AN13924

Multiple Person Detection with High-Efficient Neural Network on i.MX RT1060 and RT1170

Rev. 0 — 8 May 2023 Application note

Document Information

Information	Content
Keywords	Edge Computing Platform, RT1060, RT1170, eIQ, Machine Learning, uVITA, COCO, Pascal-VOC, Person Detection
Abstract	The crossover MCUs of NXP are ideal edge computing platforms and provide superior computing power.

1 Introduction

The crossover MCUs of NXP are ideal edge computing platforms and provide superior computing power. To further show the capability of the i.MX RT family MCUs for machine learning technology, this document introduces an example of multiple-person detection using the high-efficient neural network on the i.MX RT1060 and i.MX RT1170.

1. A lightweight person detection model is provided with an effective network architecture ShuffleNet-V2 [1], which is much faster and memory access cost friendly than most of the previous networks available on Arm platforms.
2. The given model is converted into object files through eIQ Glow tool to get increased performance and a smaller memory footprint for the Arm Cortex-M7 core on the i.MX RT1060 and i.MX RT1170. Experimental analysis is further given to demonstrate the quantization accuracy, memory usage as well as the latency on the target platform under different quantization options.
3. A *microcontroller-based vision intelligence algorithm* (uVITA) application pipeline is proposed to enable the multiple person detection solution with different microcontroller platforms. Therefore, the camera can capture the frame in real time. Meanwhile, the display shows the frame simultaneously, be the speed of the vision algorithm fast or slow on different platforms.

The contributions of this application software pack are summarized as below:

- It provides a lightweight person detection model with a highly efficient and memory access cost friendly neural network.
- The detailed steps and experimental analysis are given to demonstrate how to convert an object detection model with eIQ Glow into object files on a microcontroller.
- A microcontroller-based vision intelligence algorithm application pipeline is proposed to build the multiple person detection projects on the i.MX RT1060evk and i.MX RT1170evk.

Table 1. Glossary

Glossary	Description
ML	Machine Learning
CNN	Convolutional Neutral Network
MAC	Memory Access Cost
RAM	Random Access Memory
NMS	Non-Maximum Suppression

2 Multiple person detection neural network

Multiple-person detection plays an important role in various applications, such as, robots and security. Study shows that the deep *Convolutional Neutral Networks* (CNNs) usually have higher accuracy in these object detection tasks. Therefore, lots of CNN-based methods, including Yolo [2], ResNet [3], SSD [4], and so on, are proposed to improve the performance of object detection. Apart from the detection accuracy, computation complexity is another important factor especially for the applications on edge devices. Therefore, many lightweight CNNs like Xception, MobileNet [5], and ShuffleNet [6] are given to achieve better speed-accuracy trade-off. Among these, ShuffleNet-V2 presents a better characteristic of light weight and high accuracy [1]. Moreover, it performs lower *Memory Access Cost* (MAC) with validations on the Arm platform. As a result, the ShuffleNet-V2 architecture is applied to train the multiple person detection in our application.

2.1 Neural network with ShuffleNet-V2

To derive a lightweight ML model of a person detector, we trained a high-efficient neural network, in which the ShuffleNet-V2 architecture is applied to achieve a speed-accuracy tradeoff. ShuffleNet is a state-of-the-art network architecture, widely adopted in low-end devices such as mobiles [1].

Figure 1 illustrates the building blocks in the trained model with ShuffleNet-V2. Among these, **Block-1** and **Block-2** contribute to the main structure of the neural network. **Block-1** and **Block-2** are used to maintain many channels with neither dense convolution nor too many groups [1]. In this way, it helps reduce the MAC. Specifically, the **Block-1** helps to narrow down the feature map size and only keep the useful information. Meanwhile, A **channel shuffle** operation is then introduced to enable information communication between different groups of channels and improve accuracy [1]. **Block-2** introduces a simple operator called *channel split* to split the features into two branches. One branch remains as an identity while the other branch tries to explore more information.

