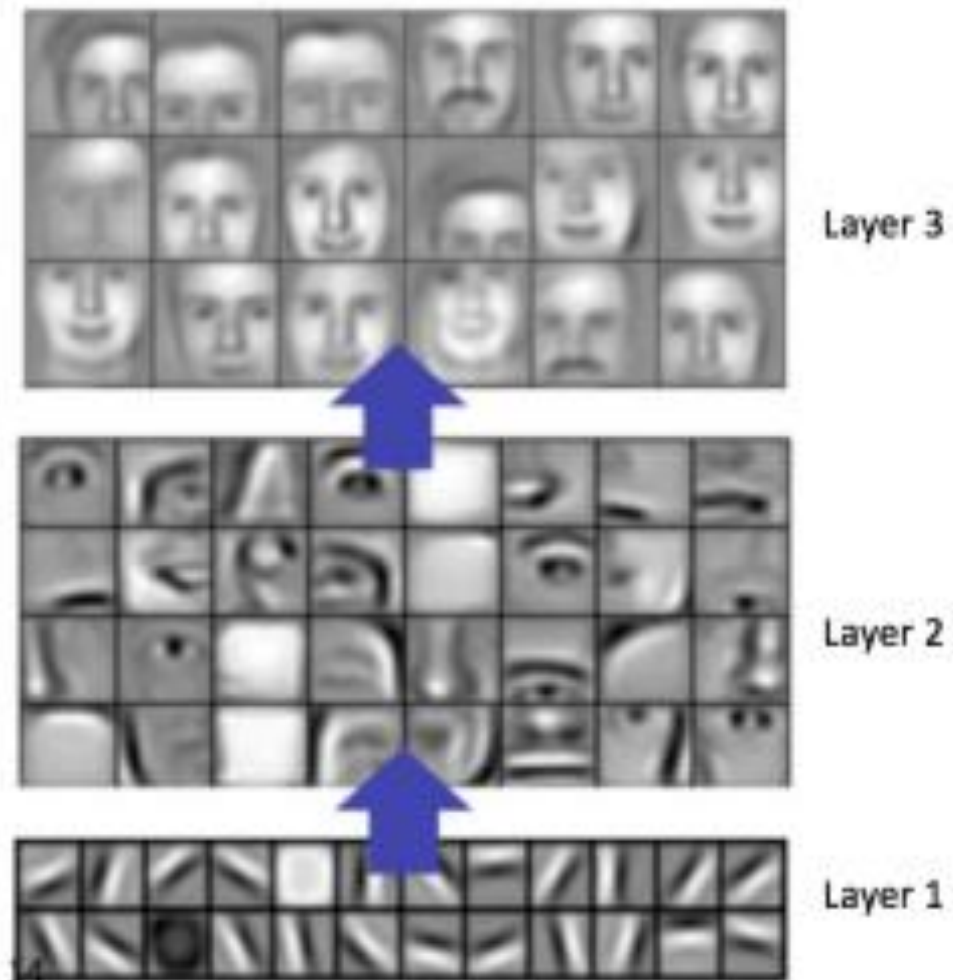
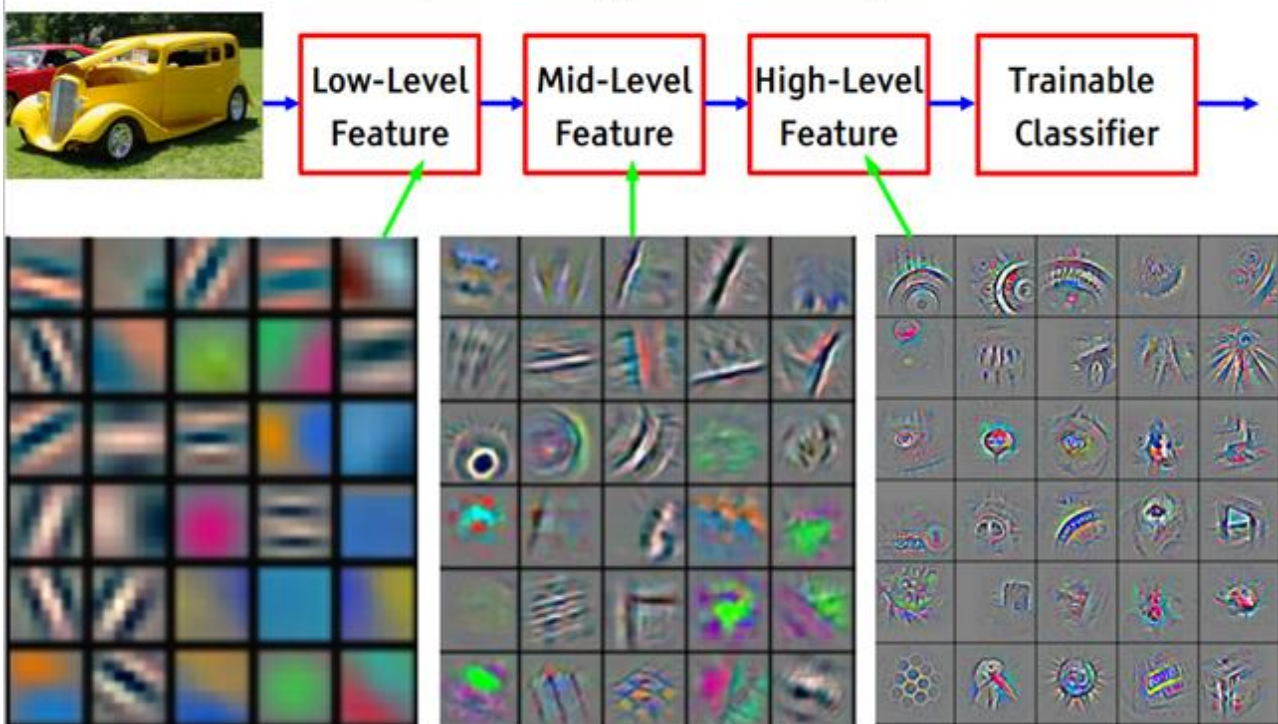
MCU上基于神经网络(NN)建模的优点



