Performance Evaluation of Embedded Image Classification Models Using Edge Impulse for Application on Medical Images

Maha S. Diab and Esther Rodriguez-Villegas

***Abstract*— This work explores the possibility of applying edge machine learning technology in the context of portable medical image diagnostic systems. This was done by evaluating the performance of two machine learning (ML) algorithms, that are widely used on medical images, embedding them into a resource-constraint Nordic nrf52840 microcontroller. The first model was based on transfer learning of the MobileNet V1 architecture. The second was based on a convolutional neural network (CNN) with three layers. The Edge Impulse platform was used for training and deploying the embedded machine learning algorithms. The models were deployed as a C++ library for both, a 32-bit floating point representation and an 8-bit fixed integer representation. The inference on the microcontroller was evaluated under four different cases each, using the Edge Impulse EON compiler in one case, and the Tensor Flow Lite (TFLite) interpreter in the second. Results reported include the memory footprint (RAM, and Flash), classification accuracy, time for inference, and power consumption.**

I. INTRODUCTION

Medical imaging plays an important role in diagnosis and treatment of various medical diseases [1]. Technologies used in medical imaging include, but are not limited to, X-rays, CT scans, MRIs, ultrasonography, and mammography [1].

Unfortunately, though, access to advanced medical systems is difficult in remote areas and/or areas with limited resources. In such circumstances, healthcare centers, and outreach field interventions could significantly benefit from portable, but also affordable imaging solutions. AI-based computer aided diagnostic (CAD) software added to the imaging systems could also further improve not just their diagnostic ability but also the diagnostic efficacy in such scenarios [2]. For example, CAD based systems could be massively beneficial in remote areas with limited availability of radiologists [3], mostly if the systems are cheap and do not require an Internet connection.

In the design of a portable medical image diagnostic system, CAD algorithms could, in principle, be either run in the Cloud or be locally embedded on the system. Machine learning (ML) models are, however, computationally expensive and typically require a connection to remote servers to run.

This work explores the feasibility of implementing ML based classification on a low power microcontroller, since this would be a first step towards having a low-cost portable CAD based imaging system suitable for deployment in low resourced and/or remote settings. Running algorithms online on a dedicated microcontroller would also offer other advantages such as privacy being better preserved and decreased latency.

The methodology used to train and deploy the image classification models is presented in section II. The architecture and performance of the two classification models developed, based on transfer learning and convolutional neural network, are covered in section III. In section IV, the results corresponding to the deployment of the two models are presented. Analysis of different performance trade-offs are discussed in section V.

II. MATERIALS AND METHODS

Embedding machine learning algorithms on resource- constrained microcontrollers has been made possible using some recently developed software tools and frameworks. In this work, one particular platform, Edge Impulse [4] is used.

A. Edge Impulse Studio

Edge Impulse provides a single platform for developers to collect or import data, visualize it, process it in preparation for training and testing of the machine learning model, and for final deployment onto the embedded system of choice. Edge Impulse uses TensorFlow and Keras for training neural network models, and sklearn for the classic machine learning ones. The studio provides the developer with two options for designing the model architecture: (1) a simple interface for non-experts is provided through a visual layered architecture; and (2) an expert view for developers needing further optimization of their architecture in Keras mode.

Moreover, Edge Impulse presents its own Edge Optimized Neural (EON) compiler, which is reported to save 25-55% in RAM and up to 35% in Flash, in comparison to TFLite for microcontroller; with reported no effect on model accuracy and time per inference [5]. This is possible because the EON compiler builds the model into the source code, as opposed to TFLite for microcontroller which requires an interpreter to run the model. However, since the model is built into the firmware, it is difficult to update the model in the field.

In this work, two image classification models, a transfer learning model and a convolutional neural network (CNN) model were trained and deployed on a Nordic nrf52840 microcontroller. The performance of both models was evaluated and compared for the EON compiler and the TFLite deployment options, using both quantized and floating point representations.

The work of M. S. Diab is supported by Imperial College London through the Department of Electrical and Electronic Engineering PhD Scholarship.

M. S. Diab and E. Rodriguez-Villegas are with the Wearable Technologies Lab, Department of Electrical and Electronic Engineering, Imperial College London, SW7 2BT, United Kingdom. Email: {*m.diab21, e.rodriguez*}*@imperial.ac.uk*

B. Dataset

The publicly available dataset, CIFAR-10 [6] for image classification was used for training and testing of the two image classification models. The dataset provides 10 classes of 32x32 colored images with 50,000 training images (5000/class), and 10,000 testing images (1000/class).

Using the free subscription for developers in Edge Impulse, there was a limitation of 20 mins for every single job. Hence, only part of the CIFAR-10 dataset was imported to the studio and used in building both classification models. The imported dataset consisted of 3 classes [bird, frog, horse] out of the original 10 classes. For each class, 500 images from the original training set were imported, and 100 images from the testing set. Twenty percent of the training data was left out for validation.