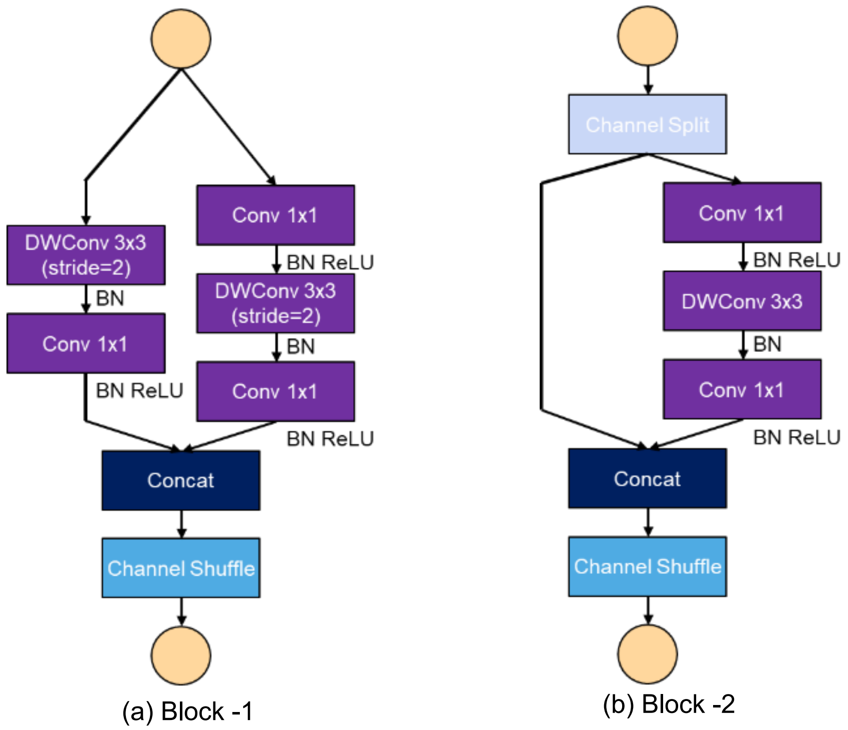

Figure 1. Main structure of the person detector model with ShuffleNet-V2

The extracted features are then sent into an Inception structure with 5 × 5 parallel convolutions, as shown in **Block-3** in Figure 2. It is expected to integrate the features of different perception fields, so that a single detection head adapts to the object detection with different scales. Finally, an anchor-free detection head with three branches is used as shown in **Block-4** in Figure 2, in which the first branch with a *sigmoid* activation layer is responsible for the detection confidence. The output of the second branch provides coordination of the detected objects. Meanwhile, the last branch with a *softmax* activation layer is in charge of detection categories. In this application, there is only one object, the human body, so the last branch actually does not work.

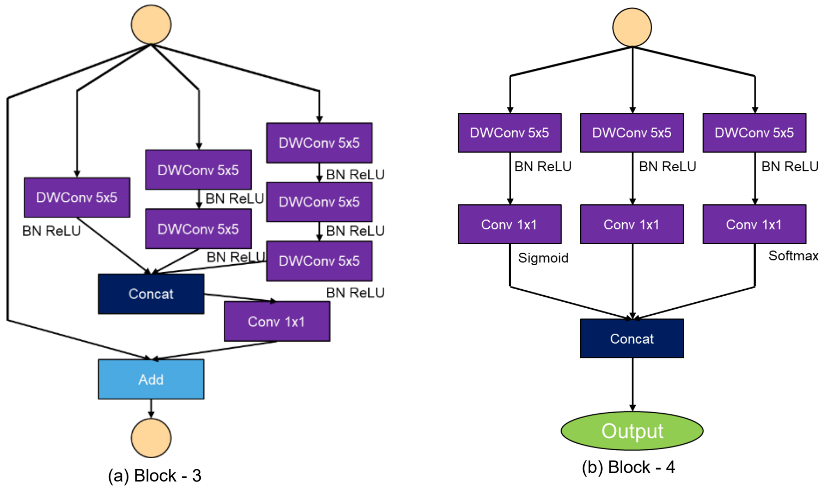

Figure 2. Building blocks of the person detector model with ShuffleNet-V2

The building blocks are repeatedly stacked to construct the whole multiple person detector. Table 2 summarizes the overall network structure. Note that the height and width of the input in the proposed person detector are set to 192 and 320 respectively, maintaining the height to width ratio of around 9:16. This is because both the height to width ratio of the cameras on RT1170EVK or RT1060EVK are around 9:16. Therefore, there would be no distortion in the matching between the camera and input of the person detector.

Table 2. Overall architecture of person detector model with layer information

Index	Layer	Output size	Kernel size	Stride	Repeat	Output channels
0	Image	192 × 320	—	—	—	3
1	Conv1	96 × 160	3 × 3	2	1	24
	MaxPool	48 × 80	3 × 3	2		
2	Block-1	24 × 40	3 × 3 and 1 × 1	2	1	48
	Block-2	24 × 40	3 × 3 and 1 × 1	1	3	48
3	Block-1	12 × 20	3 × 3 and 1 × 1	2	1	96
	Block-2	12 × 20	3 × 3 and 1 × 1	1	7	96
4	Block-1	6 × 10	3 × 3 and 1 × 1	2	1	192
	Block-1	6 × 10	3 × 3 and 1 × 1	2	1	192
5	Concat	12 × 20	—	—	—	336
	Conv2	12 × 20	1 × 1	1	1	96
6	Block-3	12 × 20	5 × 5 and 1 × 1	1	1	96
7	Block-4	12 × 20	5 × 5 and 1 × 1	1	1	6

The output size of the feature map is 12 × 20 in the given person detector network, maintaining a down-sampling ratio of 16 for the input resolution (192 × 320) of the network. Besides, there are six channels in the final output of the given network. Among these, the first channel and last channel respectively provide the confidence and category of the object. The confidence and category information are located in the corresponding grid as shown in Figure 3. The other four channels respectively correspond to the X-coordinate and Y-coordinate of the center location, as well as the width and height of the objects. Then, the candidate boxes corresponding to the interested objects are extracted, as shown in Figure 3, through the information of the six output channels. Finally, the detection results are derived by filtering the candidate boxes with a Non-Maximum Suppression (NMS) strategy.

