如何使用 Edge Impulse 训练机器学习模型并部署到

SAMA7G54-EK

Microchip Technology Inc.
MCU32 产品部

一、 简介

本文将介绍如何使用 Edge Impulse 在线工具训练机器学习模型，包括模型创建、原始数据采集与预处理、学习框架构建，以及在 Edge Impulse 云服务器上进行模型训练、测试与迭代优化。最终，我们将把训练好的模型部署到 Microchip SAMA7G54-EK 评估板上，使用 USB 摄像头实时检测目标图像，并通过浏览器查看图像识别结果。

二、 Edge Impulse 介绍

Edge Impulse 是一个可以在线训练机器学习模型并且将结果部署到各种边缘嵌入式设备上的软件平台。
Edge Impulse 针对 Microchip SAMA7 系列 MPU 做了深度优化，使得在 SAMA7 系列 MPU 上运行的模型更加轻量，推理更加快速，占用系统资源更少。
Edeg Impulse 官方网站：https://edgeimpulse.com/

三、 准备工作

(一) 电脑开发环境

本文介绍例程需要在一台安装了 Ubuntu 操作系统的电脑上构建 SD 卡镜像文件，如果您的开发电脑没有安装 Ubuntu 操作系统，请参考以下文档进行安装并配置：
HowTo_AT91_Buildroot_Setup_Based_on_Ubuntu.pdf

(二) Docker 容器环境

在构建 SD 卡镜像时会使用到 docker 容器，请先安装 docker 容器到您的 Ubuntu 开发电脑中。安装方法请参考以下链接：
https://docs.docker.com/engine/install/ubuntu/

(三) Edge Impulse 账号

本文将利用 Edge Impulse 进行机器学习模型训练及测试，请先注册账号，登录后使用其功能。账号注册链接：
https://edgeimpulse.com/

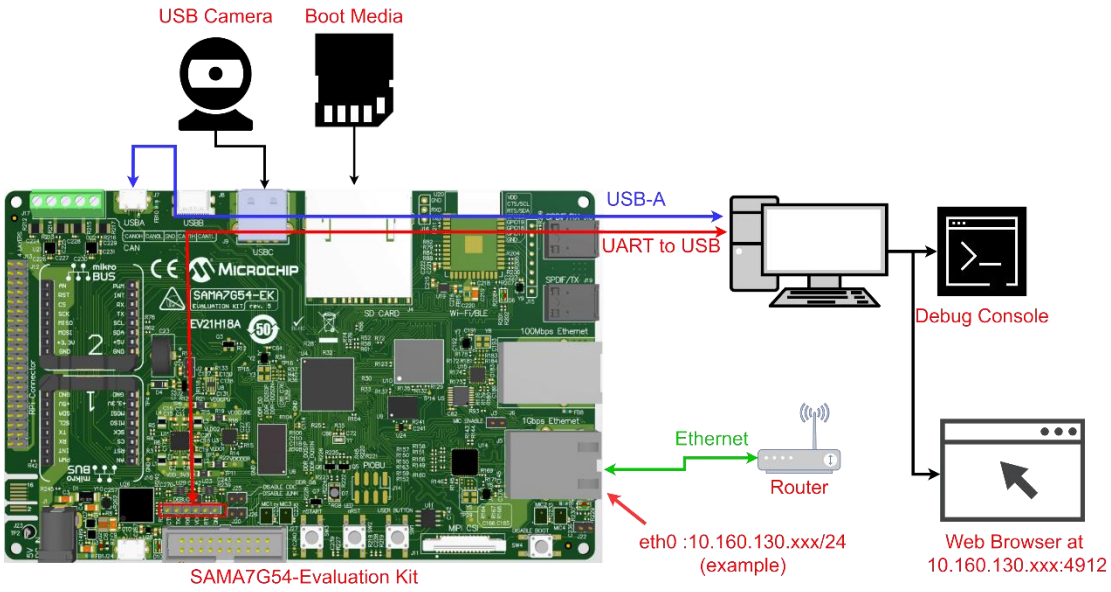
注：本文所有实验及操作使用 Community 类型账号即可完成。

四、 硬件工具和软件平台

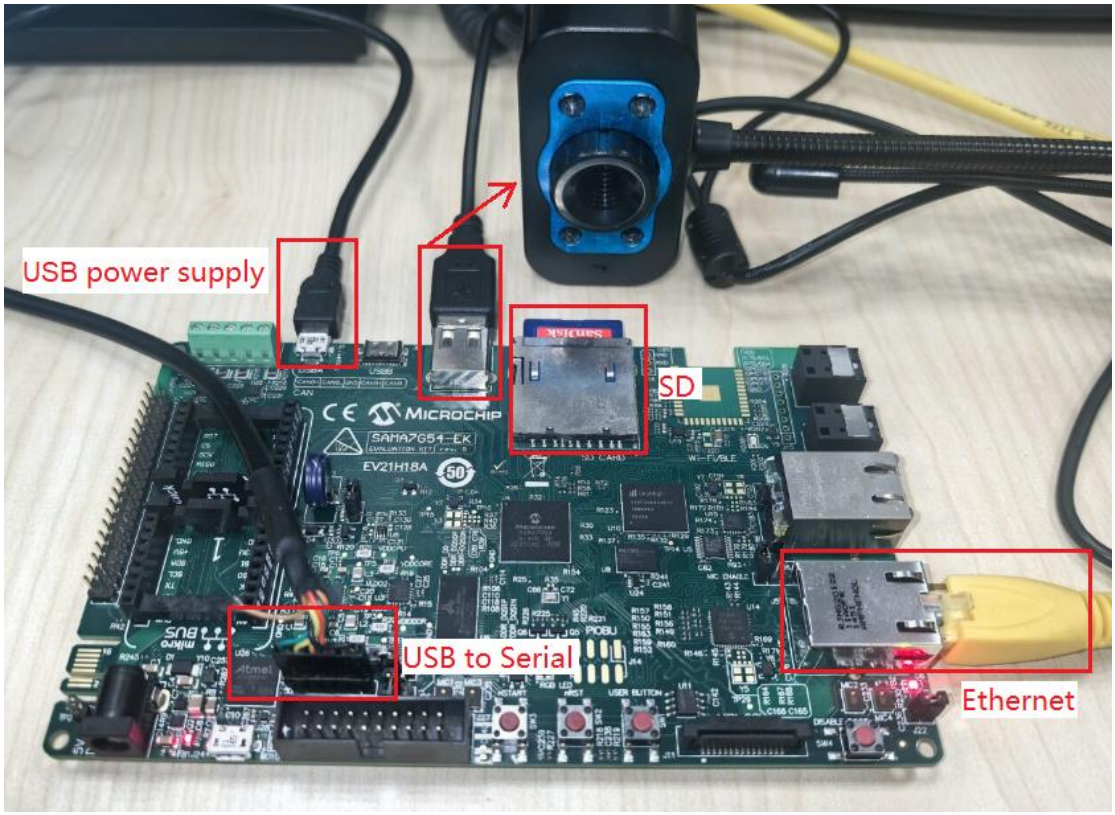
(一) 硬件

1. 硬件设备列表

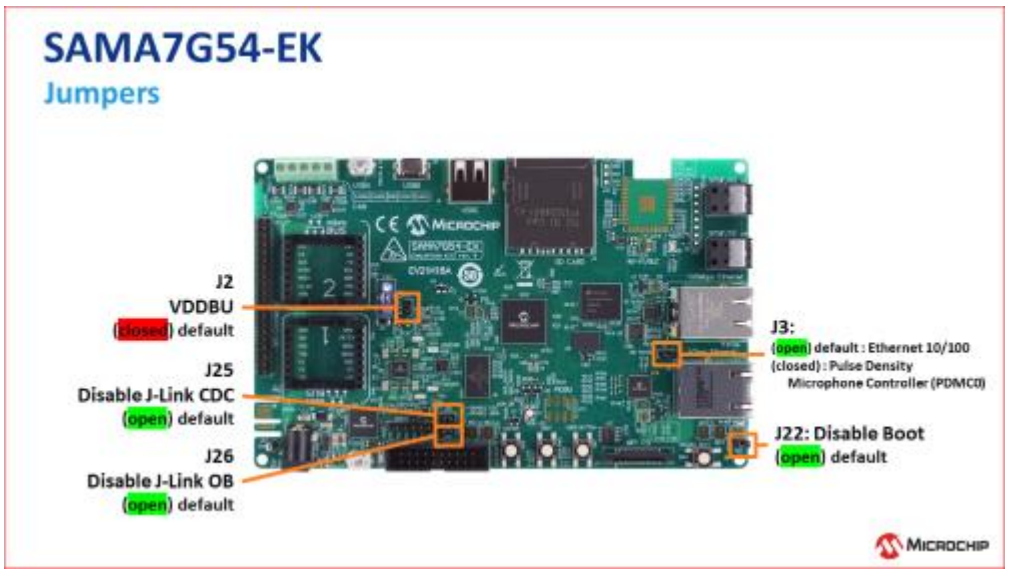
- SAMA7G54-EK 评估板
- USB 接口摄像头
- SD 存储卡
- USB 转串口线（TTL 电平）
- USB 数据线（一端 Micro USB，另一端 Type-A）
- 路由器（评估板和电脑均连接至路由器）
- 以太网线



2. 硬件连接图



3. 板子跳线设置



(二)　软件

为了方便做例程实验，本章节附上预先构建好的 SD 卡镜像链接，您可以直接下载烧录。同时也介绍了如何手动构建 SD 卡镜像的步骤，以供更深入学习。

1. 可下载 SD 卡镜像

镜像下载地址：

https://cdn.edgeimpulse.com/build-system/microchip.sama7.100724.sdcard.img.zip

2. 手动构建 SD 卡镜像

(1) 在 Ubuntu 环境中下载源码

user@at91:~/$ **git clone**

https://github.com/edgeimpulse/example-microchip-sama7g54.git

user@at91:~/$ **cd example-microchip-sama7g54**

user@at91:~/example-microchip-sama7g54$ **ls**

buildroot-config　Dockerfile　LICENSE　sharp-linux-armv7.node

Config.in　example-standalone-inferencing-linux.mk　README.md

注：本文中所有需要用户在 Linux 终端输入的命令文字均加粗以示区分。

(2) 构建 docker 容器

user@at91:~/example-microchip-sama7g54$ **sudo docker build . -t microchip**

user@at91:~/example-microchip-sama7g54$ **sudo docker run -it -v $PWD/build:/buildroot-microchip/buildroot-at91/output/images microchip**

(3) 进入 docker 容器构建 SD 卡镜像

root@a4d260547a56:/# **cd buildroot-microchip/buildroot-at91/**

root@a4d260547a56:/buildroot-microchip/buildroot-at91# **ls**

CHANGES　Config.in.legacy　Makefile.legacy　board　dl　linux　sharp-linux-armv7.node　toolchain

COPYING　DEVELOPERS　README　boot　docs　output　support　utils

Config.in　Makefile　arch　configs　fs　package　system

root@a4d260547a56:/buildroot-microchip/buildroot-at91# **make**

(4) 获取 SD 卡镜像

在上一步 make 结束之后，退出 docker，构建生成的 SD 镜像文件 sdcard.img 存储在当前目录下的 build 文件夹内。

user@at91:~/example-microchip-sama7g54$ **ls build**

at91bootstrap.bin　rootfs.tar　sama7g5ek.itb　u-boot.bin

at91-sama7g5ek.dtb　sama7g5ek_at25ff321a_click1.dtbo　sama7g5ek.its

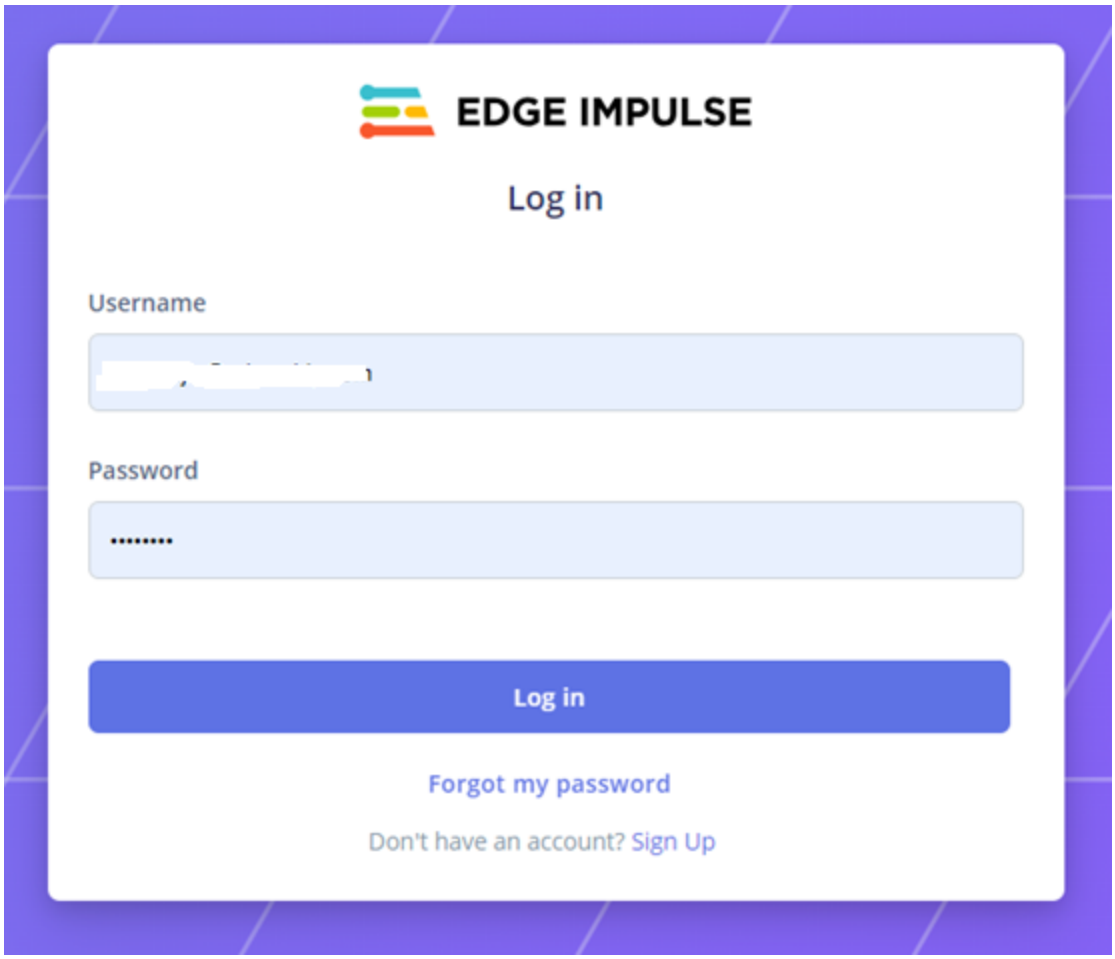
uboot-env.bin
boot.bin sama7g5ek_i2s0_pcm5102a.dtbo
sama7g5ek_pdmc0.dtbo zImage
boot.vfat sama7g5ek_i2s0_proto.dtbo
sama7g5ek_wilc3000.dtbo
rootfs.ext2 sama7g5ek_isc_imx219.dtbo
sama7g5-sdcardboot-uboot-4.0.9-rc1.bin
rootfs.ext4 sama7g5ek_isc_imx274.dtbo sdcard.img

