2019年中国嵌入式技术大会
EMBEDDED TECHNOLOGY
Conference China 2019

嵌入式系统的AI开发实践

莫志豪

MCU高级应用工程师

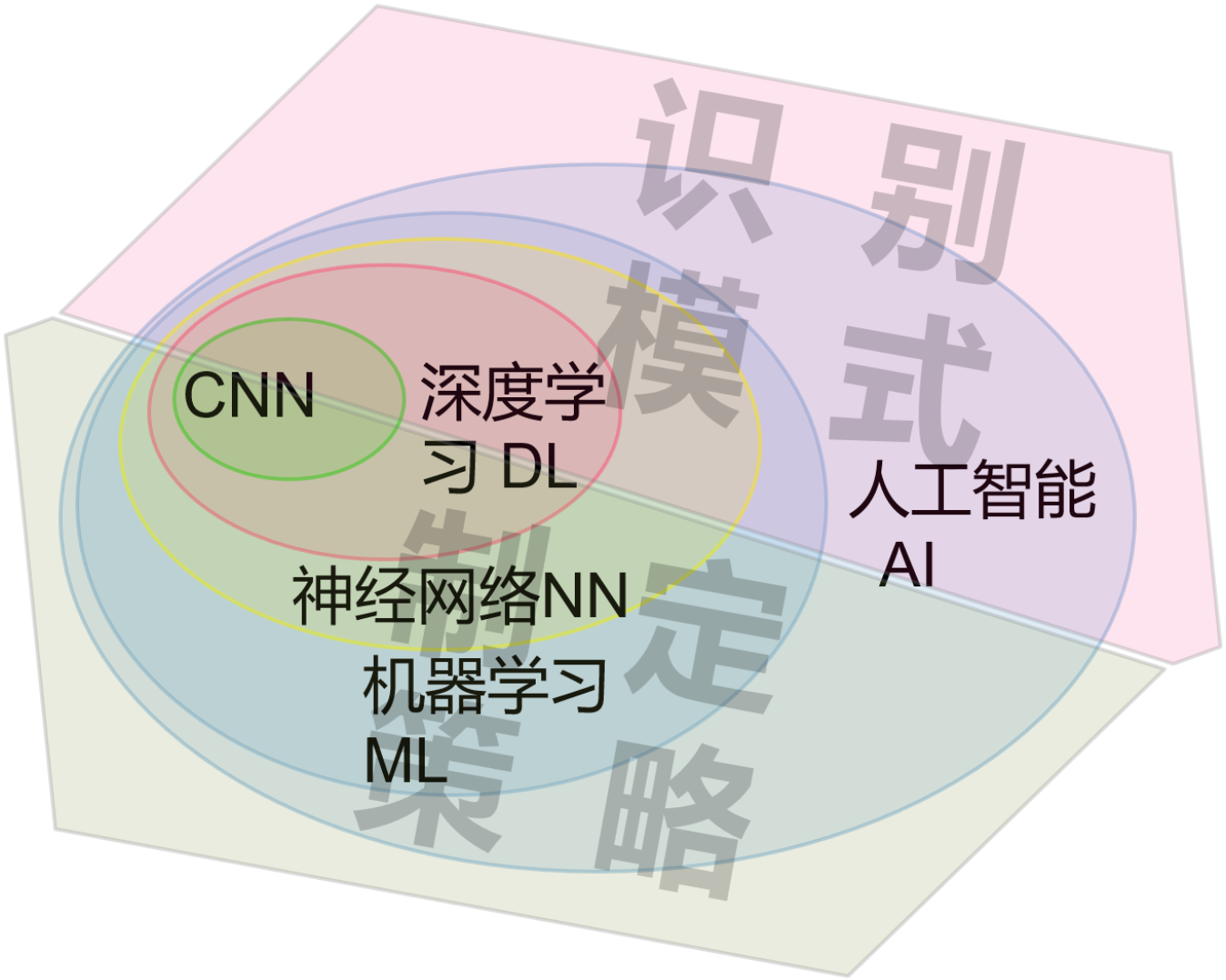
SECURE CONNECTIONS
FOR A SMARTER WORLD

内容提要

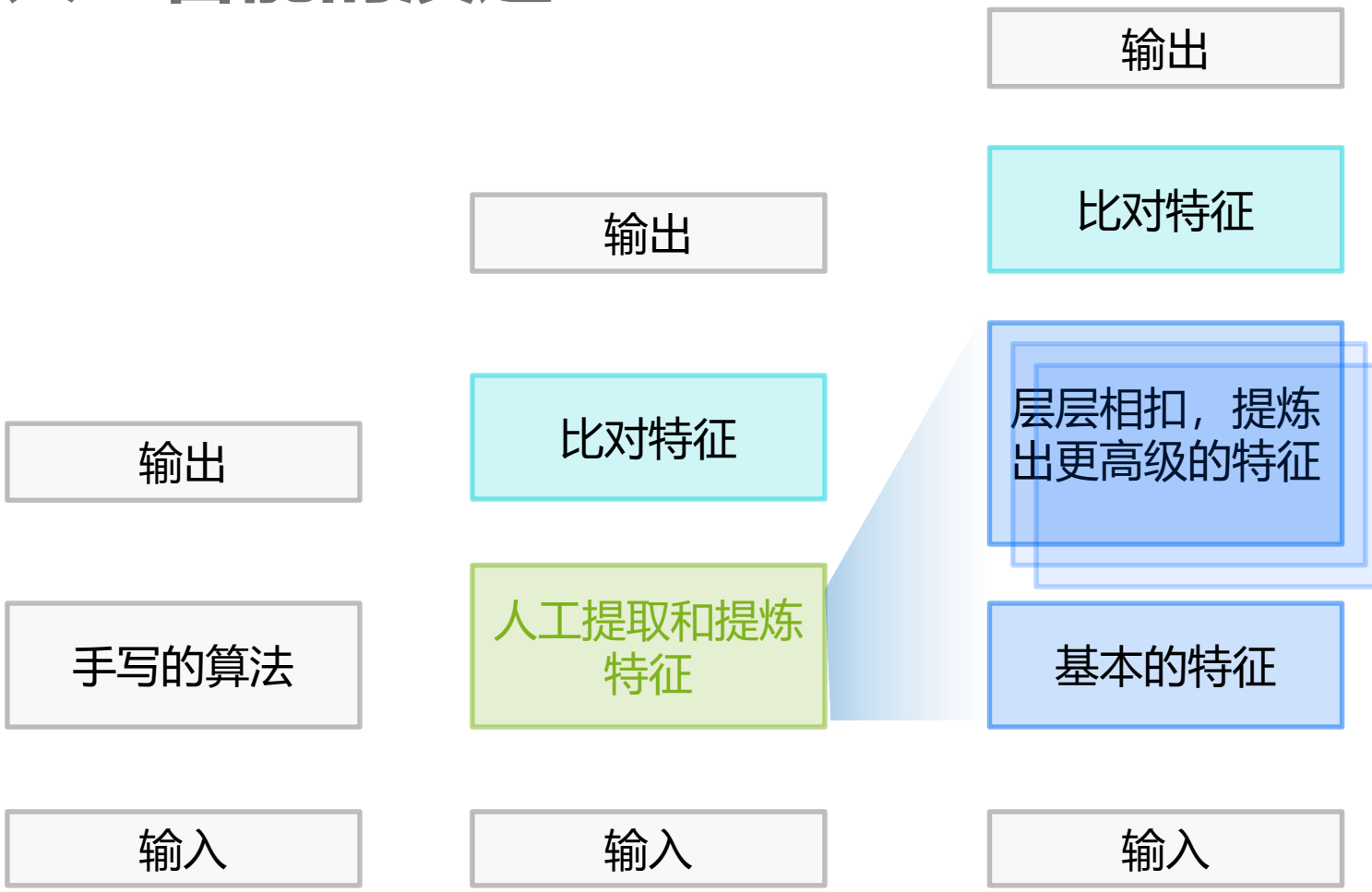
- AI概述
- 机器学习精确度
- 嵌入式系统AI应用
- NXP eIQ生态系统
- 嵌入式AI开发实例