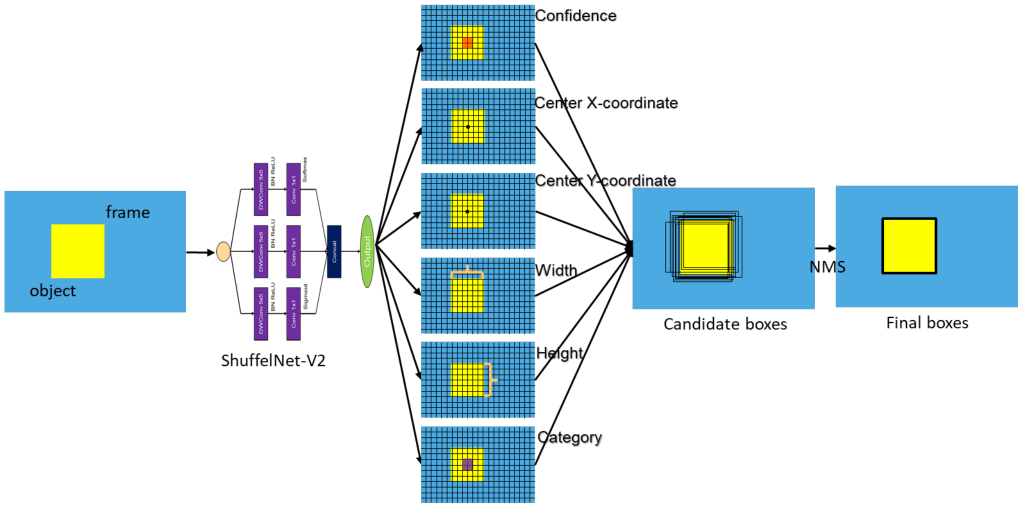

Figure 3. ML person detector algorithm pipeline with ShuffleNet-V2

2.2 Pre-process and post-process of the neural network

The given neural network was trained with COCO and PASCAL-VOC, which are two popular data sets for multiple-object detection. There are many kinds of objects in those data sets. However, the only thing we need is the category of *person* to train a person detector in this application. Therefore, a pre-process was given to prepare the labels related to the person, while all the other objects are treated as backgrounds.

To evaluate the performance of the trained model, a static image test and a dynamic video test are provided in the *Scripts* folder. There are three key points that users should pay attention to before testing a model or deploying the model onto a real edge device. The first one is the pre-process of the image data before sending it to a model. In the proposed person detector, the pre-process of the image is:

$$Input = Im / 255 \tag{1}$$

In other words, the image must be normalized between 0 and 1 before sending it to the given model. Another key point is the post-process for the output of the model. Since the given person detector extracts the candidate boxes of the object in an anchor-free way, the post-process is slightly easier than the traditional Yolo method [2]. The final mixed confidence in each output grid as shown in Figure 3 is computed as:

$$mc_{i,j} = \omega \times confidence_{i,j} + (1 - \omega) \times category_{i,j} \tag{2}$$

In Equation 2, $confidence_{i,j}$ denotes the value of the i-th row and the j-th column of the first channel in Figure 3. $category_{i,j}$ is the value of the i-th row and the j-th column of the last channel in Figure 3. $\omega < 1$ denotes the weight of the $confidence$ channel. With a given threshold to the final mixed confidence $mc_{i,j}$ in each grid, the candidate boxes of the interested object can be filtered. Then, the corresponded center coordinates are calculated as below:

$$cx_{i,j} = i + \left(\frac{2}{\left(1+\exp\left(-2\times x_{offset_{i,j}}\right)\right)} - 1 \right) \Big/ output\_w \tag{3}$$

$$cy_{i,j} = j + \left(\frac{2}{\left(1+\exp\left(-2\times y\_offset_{i,j}\right)\right)} - 1 \right) \Big/ output\_h \tag{4}$$

Meanwhile, the height and width of the interested object activated at (i, j) are given as:

$$h_{i,j} = \mathrm{sigmoid}\left(sh_{i,j}\right) \tag{5}$$

$$w_{i,j} = \mathrm{sigmoid}\left(sw_{i,j}\right) \tag{6}$$

2.3 Algorithm performance

With the given person detection model and pre/post-process, the algorithm results can be derived with some static image examples as shown in Figure 4. It illustrates the person detection results with the final confidence and coordinates of the person in each frame. These results show robust and reliable predictions for person detection in different environments. Besides, there is a video test script in this application to let users verify the performance of the given person detector with firsthand experience.