五、 详细步骤

接下来我们将从零开始一步步详细介绍如何训练一个机器学习模型，对模型进行配置、训练，并最终将其部署到 SAMA7G54-EK 上检验运行效果。

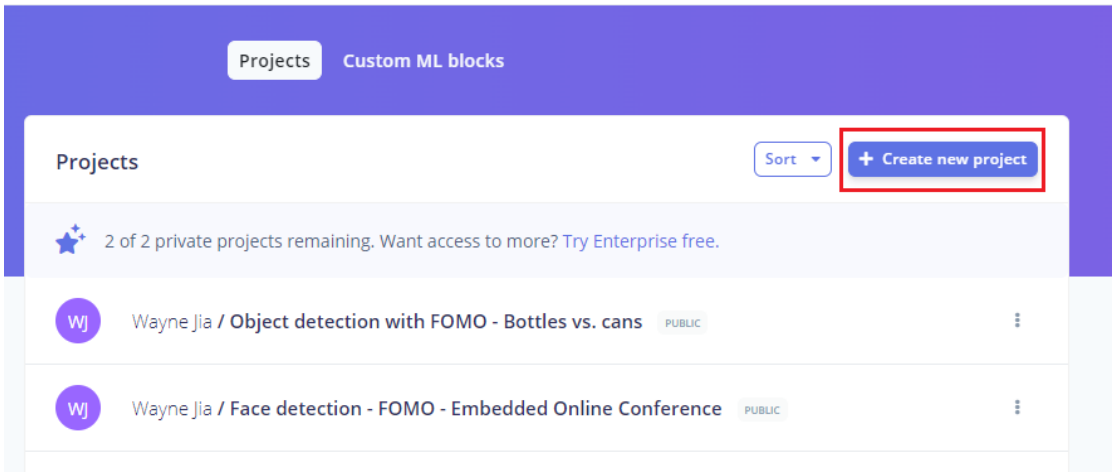
(一) 登录 Edge Impulse 网站平台

通过链接：https://studio.edgeimpulse.com/login 登录 Edge Impulse 在线平台：

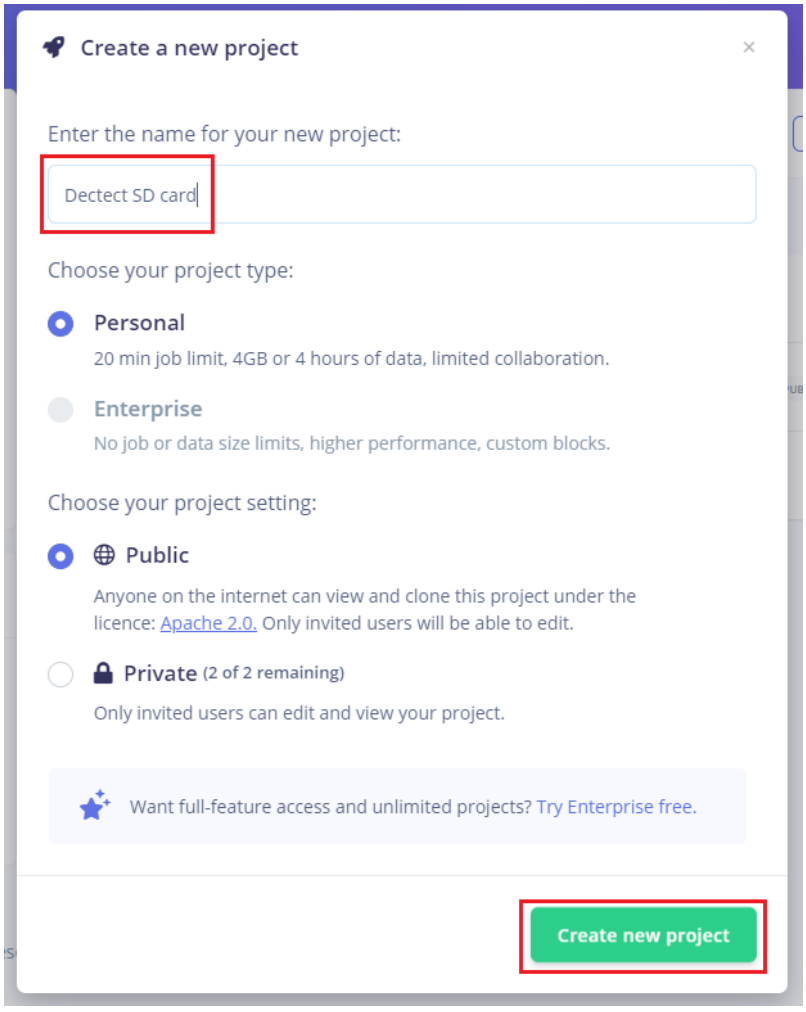


(二)　创建一个 Edge Impulse 工程

登录成功后在右上方会看到一个按钮“+ Create new project”，点击该按钮创建一个新的工程。

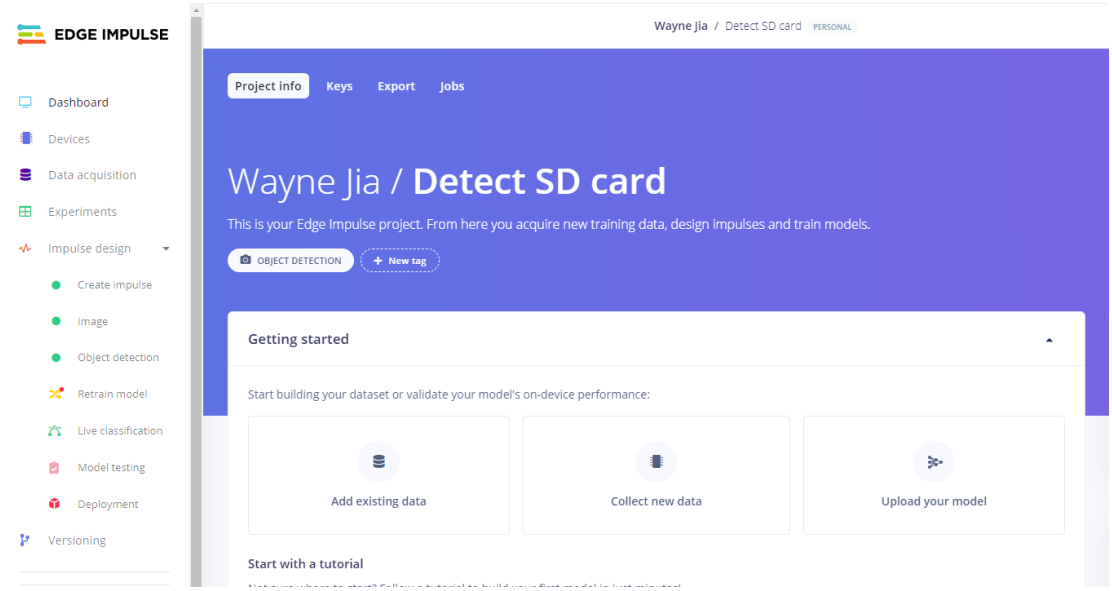


本文讲解使用的例程目标是训练一个能识别 SD 卡的模型并让其在 SAMA7G54-EK 上运行，因此将工程名字设置为：“Detect SD card”。之后点击“Create new project”按钮创建工程。



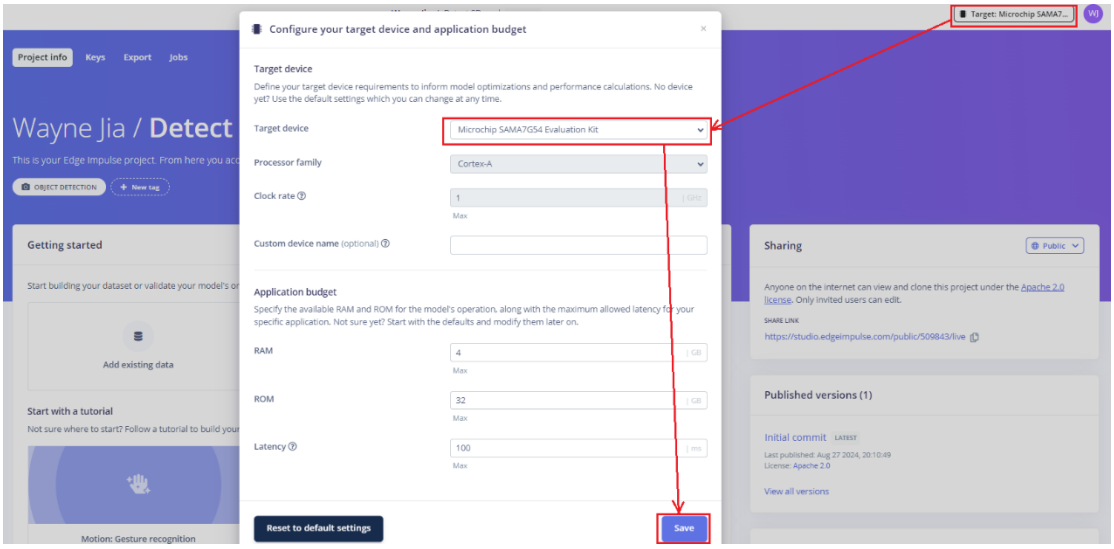
(三)　控制面板

工程创建完成后，我们可以在左侧菜单栏“Dashboard”即“控制面板”中查看当前工程的概况信息：



(四)　选取目标板型号

在控制面板页面的右上角会看到一个芯片形状的 Target 图标，点击后在弹出的对话框内选择目标设备为“Microchip SAMA7G54 Evaluation Kit”，接着点击“Save”保存。这样我们就把部署的目标设备选择为了 SAMA7G54 评估板。

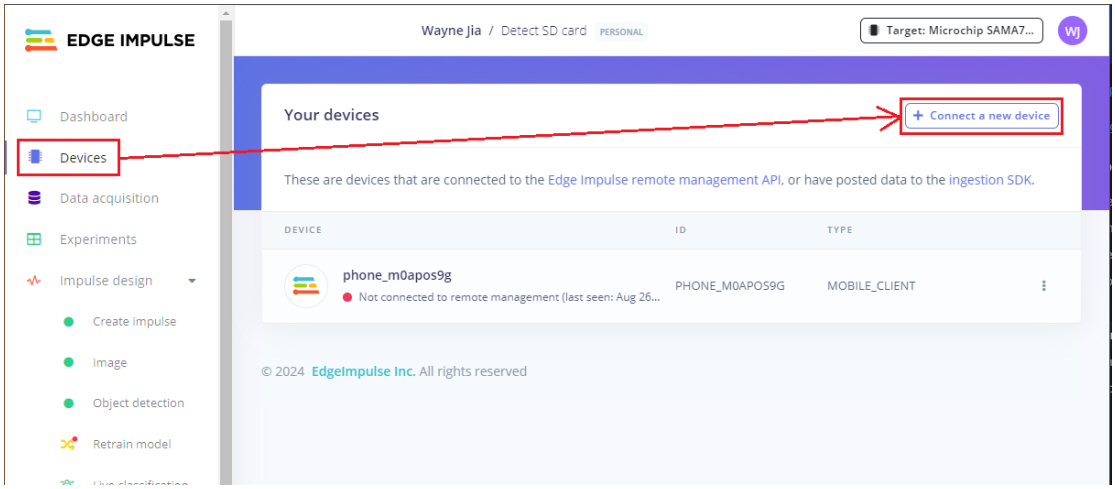


(五)　采集原始数据

在本实验中为了方便，我们使用手机摄像头进行原始数据采集。即用手机对多种 SD 卡进行拍照并上传到 Edge Impulse 云服务器进行训练。

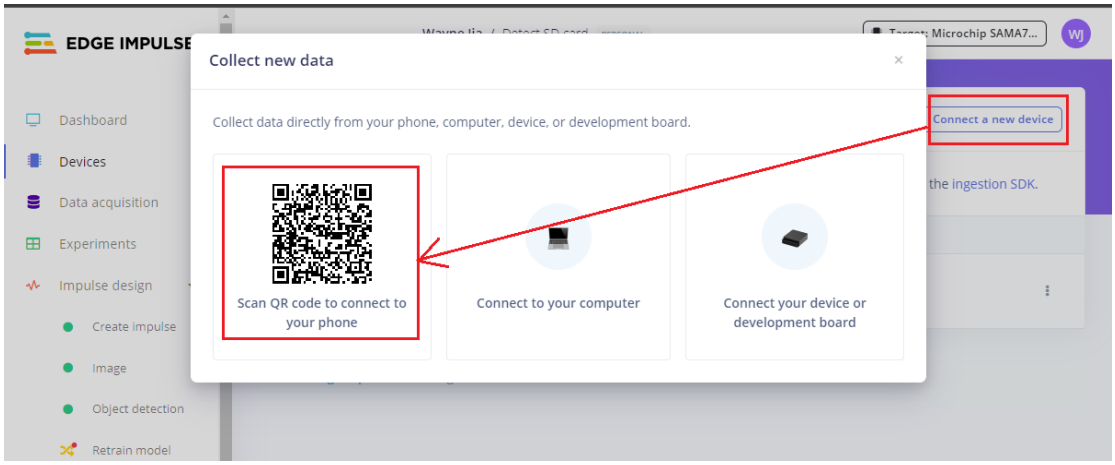
点击左侧菜单栏的“Devices”，再点击右上角按钮“+ Connect a new device”选

择采样使用的设备。

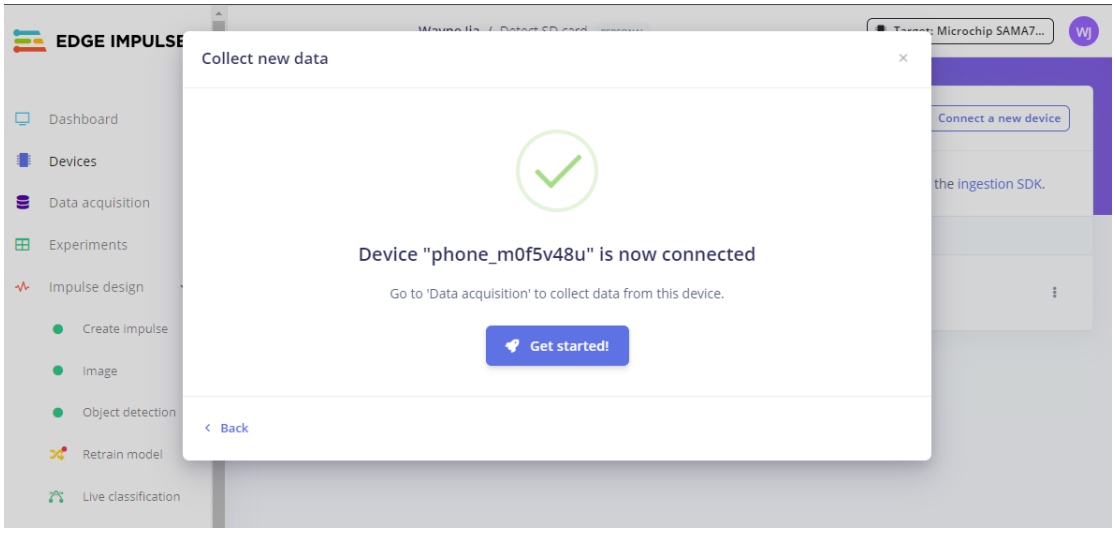


用手机或者平板设备扫描第一个选项中的二维码进行数据采集。

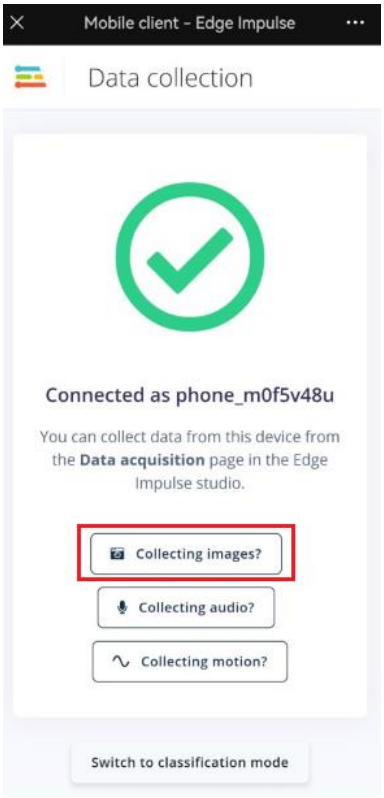
注：扫码软件可以选择手机上的浏览器，或者微信扫一扫，经测试两者都能打开 Edge Impulse 的数据采集页面。