深度神经网络工作原理模式图

- 提取初级特征：边角、线条等
- 提取中级特征：部件的轮廓
- 提取高级特征：部件
- 根据不同类别中各高级特征的重要性分类

State of the art object recognition using CNNs



适合在MCU端使用的NN基本构建块

主运算部

常规卷积层 (CNN)

CNN按空间与通道分解
(DS-CNN)

全连接层 (Dense/FC/IP)

基本构建块

先做主运算

常常再做后加工

(多数情况)后加工

激活 - 表达非线性关系

下采样/池化 - 精炼特征

Softmax - 份量转成概率

构建块的常用搭建方式

串联/直筒式：一层一层往上迭

- 最简单的结构，如同“糖葫芦”。微型/小型网络的首选

直筒 + 跳跃：某层的输出又加到后续几层后

- 胜任较复杂的问题，如人脸识别。如MobileNet V2

化整为零：大型构建块做并联/串联分解

- 处理大型问题，在MCU上耗时较长。如各代inception

预制复合结构

- 若干层构建块组成一个复合单元，后者再串在一起：简化设计，灵活多用

MCU端运行神经网络的基础软件

CMSIS-NN

- 仅是底层NN库，需另行生成上层代码或提供执行引擎
- 针对Cortex-M高度优化
- 截止至2018.11仅支持CNN, DS-CNN, FC层
- 不支持并联结构
- 仅用于Cortex-M

TensorFlow Lite

- 自带执行引擎和底层NN库
- 性能远不如CMSIS-NN
- 支持丰富的神经网络构建块与搭建方式
- 支持多种嵌入式平台

其它(制作中)

- HiFi4
- PowerQUAD
- 专用加速引擎

人工智能在机器视觉上的 示例

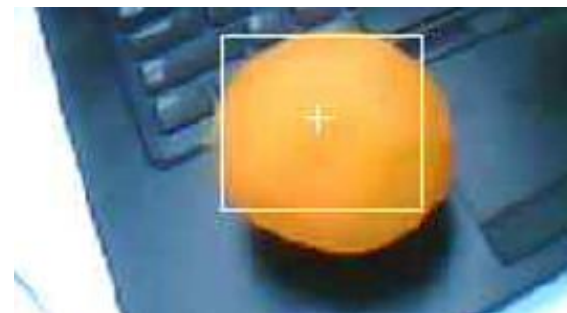
机器视觉的体验平台 – OpenMV RT

- OpenMV: 开源的可编程摄像头项目
 - 以MIT许可证开源了软件、固件、硬件
 - 支持众多机器视觉功能和卷积神经网络执行引擎
 - 可编程: 集成了micropython环境, 并把功能包装成Python模块和类
 - 专用IDE: 开发Python应用脚本, 实时监控处理结果
- OpenMV RT是在i.MX RT1050/60上的移植
 - 大幅提升算法性能
 - 接口与官方兼容, 重用官方开发工具