Figure 4. Algorithm performance of the given person detector

3 eIQ inference with Glow NN

To deploy a neural network into the i.MX RT crossover MCUs, the NXP eIQ ML software development environment provides friendly and efficient tools, such as Glow, TensorFlow Lite Micro, or DeepViewRT. In this application, it enables the ahead-of-time compilation with Glow to convert the original neural network into object files and further deploy the model on the MCUs.

3.1 Quantization and compilation with Glow NN

Glow enables the inference of the neural network model on the edge devices. To compile the given model with Glow, usually two phases convert the model into object files. In the first phase, the Glow optimizer performs quantization analysis with given calibration data and the model. To help users reproduce the quantized object files, this application provides the float model in `onnx` format and the calibration data with 132 images at resolution 192 × 320. These images are generated from a WIDER FACE dataset. Users can generate the *yml* file first with the command below:

```
image-classifier.exe -input-image-dir=Data/Calibration -image-mode=0to1 -image-
layout=NCHW -image-channel-order=BGR -model=Model/Onnx/dperson_shufflenetv2.onnx
 -model-input-name=input.1 -dump-profile=Model/Glow/dperson_shufflenetv2.yml
```

For more helpful information about the Glow operation, see the *eIQ Glow Ahead of Time User Guide* EIQGLOWAOTUG). Once the *yml* file is derived, the second phase can be applied to perform optimizations that take advantage of specialized back-end hardware features. In this application, the target platform is the Arm Cortex-M7 core. Therefore, the binary object file (bundle) can be generated by compiling a float32 model into an int8 bundle for getting Cortex-M7 support with less memory consumption and faster inference speed. To do this, the below command helps generate the 8-bit bundle.

```
model-compiler.exe -model=Model/Onnx/dperson_shufflenetv2.onnx -model-
input=input.1,float,[1,3,192,320] -emit-bundle=Model/Glow/int8_bundle -
backend=CPU -target=arm -mcpu=cortex-m7 -float-abi=hard -load-profile=Model/
Glow/dperson_shufflenetv2.yml -quantization-schema=symmetric_with_power2_scale -
quantization-precision-bias=Int8
```

Another bundle compiling option is to accelerate performance by utilizing Arm CMSIS-NN library, with the command below.

```
model-compiler.exe -model=Model/Onnx/dperson_shufflenetv2.onnx -model-
input=input.1,float,[1,3,192,320] -emit-bundle=Model/Glow/int8_cmsis_bundle -
backend=CPU -target=arm -mcpu=cortex-m7 -float-abi=hard -load-profile=Model/
Glow/dperson_shufflenetv2.yml -quantization-schema=symmetric_with_power2_scale -
quantization-precision-bias=Int8 -use-cmsis
```

Then, the glow bundle is derived from the output of the Glow compiler. Four files are generated into the directory specified by the -emit-bundle. In this application, the four files are respectively given as:

- **dperson_shufflenetv2.h** - the bundle header file (API).
- **dperson_shufflenetv2.o** - the bundle object file (code).
- **dperson_shufflenetv2.weights.bin** - the model weights in binary format.
- **dperson_shufflenetv2.weights** - the model weights in text format as C text array.

The **dperson_shufflenetv2.h** file contains the memory usage and the inference function API. The **dperson_shufflenetv2.o** file is the object file that contains the compiled model code in the form of a library. Generally, the size of the object file is larger than the flash size required by itself.

3.2 Memory footprint and latency analysis

In this application, the given person detector shows lightweight characteristics in the required memory and the latency of the model, as shown in Table 3. As is known, Glow does not allocate memory dynamically. Therefore, the required memory size of the quantized model generated by Glow is provided in the bundle header file. This information is further summarized in Table 3.

It can be found that the constant weights of the given person detector occupy 235,904 bytes and 246,848 bytes generated by Glow without and with CMSIS-NN respectively. During inferencing, the weights can be read from either Flash or from RAM, while the weights take up the specified amount of Flash. Another Flash usage is caused by the generated object code in the format of a library, which requires 76,192 bytes and 25,840 bytes respectively without and with CMSIS-NN.

The amount of memory required for both the input and output data buffers is 743,040 bytes, which must be allocated from RAM. The input/output buffer is related to the input resolution and output feature map size of the given model. For example, the given model of the person detector has an input resolution of 192 × 320 × 3 and an output shape of 12 × 20 × 6. The total buffer size is:

$$192 \times 320 \times 3 \times 4 + 12 \times 20 \times 6 \times 4 = 743040 \text{ bytes} \tag{7}$$

An activation buffer, viewed as the scratch memory required for intermediate computations, must be allocated from RAM. For the given model, the activation buffer size is 552,960 bytes and 645,120 bytes respectively without and with CMSIS-NN.