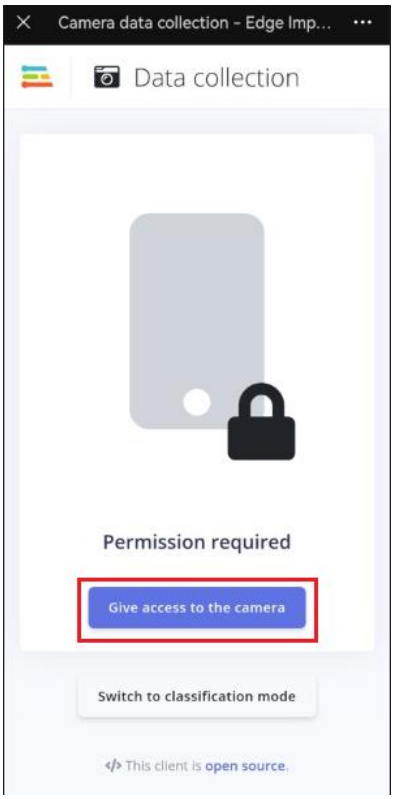
手机或平板设备成功连接后网页会显示：



在手机上，您会看到 Edge Impulse 弹出的页面，提示可以执行的操作。我们选择第一项“Collecting images?”即进行照片采集。如下图所示：



一般情况下，手机上的页面会提醒您该操作需要获取摄像头的使用权限，请点击“Give access to the camera”：

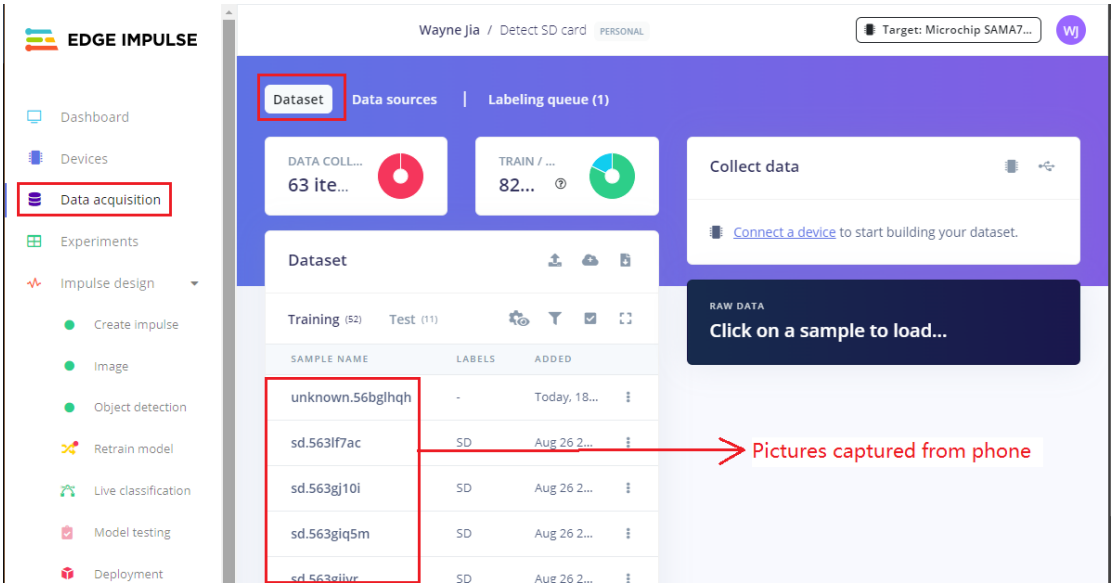


接下来就能用手机给 SD 卡拍照，点击“Capture”按钮即可将当前采集的图像上传到 Edge Impulse 的云服务器。

尽可能多拍不同角度、正反面、多品牌的 SD 卡照片到服务器以备训练使用，更多的数据将有助于训练出更好的模型。

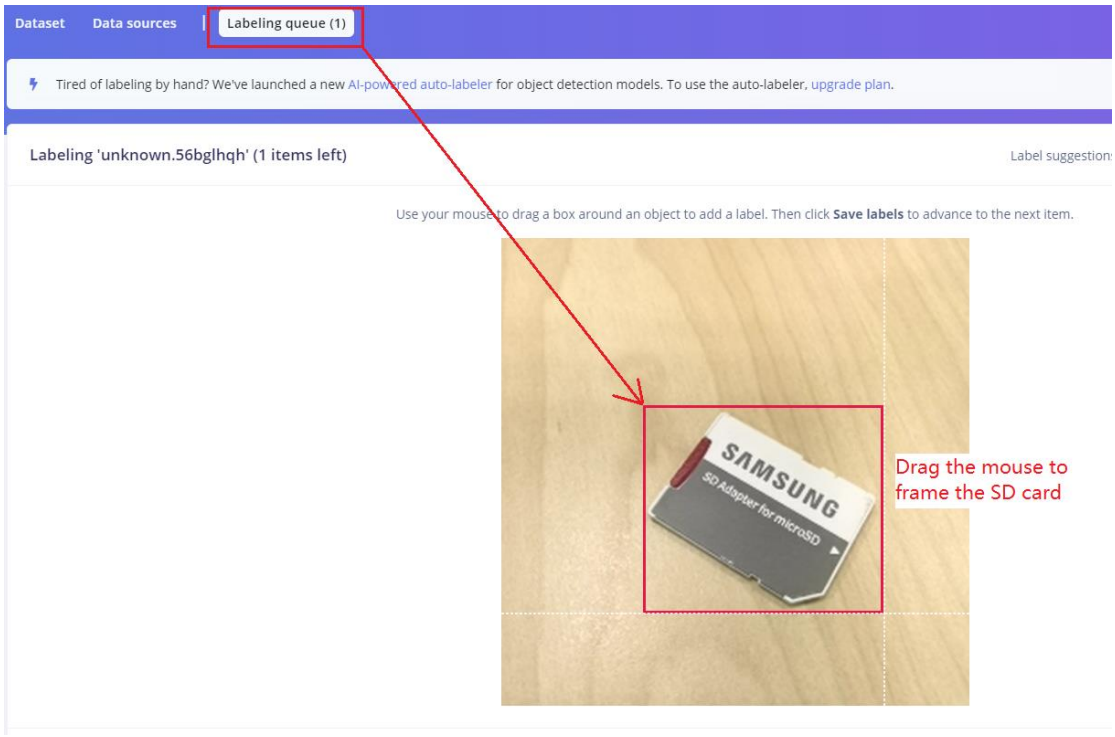
(六)　检验采集到的数据

本次实验中，共采集了 63 张不同 SD 卡的多角度照片。尽管如此，该训练数据量仍然很小。本实验为了阐述整个训练流程而没有大规模采集原始数据。在实际工程应用中，您应该采集尽可能多的数据用来训练。

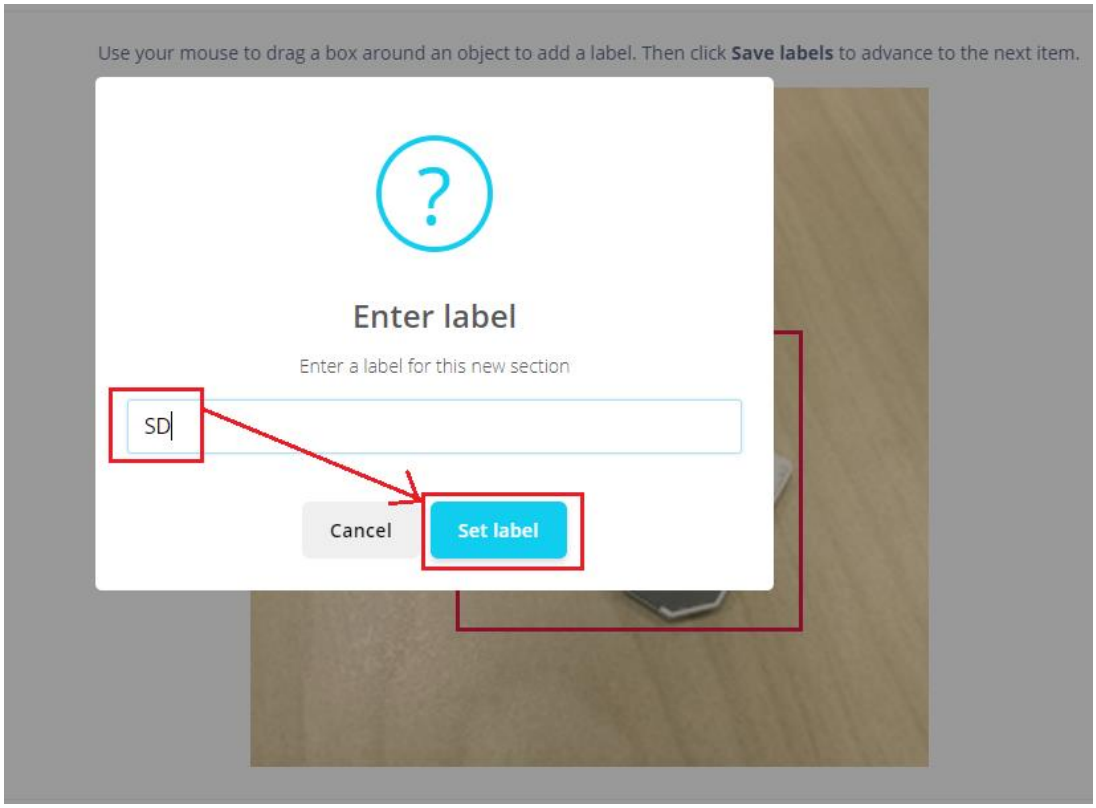


(七)　标记样本数据

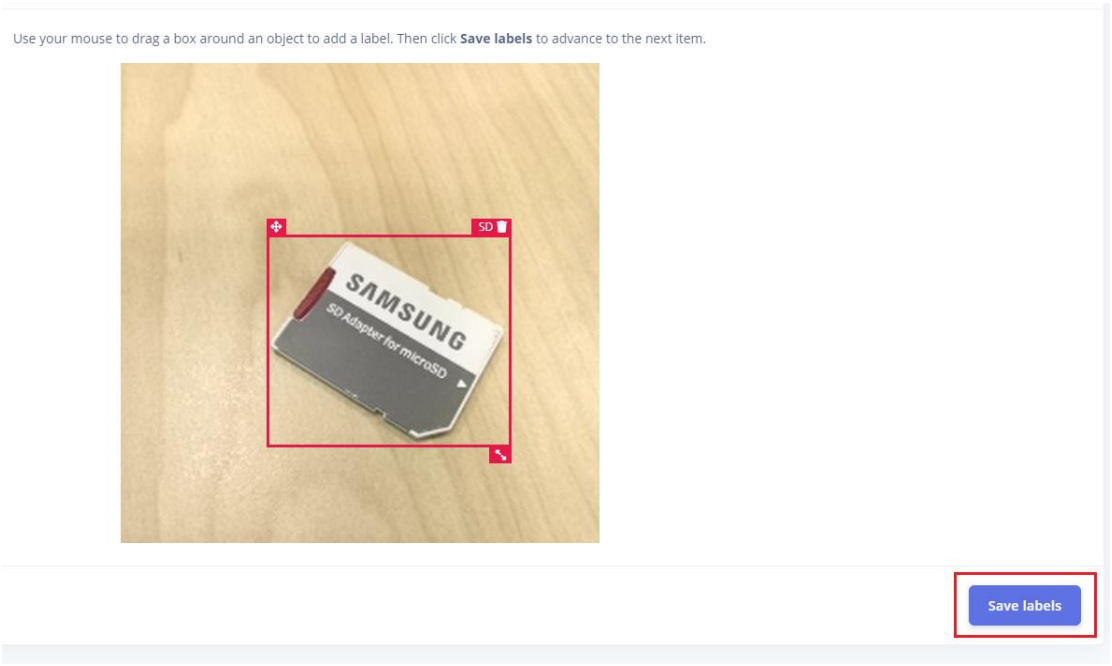
采集完成之后我们需要对样本中的目标进行标记。进入“Data acquisition”菜单后点击“Labeling queue(xx)”按钮，这里将会显示您采集到但未标记的样本数据。按住鼠标左键不放拖拽红框，使其完整框住 SD 卡全貌。



在弹出的对话框内输入标签名称，例如“SD”即可完成一张图片的标记。



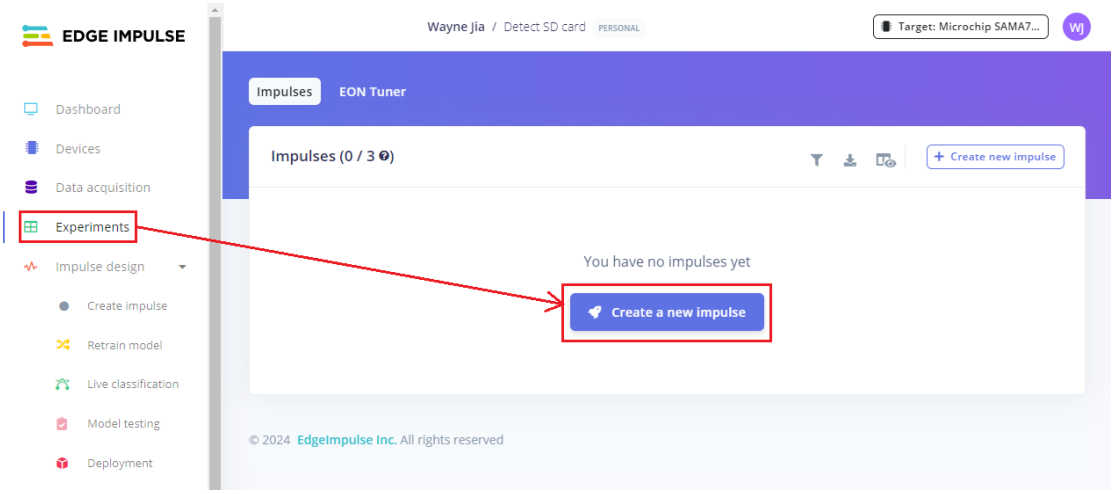
标签名设置完成后点击按钮“Save labels”保存该标记。以此类推完成所有采集照片的标记工作。



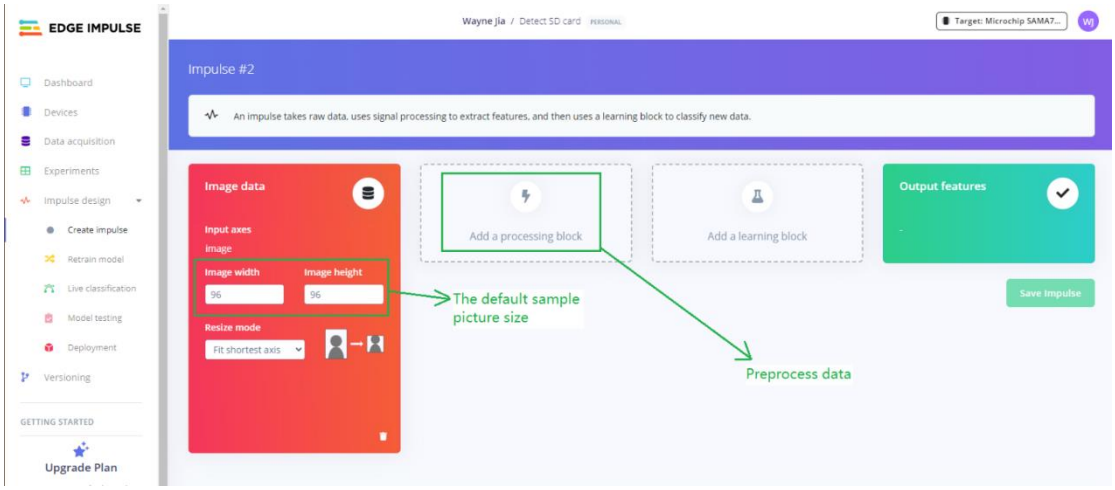
(八) 创建模型训练任务

原始数据采集完成之后即可创建模型训练任务，在这里每一个模型训练任务都被称作一个“Impulse”。

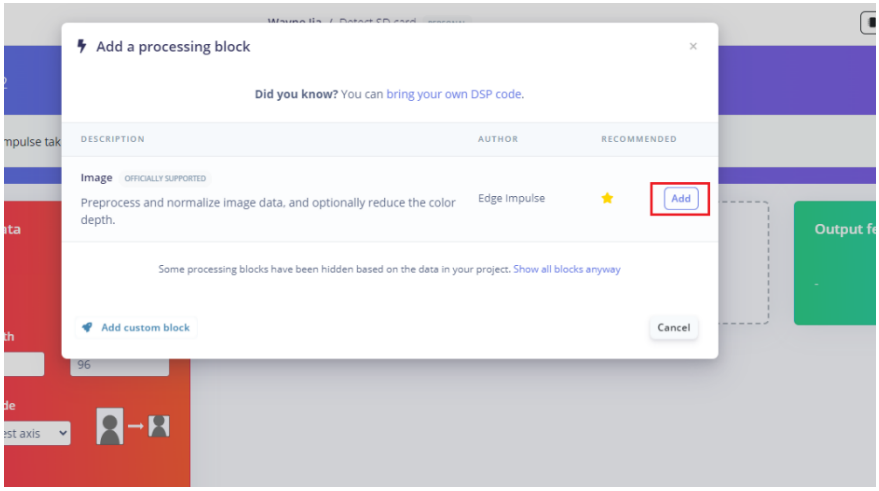
点击左侧菜单栏“Experiments”，紧接着点击按钮“Create a new impulse”来创建一个模型训练任务。



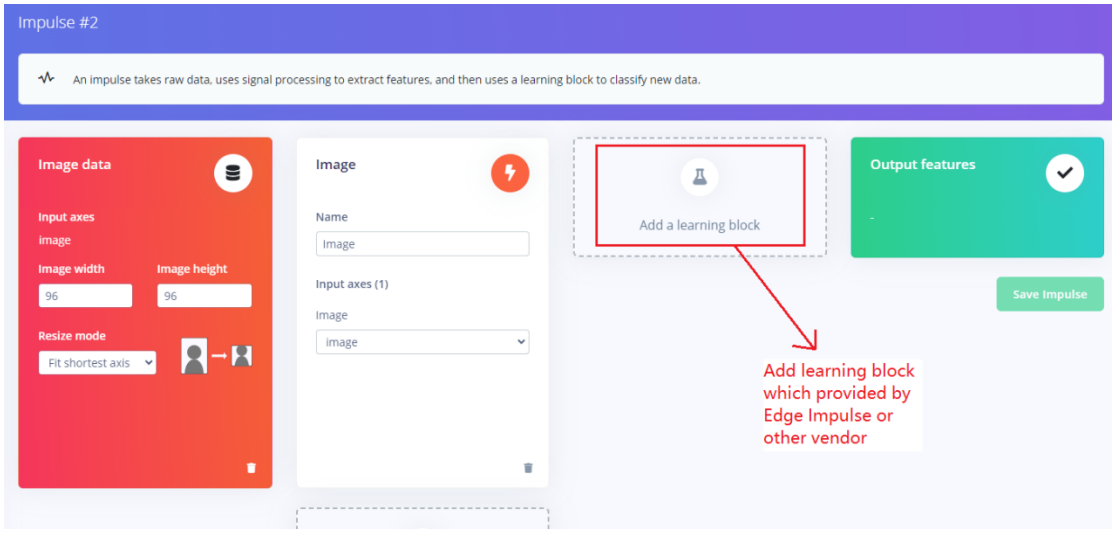
创建之后会自动进入到菜单栏的“Impulse design – Create impulse”窗口，在这里可以设置训练图片的尺寸，默认尺寸为“96x96”。接下来添加要训练的数据模块“Add a processing block”。



