Demo Abstract: SRAM Optimized Porting and Execution of Machine Learning Classifiers on MCU-based IoT Devices

Bharath Sudharsan*, Pankesh Patel*, John G. Breslin*, Muhammad Intizar Ali§
*Confirm SFI Research Centre for Smart Manufacturing, Data Science Institute, NUI Galway, Ireland
{bharath.sudharsan,pankesh.patel,john.breslin}@insight-centre.org
§ School of Electronic Engineering, Dublin City University, Ireland, ali.intizar@dcu.ie

ABSTRACT

With the introduction of edge analytics, IoT devices are becoming smarter and ready for AI applications. However, any increase in the training data results in a linear increase in the space complexity of the trained Machine Learning (ML) models, which means they cannot be deployed on IoT devices that have limited memory. To alleviate such memory issues, we recently proposed an SRAM-optimized classifier porting, stitching, and efficient deployment method in [3]. This is currently the most resource-friendly approach that enables large classifiers to be comfortably executed on microcontroller unit (MCU) based IoT devices, and perform ultra-fast classifications (1-4x times faster than state-of-the-art libraries) while consuming 0 bytes of SRAM.

In this demo, realizing our recent SRAM-optimized approach, we port and execute 7 dataset-trained classifiers on 7 popular MCUs, and report their inference performance. It is apparent from the demo results that realizing our approach makes even the slowest Atmega328P MCU perform faster unit inference than a NVIDIA Jetson Nano GPU and Raspberry Pi 4 CPU.

KEYWORDS

Offline Inference, Intelligent MCUs, Edge AI, SRAM Optimization.

1 IMPLEMENTATION

The recent studies like RCE-NN [2], Edge2Train [1] enables users to fit, deploy and execute large-high-quality neural networks (NNs) on MCUs. In contrast to NNs, in this demo, realizing our recent SRAM-optimized approach [3], we execute various datasets trained multi-class classifier models on popular MCUs. We target the MCU platform because they are low-power and low-cost chips that act as a brain for billions of IoT devices like HVAC controllers, smart plugs, smoke detectors, etc. In the following, we port the standard Python scikit-learn GPU trained ML classifier to its MCU executable C version, then stitch and flash it on the MCU of IoT devices.

Porting Classifiers to Plain C. In MCU-based tiny IoT devices, the program space (Flash memory) is always much greater than the available SRAM. So our implementation produces a C version of classifiers that will not depend on the SRAM during execution. We propose to sacrifice Flash memory in favor of the limited SRAM since it is the most scarce resource in the majority of MCUs. For example, in a Decision Tree (DT), our method *hard-codes* the DT splits in C, without storing any reference of the splits and other parameters into variables. Since no variables are allocated, 0 bytes of SRAM will be consumed during the classifier execution.

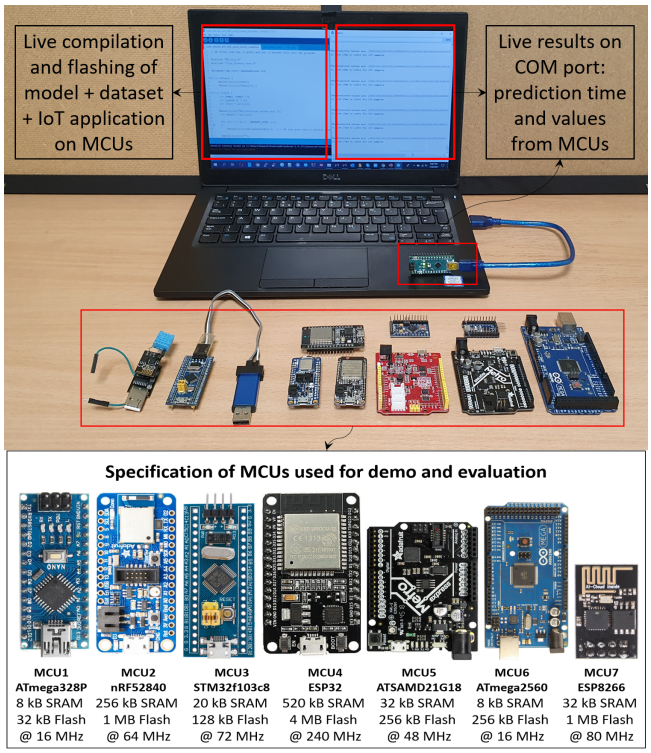

Figure 1: We port and execute various classifiers on popular MCU-boards and show the live inference results.

Executing Classifiers on IoT Devices. Here, we stitch the ported C classifier with the IoT application and perform inference as required. We write the generated C code inside a `.h` model file. During the edge application design phase, the users have to include this `.h` model as a header file at the beginning of their program. Then, this `.h` file needs to be compiled along with the main IoT edge application and flashed on MCUs. For predictions, the *predict* function inside the `.h` model file should be passed with values for which it needs predictions.