Table 3. Memory Footprint and Latency of Person Detector Model

Glow compile options	Weights (Flash)	Input/Output (SDRAM)	Activations (SDRAM)	Library (Flash)	Latency
8-bit No CMSIS-NN	235,904	743,040	552,960	76,912	778 ms (RT1060)
					495 ms (RT1170)
8-bit With CMSIS-NN	246,848	743,040	645,120	25,840	353 ms (RT1060)
					237 ms (RT1170)

The activations buffer allocated in different RAM regions could have different performance implications in terms of latency. For instance, allocating the activations buffer in On-Chip (OC) RAM reduces the latency from 230 ms to 161 ms on RT1170 as shown in Figure 5. It is understandable that the model inference computation in OCRAM has higher efficiency than that in SDRAM. More importantly, allocating the activations buffer in OCRAM can help avoid the memory access conflict between CPU and other resources like DMA and PXP. This point will be discussed in the next section.

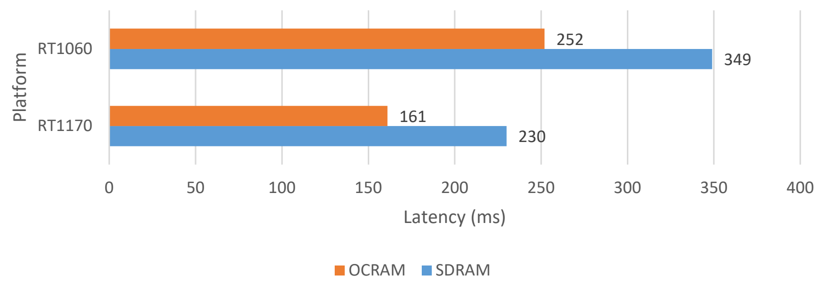

Figure 5. Model inference latency with activations buffer allocated in different RAM regions

3.3 Quantization precision verification

Before deploying a quantized model on the edge devices, the quantization precision should be verified first. Figure 6 shows the prediction results respectively given by the original float model and two different versions quantized with Glow. It can be found that the quantization results by Glow with or without CMSIS-NN are relatively consistent with that given by the float model, especially for the samples with simple backgrounds and non-overlapped persons. Since the model in the format of 8-bit has a certain loss of information, there is no wonder about the existence of the mismatch cases as represented by the red and green outlines in Figure 6 (c). Nevertheless, the overall performance of the quantized model is more reliable compared with the original float model, as shown in Figure 6 (a), (b), and (d).