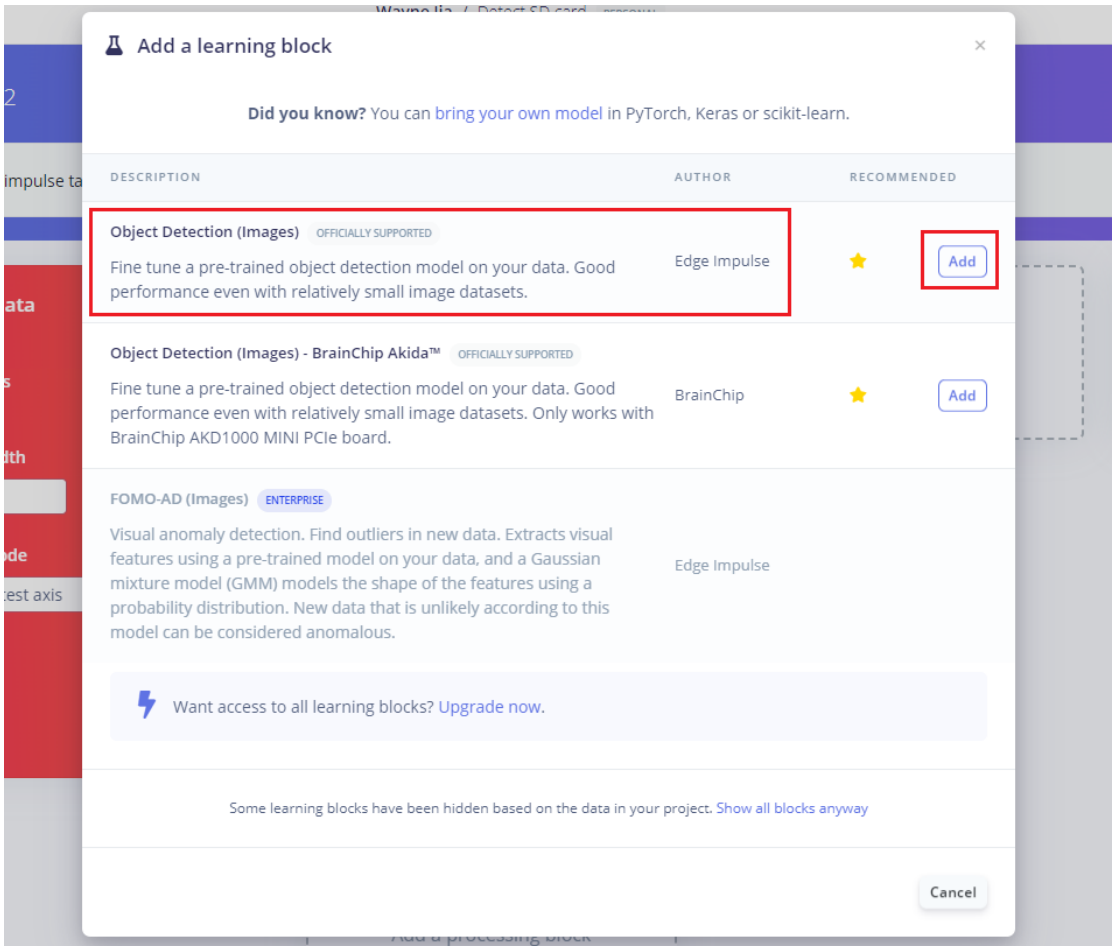
点击“Image”选项后面的“Add”按钮添加该模块。



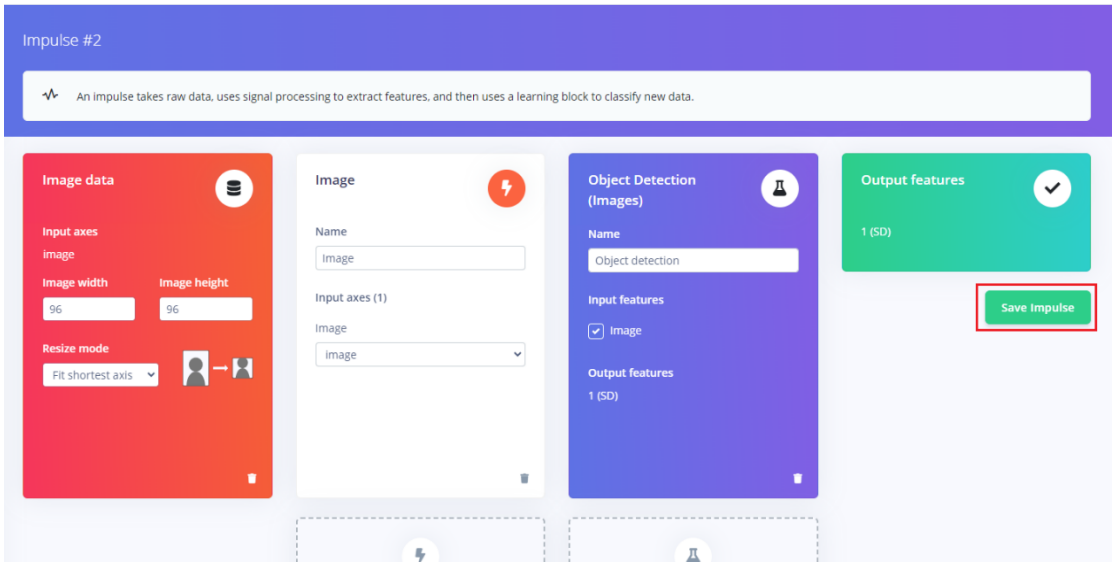
接下来点击“Add a learning block”添加一个学习模块。



在这里有很多学习模块可供选择，有些是商业版才可使用，有些需要适配特定的硬件。本次实验选择 Edge Impulse 提供的“Object Detection (Images)”模块，用作图像检测任务。

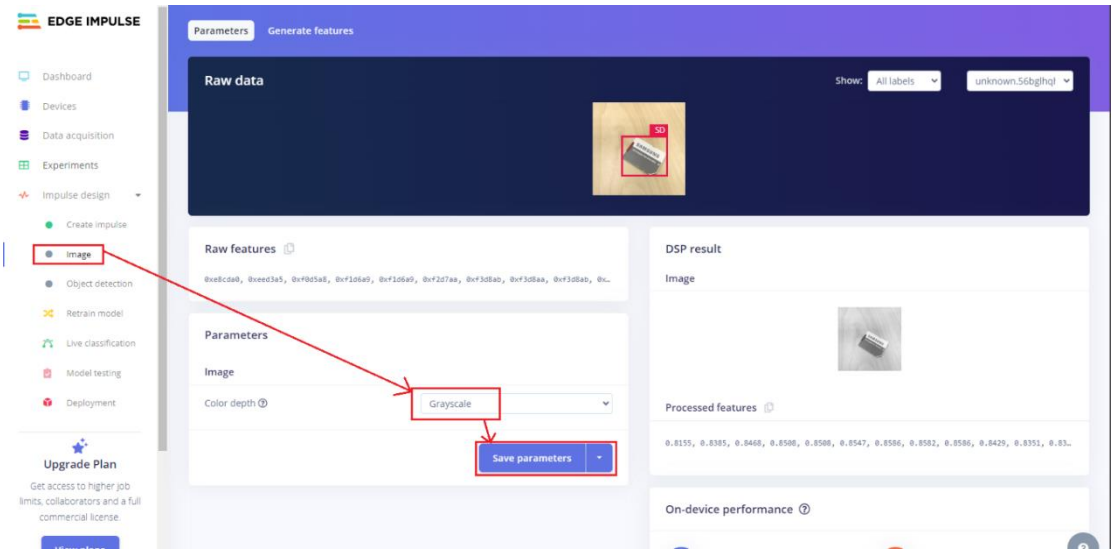


添加完成后保存当前的“Impulse”设置。

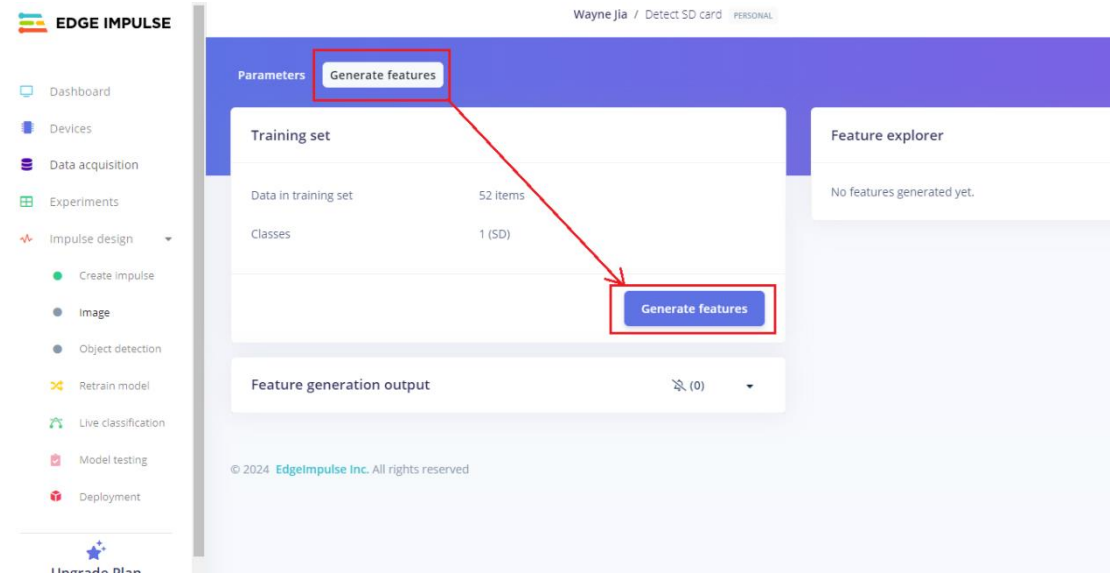


(九)　图片预处理

点击左侧菜单栏“Images”进入图片预处理的参数配置页面，在当前页面可以设置图片颜色模式为灰度或 RGB，本实验选择灰度模式：Grayscale。

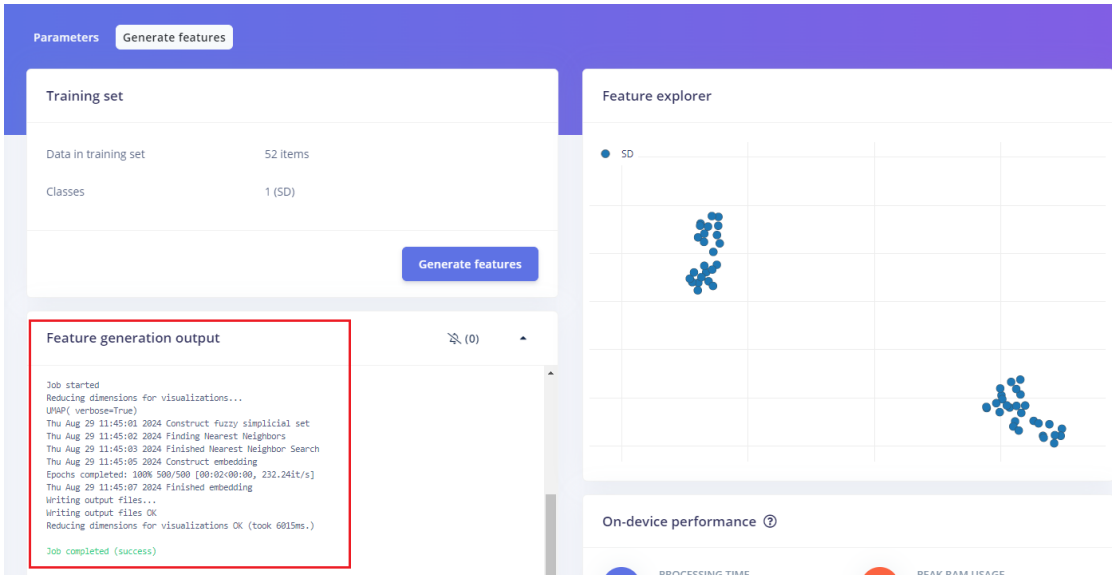


接下来点击上面的“Generate features”进入生成特征值页面，点击下面的“Generate features”按钮来生成特征值。



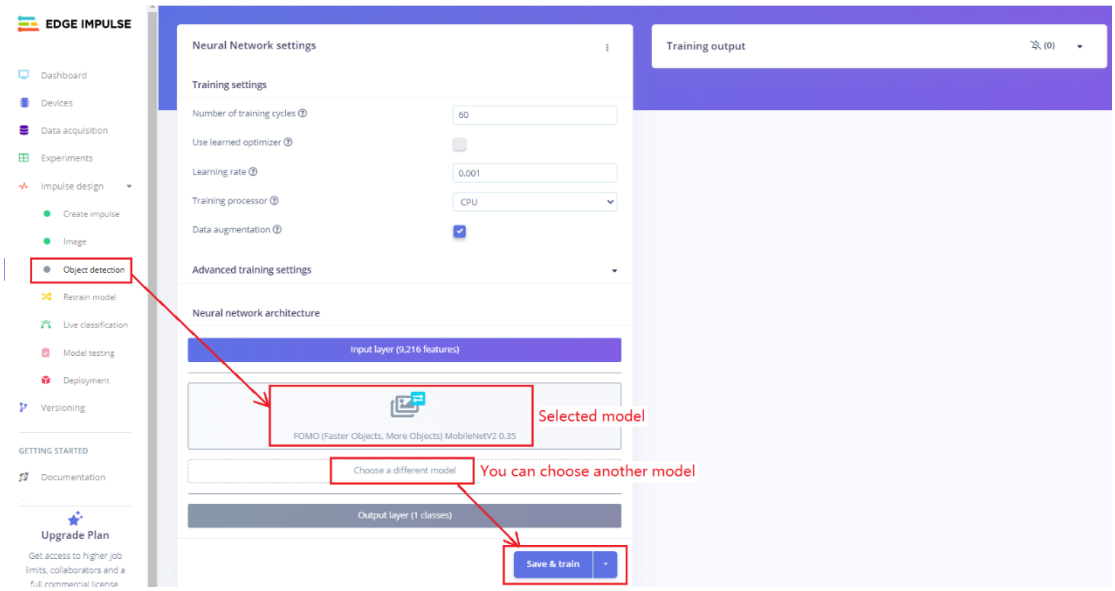
生成特征值的作用是将原始数据转换成模型可以理解的特征（features）。这些特征提取步骤对于后续的机器学习模型训练至关重要，因为它能够将原始数据（如传感器数据、音频、图像等）处理成更具代表性、能够帮助模型进行分类或预测的数值表示。它在训练工作流中是至关重要的一步，它将原始数据处理成能够被机器学习模型有效利用的特征，极大地影响模型的最终性能和效率。

生成特征值的过程可以在左下方 output 窗口查看日志输出。以及提取特征值的可视化分布图，帮助用户理解提取出来的特征值在不同类别间的分布情况。

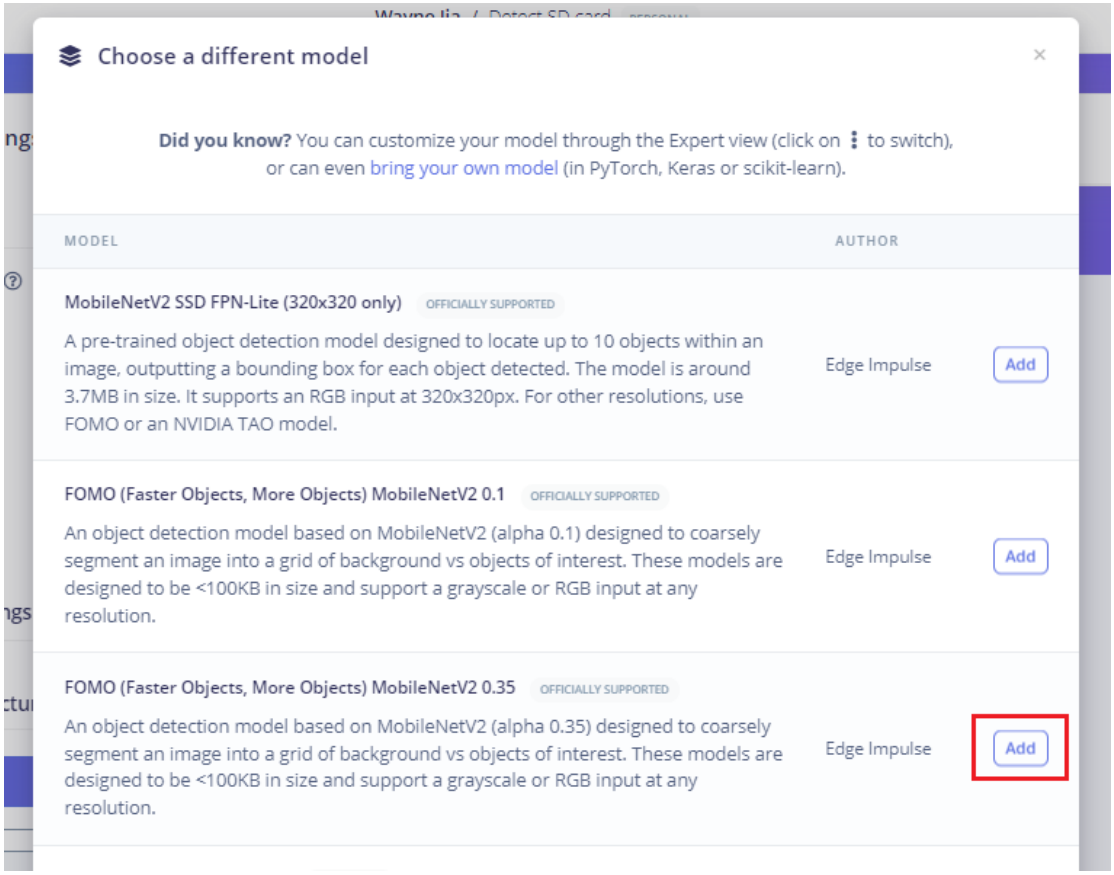


(十)　创建模型开启训练

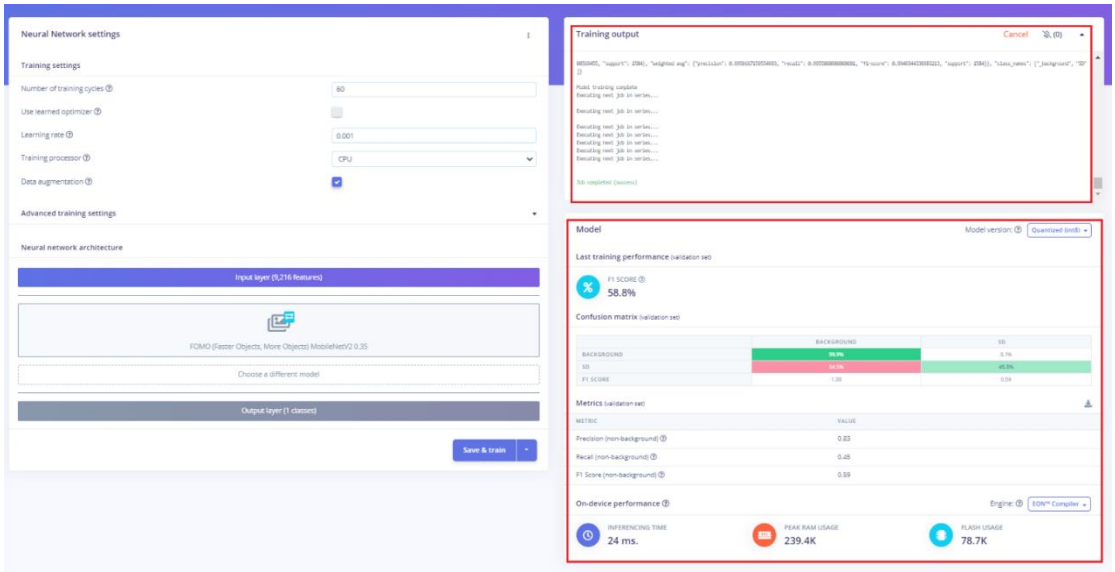
将待训练的图片数据预处理后即可创建训练模型开启训练。点击左侧菜单“Object detection”进入神经网络设置界面。在这里可以配置训练参数、选择训练框架模型，然后保存并开启训练。



