Collaboration between IoT-devices in Edge Computing hierarchical structure using TinyML

Ivan Kralj, Gordan Ježić
University of Zagreb, Faculty of Electrical Engineering and Computing
Department of Telecommunications
Zagreb, Croatia
ivan.kralj@fer.hr, gordan.jezic@fer.hr

***Abstract*—The coordinated integration of heterogeneous TinyML-enabled elements in highly distributed Internet of Things (IoT) environments paves the way for the development of truly intelligent and context-aware applications. Edge computing has emerged during the last years as a ground-breaking solution that permits to enrich regular IoT deployments with novel services and possibilities. Under this paradigm, the processing and storage capabilities of end-devices and edge-nodes are exploited in order to reduce their cloud-dependency by adding a new layer in the network architecture in charge of data aggregation, filtering, processing, and storage. On the other hand, TinyML is a recently-emerged paradigm that proposes to embed optimized Machine Learning (ML) models in units with limited computing resources, such as those powered by micro-controllers. Our goal in this doctoral research is to optimize traffic control systems through the integration of IoT-devices within an Edge Computing hierarchical structure. Specifically, the focus is on leveraging TinyML techniques in the context of IoT for traffic control applications.**

***Index Terms*—Internet of Things, Edge Computing, TinyML**

I. Introduction

The Internet of Things (IoT) is a major player in the digital revolution, enabling devices and humans to interact with each other in real time [1]. This interaction generates large amount of data whose analysis can provide insights from uncovering hidden patterns, correlations between variables, etc. However, knowledge is only valid as long as it is generated at the right time to enable the right decisions to be made. Therefore, enabling efficient analysis of this huge amount of data generated by the IoT is crucial to transform this deluge of data into meaningful information [2].

Machine Learning (ML) algorithms are showing their potential for extracting knowledge from large amounts of data. Traditionally, ML algorithms have been executed in supercomputers, where performance prevails over energy efficiency. However, when performance is not the only concern, other approaches are feasible. For instance, edge/fog computing has been approached towards decentralization, where initial computations on data are carried out in (or close to) the data capture devices. In fact, the edge computing paradigm is providing (1) energy savings by avoiding sending and processing data in the cloud, (2) highly responsive applications and services for mobile environments, (3) highly scalable systems, thanks to the distribution of processing units, (4) guaranteed privacy policies for the IoT and (5) disconnection tolerant systems as transient connection interruptions can be masked [5].

IoT devices have a limited power budget at this level of the network, as they typically rely on batteries or energy harvesters, leading to ultra-low power approaches. This limited power scenario translates into a major limitation for many components of the architecture, especially energy-intensive components such as wireless transmitters or even processing capabilities. A new trend, called TinyML [6], has recently emerged at the intersection of ML, IoT and computing platforms. This trend aims to leverage microcontroller units (MCUs) that are available in all devices across the IoT ecosystem, from sensor data collection and actuation, to information transfer and reception [3].

II. Related work

To our knowledge, there is only 1 prior work that proposes a hierarchical TinyML scheme leveraging the opportunities brought by an edge computing-based architecture.

R. Sanchez-Iborra *et. al.* [4] proposed an IoT-edge computing multi-layer intelligent architecture to enable decision making at the highest level of the hierarchy, i.e. an edge-node. This edge computing-based configuration that they proposed presents a number of advantages in comparison with typical centralized cloud-computing models. Adopting a hierarchical TinyML scheme permits end-devices to form part of the decision process as their individual decisions are considered by the higher-level instance. Besides, the system scalability and reliability is ensured given the modularity of the solution. Finally, this proposal also permits low-cost deployments as no expensive processing units or data centres are needed. This is achieved thanks to the adoption of the TinyML paradigm and the exploitation, in a distributed way, of the processing capabilities of IoT devices [4]. Figure 1 and 2 show their proposed hierarchical stacked ML scheme and TinyML workflow.

For their experimental design, they decided to use general empirical methodology for their specific use-case, in which they used a green house equipped with (i) a range of ground sensors that monitor the status of the plantation and (ii) a set of fixed sprinklers that cover certain plantation zones. The first decision level (end-device) consists of a set of p zones, and each zone contains r slave devices, so the number of elements at this level is $r \times p$. Each IoT device contains a

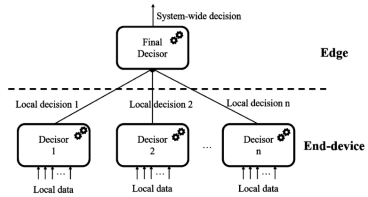

Fig. 1. Hierarchical stacked ML scheme [4].

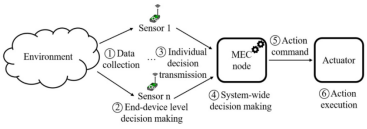

Fig. 2. Hierarchical stacked TinyML workflow [4].