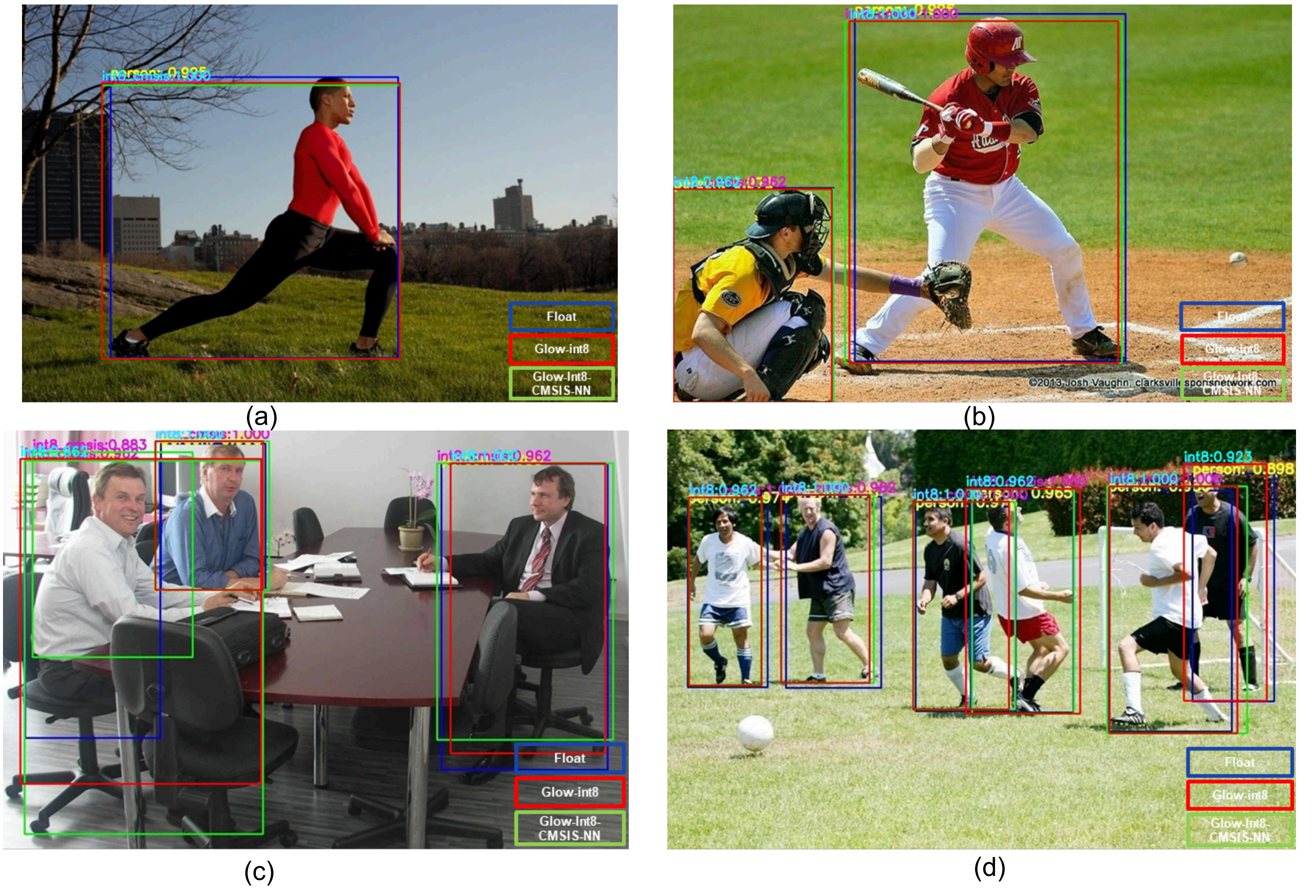

Figure 6. Quantization performance of the person detection model

4 Person detector in application

In this section, additional guidance and explanations are provided for introducing the ML person detector integrating on real edge devices. This application software pack has another *Getting Started* document to help users easily reproduce the example application.

4.1 System design

The cross-over MCUs of NXP provide high-performance intelligent capabilities with abundant hardware resources. For instance, the processor of RT1060EVK and RT1170EVK have the Arm Cortex-M7 core respectively up to 600 MHz and 1 GHz, as shown in Table 4. In addition, the given platforms provide abundant memory for vision applications. Furthermore, the generic 2D hardware acceleration (PXP) is embedded in the RT1060 and RT1170 to help achieve common image-processing functions fast and save CPU bandwidth. The supported 2D process includes image rotation, image scaling, color space conversion, and so on. As shown in Table 4, the camera used on RT1060EVK and RT1170EVK is MT9M114 and OV5640 respectively. In this application, the resolution of the camera on RT1060EVK is set as 480*272, keeping the same resolution as its display. Similar, both the resolution of the camera and display on RT1170EVK are set as 1280*720.

Table 4. Hardware resources for ML vision applications on NXP cross-over MCUs

	RT1060EVK	RT1170EVK
Processor	MIMXRT1062DVL6A 600 MHz Arm Cortex-M7 core	MIMXRT1176DVMAA 1 GHz Arm Cortex-M7 core

Table 4. Hardware resources for ML vision applications on NXP cross-over MCUs*...continued*

	RT1060EVK	RT1170EVK
		400 MHz Arm Cortex-M4 core
Memory	• 1 MB on-chip RAM • 256 MB SDRAM memory • 512 MB Hyper Flash • 64 MB QSPI Flash	• 2 MB on-chip RAM • 512 Mbit SDRAM memory • 512 Mbit Octal Flash • 128 Mbit QSPI Flash
Camera	MT9M114 or OV7725	OV5640
Display	TFT: RK043FN02H-CT Resolution: 480*272	TFT: RK055HDMIPI4M Resolution: 1280*720
Generic 2D (PXP)	• Image rotation (90°, 180°, 270°) • Image scaling • Color space conversion • ...	• Image rotation (90°, 180°, 270°) • Image scaling • Color space conversion • ...

To make the ML-based person detector easy to deploy on different development boards, we propose a cross-platform microcontroller-based Vision Intelligence Algorithms (uVITA) system to manage tasks of the camera, display as well as the algorithm. Besides, the uVITA system tries to get a better user experience in terms of ML vision applications. For example, the camera should capture the frame in real time. Meanwhile, the display should show it simultaneously, regardless if the speed of the algorithm is fast (on RT1170) or slow (on RT1060). The proposed system architecture is shown in Figure 7, in which the camera task is responsible for capturing the image frame and sending it to the algorithm task and display task with the corresponding required image format and size. Meanwhile, the algorithm task is to infer the ML model with fed data. Then, it extracts the results from the model and filters the predictions with proposed post-processing functions. Finally, the display task is responsible for showing the image frame and algorithm results on the display. Since the three tasks shown in Figure 7 run in parallel under the management of FreeRTOS, the camera and display handle their process in real time if their priority is higher than the algorithm task.