本实验选择“FOMO (Faster Objects, More Objects) MobileNetV2 0.35”框架模型进行训练。您也可以选择其他模型，甚至商业模型。



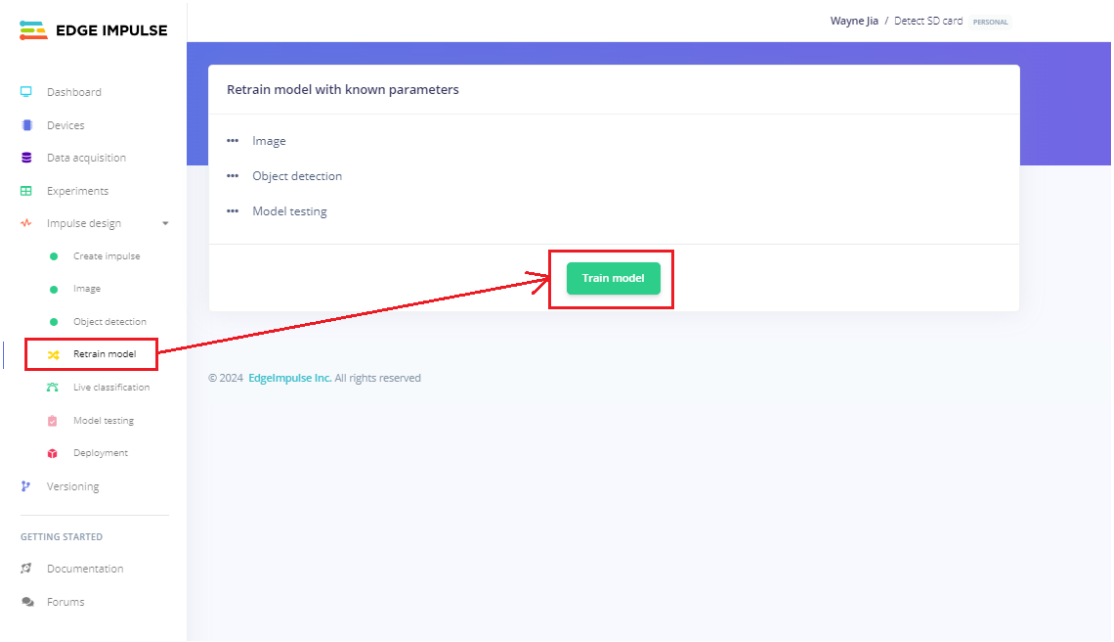
保存并开启训练后，可以在右上侧窗口查看训练日志输出。在训练结束后，在右下侧窗口可以看到模型的训练结果信息汇总。可以看到模型的识别准确率、F1分数等信息。



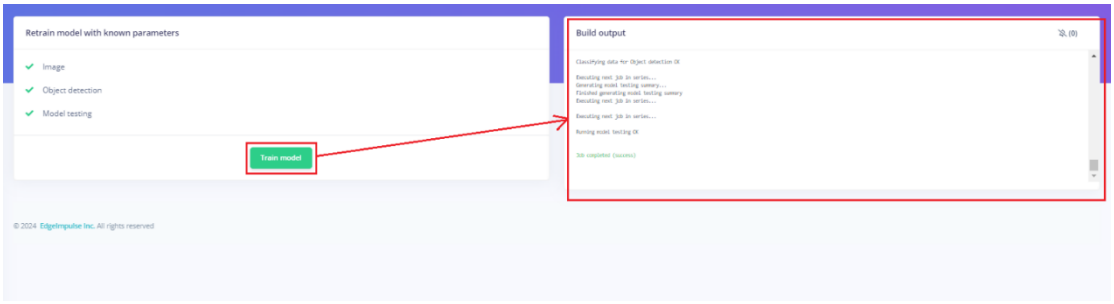
(十一) 模型再训练

初次训练之后如果添加了新数据或调整了模型参数可以进行再训练。如果没有新数据或调整模型参数，请跳过本章节。

点击左侧菜单“Retrain model”进入再训练页面，然后点击“Train model”按钮再次开启训练。

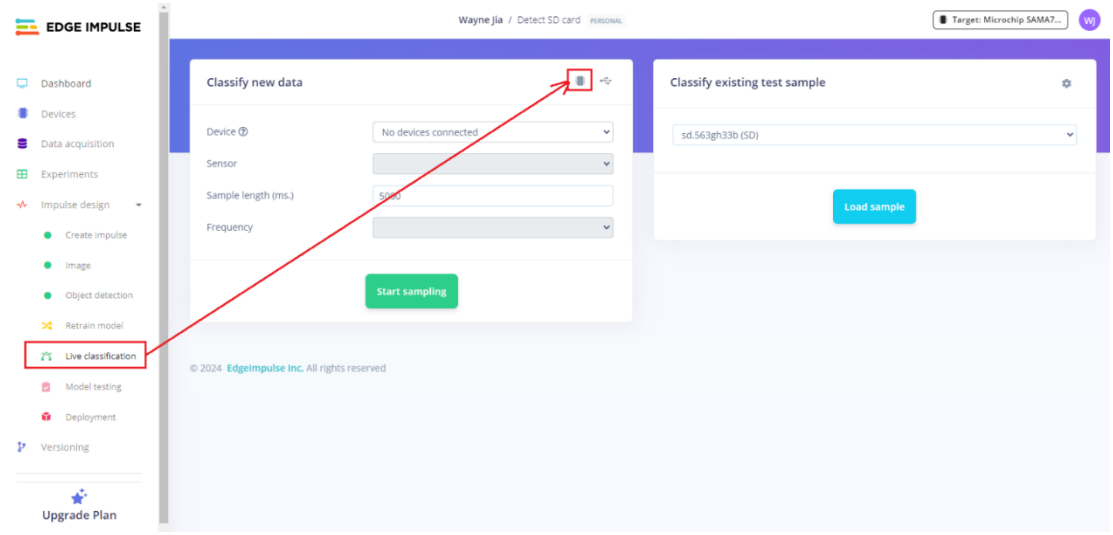


同样在右侧窗口可以看到训练输出日志信息。

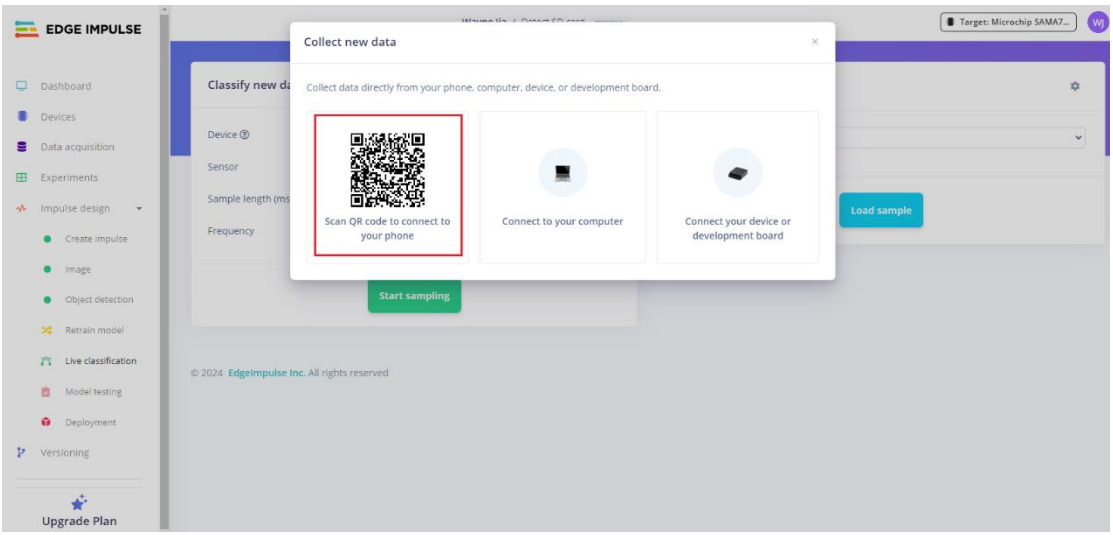


(十二) 实时测验

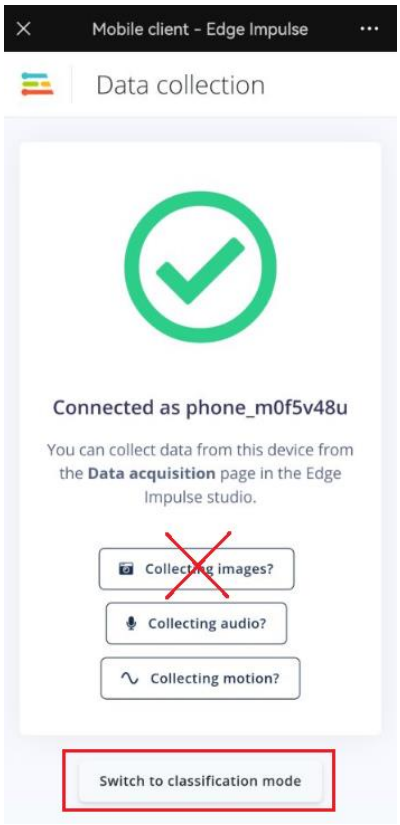
训练完成后我们可以快速测验一下模型的效果。点击菜单“Live classification”进入实时分类页面，在此使用数据采集设备来捕获新数据测验。



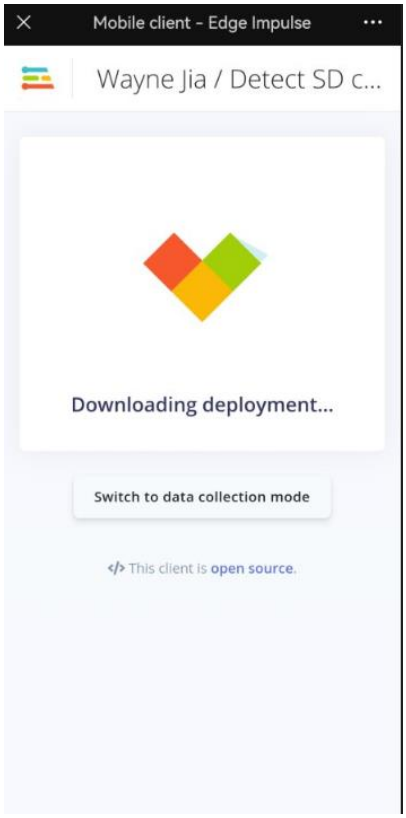
这里为了方便我们仍然使用手机摄像头来做实时测验，在弹出的选项中扫描二维码进入手机端。



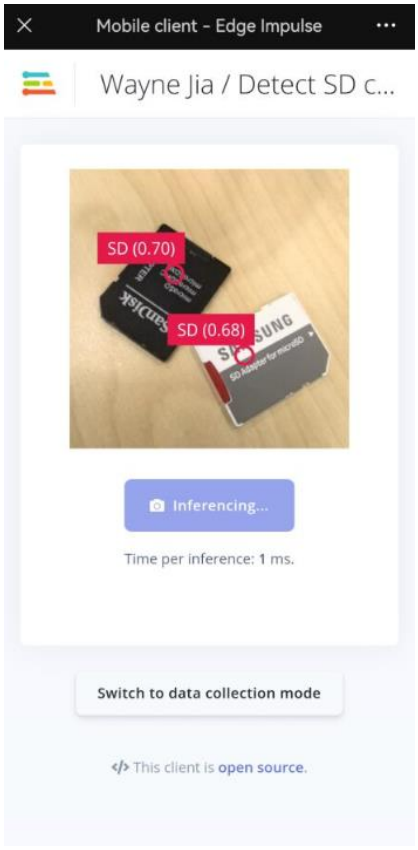
在手机上看到连接成功的界面后不要点击“Collecting images?”按钮，此时要点击最下方的“Switch to classification mode”按钮进入分类功能。



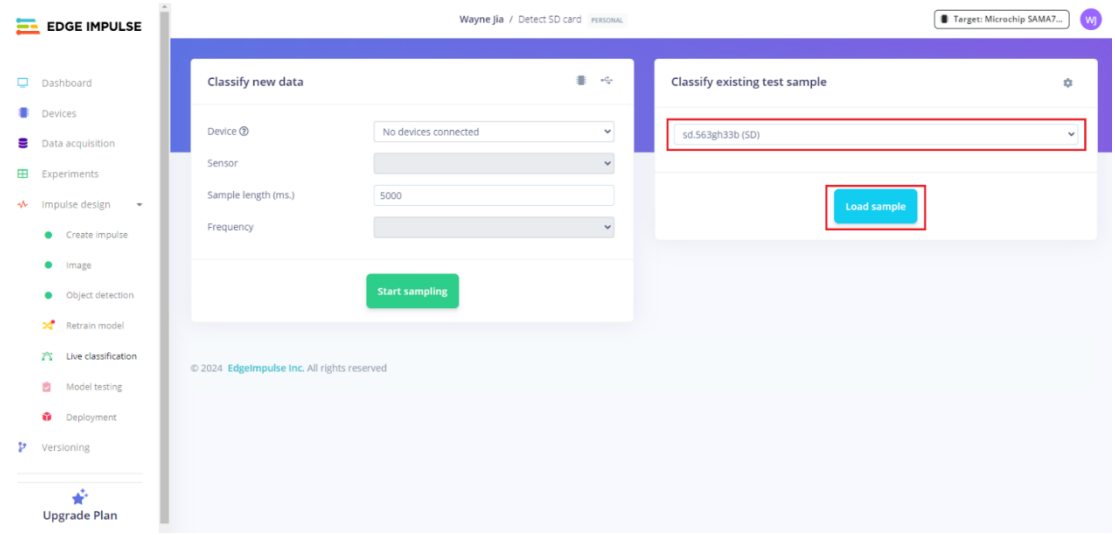
这时会下载我们训练好的模型到手机上进行部署：



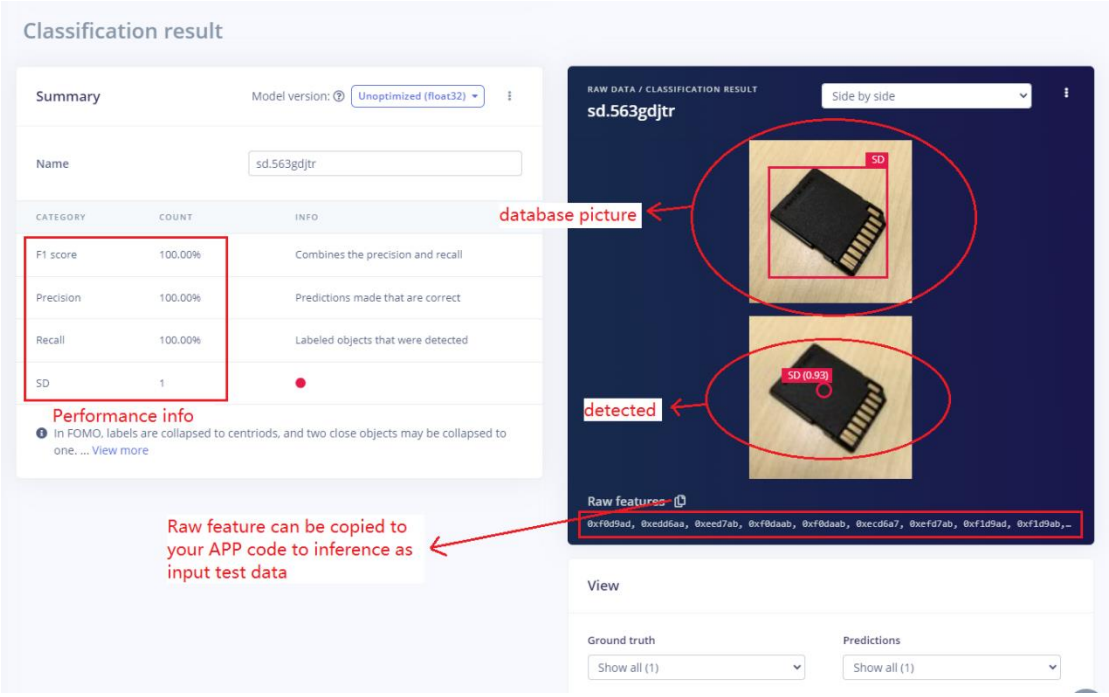
部署完成后，我们在手机端即可进行实时图像检测任务。将手机镜头对准 SD 卡，将会看到根据我们训练的模型实时识别出的效果。识别成功后在 SD 卡上显示了标签和 F1 分数。下方显示了在手机上的性能，例如每次推理时间是：1ms。