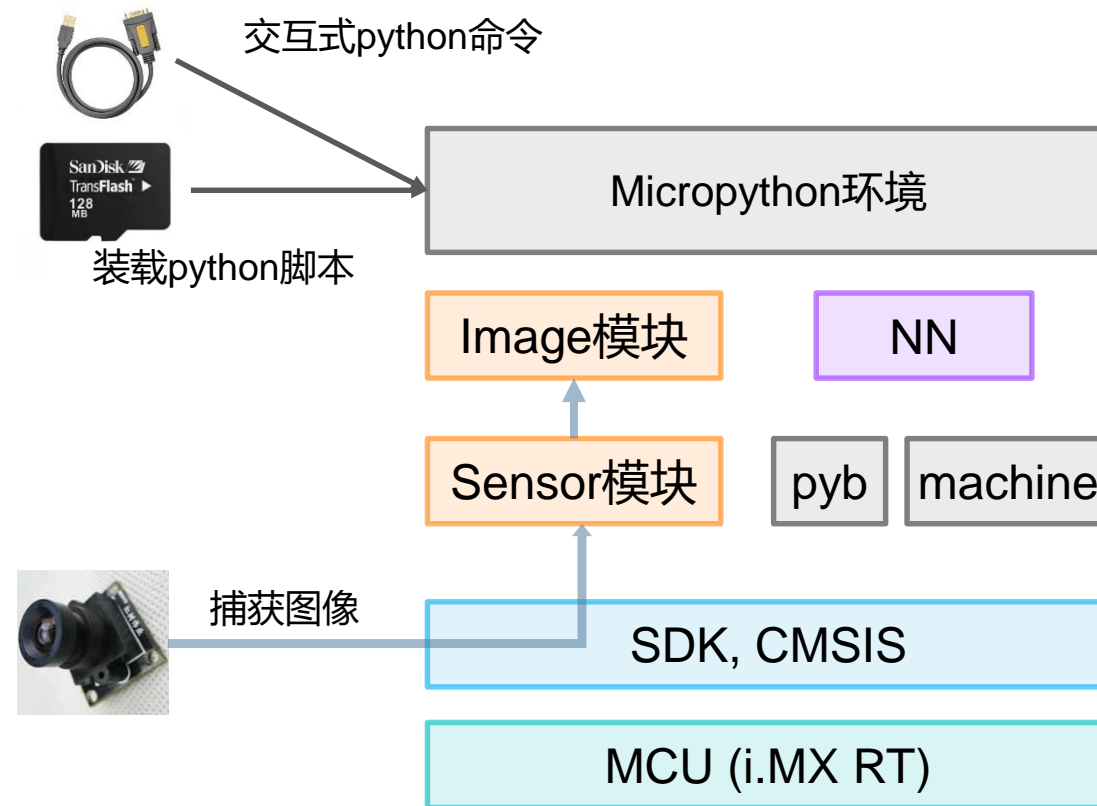
OpenMV 的主要功能

- 使用Python编程MCU
- 基本绘图
- 图像采集与视频录制
- 数字图像处理
- 特征检测
- 脸部检测
- 色块检测
- 条码/二维码检测
- **神经网络执行引擎**