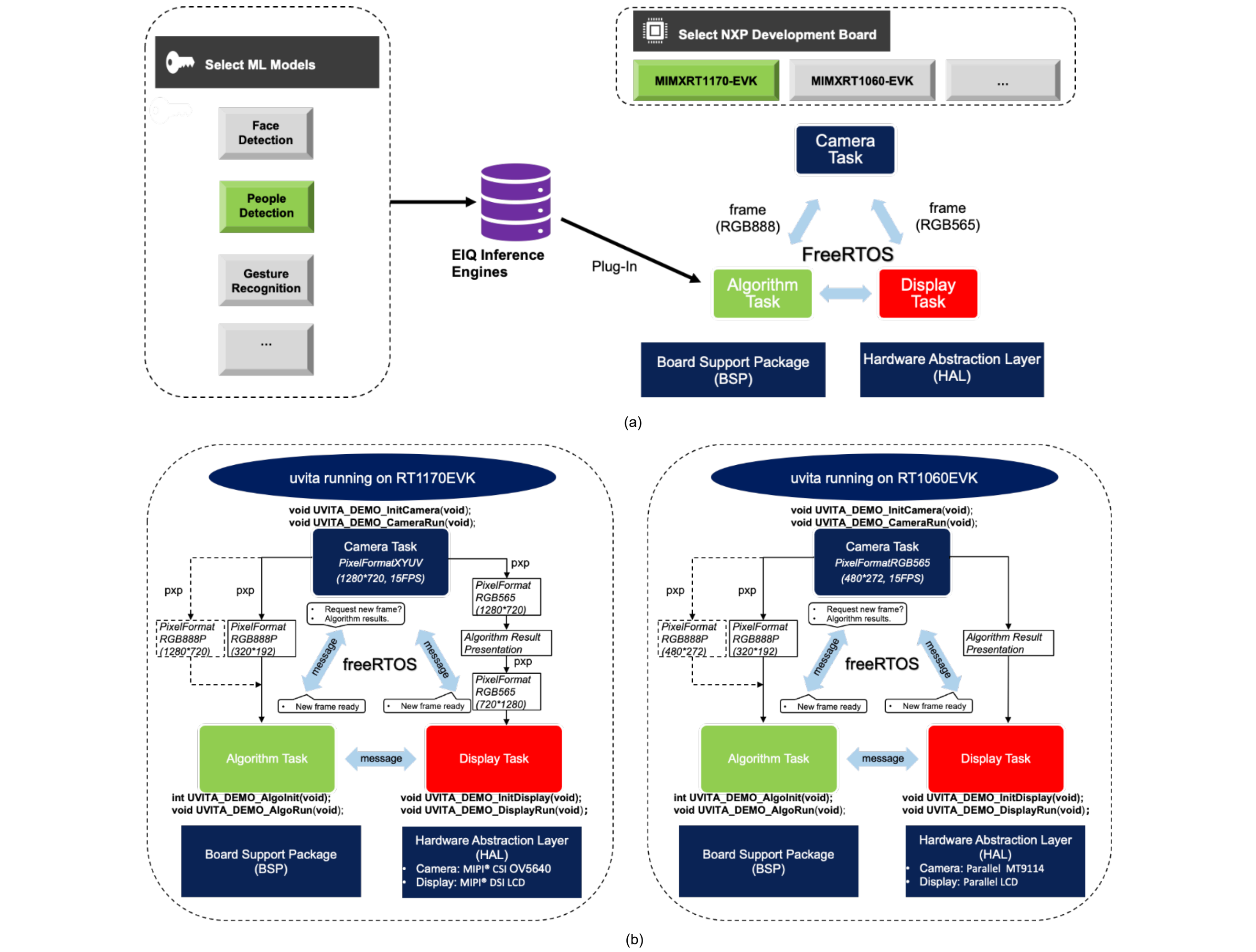

Figure 7. Microcontroller-based Vision Intelligence Algorithms (uVITA) system

The conversion of the image scale and image format is realized by the PXP accelerated functions supported on the cross-over MCUs of NXP. According to the requirements of the input of the person detector, the image frame received from the camera is directly converted to the RGB888 format at a resolution of 320*192 by the PXP function. Besides, the image frame captured by the camera is converted to the frame shown in the display at 15PFS in the format of RGB565 by the PXP function. Therefore, the CPU resources are saved as much as possible so that it can infer the neural network of the person detector with more bandwidth. It needs a 90° rotation before showing the frame on the display of RT1170EVK since the display panel is in vertical mode.

4.2 Overall performance

In this application, the memory requirement impact of the person detector in the MCUXpresso IDE project is discussed first. As shown in Table 5 with the application project on RT1060, all buffers have been set to zero to start with. Then, the camera, display, and FreeRTOS support are added based on the SDK project. The goal is to focus on the memory requirement impact of the application project. Users can determine if a particular ML model could fit on a particular board. In addition to the required memory listed in Table 5, an extra 1020 K-bytes memory is required for bearing the data of capturing the frame by a camera and showing it on the display. Specifically, the camera resolution is set as 272*480 in the format of RGB565, so its data buffer occupies 2*272*480*2 bytes. Besides, the display resolution and format are set as 272*480 and RGB565 respectively, so its data buffer occupies 2*272*480*2 bytes. Therefore, all the above buffers are given in the SDRAM with a total of 1020 kB.