另一种实时检验模型的方法是利用采集的数据集中的 TEST 数据集（该数据集并未参与模型训练，它们专门用于模型测试）。在当前页面选择“Classify existing test sample”卡片的下拉框选择一张图片数据，然后点击“Load sample”按钮。



可以看到实时推理结果，该模型将 TEST 数据集中的图片 sd.563gdjtr 识别为 SD，F1 分数：0.93。

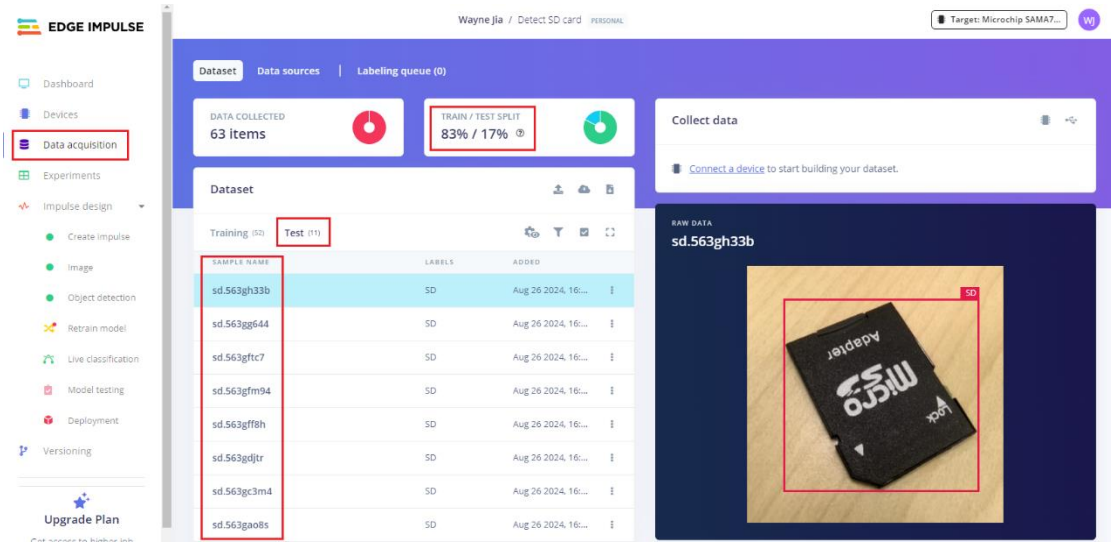


(十三) 模型测试

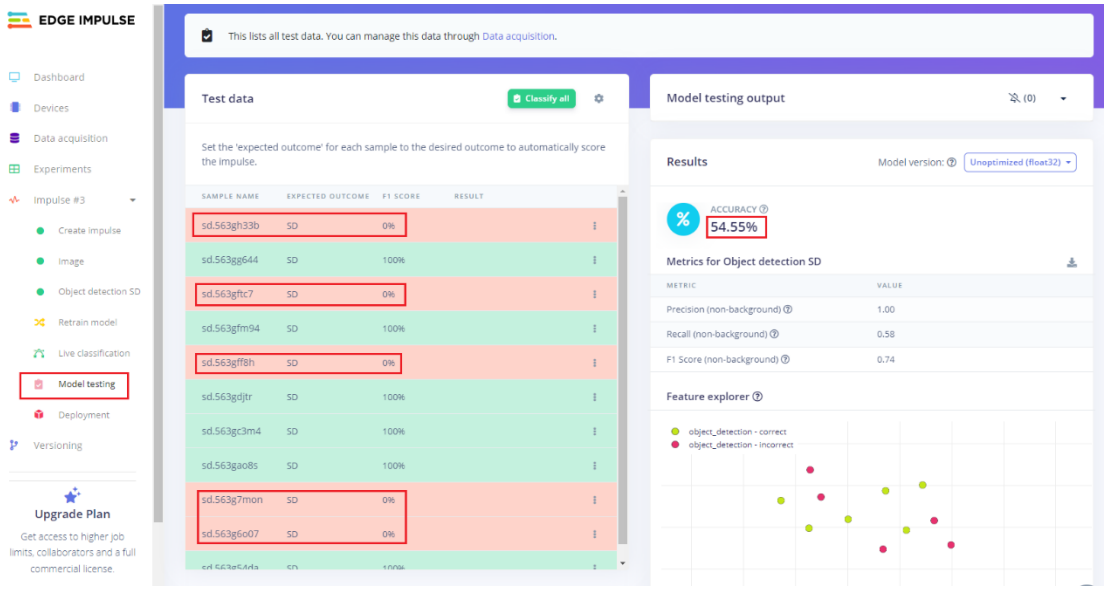
让我们回顾之前数据采集页面，我们采集的 63 条数据被分成两份，其中 83%用作训练，剩下的 17%用作模型测试。

- TRAIN: 83%
- TEST: 17%

在模型测试环节，我们将利用 17%的测试数据集来测试模型的性能。



点击左侧菜单“Model testing”之后进入模型测试页面。如下图所示，当前模型的识别准确率：54.55%。左侧的红色背景为识别失败的图片，绿色背景为正确识别的图片。如果没有执行过再训练，可以点击“Classify all”按钮来启动模型测试并查看结果。



(十四) 部署到 SAMA7G54-EK

当训练完一个模型之后我们就可以将其部署到目标板 SAMA7G54-EK 上进行测试。

在软件平台章节我们得到了 SD 卡镜像并且利用 Etcher 等烧录工具将其烧录到了启动 SD 卡，将 SD 卡插入 SAMA7G54-EK 板后启动进入 Linux 终端，使用以下用户信息登录：

- 用户名：**root**
- 密码：**edgeimpulse**

如果您想使用 SSH 方式访问评估板，请参考以下步骤：

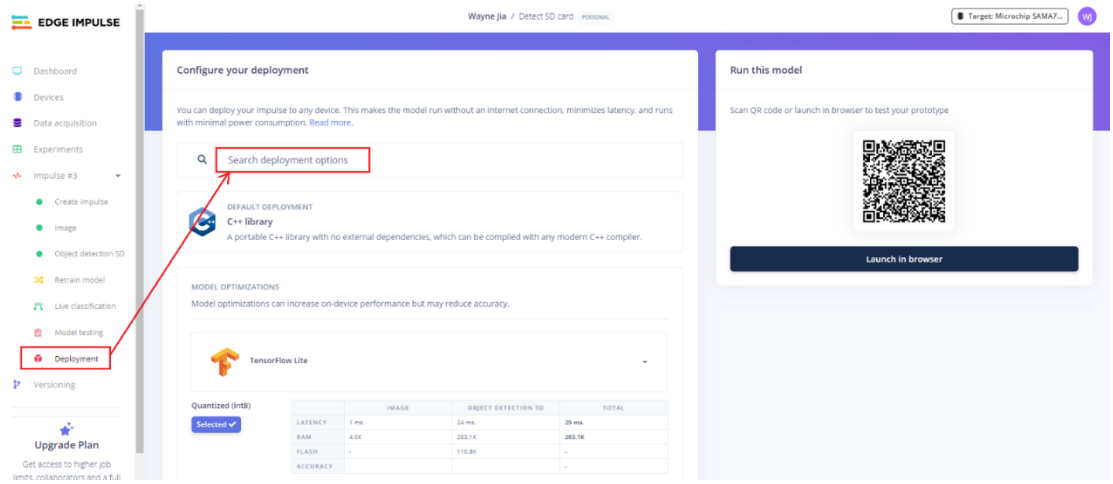
- sama7g54# **cd /etc/ssh/**
- sama7g54# **nano sshd_config**
- 取消该行注释并修改为：
    - **PermitRootLogin prohibit-password → PermitRootLogin yes**
- 取消该行注释：
    - **PasswordAuthentication yes**
- CTRL+X 然后输入 Y 再按回车键
- 重启评估板使 SSH 功能生效
- 在 Linux 命令行终端输入 **ifconfig** 获取板子 IP 地址
- 在你的主机终端上输入 **ssh root@ip** 来访问评估板

接下来我们介绍两种部署模型到评估板的方式，一种是在 Edge Impulse 云服务器构建.eim 文件，上传到评估板执行测试。另一种我们会将模型构建为 C++库，将该库复制到 SD 卡镜像的编译环境中使用 Buildroot-at91 重新编译 SD 卡镜像进

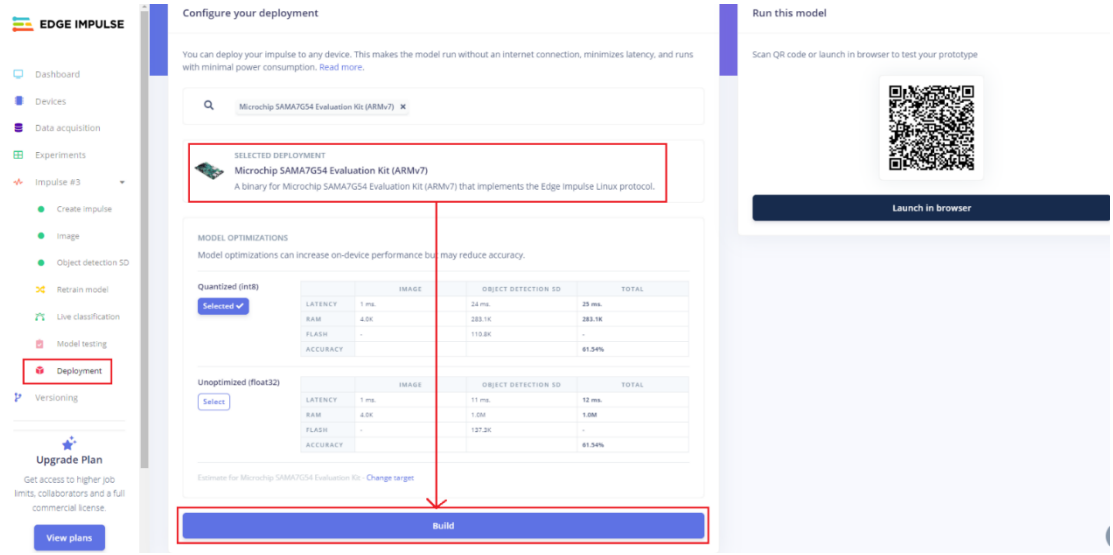
行部署。前者是 Edge Impulse 默认构建的.eim 程序，无法修改程序行为；后者灵活性更高，用户可以修改包含模型的 C++源码，修改编译构建自己想要的程序流程。

1. 部署.eim 文件

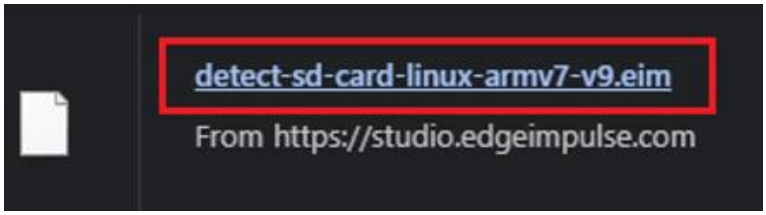
点击左侧菜单“Deployment”进入部署页面，在搜索框输入“Microchip SAMA7G54 Evaluation Kit (ARMv7)”搜索 SAMA7G54-EK 评估板。



找到 SAMA7G54-EK 评估板后点击最下方的“Build”按钮进行构建。



构建成功后浏览器会自动下载生成的.eim 文件到浏览器的默认下载目录，您可以查看浏览器下载记录：



生成的.eim 文件全名是：detect-sd-card-linux-armv7-v9.eim。

注：.eim 文件格式是 Edge Impulse 平台用于保存机器学习模型的专用格式。它是一个经过 Edge Impulse 优化后的模型文件，通常包含了模型结构、模型权重、特征提取器以及用于在设备上进行推理的必要信息。在目标板上可以使用 Edge Impulse 的 js 程序文件直接调用。

将 detect-sd-card-linux-armv7-v9.eim 文件上传到 SAMA7G54-EK 评估板的/root 目录并查看：

sama7g54# **ls**

detect-sd-card-linux-armv7-v9.eim video-capture-at91

utils.sh

使用以下命令启动该模型文件：

sama7g54# **edge-impulse-linux-runner --model-file**

detect-sd-card-linux-armv7-v9.eim

[RUN] Starting the image classifier for Wayne Jia / Detect SD card (v9)

[RUN] Parameters image size 96x96 px (1 channels) classes ['SD']

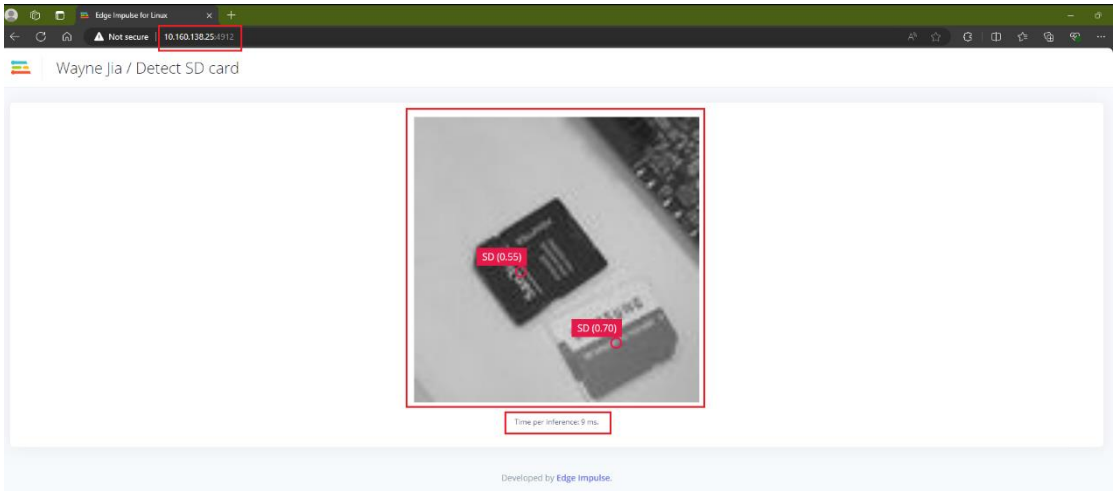
[GST] checking for /etc/os-release

[RUN] Using camera /base/soc/ehci@500000-3 starting...

[RUN] Connected to camera

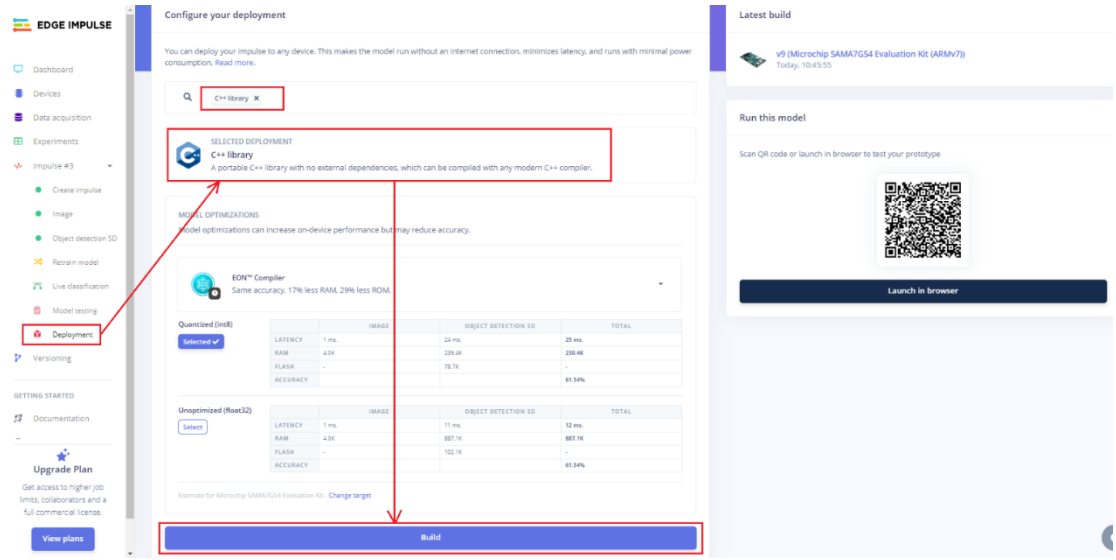
Want to see a feed of the camera and live classification in your browser? Go to http://10.160.138.25:4912

在电脑上打开浏览器输入地址和端口：http://10.160.130.25:4912，这时就能看到摄像头实时捕捉的画面。将摄像头对准 SD 卡，可以看到该模型在 SAMA7G54-EK 评估板上运行推理的实时检测效果，例如下图检测到两个 SD 卡目标以及各自的 F1 分数，在图像框的下方实时显示了在评估板上推理一次所需要的时间。