2 DEMONSTRATION DESCRIPTION

We show the demo setup in Figure 1. During the demo, by following instructions from Section 1, we perform live porting, stitching, and

ICCPS '21, May 19–21, 2021, Nashville, TN, USA

ACM ISBN 978-1-4503-8353-0/21/05…$15.00
https://doi.org/10.1145/3450267.3451999

Figure 2: Demo results: comparing the time consumed by MCUs 1-7 (using our implementation) and CPUs 1-5 (using Python scikit-learn) when performing inference for datasets of various feature dimensions (3-64) and class counts (1-10).

execution of a classifier model (model shall be chosen by demo viewers) on all the displayed MCUs.

Devices, Datasets and Demo Procedure. We first select 7 datasets (shown in Figure 2) with feature dimensions ranging from 4 to 64 features and class counts from 2 to 10 classes. Using these datasets, we then train DT and RF classifiers using Python scikit-learn (80/20 training/testing split for each dataset) to produce 14 different models. We then ask the demo viewers to select any one model of their choice, which we quickly port to plain C, stitch it with an IoT application and execute it on each of the 7 displayed MCUs. We also show the live prediction results with prediction time for 100 data samples on the COM port of the connected MCU.

We use the same demo setup to perform an extensive evaluation. Here, similar to the MCUs, we selected various CPUs, where CPU1 is a i7-8650U Windows laptop, CPU2 is a NVIDIA Jetson Nano, CPU3 is a i5-825U Windows laptop, CPU4 is a i7-5500 Ubuntu laptop, and CPU5 is a Raspberry Pi 4. We execute the same 7 datasets trained classifiers on CPUs 1-5 and report the inference time in Figure 2, next to the time taken by MCUs 1-7 to infer using same models and data samples. For statistical validation, the plotted time corresponds to the average of 5 runs. In the following, we analyze the results and compare the inference performance of MCUs with CPUs.

Inference on MCUs and CPUs. Here, we perform an onboard test for accuracy, and simultaneously record the time taken by each MCU to perform unit inference and inference for 100 samples for each model and show the results in Figure 2 (y-axis in base-10 log scale), and report the following observations: (i) All the MCUs, irrespective of their specifications, for all datasets, performed unit inference in less than 1 ms; (ii) Even the slowest Atmega328P MCU performed faster unit inference than the Jetson Nano and Raspberry Pi 4; (iii) The ported models, during execution on MCUs, show the same level of accuracy and F1 score as its original models (before porting) when evaluated on high-resource lab setups; (iv) For the largest 64-feature Digits dataset, the 3$ ESP32 (MCU 4) inferred for 100 samples in 7 ms, which is only ≈ 5 ms slower than CPUs 3 & 4, which are 200x times more costly than MCUs.

Features Count vs Inference Time. From Figure 2, it can be observed that using high-feature data as input has only a few ms impact on the unit inference time. Whereas for 100 input samples, the slower MCUs 1,5 & 6 show a logarithmic growth in inference time, the fast MCUs 2, 3, 4 & 7 shows time growth only for the Cancer and Digits dataset. When users deploy their IoT use case models on advanced MCUs or AIoT boards, they will obtain much faster inference results due to the FPU, KPU, and FFT support.

3 SUMMARY

In this demo, we implemented our SRAM-optimized ML classifier porting, stitching, and efficient deployment method [3] and executed multiple classifiers on popular MCUs that have limited memory footprint. When users apply our method to port and deploy any use-case models on their IoT devices/products, similar to the demo results, they will get benefits from the memory-friendly and ultra-fast classifier implementations.

ACKNOWLEDGEMENT

This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) under Grant Number SFI/16/RC/3918 (Confirm) and also by a research grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289_P2 (Insight), with both grants co-funded by the European Regional Development Fund.

REFERENCES

[1] Bharath Sudharsan, John G. Breslin, and Muhammad Intizar Ali. 2020. Edge2Train: A Framework to Train Machine Learning Models (SVMs) on Resource-Constrained IoT Edge Devices. In *10th International Conference on the Internet of Things.*

[2] Bharath Sudharsan, John G Breslin, and Muhammad Intizar Ali. 2020. RCE-NN: a five-stage pipeline to execute neural networks (cnns) on resource-constrained iot edge devices. In *10th International Conference on the Internet of Things.*

[3] Bharath Sudharsan, Pankesh Patel, John G. Breslin, and Muhammad Intizar Ali. 2021. Ultra-fast Machine Learning Classifier Execution on IoT Devices without SRAM Consumption. In *2021 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops).*