Energy efficient Firmware Over The Air Update for TinyML models in LoRaWAN agricultural networks

1st Chollet Nicolas
ECE, LISV
Paris, France
nicolas.chollet@ece.fr

2nd Bouchemal Naila
ECE
Paris, France
naila.bouchemal@ece.fr

3rd Ramdane-Cherif Amar
LISV
Versailles, France
amar.ramdane-cherif@uvsq.fr

2022 32nd International Telecommunication Networks and Applications Conference (ITNAC) | 978-1-6654-7103-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ITNAC55475.2022.9998338

***Abstract*—Current agricultural practices fail to feed everyone correctly while being harmful to the environment and highly sensitive to climate change. Therefore, new and modern agriculture needs to be developed and overcome numerous challenges. Artificial Intelligence (AI) is one of the tools widely used in this new type of Agriculture called Precision Agriculture (PA) or Smart Farming (SF). Thanks to the Internet of Things (IoT) technologies, AI algorithms are fed with a vast amount of data to provide valuable insights for farmers, such as weather prediction, pest development detection, irrigation management, etc. AI algorithms are often executed in cloud servers, thus requiring IoT devices to offload their data to process. This creates privacy, latency, and security issues, but mostly, it requires a large quantity of energy for transmission. To overcome those issues, recent research brought new tools like Tiny Machine Learning (TinyML) to perform AI directly on the IoT devices and break free from the cloud. Despite promising results, such Smart devices cannot train new models on their constrained hardware and therefore need frequent updating to increase the model's accuracy over time, regarding the specific environment where the sensor is deployed. In the Agricultural domain, sensor devices are numerous and usually spread over vast geographical areas while running on battery. For this reason, Farms Wireless Sensor Network (WSN) use mostly Low Power Wide Area Network like LoRaWAN to communicate. Therefore Firmware Update Over the Air (FUOTA) is required. In this context, this paper proposes a study of the FUOTA process for a TinyML model using LoRaWAN in a specific agricultural scenario. A TinyML sensor prototype was built to evaluate the feasibility of FUOTA for tinyML devices using LoRaWAN. The system's energy consumption and Packet delivery ratio are then analyzed in a simulator with different network scenarios.**

***Index Terms*—Agriculture, AI (Artificial Intelligence), IoT (Internet of Things), LoRaWAN, LoRa, Smart Farming, TinyML (Tiny Machine Learning), FUOTA (Firmware Update Over the Air), LPWAN (Low Power Wide Area Network)**

I. Introduction

The Food and Agricultural Organization estimates that food production must increase by 60 % to feed the 9.8 Billion humans on the globe by 2050[1]. However, agricultural land is already covering 38 % of the overall emerged land worldwide, and modern agriculture uses harmful environmental methods [2]. For example: the intense usage of chemicals resources that are present in low quantity on earth and pollute the soil and water stocks while destroying the biodiversity [3]; the usage of oil hungry heavy machinery that accelerates soil compaction [4]; the massive needs for irrigation water that dry up groundwater supply [5].

To change how humanity produces food, researchers have brought numerous new tools using Information and Communication Technologies (ICT) to the farm fields, creating a new research domain called Smart Farming (SF). SF, also called Precision Agriculture (PA), uses data produced by wireless sensor networks (WSN) and Artificial Intelligence (AI) to understand better the environment and its interaction with crops and livestock to make food more efficient with fewer resources [6].

In this domain, Artificial Intelligence (AI) powered Computer Vision algorithms are of particular interest as they can, for example, identify a plant's different stages of growth to detect specific situations such as disease development, pest attack, or fruit maturity. This knowledge is crucial as it allows the farmers to take the necessary actions in time (Crop thinning, fertilizer application, call for harvest, etc.) [7]. Unfortunately, such algorithms require significant processing power. Thus such programs are usually performed in the cloud and not directly on the device taking the picture [8]. However, sensors in the Agricultural domain mainly use Low Power Wide Area Network (LPWAN) to transmit data to save battery lifetime [9]. Those type of network allows transmission over long distances at minimal energy cost but with a reduced data throughput, making difficult the communication of heavy files such as images[10].