AI概述

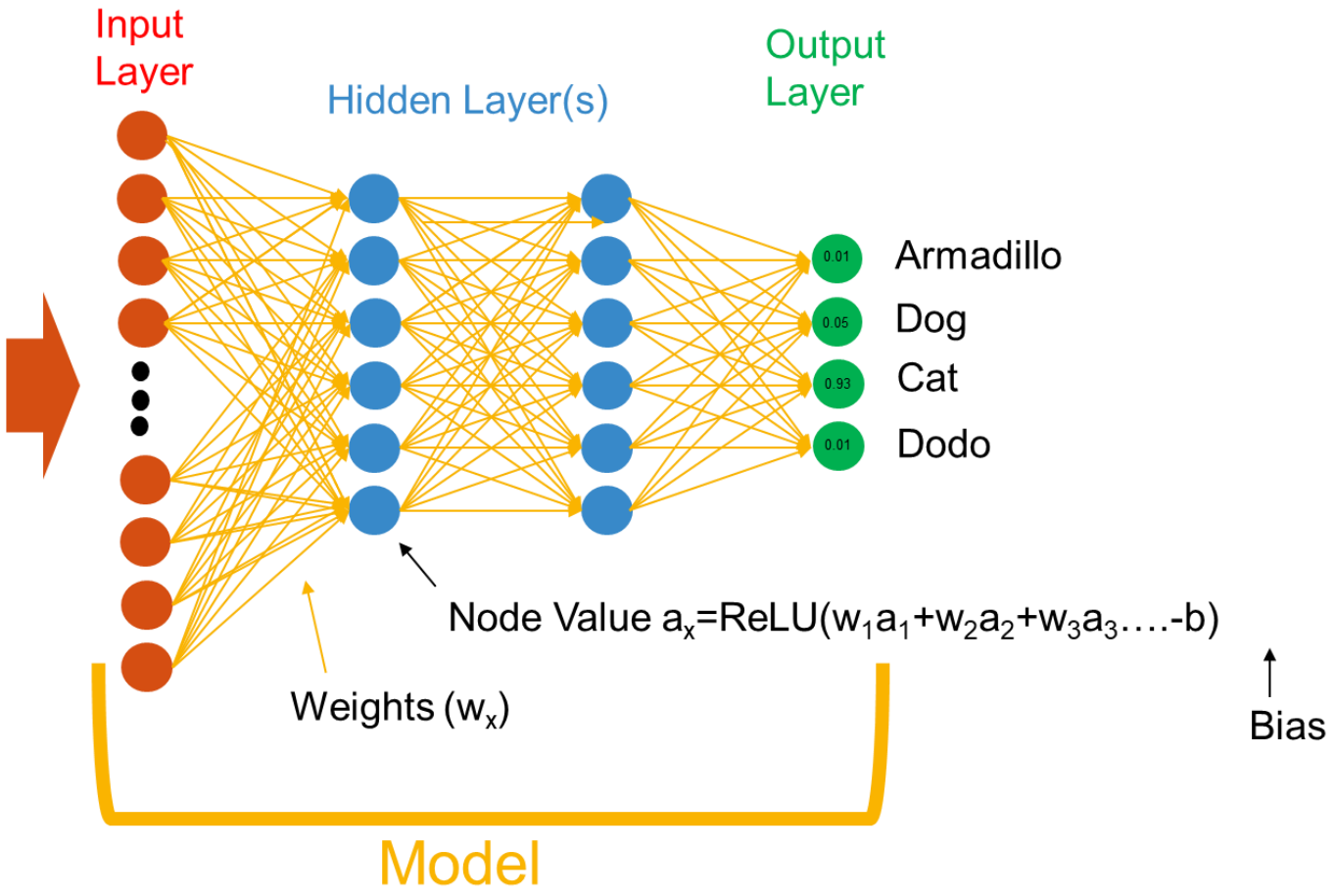
AI – 人工智能



人工智能的演进

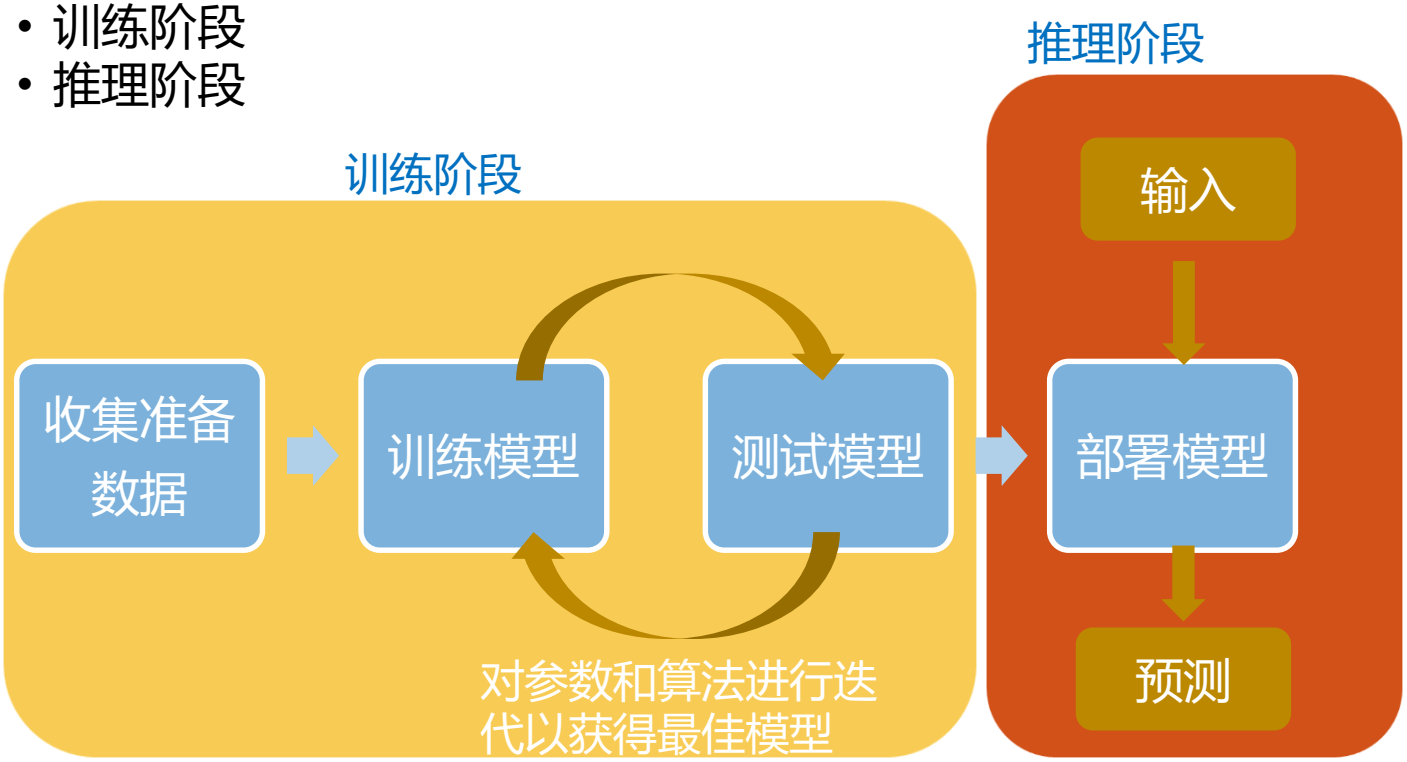

输出
比对特征
层层相扣，提炼出更高级的特征
基本的特征
输入
输出
比对特征
人工提取和提炼特征
输入
输出
手写的算法
输入
DL常用基于NN的方法，也可沿用传统ML
DL模型**自动**找出特征，多达成百上千上万!
DL模型基于多层**神经网络**
基于规则的专家系统
传统机器学习(ML)
深度学习(DL)

人工智能神经网络模型


Input Layer
Hidden Layer(s)
Output Layer
0.01
Armadillo
0.05
Dog
0.93
Cat
0.01
Dodo
Node Value a_x=ReLU(w_1a_1+w_2a_2+w_3a_3….-b)
Weights (w_x)
Bias
Model

机器学习流程

- 训练阶段
- 推理阶段



机器学习精确度

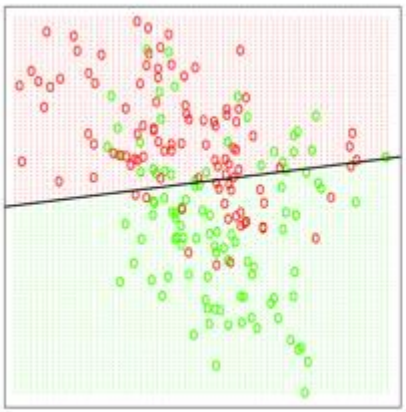
