Bringing Machine Learning to the Deepest IoT Edge with TinyML as-a-Service*

Preprint · July 2020

CITATIONS
0

READS
3,882

3 authors, including:

Roberto Morabito
Ericsson
35 PUBLICATIONS **1,411** CITATIONS
SEE PROFILE

Jan Höller
Ericsson
56 PUBLICATIONS **927** CITATIONS
SEE PROFILE

Bringing Machine Learning to the Deepest IoT Edge with TinyML as-a-Service*

*This text was published in the IEEE IoT Newsletter - March 2020

Hiroshi Doyu
Ericsson Research
hiroshi.doyu@ericsson.com

Roberto Morabito
Ericsson Research
roberto.morabito@ericsson.com

Jan Höller
Ericsson Research
jan.holler@ericsson.com

I. Introduction

The power of machine learning can have a remarkable technological impact on the core of constrained and embedded Internet of Things (IoT). Yet various technological barriers have so far made it challenging to realize the full value of ML-driven IoT at the edge. TinyML holds promise for a solution. At Ericsson Research, we are currently exploring the potential and challenges of TinyML, as well as introducing the concept of TinyML as-a-Service (TinyMLaaS) to address some of the challenges. Machine Learning (ML) has profoundly revolutionized and enhanced the last decade of computer technologies. By extension, it has impacted several application domains and industries ranging across medical, automotive, smart cities, smart factories, business, finance, and more. Remarkable research efforts are still ongoing today, across both industry and academia, to bring the full advantage of the ever-growing number of ML algorithms. Here, the aim is to make computing machines, of every size factor, smarter and able to deliver sophisticated and reliable services. ML applied in the context of the IoT is, without doubt, an application domain that has attracted a large amount of interest from across the enterprise, industrial and research communities. Today, researchers and industry experts are working extensively to advance existing ML-driven IoT to boost the quality of experience for users of smart devices and the improvement of industrial processes. It is worth noting that the use of ML in IoT has multiple opportunities and interpretations. In our view, taking advantage of intelligent algorithms in the IoT context includes also having the possibility of equipping small IoT end-devices running on micro-controllers, with capabilities to benefit from ML algorithms. This thus extends the use of ML in IoT beyond the cloud and more capable devices running e.g. Linux.

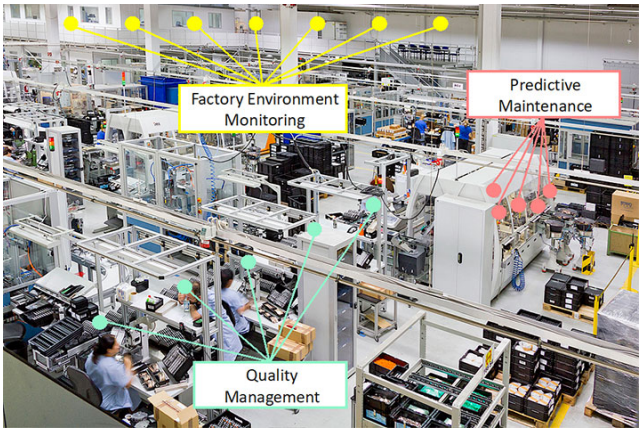

Fig. 1. Example usages of TinyML in industrial IoT.

II. What is TinyML?

Using the above definition of "constrained IoT device" as a starting point, it is crucial to characterize the distinction between "serving" ML to IoT devices, and "processing" ML within IoT devices.

In the "serving" case, all the ML-related tasks like training are “outsourced” to the Edge and Cloud, meaning that an IoT device is somehow "passively" waiting to receive the rendered ML model algorithm. In the "processing" case, an IoT device effectively uses the ML model for local inferencing on sensor data.

Figure 2 illustrates the overlaps of different technology areas in this context and where our research focus is. One can note several overlapping areas representing the common grounds of interest. As an example, the world of embedded Linux can be considered a rallying point between "Linux" technologies and "constrained IoT", thus also acknowledging that IoT capabilities stretch across the device-edge-cloud realms. "TinyML" represents the intersection between "Constrained IoT" and "ML" and disjoint with "Linux", the latter feature being a crucial aspect of our research focus [1].