OpenMV RT + CNN执行平台

- 使用Micropython开发应用，包含以下关键Python模块
 - (micropython & pyboard 标准 API) **pyb,machine**: 控制MCU通用资源
 - (openMV) **sensor**: 控制摄像头
 - (openMV) **Image** : openMV核心APIs
 - (OpenMV) **NN**: 神经网络模块.
- NN模块读取并执行SD卡中的模型，输入数据来自摄像头。支持串行CNN模型。



mpy porting repository
<https://github.com/RockySong/micropython-rocky>



OpenMV RT上“CIFAR-10”十项物体分类CNN演示

nn.py - OpenMV IDE

```
nn.py*
4 sensor.set_contrast(3)
5 sensor.set_pixformat(sensor.RGB565) # Set pixel format to RGB565 (or GRAY)
6 sensor.set_framesize(sensor.QVGA) # Set frame size to QVGA (320x240)
7 sensor.set_windowing((192, 192)) # Set window
8 sensor.set_framerate(2<<9|2<<11)
9 sensor.skip_frames(time = 100) # Wait for settings take effect.
10 #net = nn.load('/cifar10.network')
11 net = nn.load('/cifar10_fast.network')
12 labels = ['plane', 'auto', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'st
13 clock = time.clock()
14 tAvg = 40
15 while(True):
16     clock.tick()
17     img = sensor.snapshot()
18     t0 = time.ticks()
19     lst = net.search(img, threshold=0.595, min_scale=1, scale_mul=0.8, \
20 x_overlap=-1, y_overlap=-1, contrast_threshold=0.5)
21     t1 = time.ticks() - t0
22     tAvg = (tAvg * 200 + t1 * 56) >> 8
23     for obj in lst:
24         print(' %s - Confidence %f%%' % (labels[obj.index()], obj.value()))
25         rc = obj.rect()
26         img.draw_rectangle(rc, color=(255,255,255))
27         img.draw_string(rc[0], rc[1], labels[obj.index()])
28     print('infer fps=%f' % (1000.0 / tAvg))
29
```

识别结果

bird

直方图 LAB色彩空间 Res (w:192, h:192)

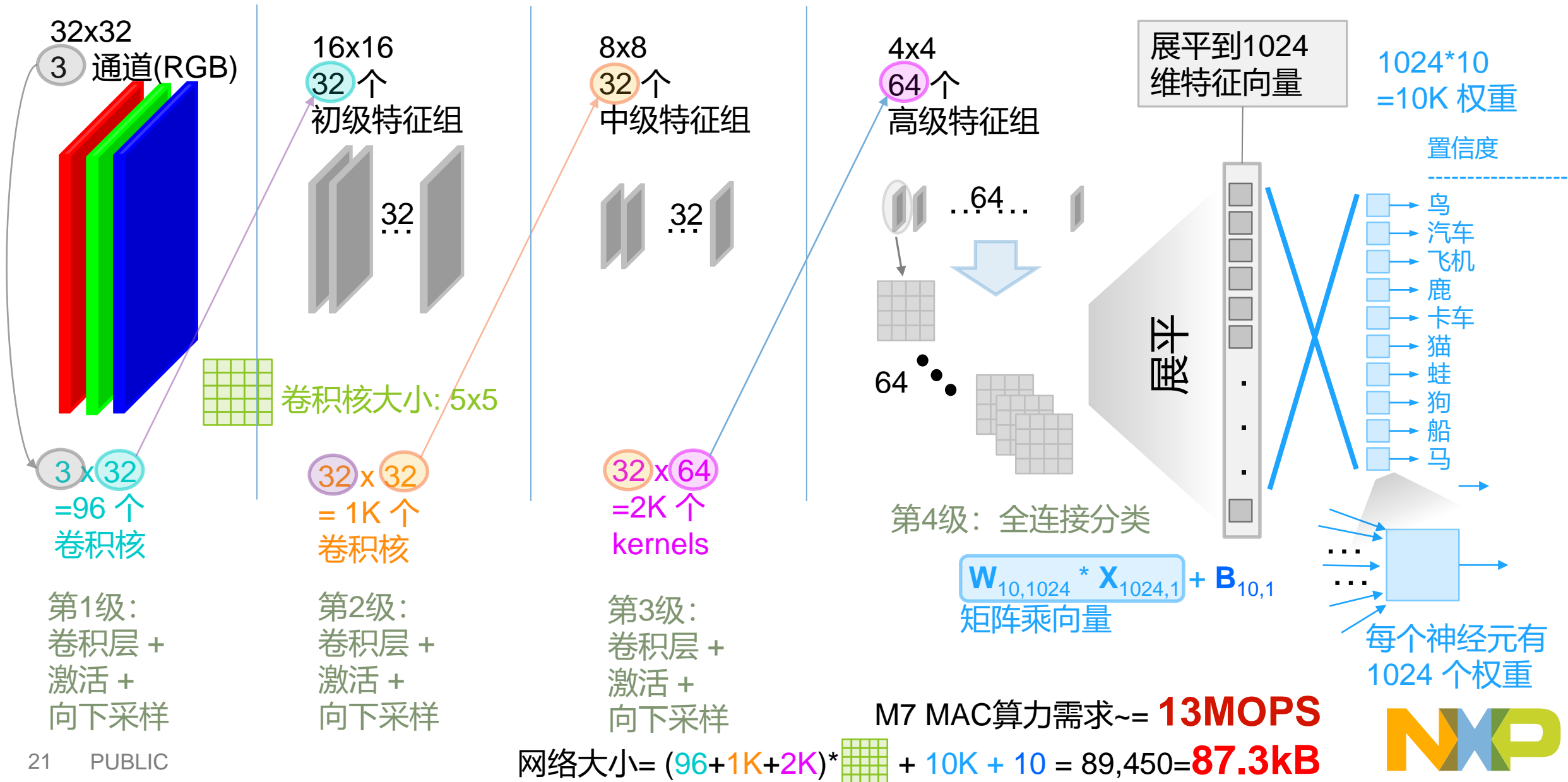
平均数	44	中位数	41	众数	36	StDev	21
最小	0	最大	100	LQ	29	UQ	57