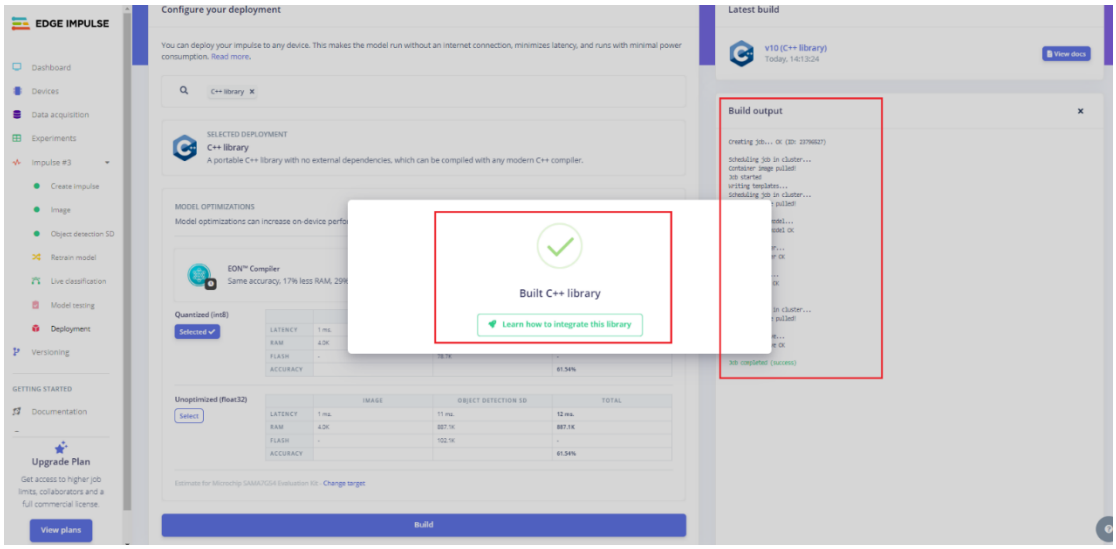


2. 部署 C++库文件

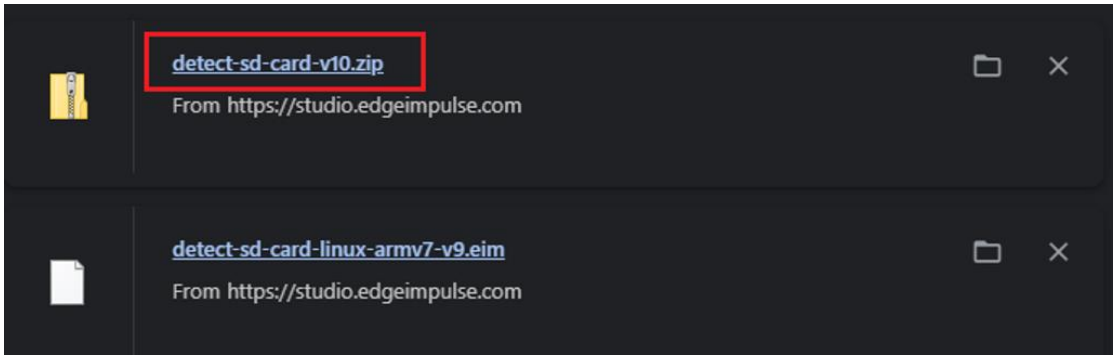
首先将模型编译构建到 C++库中。让我们回到 Edge Impulse 的部署页面。这次选择 C++ library 选项，然后点击“Build”按钮开始构建 C++库文件。



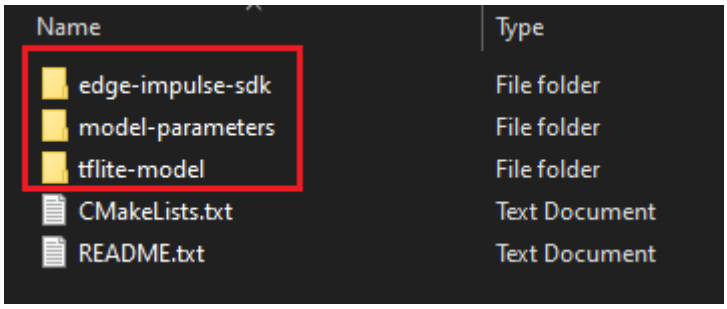
构建过程可以在右侧窗口查看输入日志信息，构建完成会弹出窗口提示。



当 C++库编译构建成功后浏览器会自动下载到默认目录。



生成的 C++库文件为 detect-sd-card-v10.zip。请注意每次构建时生成的文件名末尾版本号都会加 1。解压该 zip 文件，我们可以看到有 3 个文件夹如下图红框所示：



首先将这 3 个文件夹复制到 Ubuntu 主机的编译环境 build 目录，然后通过 build 共享目录转移 3 个文件夹到 docker 的 Buildroot-at91 编译环境中。

user@at91:~/example-microchip-sama7g54$ **sudo cp -r edge-impulse-sdk model-parameters tflite-model build**

user@at91:~/example-microchip-sama7g54$ **ls build**

at91bootstrap.bin rootfs.tar
sama7g5ek_pdmc0.dtbo
at91-sama7g5ek.dtb sama7g5ek_at25ff321a_click1.dtbo
sama7g5ek_wilc3000.dtbo
boot.bin sama7g5ek_i2s0_pcm5102a.dtbo
sama7g5-sdcardboot-uboot-4.0.9-rc1.bin
boot.vfat sama7g5ek_i2s0_proto.dtbo sdcard.img
edge-impulse-sdk sama7g5ek_isc_imx219.dtbo tflite-model
model-parameters sama7g5ek_isc_imx274.dtbo u-boot.bin
rootfs.ext2 sama7g5ek.itb uboot-env.bin
rootfs.ext4 sama7g5ek.its zImage

如果您已经退出了 docker 环境，可以使用 docker ps 命令查看之前创建的 docker id 并重新进入 docker：

user@at91:~/example-microchip-sama7g54$ **sudo docker ps**

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
a4d260547a56 microchip "/bin/bash" Less than a second ago Up 4 seconds gallant_jones

查询到了 docker id 为 a4d260547a56，使用以下命令重新进入 docker：

user@at91:~/example-microchip-sama7g54$ **sudo docker exec -it a4d260547a56 /bin/bash**

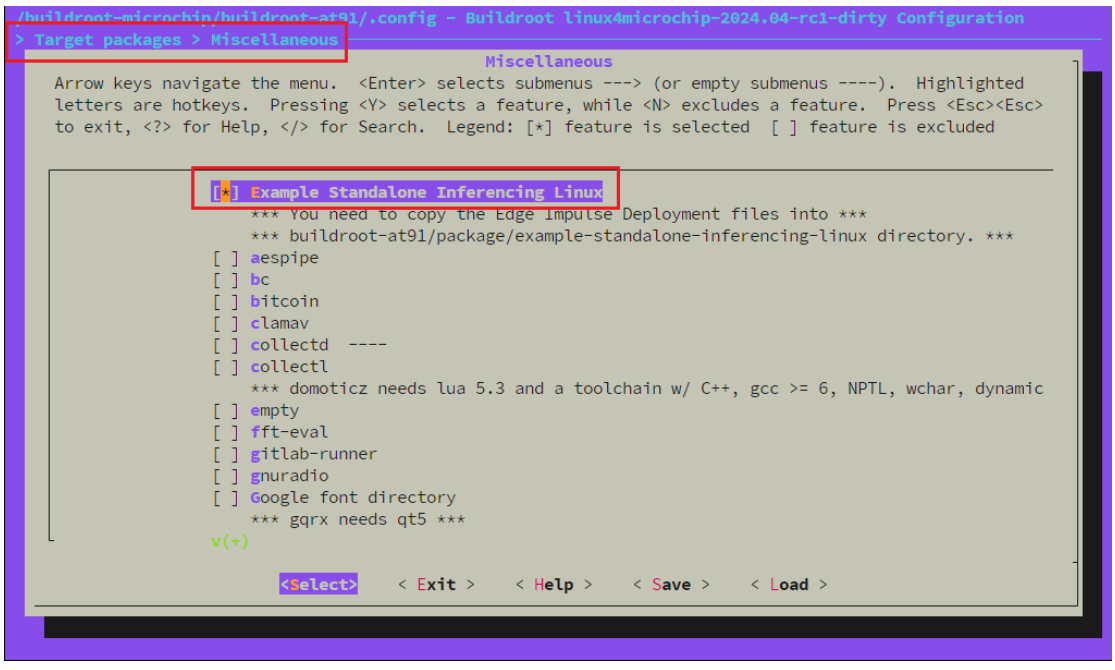
在 docker 环境中，将以上 3 个文件夹复制到 /buildroot-microchip/buildroot-at91/package/example-standalone-inferencing-linux 目录下进行编译构建。

root@a4d260547a56:/# **cd /buildroot-microchip/buildroot-at91/output/images**

root@a4d260547a56:/buildroot-microchip/buildroot-at91/output/images# **cp -r edge-impulse-sdk model-parameters tflite-model ../../package/example-standalone-inferencing-linux/**
root@a4d260547a56:/buildroot-microchip/buildroot-at91/output/images# **cd ../../package/example-standalone-inferencing-linux/**
root@a4d260547a56:/buildroot-microchip/buildroot-at91/package/example-standalone-inferencing-linux# **ls**
Config.in build-opencv-linux.sh inc tensorflow-lite tidl-rt
LICENSE build-opencv-mac.sh ingestion-sdk-c tflite utils
Makefile edge-impulse-sdk model-parameters tflite-model
README.md example-standalone-inferencing-linux.mk source third_party

复制完 3 个文件夹后，如果您要修改程序，请进入 source 目录，找到 custom.cpp 进行修改，该文件编译后最终在评估板上执行。里面包含了调用 Edge Impulse 生成的模型及其他 API 调用。
root@a4d260547a56:/buildroot-microchip/buildroot-at91/package/example-standalone-inferencing-linux# **ls source**
audio.cpp camera.cpp collect.cpp custom.cpp eim.cpp

接下来在 menuconfig 中将该 C++模块添加到 SD 卡镜像文件中：
root@a4d260547a56:/buildroot-microchip/buildroot-at91/package/example-standalone-inferencing-linux# **cd ../..**
root@a4d260547a56:/buildroot-microchip/buildroot-at91# **make menuconfig**



打开 menuconfig 后进入目录：Targe packages → Miscellaneous。然后在“Example

Standalone Inferencing Linux”选项的方框中按空格键选择该选项，保存退出，在命令行执行构建任务：

root@a4d260547a56:/buildroot-microchip/buildroot-at91# **make**

make: Warning: File 'docs/manual/manual.mk' has modification time 1087962 s in the future

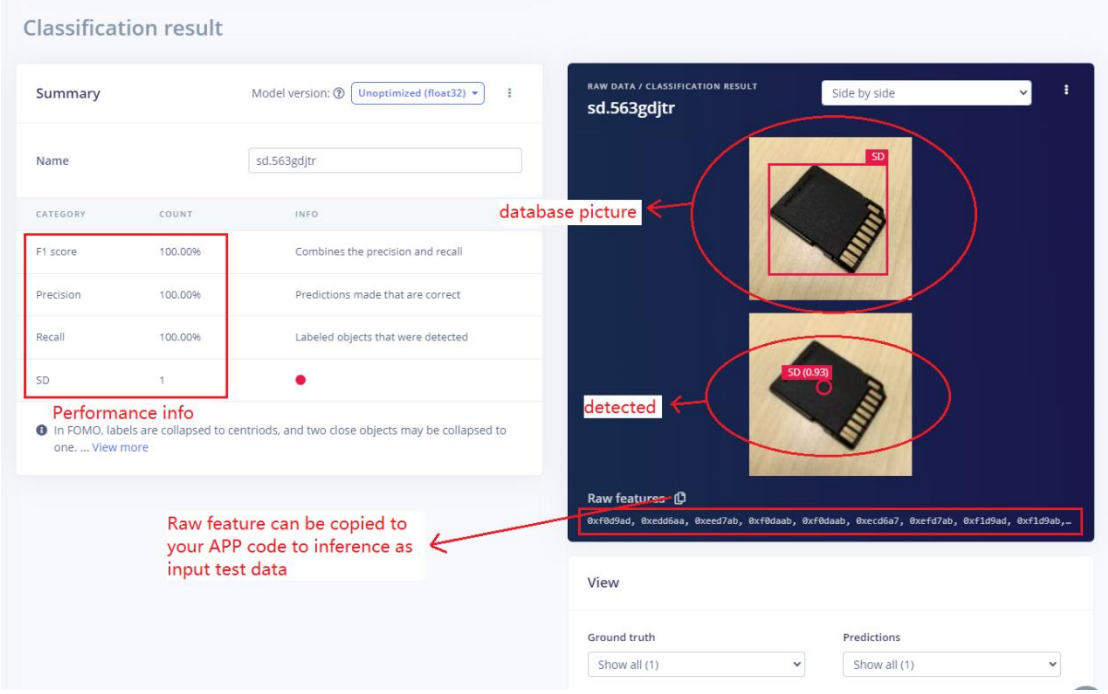
……

构建完成后将生成的 sdcard.img 文件烧录到 SD 卡并插入 SAMA7G54-EK 评估板进行测试准备。板子启动后在/home 目录下会发现一个名为 custom 的可执行程序：

sama7g54# **ls /home/**

custom

该程序即由前面讲过的 custom.cpp 编译而来。它在运行时需要一个输入文件，文件内容为需要测试推理的图片原始特征数据，该数据我们可以在数据集中快速获取。在第(十一)实时测验章节我们看到了图像原始特征的复制按钮，点击该按钮即可复制该图片的原始特征数据，将其粘贴到一个名为 raw_features.txt 的文本中保存并上传到板子的/home 目录。



接下来就可以使用 custom 程序调用该特征数据进行推理实验：

sama7g54# **./custom raw_features.txt**

Predictions (DSP: 5 ms., Classification: 80 ms., Anomaly: 0 ms.):

#Object detection results:

SD (0.886719) [x: 48, y: 40, width: 8, height: 8]

sama7g54# **ls**

custom debug.bmp raw_features.txt

从以上推理结果看，利用该 C++模型库推理该图片为 SD 标签，F1 分数：0.886719，

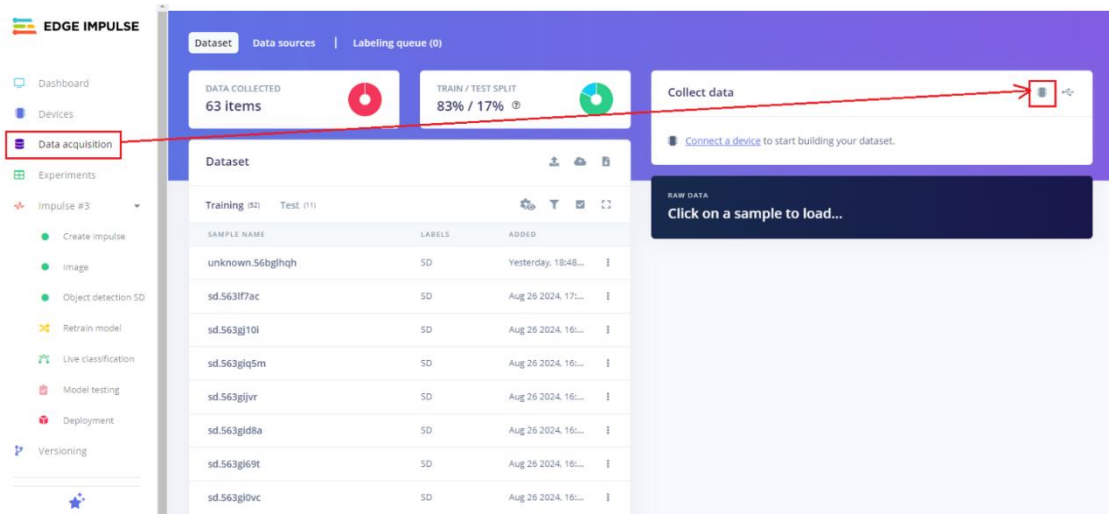
以及图片尺寸位置信息。同时运行完还生成了一张检测的 BMP 图片：debug.bmp。

六、 进阶

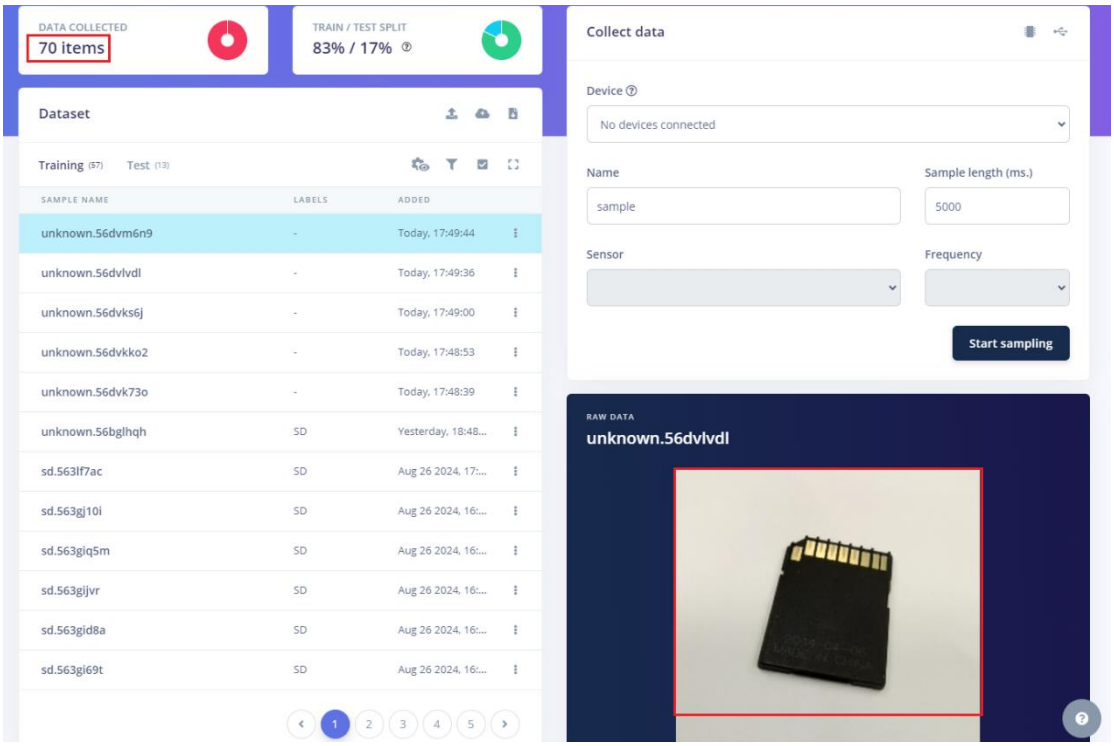
本文介绍了利用 Edge Impulse 训练模型并部署到 SAMA7G54-EK 评估板的过程，同时我们发现模型性能不高，准确率只有：54.55%，本章节介绍两种方法来优化我们的模型。