To avoid relying on the cloud for image analysis, researchers have looked into bringing the computation of AI algorithm directly to the end device with constraint hardware. Those techniques are called embedded AI or Tiny Machine Learning (TinyML) [11]. Performing the AI computing task on the device allows the vision sensor to communicate only the result in small messages through LPWAN, saving battery life while increasing privacy and security and reducing the latency [12].

Despite promising results, TinyML methods only allow the end device to perform inference on its hardware and are not able to learn and evolve after its deployment into a specific environment. Therefore Smart Edge Devices need to be updated frequently to ensure their accuracy [13]. However, as said previously, WSNs in agriculture are deployed over vast geographical areas making it challenging to update devices through wired connections requiring remote updating. The Firmware Update Over The Air (FUOTA) problem is well known in the IoT field as every device will probably need bug fixes and security maintenance while deployed [14]. Multiple researchers have also studied its feasibility for LPWAN network [15] as their characteristics regarding data throughput and regulations involve extra layers of complexity to take into account.

In this paper, we propose a study of the FUOTA process for LoRaWAN network as it is a widely used network in agriculture applications [16]. We then analyze its usage in the context of a TinyML firmware update. Afterward, we build a prototype of an intelligent sensor using TinyML for fruit presence detection communicating its result through a LoRaWAN network for energy saving. The sensor feedback will let the farmer know when and where fruits are present to take the necessary actions (harvest, fertilizer application, etc.). After deploying the first model in a specific environment, we train a new model and perform a FUOTA with LoRaWAN to validate the feasibility of the process. Finally, we study the energy consumption of our prototype and the packet delivery ratio in different network scenarios.

II. State of the Art

A. *Firmware Update for Internet of Things devices*

IoT Technologies are being used exponentially in every domain of modern life, and as related protocols and applications evolve, the need for firmware updates also [17]. An update allows a developer to add functionalities to the IoT devices after deployment, fix bugs, and patch security issues. Good surveys on fundamental principles and programming techniques of an IoT device firmware update process have been made by the authors in [18] and [19]. Key points of the IoT Firmware Update process are:

- **Over the Air:** Devices are spread over various geographical areas and might be difficult to reach. The update process must be performed wirelessly. In the agricultural domain, we talk about hundreds of devices in vast crops field [6].
- **Security :** Ensuring that an update is performed only on the request of the authorized user is critical. Otherwise, hackers could take control of the device by remotely updating corrupted firmware. [20].
- **Energy Consumption:** IoT devices are mostly battery-powered. When it is the case, it should minimize communication as wireless transmission is inherently power-hungry. However, a firmware update can be a large file [21].

B. *Firmware Update over the air for LPWAN*

If the devices communicate through a LPWAN, the FUOTA process will be impacted accordingly, as the characteristics of such networks make the transfer of large file complex. The authors in [22] and [23] have surveyed existing issues and methods for different protocols. In LPWAN, the biggest constraint is energy consumption; Various strategies exist to optimize battery life and can be found in [21]. The second most significant issue is the data rate which can be low compared to the update's size, as discussed in [24].