模型精确度

- 所有的预测，大多数模型会给出一个精确度。一般地，在模型的最后一层转换成“真”或“假”作为最终结果输出。
- 所有的模型都有一定的局限性。
- 机器学习的目标
  - 提高精度
  - 降低尺寸
  - 降低预测时间

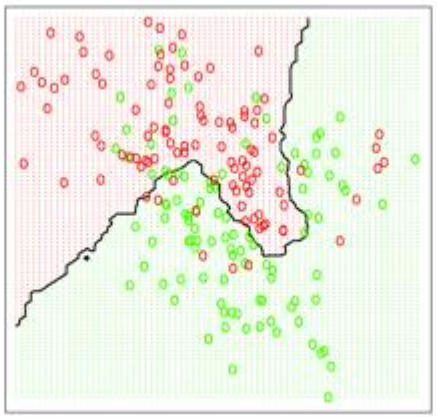
机器学习两大问题: 欠拟合与过拟合

- 都会影响模型在训练样本外的精度：泛化能力
- 欠拟合: 模型过于简陋，不足以识别丰富的特征
- **过拟合**: 模型过于精致或训练集过小，把训练集的随机特点/噪声也当共性了

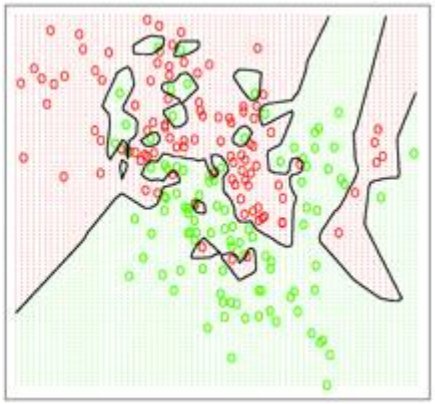
  -更普遍



欠拟合



刚刚好



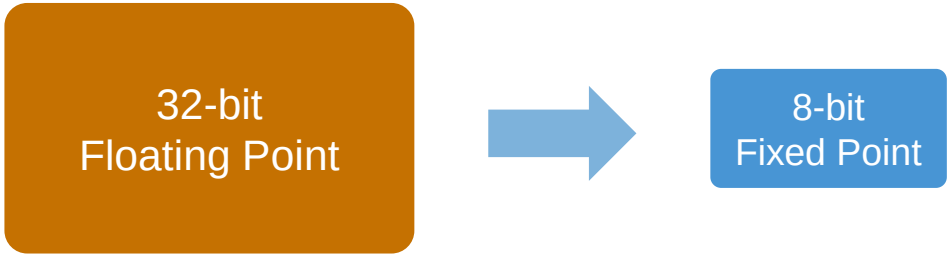
过拟合

影响模型精确度的事项

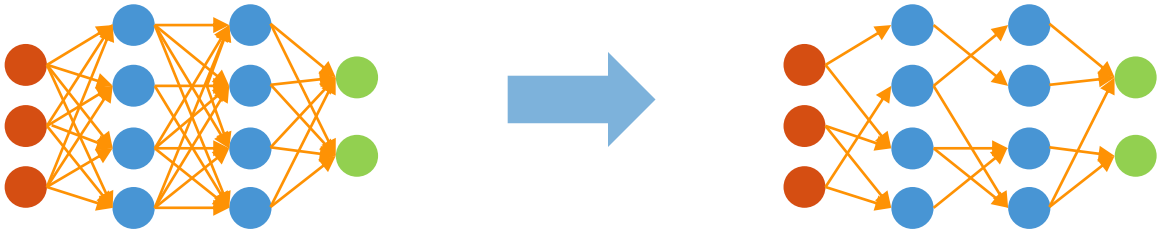
- 训练集的质量
- 训练集的数量
- 模型架构和训练方法
- 模型在嵌入式系统上的转换效率
  - 量化和修剪
- 输入测试数据的质量

