Neural networks on microcontrollers with TensorFlow & the Pelion IoT platform

Yue Zhao*, Austin Blackstone*, Neil Tan, Vikas Kache, Damon Civin
Arm, San Jose, CA

Neural networks (NNs) have demonstrated state of the art performance in a variety of complex tasks, such as computer vision and speech recognition. In many applications, low hardware cost, low communication cost, energy-efficiency, and/or privacy concerns are desired. Microcontrollers provide these properties and are widely used in industrial settings. This work combines the strengths of the TensorFlow ML framework and the Pelion IoT Platform to demonstrate an easy-to-use workflow for developers wishing to run NNs on microcontrollers in the wild.

In spite of their limited memory and compute resources, many studies have shown the possibility of deploying complex NNs on microcontrollers. This is achieved through model compression methods like pruning and quantization. For example, popular architectures such as CNN and LSTM have deployed on microcontrollers for keyword spotting and image recognition tasks. CMSIS-NN, a library of optimized kernels for NNs, has been developed to maximize performance and minimize memory footprint on Arm Cortex-M processors targeted for IoT devices.

However, this is only one side of the problem. In practice, one needs to update the NN or other software, often remotely. In this work, we demonstrate a novel system to enable scalable remote updates from the cloud to NNs deployed on microcontrollers. This system is based on TensorFlow, MbedOS enabled microcontrollers and the Pelion Device Management (Pelion DM) platform.

MbedOS is an open-source real-time operating system (RTOS). It is designed specifically for IoT devices. The associated toolchain allows for straightforward compilation of C/C++ applications. Pelion DM securely connects and recognizes these devices with standards-based communication using Open Mobile Alliance's LWM2M model and Constrained Application Protocol (CoAP). It also offers an end-to-end management and monitoring of remote updates. The Device Management Portal provides visualization of raw data and results of inference.

In this work, we demonstrate a complete workflow. We choose a simple use case of anomaly detection with a simple model to demonstrate the workflow, though more complex models are supported. We begin by posting raw temperature data from microcontrollers to Pelion Data Management (Treasure Data) database. We then train a linear regression model as a single-layer NN with mean square error (MSE) in TensorFlow.

The TensorFlow model is then frozen in a protobuf file and the embedded C++ code is generated by uTensor (an extremely light-weight machine learning inference framework built on Mbed and TensorFlow). This program is compiled with MbedOS and flashed to the devices. The devices then securely register with Pelion DM using Mbed Cloud/Client API keys. This allows the remote update. After more data is gathered, the model is re-trained and the weights of the improved model are pushed to the microcontroller, again with Pelion DM.

The standard deviation and confidence interval are computed and combined with the predicted temperature to give an interval estimation. Data points lying outside the 95% confidence interval are flagged as anomalies. The MSE on training and test datasets is 0.249 and 1.03 respectively. There is no loss of accuracy as compared with running on a desktop computer. Likewise, a classifier can be trained by using logistic regression by replacing the MSE loss function with cross entropy. Again, there no loss of accuracy on this simple task. In terms of memory usage, the total Static RAM of the entire program is 39.38 kB and the total flash memory is 607.68 kB (including Mbed OS, uTensor and NN). The demo will run on a DISCO-F413ZH (Cortex M4 based running at 100MHz with 320 kB SRAM and 1.5 MB Flask) board connecting to Pelion through WIFI.

*These authors contributed equally to the work. Please send further questions to yue.zhao@arm.com