III. CLASSIFICATION MODELS

A. MobileNet Model

MobileNet has been used for classification of medical images in various applications such as breast cancer detection [7], and diabetic retinopathy detection [8]. Among the pre-trained models used in transfer learning, MobileNet was chosen due to its relatively smaller size, hence enabling the possibility of embedding it on the edge device. The MobileNet- V1 0.25 model in Edge Impulse was used as the base model, with input images resized to 96x96 for optimum performance. An additional dense layer (16 neurons), and dropout (0.1) was added before the final classification layer (3 neurons). The model was trained for 20 epochs with a learning rate of 0.0005. The classification accuracy achieved was 84.7% on the validation set, and 83.0% on testing set. The resulted confusion matrix for classifying the 300 testing images is given in Table I.

TABLE I
CONFUSION MATRIX FOR CLASSIFICATION OF THE TESTING SET USING MOBILENET-V1 MODEL.

	Bird	Frog	Horse	Uncertainty
Bird	68%	8%	4%	20%
Frog	2%	92%	1%	5%
Horse	0%	3%	89%	8%
F1 Score	0.80	0.91	0.92	

B. Convolutional Neural Network Model

Convolutional neural networks have also been used in the analysis of medical images for various medical problems, such as diagnosis of breast cancer, and classification of Alzheimer's disease [9]. A simple convolutional neural network model was designed for comparison against the deeper architecture of the MobileNet model. The model consisted of three 2D-Convolutional layers, with input image size of 32x32. The model convolutional layers had filter sizes (16, 32, 64) with kernel size 3, followed by a flatten layer, dropout (0.25), dense layer (64 neurons), dropout (0.25), and final classification layer (3 neurons). The model was trained for 50 epochs with learning rate of 0.0005. The classification accuracy achieved for validation was 76.7%, and 74.33% on the testing set. The resulted confusion matrix for testing classification is given in Table II.

TABLE II
CONFUSION MATRIX FOR CLASSIFICATION OF THE TESTING SET USING THE CNN MODEL.

	Bird	Frog	Horse	Uncertainty
Bird	59%	11%	13%	17%
Frog	8%	82%	2%	8%
Horse	8%	3%	82%	7%
F1 Score	0.67	0.84	0.83	

It was noticed that in both models, the uncertainty for classification of bird images was high (MobileNet - 20% and CNN - 17%), which negatively affected the overall performance in both cases. This issue could be addressed by providing more training data, allowing both models to further learn specific features of this class. However, due to the constraints associated to the free license for Edge Impulse, and accuracy in classification not being the main objective of this work, these non-optimized models were considered adequate for deployment.

IV. RESULTS

Edge Impulse provides a deployment suite that offers two deployment options: creating a library source code with a language of choice, or a binary file that can be directly built into the board's firmware. The first option was used, to create a C++ library suitable for a Nordic nrf52840 development kit. An ARM Cortex-M4 32-bit based processor, the nrf52840 MCU, was chosen as the target embedded device. This has 1 MB of Flash memory and 256 kB of RAM, operating at 64 MHz with flexible power management for low power operation. The C++ library was created for each model (MobileNet, and CNN), for both 8-bit quantized and 32-bit floating point representation, using either the EON compiler or the TFLite interpreter. This resulted in four different cases per model.

A. MobileNet Model

The deployment suite within Edge Impulse provides an estimate of memory usage per model and an estimated time per inference based on the selected deployment board. This is provided for both, the floating-point model (F) and the optimized (quantized) model (Q) using either EON or TFLite

interpreter. The C++ library for each of the four different cases of the MobileNet model was downloaded and deployed separately for performance evaluation on the board. Using the stand-alone inference option, the raw data for a test image was included into the library for measurement of both, time and power per inference. The inference results on the nrf52840 microcontroller are summarized in Table III. In the case of the floating-point model, for both EON and TFLite, the required RAM exceeded the available one. Hence, none of the models could be tested on board for time and power measurements.

TABLE III
INFERENCE RESULTS OF THE DEPLOYED MOBILENET-V1 CLASSIFICATION MODEL ON THE NRF52840.

	EON(Q)	EON(F)	TFLite (Q)	TFLite (F)
RAM (KB)	131.1	327.9	125.0	389.6
Flash (KB)	311.9	887.3	395.4	926.2
Accuracy (%)	80.33	83.0	80.33	83.0
Time (ms)	738	-	728	-
Power (mW)	12.99	-	12.84	-

B. Convolutional Neural Network Model

A similar setup to the MobileNet deployed model was used for testing the CNN model on board of the nrf52840. Each C++ library of the four different cases was run separately for on-board inference and measurements. Due to the smaller size of this model in comparison to the MobileNet model, the floating point models (EON, and TFLite) were successfully deployed and measurements for time and power were obtained. A complete summary of results is shown in Table IV. Power measurements were obtained by running the development board on a 3V coin-cell battery and measuring the current per inference.