量化和修剪

- 量化
  - 将32位浮点权重转换成8位整型
    - 缩小模型尺寸为四分之一



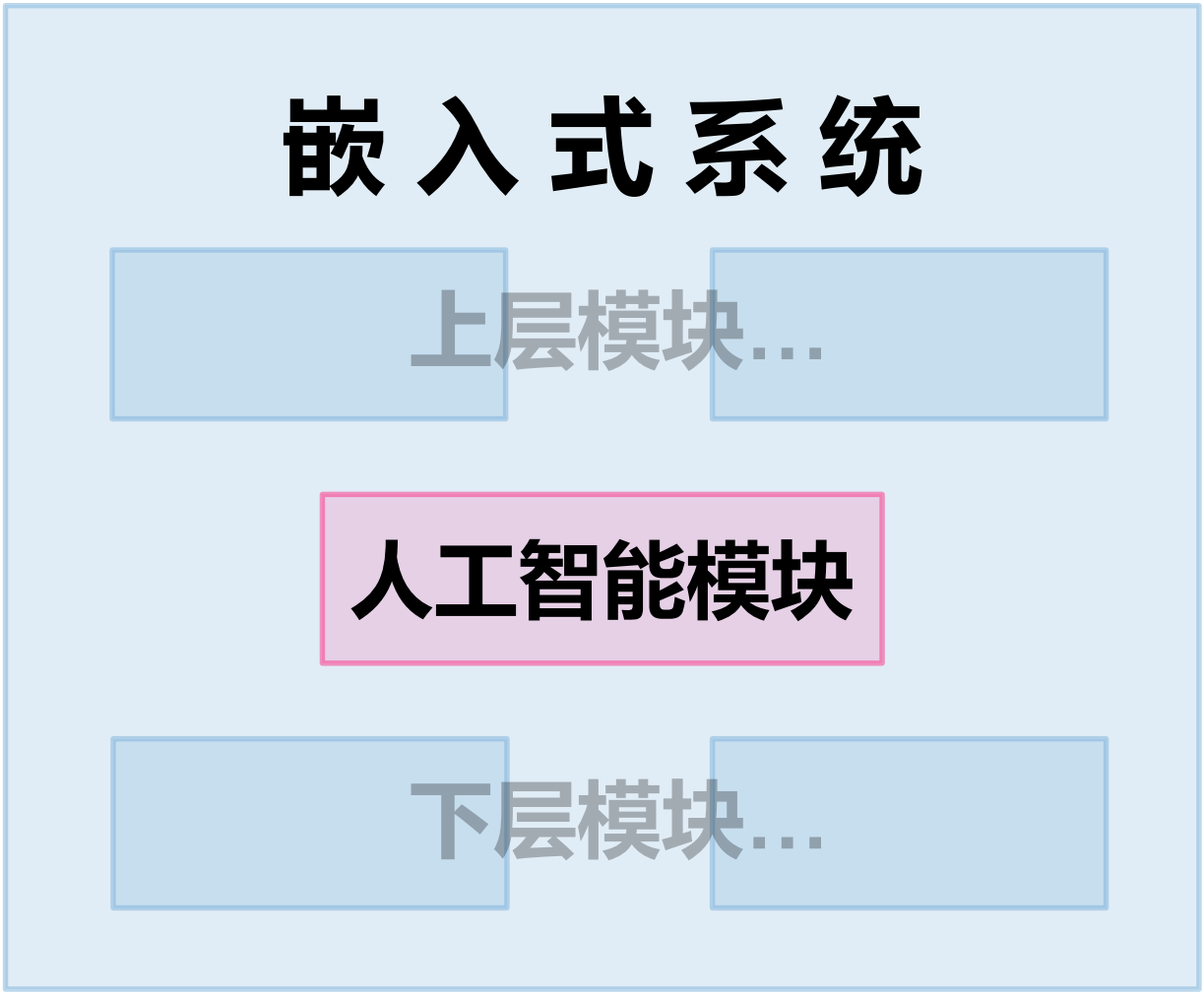
- 修剪
  - 从神经网络中去除地重要性的权重和偏差
    - 建议修剪模型后重新训练模型



嵌入式系统AI应用

人工智能与嵌入式系统

- 系统是主体
- 人工智能是“装备”
- 强大的“属性加成”
- 以模块来呈现
- 提供新功能
- 改进现有功能



嵌入式AI应用

图像/物体识别

语音识别

异常监测

智能穿戴

智慧工厂

医疗

AR增强现实

“轻型智能”，轻在何处?

- 规模小：模型尺寸和算力要求低
- 自包含：可独立运行AI模块，无需云端连接
- 应用专：对重点应用量身定制
- 优化高：针对计算平台高度优化
- 功耗低：实现IoT设备的长期工作
- 响应稳：多有实时性要求，快速而确定