串行终端

```
bird - Confidence 0.798235%
infer fps=45.5
bird - Confidence 0.768627%
infer fps=45.5
bird - Confidence 0.760784%
infer fps=45.5
```

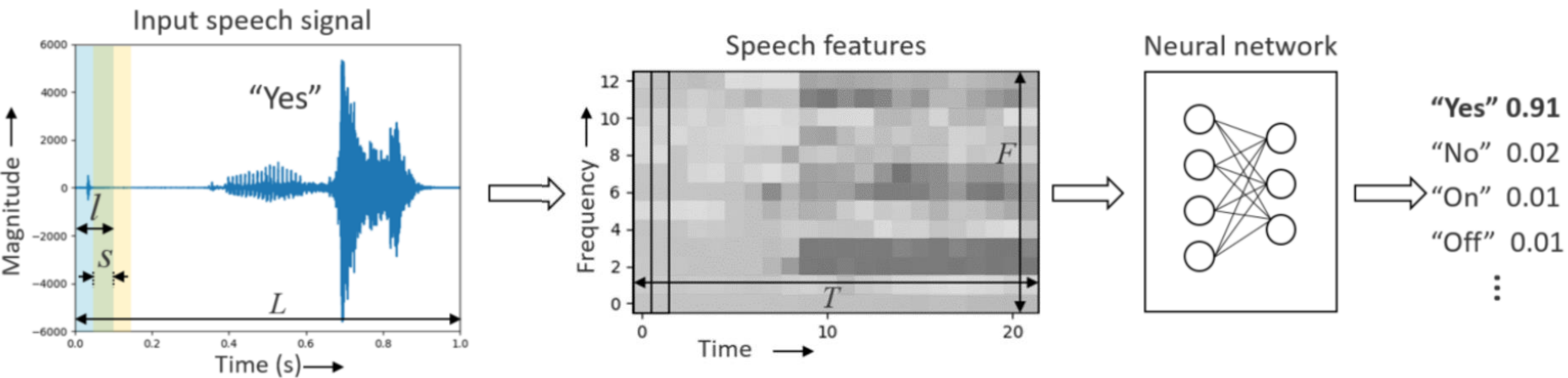
这是使用经典的CIFAR-10识别10种彩色物体的例子。使用基于i.MX RT的OpenMV系统，无论是否识别出物体，帧率都在45.5帧/秒左右。改变训练的数据集，CIFAR-10就可以识别其它物体。