TABLE IV
INFERENCE RESULTS OF THE DEPLOYED CNN CLASSIFICATION MODEL ON THE NRF52840.

	EON(Q)	EON(F)	TFLite (Q)	TFLite (F)
RAM (KB)	26.0	84.0	29.2	101.1
Flash (KB)	122.1	381.1	139.9	396.6
Accuracy (%)	74.33	74.33	74.33	74.33
Time (ms)	223	3726	229	5478
Power (mW)	12.84	13.35	13.05	13.20

V. DISCUSSION

The performance of a system using embedded machine learning, as shown in this work, is affected by different factors. The MobileNet model with its pre-trained weights provided classification with higher accuracy: 8.67% greater than the CNN model for floating point representation, and 6% higher for quantized representation. However, this gap in performance came at the cost of memory footprint and inference time.

In the case of floating point representation, it was not possible to deploy the MobileNet model as it required x1.52 and x1.28 of available RAM for the TFLite and EON models respectively. Moreover, the size of the model was close to the limit to the available Flash memory, requiring 93% and 89% of it for TFLite and EON respectively. The CNN model, though, with its floating point representation still occupied less than 50% of the available Flash and RAM for both EON and TFLite models. Nevertheless, for optimum embedding of machine learning algorithms on microcontrollers, the focus would be centered around the performance of the quantized model. After quantization, the MobileNet model size was compressed by a factor of 2.34 and 2.84 for TFLite and EON respectively. As for the RAM usage, almost half of it was occupied, with noticeable decrease after quantization by a factor of 2.5 for EON and 3.12 for the TFLite model, which allows for on-board inference. As for the CNN model after quantization, the compression further decreased the memory footprint to 10% of RAM and 12% Flash for EON. While for the TFLite model, this was 14% of Flash and 11% of RAM. Furthermore, although the quantization of the CNN model decreased the memory footprint, it did not affect the accuracy of the model (74.33%); as it was the case of MobileNet which decreased it 2.67%. The required memory footprint in terms of ratio for the available Flash and RAM of the eight different cases is visualized in the graphs of Fig. 1 and Fig. 2.

As for the use of EON or TFlite interpreter for model deployment and on-board inference, it is clearly noticed that EON requires less memory footprint than TFLite. However, the drawback of using EON is the difficulty on updating the model, in contrast with TFLite. The use of EON does not affect the accuracy nor inference time as it still runs TFLite in the background of its compiler [5]. For both models, accuracy was the same, while the time showed a difference of a few ms between the EON and TFLite quantized models. In the MobileNet model, there was a difference of 10 ms, with TFLite being faster; whilst the CNN model had a 6 ms difference, with EON being faster. This percentage difference of 1- 2.6% could be related to the model's size. However, there was a noticeable difference in inference time between EON and TFLite for the floating point CNN model: around 1.7 seconds per inference. As for the memory footprint, the MobileNet model saved around 9% of Flash and increased the RAM usage by 2.38%. As for the CNN model, 1.78% of Flash and 1.25% of RAM were saved.

It is worth noting that even with the decrease in memory footprint for the MobileNet model due to the use of EON, its memory footprint is still higher than that of the CNN model. Having the EON quantized model of CNN occupying 18.9% less Flash and 41% less RAM than the quantized EON MobileNet model. In addition, the time required for an inference for the MobileNet-EON is x3.3 more than the CNN-EON. As for the quantized TFLite model, the CNN requires 37.4% less RAM and 25.5% less Flash with almost the same time difference for inference as EON. This comes at the cost of lower accuracy of the CNN compared to the MobileNet model, as mentioned previously.

In terms of power consumption, all of the six deployed cases measured current per inference in the range of 4.28 mA - 4.45 mA, as shown in Fig. 3. The lowest current consumption of 4.28 mA (12.84 mW) was measured for the CNN model with quantized EON deployment and the MobileNet quantized TFlite model. In the case of the MobileNet model, it was only possible to measure the quantized cases. These had an increase in current consumption of 50 µA for the EON deployment. As for the CNN current measurement, the current consumption increased by 70 µA, 50 µA, and 50 µA between EON(Q) - TFLite(Q) - TFlite(F) - EON(F).

In summary, embedding machine learning algorithms onto microcontrollers is bound by different factors related to the performance of the classification model and the microcontroller available resources. In the context of medical imaging, this work has proved it is possible to embed quantized models that could be suitable for a portable medical system. Once the specific application is fully defined, depending on the intended use and the usability requirements accuracy or time will have to be prioritized. If the time of inference is not critical for the specific application, quantized MobileNet might be the best choice in terms of classification performance as long as enough memory is available. Where faster inference is needed, a simpler CNN model would be the better choice as it can process x3 more images than MobileNet in the same time. Further training with a larger dataset could improve the performance while maintaining a relatively lower time per inference and memory footprint.