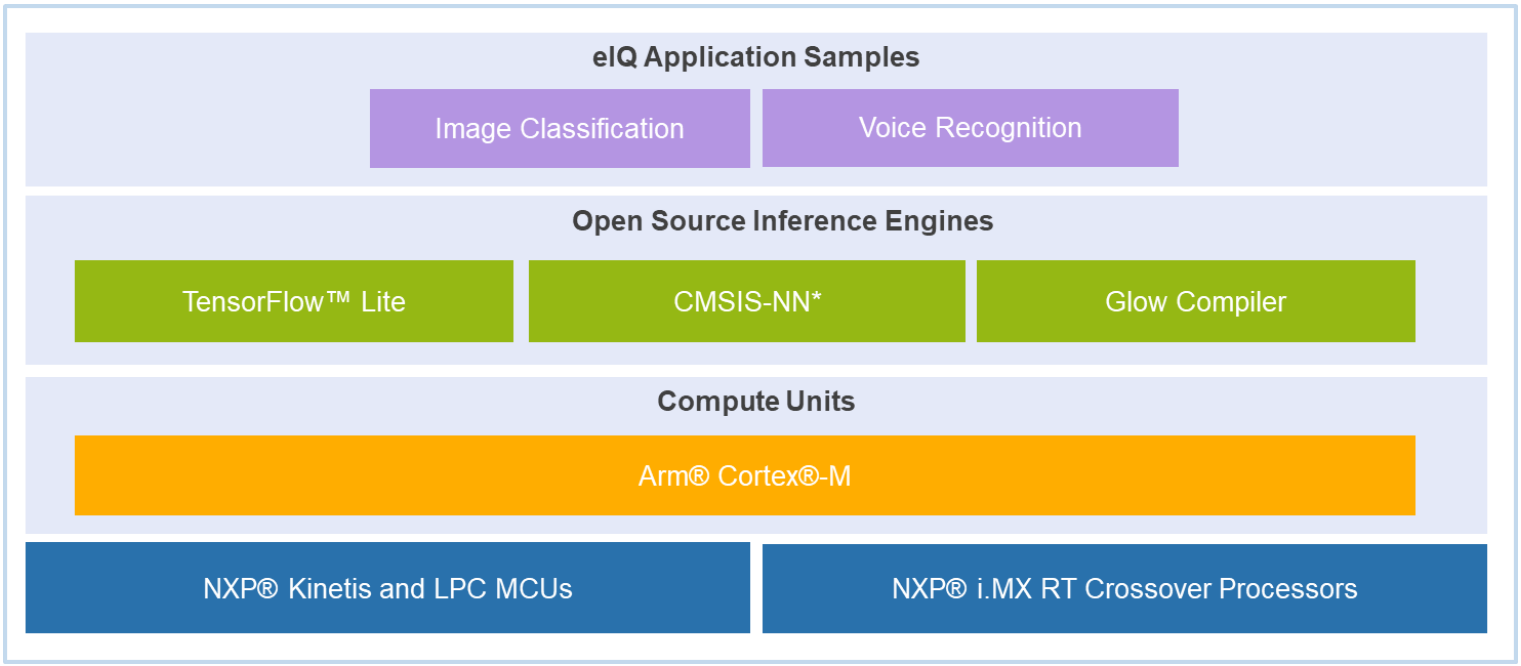
适合在MCU风格的平台上使用

MCU上AI应用的特点

NXP EIQ生态系统

NXP MCU + AI 工具计划

- “eIQ”软件包
- CMSIS-NN配套工具
- GLOW模型编译器
- 性能优化的TensorFlow-Lite
- 更多模型格式
- 更多NXP器件

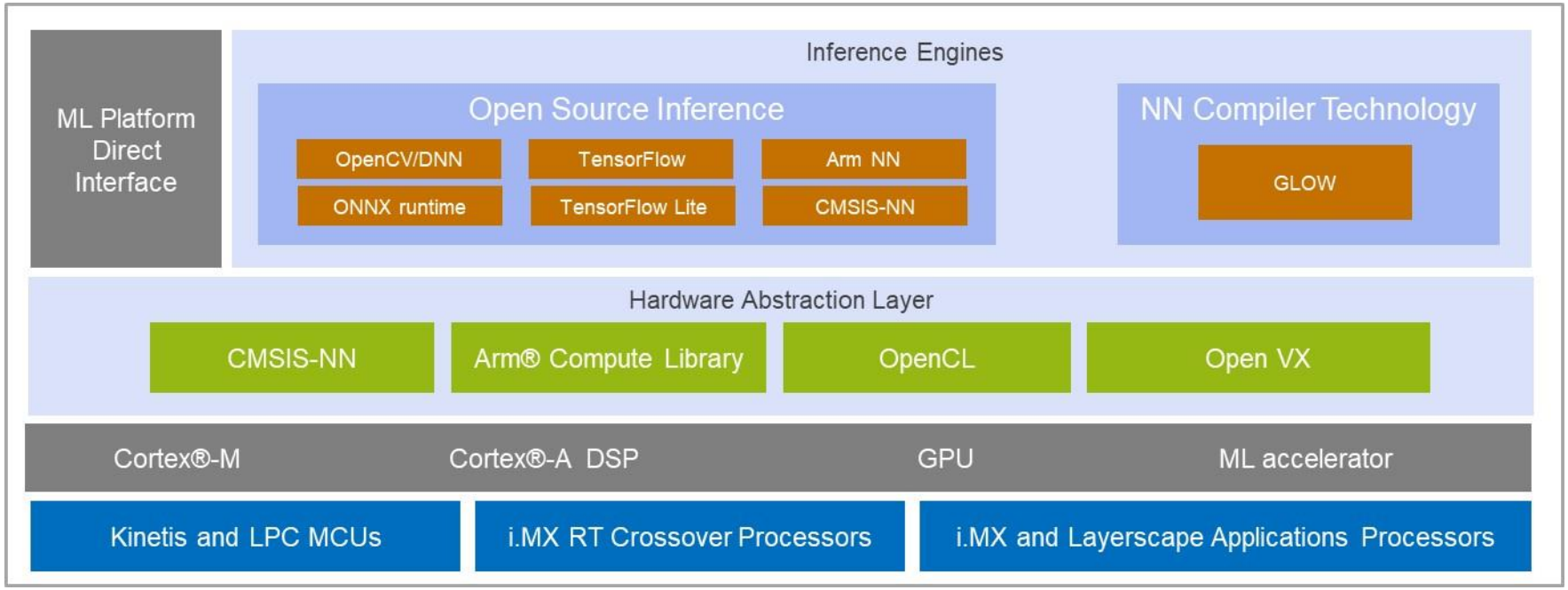


eIQ初探

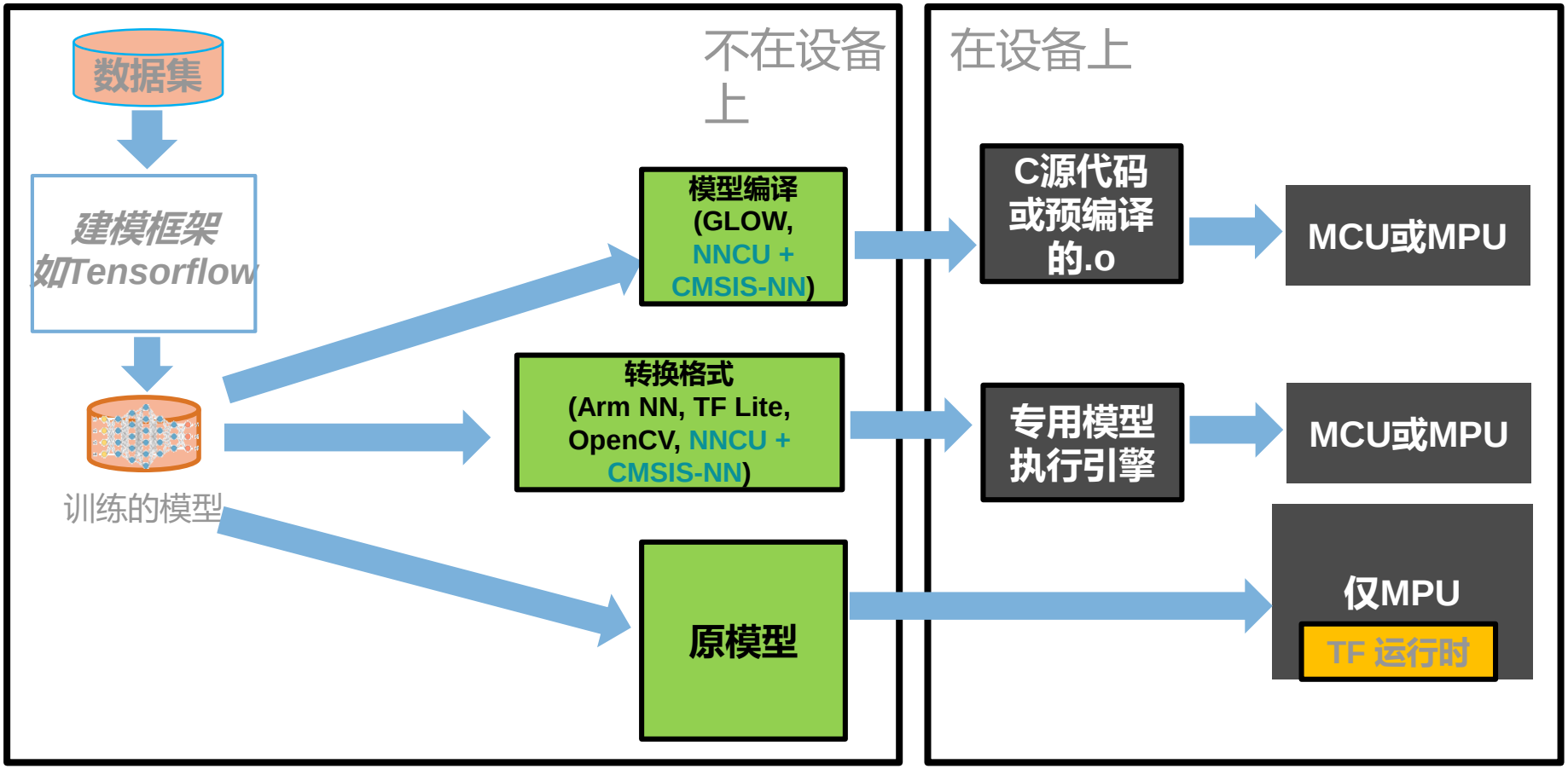
- 是在NXP MCU和MPU上运行AI模型所需的软件集合
- 开发中的工具:
  - 推理引擎: Arm NN, OpenCV, Arm CMSIS-NN, TensorFlow Lite, 等等
  - 在线实时示例，演示典型使用场景
  - 正在支持前沿的NN编译技术，如GLOW
  - 为传统机器学习开发独立的工具 (SVM, 随机森林等)
  - 为MCU和MPU各开发1套工具