C. *Firmware Update over the air for LoRaWAN network*

This paper will focus on implementing a FUOTA process in LoRaWAN as it is the primary LPWAN protocol used in Agricultural applications [9]. The reason for this is its good technical abilities (data rate, distance, energy consumption, etc.) and its ability to create an ad-hoc network without relying on already existing network provider coverage [25] facilitating LoRaWAN deployment and usage in rural and remote areas. The LoRaWAN firmware update technical recommendations are described by the LoRa Alliance, the organism in charge of maintaining the LoRa technology, and can be found here [26]. The main issue with LoRaWAN FUOTA process is the size of the update to perform. LoRaWAN has a low data throughput and duty cycle and is subject to interference [29]. Therefore, sharing large data files is complex. To do so, three mechanisms must be applied: Multicast update (update all the device that needs together), clock synchronization (choose the right moment to perform the update), and Fragmentation (divide the update file into smaller ones). A detailed description of the methods applied to perform FUOTA is made by the authors in [27], and [28]. Regarding practical implementation, there are various attempts in the literature to propose a suitable solution for FUOTA in LoRaWAN. The leading available solution for programmers is to use The Chirp Stack LoRaWAN open source server [30] with the open source Embedded Device OS Mbded OS [31]. The example code can be found here [32]. There is also proprietary solution proposed like Multi-Connect Conduit [33] and Pycom [34] but only work with their specific hardware. On the research side, applications are relatively new and scarce: Authors in [35] have tried to implement FUOTA with multiple gateways, while the authors in [36] have looked into adaptive data rate techniques to optimize the process. Finally, the authors in [37] have developed a small test bench to validate the usage of FUOTA for LoRa in the agricultural domain, where the distance and energy factors have been studied to validate the practical application of FUOTA in Farms.

D. *Firmware Update for TinyML Model*

TinyML adaptation capabilities are a well-known issue as the inference model must be pre-trained before being embedded in a hardware-constrained IoT node [13]. To allow a device to adapt its model over time regarding the specific environment where it is deployed, researchers have proposed methods to allow on-device learning, like the author in [38] and [39]. However, we assert that TinyML model could still be pre-train on dedicated servers and then remotely updated on the device. Literature on this topic is scarce, and we found only one approach proposed by the author in [40] to update the model remotely through WiFi. They use a web platform to allow the end device to fetch the model it needs for different applications allowing the end device to full fill various objectives. Therefore, our approach to updating a TinyML model through LPWAN is novel.

III. Architecture Proposal

A. Scenario

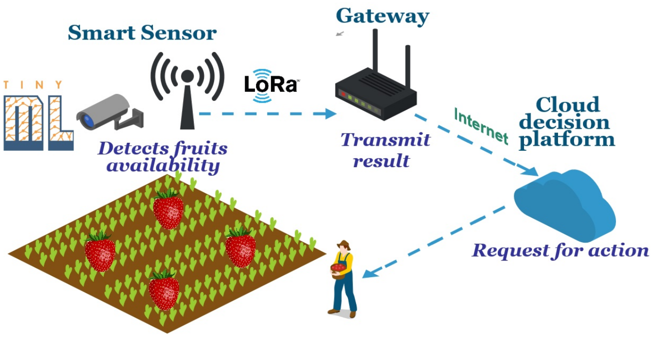

Fig. 1. Use case scenario

An intelligent camera sensor is deployed in a field to detect the number of fruits thanks to a TinyML algorithm. It then communicates the number of fruit detected to a cloud decision platform through a LoRaWAN network. A decision is then taken regarding the action to perform, for example, fertilizer application, irrigation, or harvest. This decision is then communicated to the performer, for example, the farmer. This use case scenario is depicted in Figure 1.

When the accuracy of the inferences drops below a certain point, the sensor device requires an update from the server with a specific message sent through the LoRaWAN network. The application server must then acquire new picture data to retrain the TinyML Model of the device. Once done, the update is transmitted from the application server to the LoRaWAN server, which will deploy the update to the defined devices with a FUOTA process. This process is describe in Figure 2.

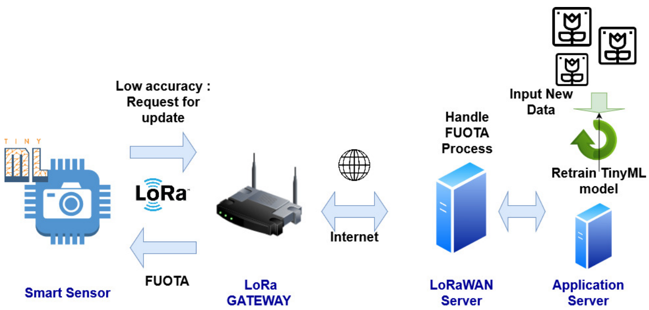

Fig. 2. FUOTA with LoRaWAN network: Overall process

Our architecture is therefore made of four main components:

*1) **Smart sensor**:* It uses TinyML to infer the number of fruit in its sight of view. If the accuracy of its inferences is too low, it requests a firmware update over the air.