CIFAR-10示例的结构



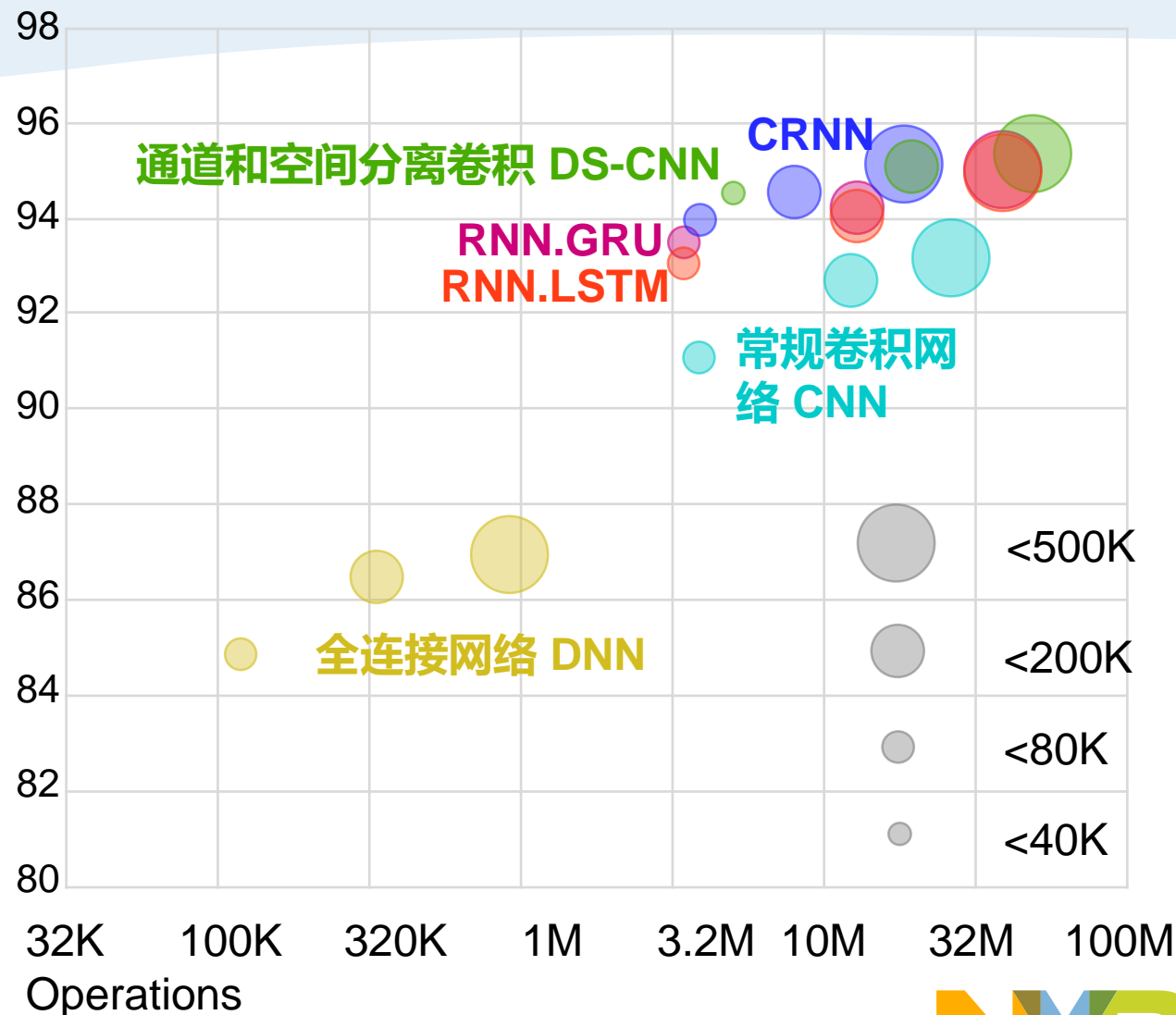
使用DL实现语音口令检测 (KWS)

- 接收短口令，并映射到预定义的命令
- 广泛用于语音控制的系统中，作为永久在线的低功耗唤醒模块
- 把时域信号分割变换成多段频谱，形成灰度图，再应用NN技术



关键词检测模型的资源需求

- 显著的边际效应递减
- 全连接神经网络(DNN) 精度最低, 但计算量也最低.
- 常规CNN性能稍有不足
- **DS-CNN**有效减少了模型尺寸和提高性能, 只是对算力要求稍有提高.
- RNN比常规CNN效果好, CRNN融合了CNN与RNN的优点.



小结

- 人工智能可以提升嵌入式系统的价值
- 在MCU上应用人工智能有独特的优点
- 神经网络是在MCU上实现人工智能的有效技术
- OpenMV RT可以在MCU上体验机器视觉和人工智能



SECURE CONNECTIONS
FOR A SMARTER WORLD