- eIQ最终将以中间件纳入Yocto linux BSP和MCUXPresso SDK的发布包

eIQ整体框架图

- eIQ是一套软件集合，用于在NXP MCU和MPU上运行ML模型
- 大量使用开源软件

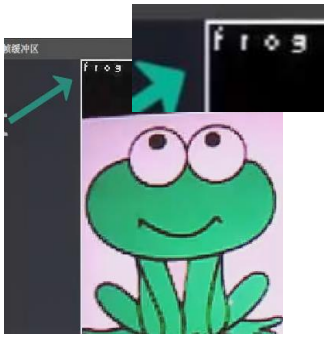


eIQ下的三种工作流程

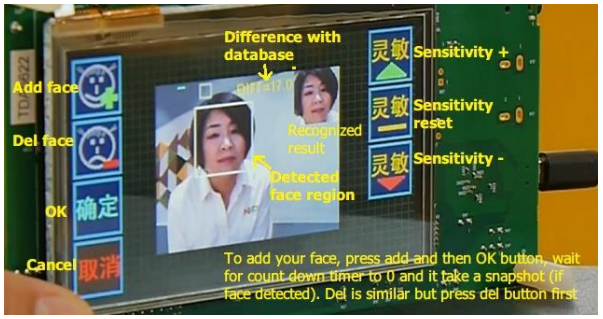


嵌入式AI开发实例

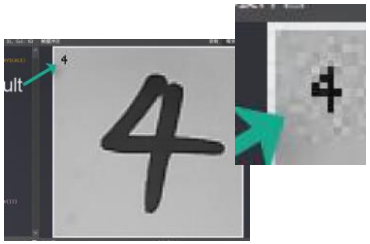
MCU AI/MV 演示一览



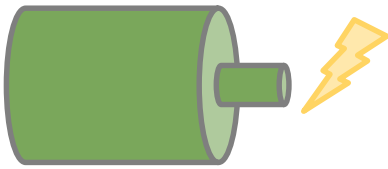
10/100 分类



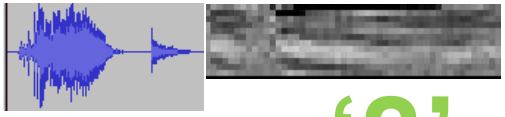
人脸检测与识别



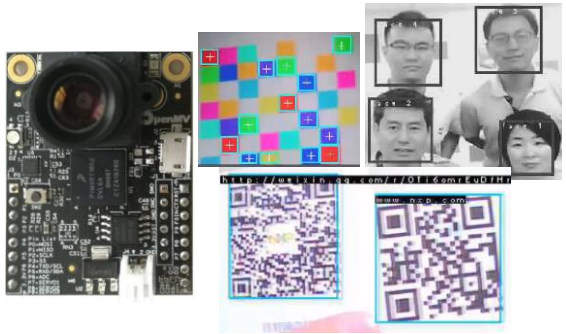
手写数字识别



电机控制异常检测
运动/震动异常检测



语音口令识别 ‘8’