*2) **Gateway**:* In charge of the communication between the LoRa device and the LoRaWAN server on the internet

*3) **LoRaWAN Server**:* In charge of receiving and dispatching the message from the device and the Application Server. Also in charge of the FUOTA process in LoRa.

*4) **Application Server**:* In charge of the model retraining when enough new data are acquired.

B. Environment development

The prototype is built with the following hardware and software parts: For the microcontroller with the camera sensor, performing the TinyML algorithm, and getting remotely updated, we use an Arduino Portenta H7 microcontroller with a Lora Vision shield. It is a 32-bit Microcontroller with a suitable specification to perform TinyML. We already proposed a prior TinyML application with this sensor [12]. The Arduino Portenta is running ARM Mbded Operating System [31] to perform the update when new firmware is received. Using the Arduino Ecosystem should help replicate our experiment for other researchers. To create and update the TinyML algorithm, we used Edge Impulse, the development platform based on tensor flow lite proposed by Google [41]. Finally, to communicate with LoRa, we use a Laird RG1868 gateway with the ChirpStack open source LoRaWAN Server that has already included tools for FUOTA [30].

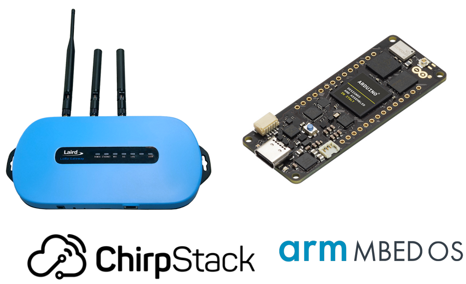

Fig. 3. Laird RG1 running ChirpStack (left) and Arduino Portenta running MBEDOS (right)

C. Phases

In this section, we describe the end-to-end phases implemented in our prototype.

*1) **Phase 1 - Firmware update request**:* After the sensor deployment, the farm ecosystem around the sensor might change, making the accuracy of its inferences drop beyond an acceptable point (value to define regarding the application criticality). When the sensor notices that it has reached this point, it sends a special message to the server to inform of its situation and request a new model update.

*2) **Phase 2 - New data acquisition** :* Once the server is noticed that an update is required, it should gather new data. In our scenario, farmers could take new pictures in the field. However, it could be interesting in future work to ask sensors to offload punctual pictures to increase the variety and quantity of data allowing the model to evolve.

*3) **Phase 3 - Model re-training**:* We use Edge Impulse FOMO (Faster Objects, More Objects) algorithm to make our fruit detection [41]. With it, the device can count objects, find the location of objects in an image, and track multiple objects in real-time using up to 30x less processing power and memory than other similar dedicated algorithms such as MobileNet SSD or YOLOv5 [42].

*4) **Phase 4 - FUOTA**:* The FUOTA procedure is described precisely in [26] [27] [28] . Its architecture is shown in Figure 4.

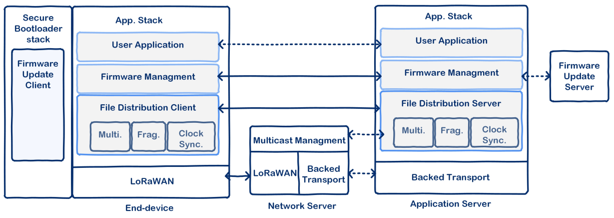

Fig. 4. LoRaWAN FUOTA specifications architecture

It is made of three components:

- **The Network Server (NS)** is at the core of the FUOTA process. It handles the communication of the unusually large update file by choosing the gateways used to broadcast the messages in compliance with the local regulations and limiting-self interference.
- **The End Devices (ED)** must implement the file distribution client (FDC) on top of the LoRaWAN link layer protocol stack. The FDC comprises three different protocols: Clock synchronization, multicast, and fragmentation.
- **The Application Server (AS)** handles the application encryption and decryption of the messages while also processing the application data mechanics. The interactions between AS and NS are specific to the application. The AS communicates with the NS to gather multicast settings to prepare the update when needed.