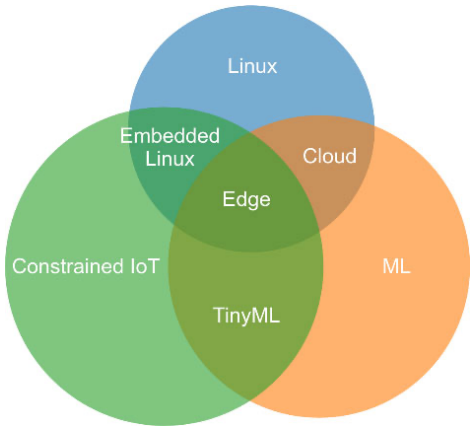

Fig. 2. Intersections between Constrained IoT, ML, and Linux.

Here we define TinyML as the technology area which concerns the running of ML inference ("processing") on Ultra-Low-Power (ULP ~1mW) micro-controllers found on IoT devices. TinyML is not only a general technical concept but also it has an emerging community of researchers and industry experts. tinyML Summit is held annually and tinyML meet-up is held monthly at Silicon Valley [6, 7].

III. The Challenges of TinyML

Now we elaborate a little on two key challenges of TinyML itself, the first being related to development, the second related to applicability of ML frameworks.

- **The gap between general software development and embedded development**: general software development and execution usually target environments of a fleet of Linux machines with Gigabytes of RAM, Terabyte of storage (HDD/SSD), GHz of processing and multicore

64-bit processors, and where Linux Container orchestration is used. On the other hand, embedded development and execution target a variety of micro-controllers, a variety of Real Time Operating Systems (RTOS), with 100s of kB of SRAM, a few Megabytes of flash memory, without any standard orchestration. Those two target environments, as illustrated in Figure 3, are totally different. We cannot migrate cloud-native software onto constrained IoT devices.

Fig. 3. Web vs Embedded software environments.

- **Applicability of ML frameworks**: as mentioned, ML typically has two phases, one for training and another for inferencing. ML training is usually done in the cloud with popular python-based ML frameworks, e.g. TensorFlow, PyTorch, etc., and its produced model is stored and archived in repositories called model zoo. Thanks to the latest introduction of ONNX (Open Neural Network eXchange format), each ML framework can make use of a model that is trained on another framework easily. But this cannot be applied to embedded IoT. Any of those frameworks and models are too big to run on IoT devices (Figure 4).

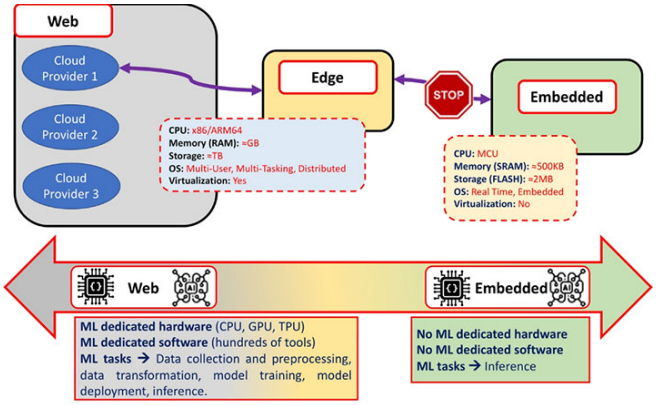

Fig. 4. ML, software and hardware specifics across cloud, web and embedded domains.

The above limitations are further explained in one of our Ericsson Blog articles [2]. In summary, we propose to build a higher-level abstraction of TinyML software that is as hardware and software agnostic as possible to hide the heterogeneity of ML-enabled chips and compilers, and further to support this in an "as a Service" fashion. This is what we call TinyML as-a-Service.

IV. What is TinyML as-a-Service?

So, what is our TinyML as-a-Service concept (TinyMLaaS) and how can it solve TinyML problems?

A typical and traditionally pre-trained ML inference model cannot be run on constrained IoT devices as it is, because the computing resources of those constrained devices are not enough. Such models must be converted into the appropriate size fitting the target device resources. An ML compiler can convert a pre-trained model into an appropriate one for the target IoT device platform. They use techniques to squeeze the model size, for example, "quantizing" with fewer computing bits, "pruning" less important parameters, "fusing" multiple computational operators into one. Since popular ML frameworks cannot run in the targeted IoT devices, an ML compiler also needs to generate a specialized small runtime, optimized for that specific model and for the embedded hardware accelerators that the device is featuring. The latter is typically chip vendor-specific, and we consider those steps as a customization service per device features.