(一) 改进图片数据的质量和数量

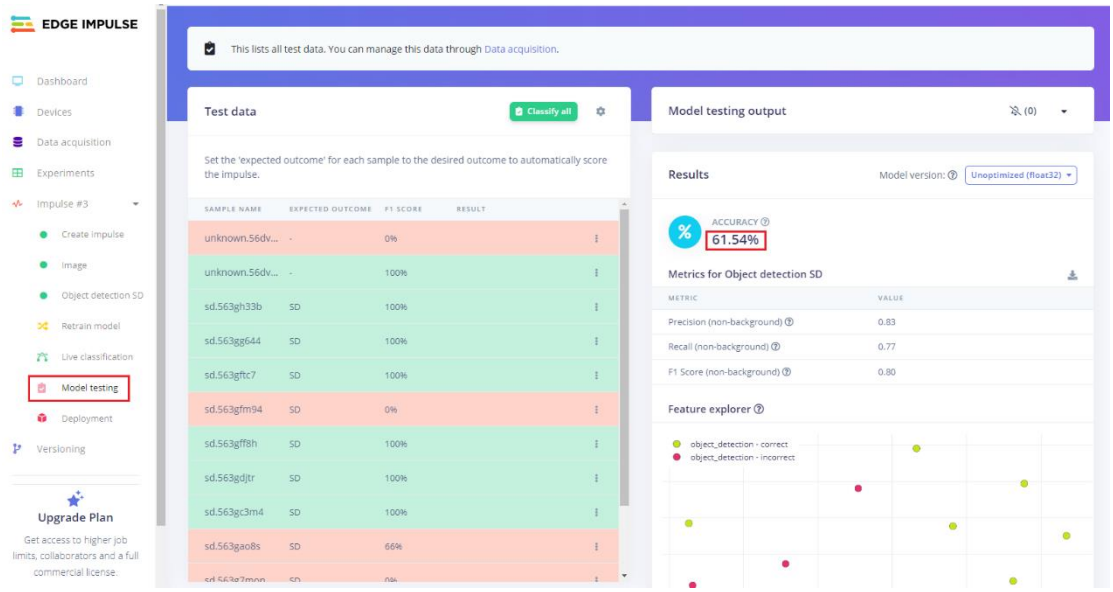
增加训练图片的数量，同时在采集时避免噪声干扰。进入菜单“Data acquisition”页面，点击采集设备图标采集更多图片数据。



选取简单的背景拍摄目标对象，采集更多的新鲜数据以备训练。



采集完成后进入“Retrain model”菜单进行再训练，待重新训练结束，进入“Model testing”检查模型测试效果。我们可以看到准确率相比之前略有提升：



但这个样本量对于机器学习来说非常小，我们仅仅用它来讲解训练的流程，实际应用中需要采集大量的原始数据进行训练才能得到比较好的模型。

(二) 使用 EON Tuner

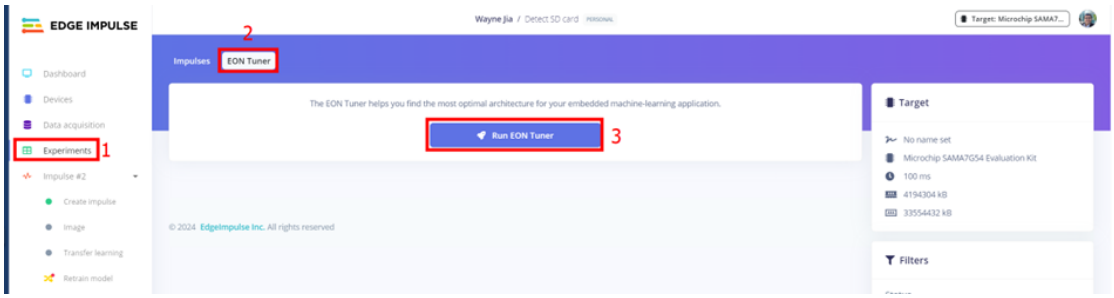
经过上面改进图片质量、数量的优化步骤后，模型的准确率提升到了 61.54%，

但仍然不够理想，最终结果实际上取决于训练阶段使用的模型的设置。这就是为什么我们需要找到一种方法来微调这些设置。

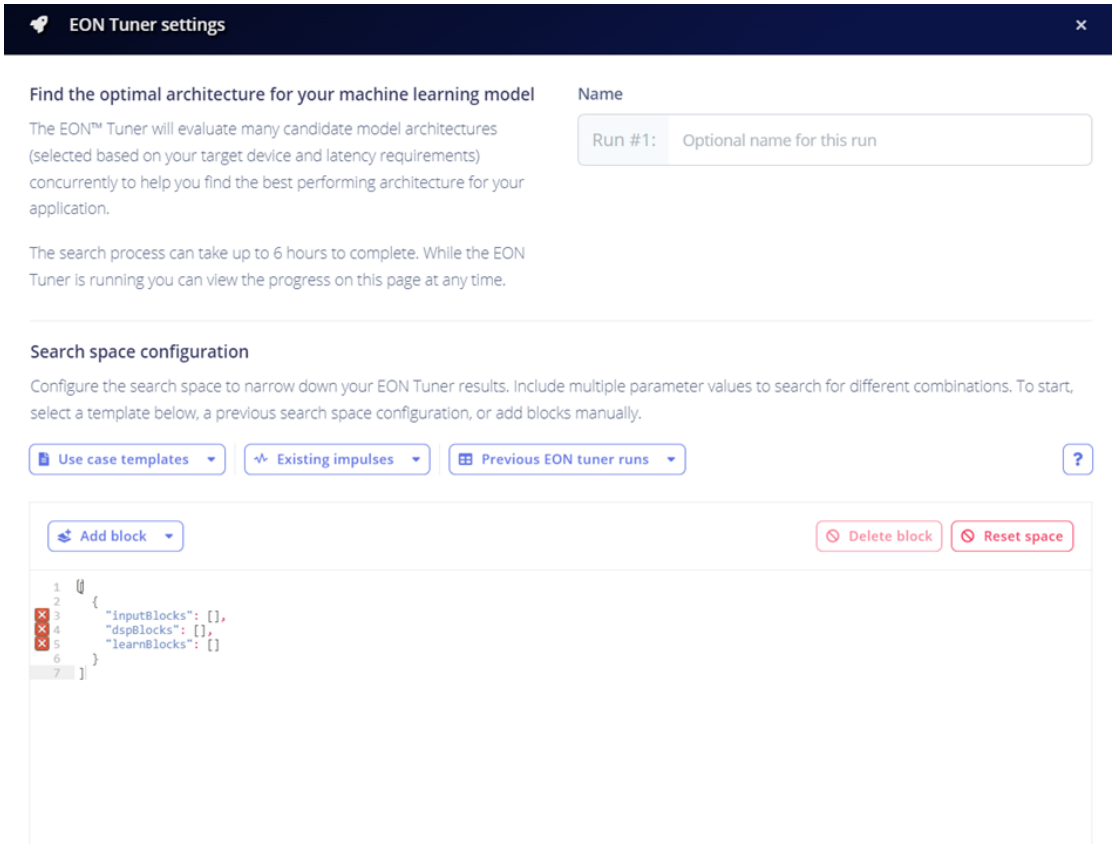
Edge Impulse 提供的最强大工具之一，正是专门用于此任务的。这就是 EON Tuner，一款可以自动测试不同模型和不同训练设置的工具，并向用户展示最佳结果。

注：EON Tuner 是 Edge Impulse 的“专业”和“企业”计划中的一项功能。您可以免费试用 14 天（无需绑定信用卡）。通过以下链接注册免费试用：
https://studio.edgeimpulse.com/trial-signup

点击左侧菜单栏“Experiments”进入实验界面，然后点击上方的 EON Tuner 菜单，接着点击“Run EON Tuner”开始运行：

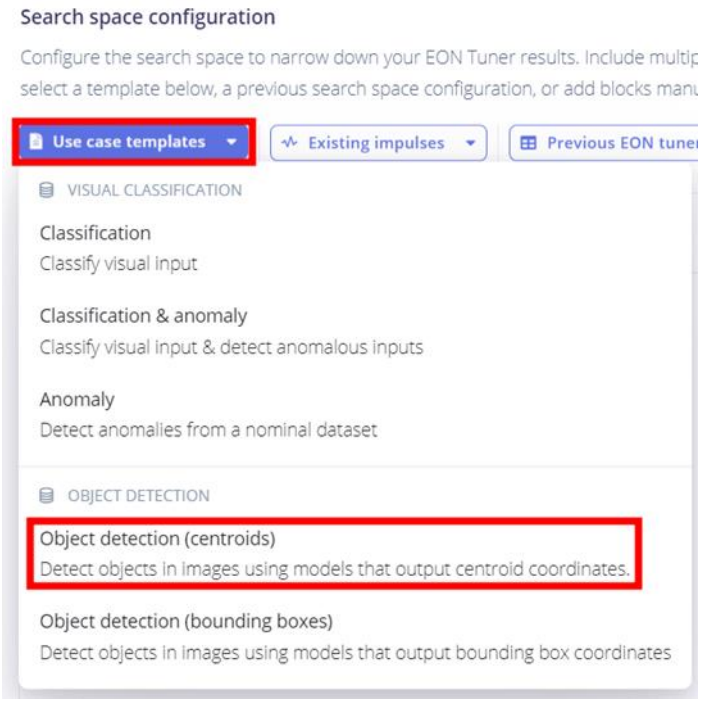


运行之后会看到以下窗口：

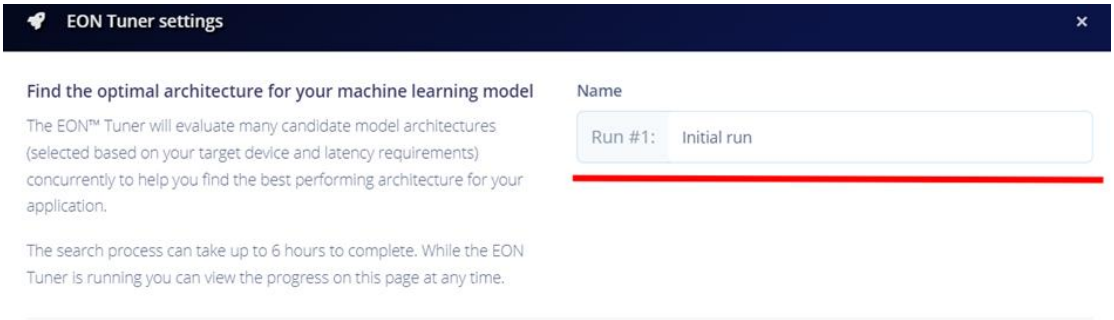


可以深入自定义 EON Tuner 的运行方式，但目前我们将重点放在如何从模板运行。

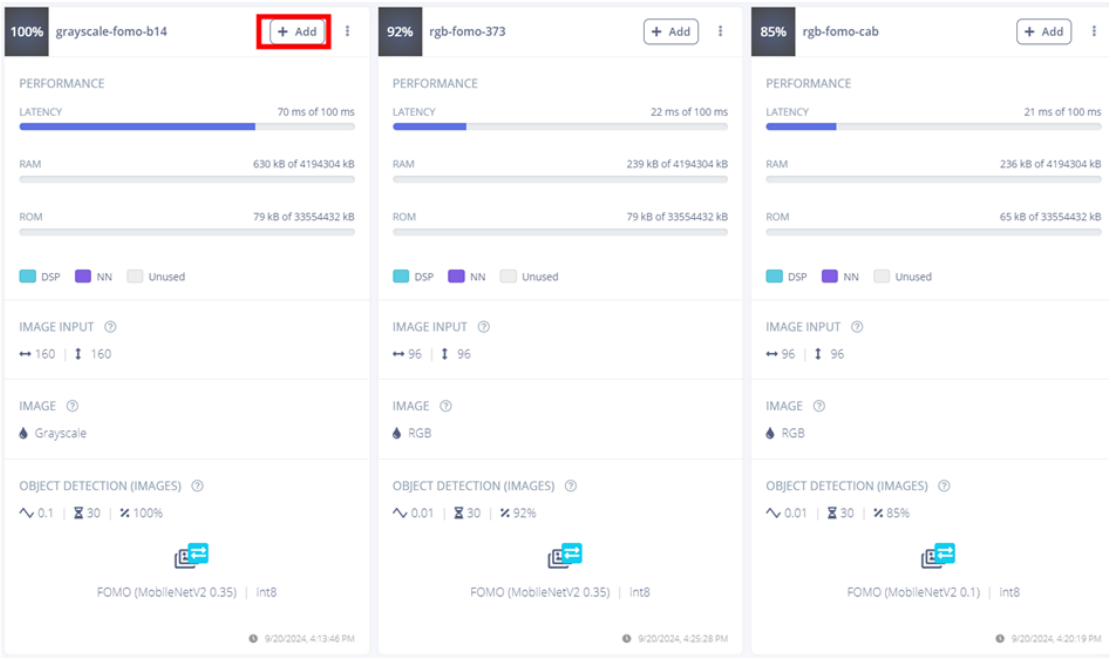
点击“Use Case Templates”然后选择“Object Detection (centroids)”选项：



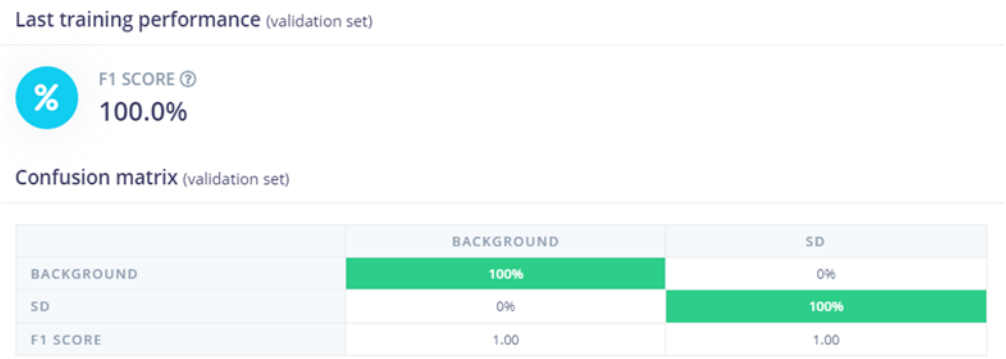
如您所见，“Search space configuration”已更新为所选模板。您可以在右上角的文本字段中为每次 EON Tuner 运行指定一个特定名称，例如：



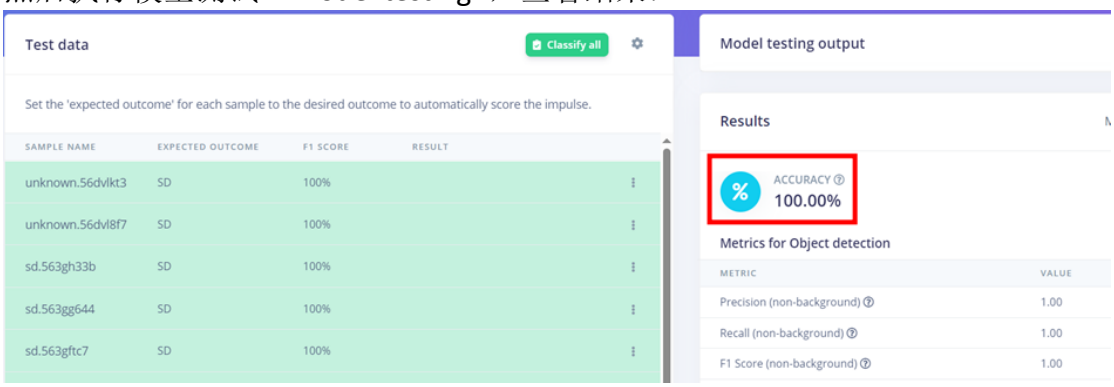
准备就绪后，您可以启动 Tuner。这将花费相当长的时间运行。完成后，您可以查看不同的结果，并根据应用需求选择您喜欢的模型。在本例中，我们将选择第一个：



现在，您可以在“Object Detection”选项卡中看到，准确率已显著提高：



然后执行模型测试“Model testing”，查看结果：



使用 EON Tuner 是一种轻松测试各种模型架构的好方法。您可以自定义其工作方

式，例如测试更多或不同的设置。请查阅 Edge Impulse 文档了解更多信息。
https://docs.edgeimpulse.com/docs/edge-impulse-studio/eon-tuner

七、 总结

本文介绍了机器学习模型训练工具 Edge Impulse 在 Microchip SAMA7G54-EK 评估板上使用的方法，通过介绍图像检测模型训练的流程，用户可以学到如何构建一个图像检测模型的完整过程，并将该模型最终部署到嵌入式设备 SAMA7G54-EK 上运行推理测试。文章还介绍了优化模型的方法，使得用户可以根据项目应用需求对模型进行优化改进，最终在 SAMA7G54-EK 上运行一个理想的人工智能应用。