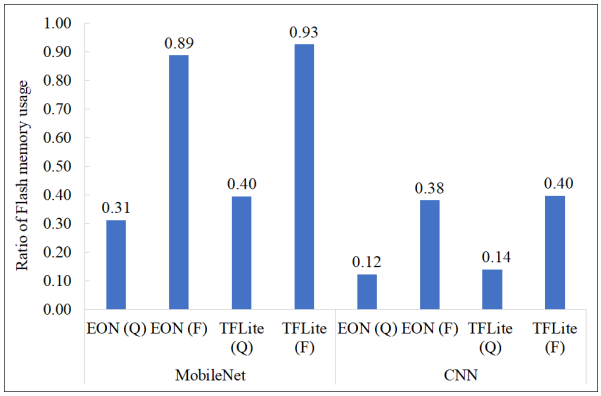


Fig. 1. Ratio of nrf52840 Flash memory usage for MobileNet and CNN models using EON and TFLite interpreter in floating point (F) and quantized (Q) representation.

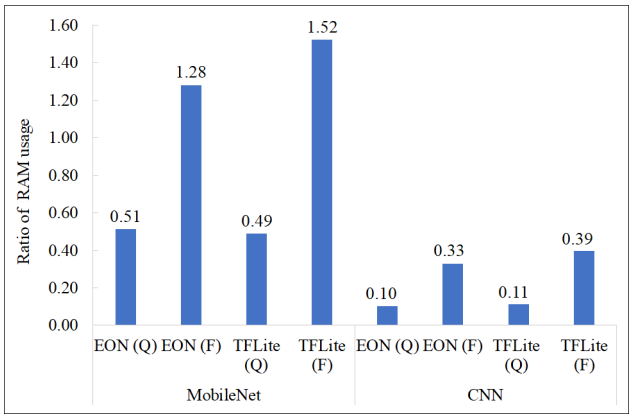


Fig. 2. Ratio of nrf52840 RAM usage for MobileNet and CNN models using EON and TFLite interpreter in floating point (F) and quantized (Q) representation.

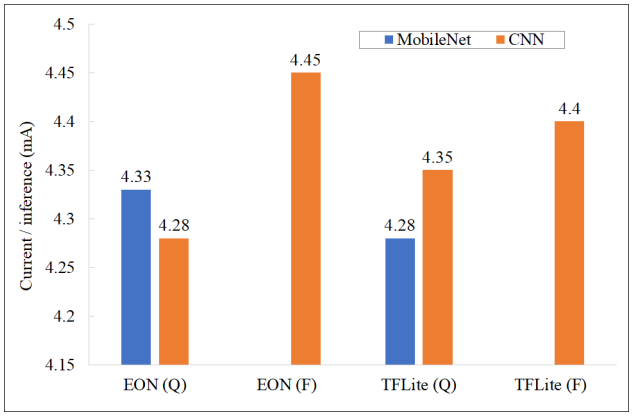


Fig. 3. Measured current consumption per inference on the nrf52840 development board for the six deployed cases.

REFERENCES

[1] World Health Organization, "Strengthening medical imaging," 2022. [Online]. Available: https://www.who.int/activities/strengthening-medical-imaging.

[2] World Health Organization, "Portable digital radiography system: technical specifications," 2021. [Online]. Available: https://www.who.int/publications/i/item/9789240033818.

[3] M. Harris et al., "A systematic review of the diagnostic accuracy of artificial intelligence-based computer programs to analyze chest x-rays for pulmonary tuberculosis," PLOS ONE, vol. 14, no. 9, p. e0221339, 2019. [Online]. Available: 10.1371/journal.pone.0221339

[4] Edge Impulse. https://www.edgeimpulse.com/

[5] J. Jongboom, "Introducing EON: Neural networks in up to 55% less RAM and 35% less ROM," Available Online: https://www.edgeimpulse.com/blog/introducing-eon, Sep 2020.

[6] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, 2009.

[7] W. Ansar, A. R. Shahid, B. Raza, and A. H. Dar, "Breast cancer detection and localization using MobileNet based transfer learning for mammograms," in Intelligent Computing Systems, pp. 11–21, 2020.

[8] S. Patel, "Diabetic retinopathy detection and classification using pre-trained convolutional neural networks," International Journal on Emerging Technologies, vol. 11, no. 3, pp. 1082–1087, 2020.

[9] S. M. Anwar, M. Majid, A. Qayyum, M. Awais, M. Alnowami, and M. K. Khan, "Medical image analysis using convolutional neural networks: a review," Journal of medical systems, vol. 42, no. 11, pp. 1–13, 2018.