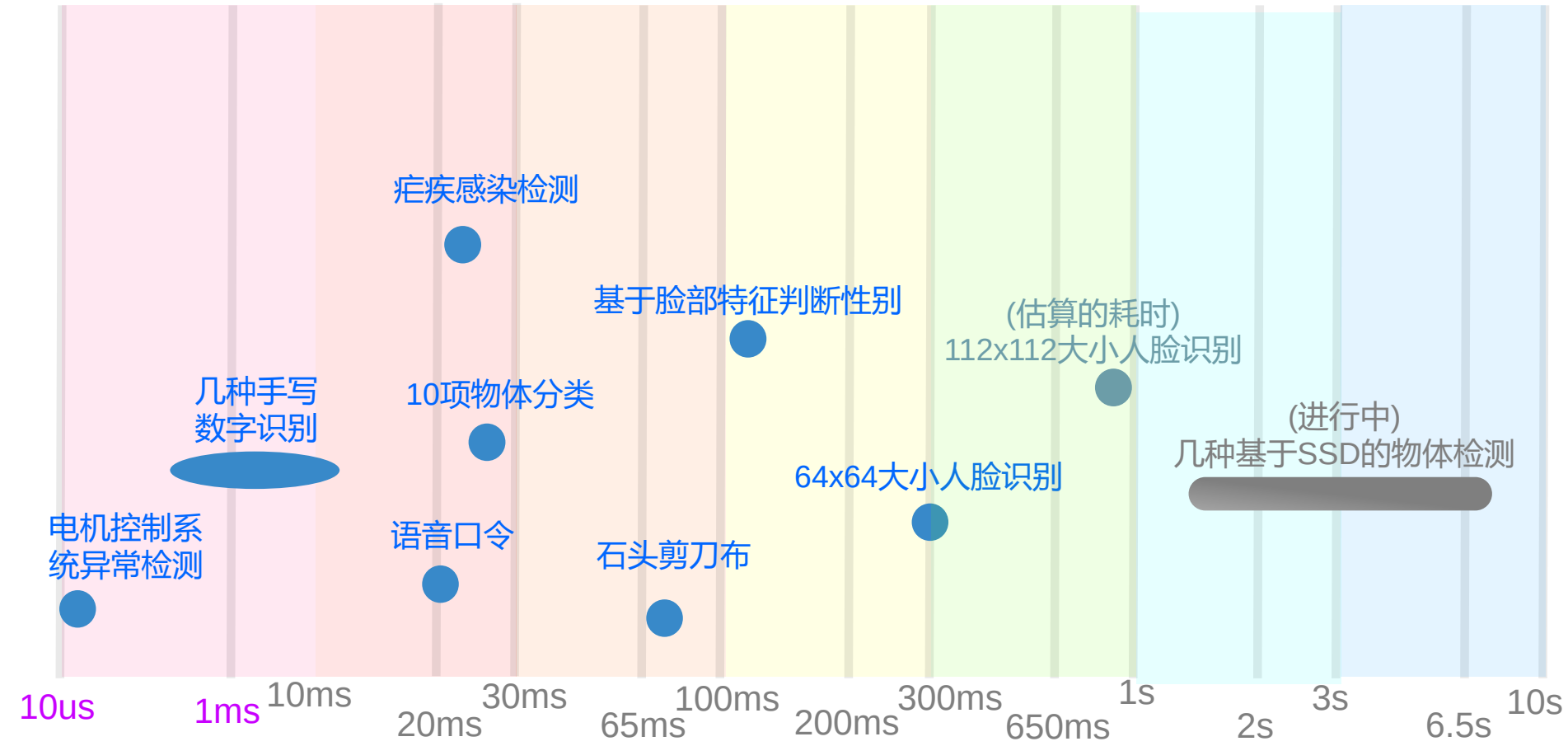
OpenMV-RT 多项演示

红细胞感染疟疾检测

基于面部的性别识别

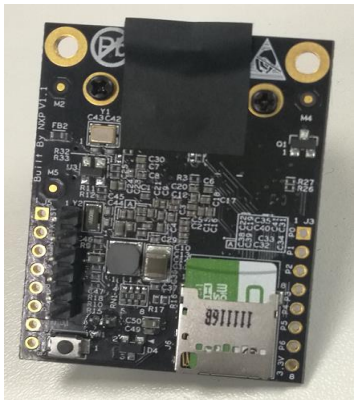
石头剪刀布

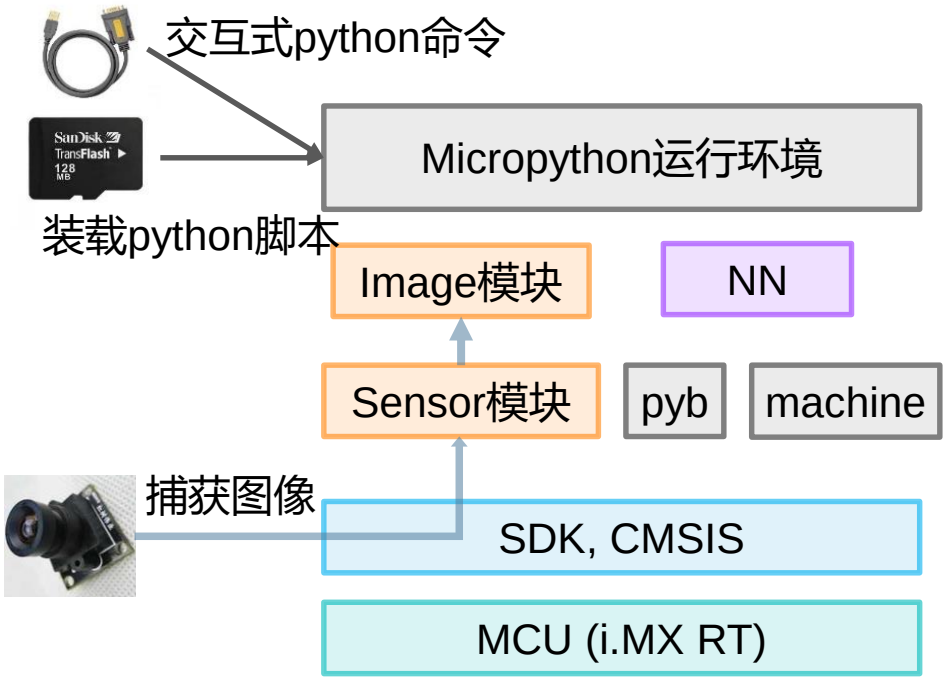
i.MX RT1050上demo模型运算耗时



可编程视觉模块

- 基于NXP i.MX RT1062
- OV7725 + M12系列镜头
- 高度兼容OpenMV Cam
- 使用Micropython开发
- 一键部署脚本到RAM
- 支持多种机器视觉算法
- 支持神经网络的运算
- 板载9轴运动传感器
- 可通过SPI, I2C, UART连接外设或底板





