From Collection To Classification: An End-to-End Software Solution for tinyML on Resource Constrained Embedded Devices

The rapid drop in the price MEMS sensors combined with improvements in power-performance of SoC microcontrollers is opening the door to a wave of new IoT applications. At the moment, most IoT use cases involve simple connected sensors which stream data to a central repository for further processing. However, there are many applications in the industrial, agricultural, consumer and medical fields where network connectivity or bandwidth are limited and this approach is not possible. In these cases, it is necessary to deploy tinyML classification algorithms locally at the sensor node to generate meaningful metadata for local decision making or transmission.

A major challenge in developing tinyML algorithms is a lack of software tools for generating suitable firmware code from machine learning models. Data Scientist often create algorithms using Python or R, which are then shipped to a cloud service for deployment. With tinyML they find themselves working with a firmware engineer to continuously refine their code until it meets the accuracy vs power and memory requirements. This is a challenging process that can easily take months. There is an obvious need for software tools that are designed from the ground up to target deploying machine learning models to resource constrained embedded devices so that when a data scientist finishes developing an algorithm, they can immediately generate production grade firmware code just as easily as they can deploy a model to the cloud.

At SensiML we have developed an end-to-end software application which enables developers to rapidly build models consisting of data preprocessing, feature extraction and classification steps in the cloud which can then be automatically compiled to a target embedded devices. By using a cost report consisting of latency, RAM and ROM usage accuracy vs performance tradeoffs can be made early in the algorithm building process. The final pipeline is turned into firmware code for the target device and requires only the raw sensor data as input. The preprocessing, feature extraction and classification algorithms are all generated as part of a library. Developers can also combine multiple models consisting of entirely different preprocessing, feature extraction and classifier or hierarchical models which consist of shared resources but different classifier and feature extraction steps into a single library. This allows developers to generate a single resource efficient algorithm capable of performing classification in different contexts.

Our contribution is significant to the tinyML community as we have developed a software that rapidly speeding up time to market for deploying tinyML to IoT devices as well as lowering the bar of entry for teams that lack expertise in firmware and/or data science. Beyond that we are working to integrate support for new ML HW accelerators and FPGA fabrics so that we can help support the ecosystem by allowing developers to use a familiar toolchain but still reap the benefits of acceleration on the edge devices. Our main goal is to empower application developers with the ability to rapidly create production ready algorithms for embedded devices without having to write a single line of firmware code.