TinyML model that outputs its individual decision using a set of inputs $\{x_1,x_2,...,x_n\}$ collected from the environment. In turn, the second level consists of a single master device (edge-node) which provides a final-decision TinyML model using as input a categorical vector of r dimensions $\{s_1,s_2,...,s_r\}$, which represents the outputs of the individual devices within a zone, and provides as output a final decision for that zone.

The training and evaluation of the different ML models have been carried out in a Python non-constrained environment and used the criterion of accuracy (for balanced data) and the balanced accuracy (for unbalanced data) as performance metrics. For evaluation, the following they used the following criteria: Flash memory, SRAM, and latency.

For their end-device-level decision, TinyML model at this level infers the most advantageous action for the monitored plant(s) and outputs (o) it, o $=\{a_0,a_1,a_2\}$, where a_0 indicates "no action", a_1 represents the "irrigation action" (watering), and a_2 symbolizes the "fertigation action" (watering with nutrients).

For their edge-level decision, they consider that each sprinkler covers a zone monitored by 4 sensors. Thus, the meta-model placed in the edge-node receives 4 input parameters $\{s_1,s_2,s_3,s_4\}$, where s_i represents the decision made in the previous step by each of the 4 sensors within a certain zone. Finally, this model generates a single output (O) with three possible commands $\{A_0,A_1,A_2\}$, i.e. "no action", "irrigation", and "fertigation", respectively.

For TinyML models generation, they used the following ML algorithms: Multi-Layer Perceptron (MLP), Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM). In the end, they concluded that DT algorithm is the most adequate one to implement at the end-device-level, with it's 99.9% accuracy, 13 $\pm$ 2 µs latency, using 346 B of SRAM, and 3392 B flash memory. For the edge-level, they concluded that DT algorithm is, once again, the most adequate one to implement, given its near perfect accuracy with low impact on the device.

III. Problem and motivation

The primary challenge regarding traffic control using real footage of the traffic lies in establishing efficient collaboration and communication between diverse IoT devices placed at different levels of the Edge Computing hierarchy. The solution should encompass hierarchical collaboration between IoT-devices operating at various hierarchical levels, integrating TinyML capabilities into IoT-devices to perform real-time analysis and decision making based on the video data and to address challenges posed by the limited processing power, memory, and communication bandwidth in IoT-devices, while still maintaining the responsiveness and accuracy of the traffic control system.

One idea proposed in this doctoral research is to implement Convolutional Neural Network (CNN) on an ESP32, where data is generated. CNNs are computationally intensive and can demand significant processing resources. The ESP32, being a resource-constrained device, may struggle to execute the entire CNN model with a satisfactory level of accuracy and speed. So, to address the computational limitations of the ESP32 while still leveraging the benefits of CNNs for edge processing, a hierarchical edge processing approach can be adopted. In this approach, the workload of the CNN is split between the ESP32 and a more powerful edge device (the 2nd edge layer). By offloading the resource-intensive portions of the CNN to the 2nd edge layer, the system can still achieve high accuracy and performance without overwhelming the ESP32.

The ESP32, as the 1st edge layer where data is generated, can preprocess and extract important features from the data using a small CNN segment, typically consisting of a few initial layers. This lightweight processing can help reduce the data size and complexity before passing it on to the next edge layer.

The 2nd edge layer, which is a more capable edge device or a cloud-connected resource, receives the preprocessed data from the ESP32. This device has greater computational power and memory, enabling it to handle the more complex layers of the CNN.

References

[1] Feki, M.A., Kawsar, F., Boussard, M., Trappeniers, L.: The internet of things: The next technological revolution. Computer **46**(2), 24–25 (2013). https://doi.org/10.1109/MC.2013.63

[2] Juan Morales-García, Andrés Bueno-Crespo, R.M.E.J.L.P.P.M.J.M.C.: Evaluation of edge computing platforms through tinyml workloads. The Journal of Supercomputing (2022)

[3] Portilla, J., Mujica, G., Lee, J.S., Riesgo, T.: The extreme edge at the bottom of the internet of things: A review. IEEE Sensors Journal **19**(9), 3179–3190 (2019). https://doi.org/10.1109/JSEN.2019.2891911

[4] Sanchez-Iborra, R., Zoubir, A., Hamdouchi, A., Idri, A., Skarmeta, A.: Intelligent and efficient iot through the cooperation of tinyml and edge computing. Informatica **34**(1), 147–168 (2023). https://doi.org/10.15388/22-INFOR505

[5] Tahsien, S.M., Karimipour, H., Spachos, P.: Machine learning based solutions for security of internet of things (iot): A survey. Journal of Network and Computer Applications **161**, 102630 (2020). https://doi.org/https://doi.org/10.1016/j.jnca.2020.102630

[6] tinyML Foundation: Tinyml, `https://www.tinyml.org/`