项目地址:
https://github.com/RockySong/micropython-rocky

基于i.MX RT可视模块openmv搭建

- 项目工程
  - 下载基于micropython的RT工程
  - 编译并烧录代码至目标板
- openmv IDE
  - 下载openmv IDE并安装驱动
- 通过usb将可视模块接到电脑端
- 运行openmv的例程或mpy代码

openmv物体识别示例

nn_cifar10_search_whole_window_1.py - OpenMV IDE

File Edit Tools Window Help

nn_cifar10_search_whole_wi... Line: 48, Col: 1

```
# CIFAR-10 Search Whole Window Example
#
# CIFAR is a convolutional nueral network designed to classify it's field of view into several
# different object types and works on RGB video data.
#
# In this example we slide the LeNet detector window over the image and get a list of activations
# where there might be an object. Note that use a CNN with a sliding window is extremely compute
# expensive so for an exhaustive search do not expect the CNN to be real-time.

import sensor, image, time, os, nn

sensor.reset()                         # Reset and initialize the sensor.
sensor.set_pixformat(sensor.RGB565)    # Set pixel format to RGB565 (or GRAYSCALE)
sensor.set_framesize(sensor.QVGA)      # Set frame size to QVGA (320x240)
sensor.set_windowing((128, 128))       # Set 128x128 window.
sensor.skip_frames(time=750)           # Don't let autogain run very long.
sensor.set_auto_gain(False)            # Turn off autogain.
sensor.set_auto_exposure(False)        # Turn off whitebalance.

# Load cifar10 network (You can get the network from OpenMV IDE).
net = nn.load('/cifar10.network')
# Faster, smaller and less accurate.
# net = nn.load('/cifar10_fast.network')
labels = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

clock = time.clock()
while(True):
    clock.tick()

    img = sensor.snapshot()

    # net.search() will search an roi in the image for the network (or the whole image if the roi is not
    # specified). At each location to look in the image if one of the classifier outputs is larger than
    # threshold the location and label will be stored in an object list and returned. At each scale the
    # detection window is moved around in the ROI using x_overlap (0-1) and y_overlap (0-1) as a guide.
    # If you set the overlap to 0.5 then each detection window will overlap the previous one by 50%. Note
```

Serial Terminal

```
2.602811
Detected airplane - Confidence 0.609804%
2.604167
Detected airplane - Confidence 0.603922%
2.604167
Detected airplane - Confidence 0.607843%
2.60078
Detected airplane - Confidence 0.603922%
2.601908
```

Frame Buffer Record Zoom Disable

Histogram RGB Color Space

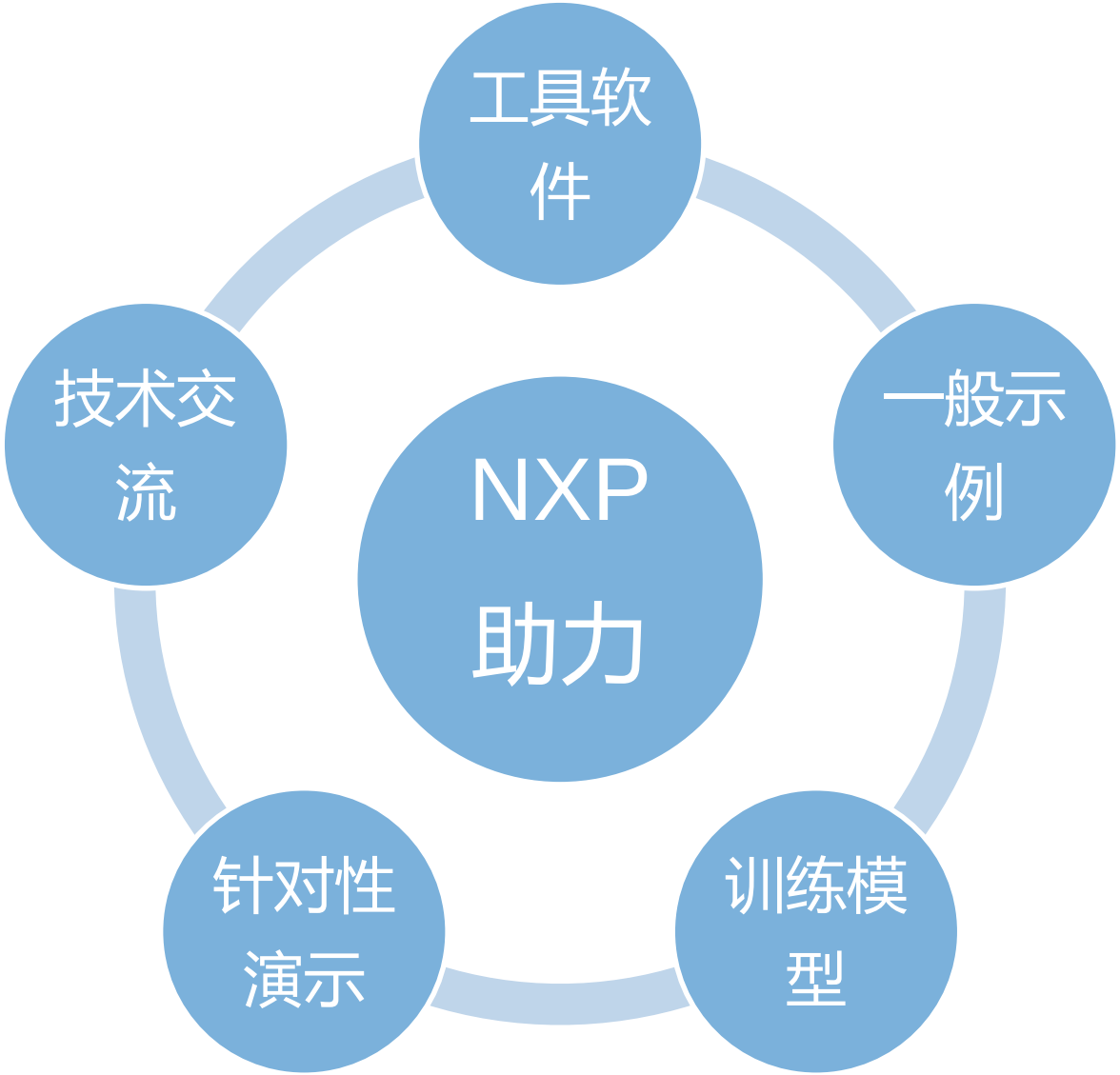
Res (w:128, h:128)

Search Results Serial Terminal Firmware Version: 3.15.0 - [latest] Serial Port: COM29 Drive: D:/ FPS: 2.6

视觉模块在2019年全国大学生智能车竞赛应用

- April tag
  - 标签识别与定位
- 色块
  - 信标灯识别
- 线条与形状
  - 赛道识别
- 图像预处理
  - 有助于后续处理
- 通用MCU功能

携手前进，共创辉煌



SECURE CONNECTIONS
FOR A SMARTER WORLD