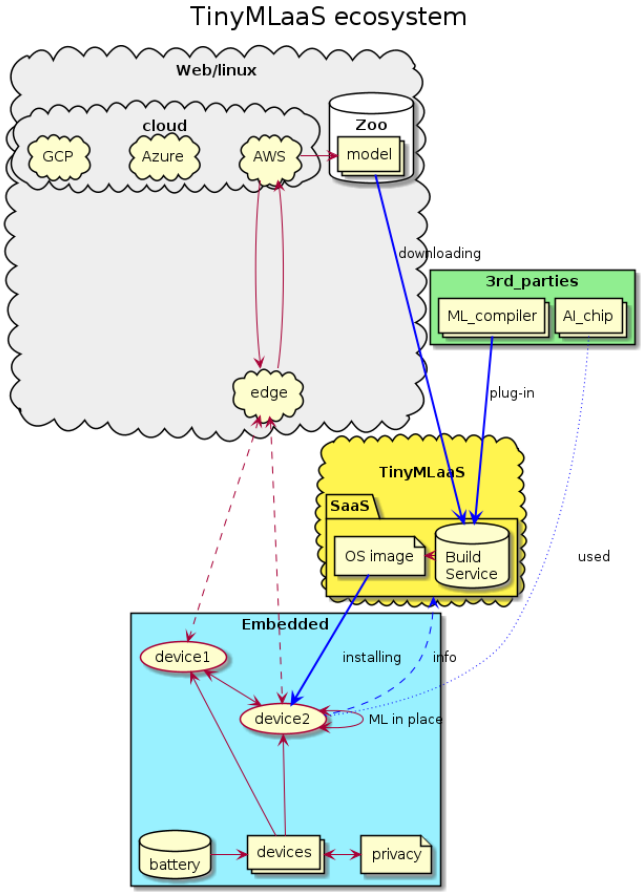

Fig. 5. TinyML as a Service overview.

TinyML as-a-Service is proposed as an on-demand customization service in the cloud. It can host multiple ML compilers as its backends, firstly gather device information from a device, e.g. using LwM2M [8]. Secondly, it can generate an appropriate ML inference model from model Zoo, and then install it onto devices on-the-fly, e.g. again using an LwM2M Software Over The Air update (Figure 5).

Usually, embedded developers and ML developers have different and often complementary development skills. This means that introducing ML into the embedded world can represent a challenge for embedded developers. However, with the use of TinyMLaaS, embedded developers can easily introduce ML capabilities onto their devices and, vice versa, ML developers can also target constrained IoT devices when designing their algorithms and models. Looking at the high-level picture, TinyMLaaS can potentially enable any service providers to start their AI business with devices more easily.

To learn more about the TinyMLaaS approach and the impact it can generate, please refer to our Ericsson blog article [3].

V. Conclusion

The TinyML community has rapidly evolved during the last year. TinyMLaaS is an ecosystem around TinyML. Other ecosystem players, like chip vendors, compiler companies, service providers, etc. have an opportunity to both influence and accelerate the development of such an ecosystem. Here at Ericsson, we very much encourage and invite this level of cross-industry collaboration, to make ML at the deepest IoT Edge possible.

References

[1] H. Doyu, R. Morabito. "TinyML as-a-Service: What is it and what does it mean for the IoT Edge?" [Online]. Available at: https://www.ericsson.com/en/blog/2019/12/tinyml-as-a-service-iot-edge

[2] H. Doyu, R. Morabito. "TinyML as a Service and the challenges of machine learning at the edge." [Online]. Available at: https://www.ericsson.com/en/blog/2019/12/tinyml-as-a-service

[3] H. Doyu, R. Morabito. "How can we democratize machine learning on IoT devices?" [Online]. Available at: https://www.ericsson.com/en/blog/2020/2/how-can-we-democratize-machine-learning-iot-devices

[4] C. Bormann, M. Ersue, A. Keranen. "Terminology for Constrained-Node Networks". Internet Requests for Comments (RFC), No. 7228, 2014.

[5] P. Warden, D. Situnayake. "TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers". O'Reilly Media, 2019.

[6] tinyML Summit 2020. [Online]. Available: https://www.tinyml.org/summit/

[7] tinyML - Enabling ultra-low Power ML at the Edge. [Online]. Available: https://www.meetup.com/tinyML-Enabling-ultra-low-Power-ML-at-the-Edge/

[8] Open Mobile Alliance (OMA). "Lightweightm2m technical specification v1.0". [Online]. Available: http://www.openmobilealliance.org