Table 5. MCUXpresso SDK compiled project size for ML person detector on RT1060EVK

Description	Flash (bytes)	RAM (bytes)	Change (bytes)	Details
Bare-Bones	109,144	26,372	Baseline	Baseline SDK project with camera, display, and FreeRTOS Support.
Adding memory for static input image in the algorithm task	109,144	211,292	+184,320 RAM	The frame buffer of algorithm task is in format of RGB888 with resolution 192*400, so it occupies 192*320*3 = 184320 bytes.
Adding in *.o* library	134,984	211,292	+25,840 Flash	The Glow compiled person detection library *.o* file. ***Note:*** *This is less than the size on the .o file on the PC hard drive.*
Adding input/ output and activations buffers	134,984	1,599,604	+1,388,312 RAM	Statically allocate memory for mutable weights (model input/output data, 743,040 bytes) and activations (model intermediate results, 645,120 bytes).
Adding weights in Flash	381,832	1,599,604	+246,848 Flash	If weights are read from Flash, this does not affect RAM usage but requires 246,848 bytes of nonvolatile memory.
Adding weights in RAM	381,832	1,846,452	+246,848 RAM	If weights are read from RAM, the project requires 246,848 additional bytes of RAM. This is optional but may decrease inference time.

The similar memory requirement impact of the person detector can be found on the RT1170EVK, where the main difference exists in the size of the extra buffer for bearing the data of the camera frame and display frame whose resolution is higher than that on RT1060EVK. For the RT1170EVK, both the resolution of the camera and display is 1280*720 in the format of YUYV so that its data buffer occupies 2*720*1280*4 bytes. Meanwhile, the display resolution and format are set as 720*1280 and RGB565 respectively, so its data buffer occupies 2*720*1280*2 bytes. Besides, there is an extra buffer for saving a single frame to show the algorithm results before sending it to display, and it needs 720*1280*2 bytes. Therefore, all the above buffering is handled in the SDRAM with a total of 12600 kB.

Another aspect to be addressed is the latency impact of the person detector in real edge applications. The CPU resources and memory access bandwidth are occupied by the multi-task system, so they may not fully serve the model inference task. For instance, when the activation buffer of the compiled model by Glow with CMSIS-NN optimization is allocated in the SDRAM, the ideal latency of the compiled model is 230 ms on the RT1170EVK. However, when the camera task and display task are running at the same time, the latency of compiled model in the algorithm task increases to 280 ms. The key reason exists at the memory access bandwidth limitation between CPU and other hardware accelerators like PXP and DMA. Therefore, a more ideal memory configuration is to make the activation buffer of the compiled model allocated in the OCRAM, meanwhile, put the data buffer of the camera and display in the SDRAM. In this way, the memory access conflict can be avoided. As shown in Table 5, the latency impact can be reduced when the activation buffer is allocated in OCRAM.

Table 6. Latency impact of the person detector on RT1170EVK

Model	Weights (246,848)	Activations (645,120)	Latency (ideal)	Latency (real application)
Shufflenetv2 EIQ-Glow 8-bit with CMSIS-NN	Flash	SDRAM	230 ms	281 ms
	Flash	OCRAM	161 ms	165 ms

5 Conclusion

In this application, the multiple person detector is proposed on the cross-over MCUs of NXP, i.MX RT1060 and RT1170. The given person detector is first achieved with a high-efficient neural network based on ShuffleNet-V2 architecture with a speed-accuracy tradeoff. The quantization and compilation procedures by eIQ Glow are then introduced for the trained person detector so that the corresponding executable codes on the MCUs can be obtained. Meanwhile, the memory usage, latency, and quantization precision of the converted model are analyzed. Finally, the proposed uVITA system is demonstrated for building a person detector on the RT1060EVK and RT1170EVK respectively. Therefore, the camera can capture the frame in real time. Meanwhile, the display shows it simultaneously, regardless whether the speed of the algorithm is fast or slow. This application serves as a prototype from which users can build their own ML vision programs with the cross-over MCUs of NXP. With their own developed ML models in hand, customers can build intelligent products similar to this application based on the eIQ ML software development environment.

6 Reference

1. Ma N, Zhang X, Zheng H T, et al. Shufflenet v2: Practical guidelines for efficient cnn architecture design[C]//Proceedings of the European conference on computer vision (ECCV). 2018: 116-131.
2. Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.
3. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European Conference on Computer Vision. pp. 630–645. Springer (2016).
4. Liu W, Anguelov D, Erhan D, et al. Ssd: Single shot multibox detector[C]//European conference on computer vision. Springer, Cham, 2016: 21-37.
5. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017).
6. Zhang, X., Zhou, X., Lin, M., Sun, J.: Shufflenet: An extremely efficient convolutional neural network for mobile devices. arXiv preprint arXiv:1707.01083 (2017).

7 Note About the Source Code in the Document

ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

8 Revision history

Table 7 summarizes the revisions to this document.

Table 7. Revision history

Revision number	Date	Substantive changes
0	08 May 2023	Initial release

9 Legal information

9.1 Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

9.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this data sheet expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. - NXP B.V. is not an operating company and it does not distribute or sell products.

9.3 Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, μVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

eIQ — is a trademark of NXP B.V.

i.MX — is a trademark of NXP B.V.

Contents

Date of release: 8 May 2023
Document identifier: AN13924