The overall process of FUOTA is not standardized by the LoRa Alliance, which only provides recommendations. Here we will describe the process applied in the MBOS/ Chirp Stack project [32] that we implemented later with a TinyML update. This process can be seen in Figure 5. LoRaWAN devices can be of three classes: A, B, or C. A and B are the most frequent as they are less energy-consuming than C, but class C allows better transmission reliability. For more information, please refer to [43]. We take the case of a FUOTA for a device in Class A, meaning the device is always asleep unless it sends an uplink message or during a short receiving window after the uplink is sent. The FUOTA starts when the Application Server request the Network Server to update multiple devices in class A. The NS first needs to prepare every device for the multicast session. For that, every time one device sends an uplink, the Network server sends an acknowledgment and a multicast address. The same mechanism is employed to set up the fragmentation. Afterward, the device will send a synchronization request. When satisfied, the NS will request the device to go in C class at a precise hour, so every device to be updated goes in C class simultaneously; C class means the device is never asleep and constantly listening for messages. All the devices go in C class at the defined time and listen for the multicast transmission. The End device reconstructs the fragmented firmware update file thanks to various error correction mechanisms. Then it restarts itself, and the bootloader will load the new firmware image if it is correct, discarding the previous one. Finally, an acknowledgment of success is sent to the NS. If anything goes wrong during the process, the devices that are not correctly updated will be flagged for a new cycle of FUOTA.

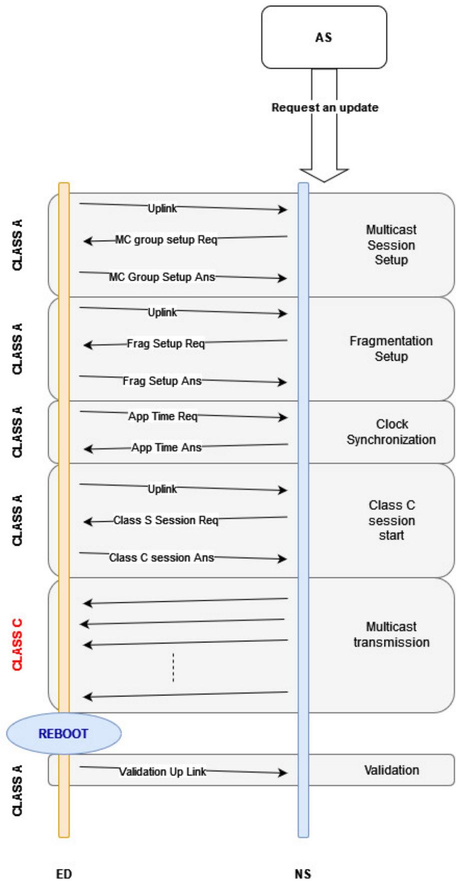

Fig. 5. LoRaWAN FUOTA session

IV. Experimentation

A. Firmware update over the air performance Evaluation

To test our prototype, we train a first model with a dataset of 100 pictures containing between 1 and 5 strawberries. The accuracy of the FOMO algorithm reaches 90.2%, the RAM usage is 243.9 kb, and the size of the firmware is 77.5 kb, respectively 24.39% and 3.88% of the Arduino's hardware capabilities, leaving room for improvement. Afterward, we train another model with 100 different pictures for the same application. This time the accuracy is 89.2%, the RAM usage is 252.2 kb, and the firmware size is 83.6 kb. Both models seem equivalent in performance when tested in real life directly on the microcontroller. We then implement Mbed OS 5.11.1 and flash the first TinyML model. Finally, we perform a FUOTA session to update the device with the other firmware containing the new model, as described previously and in [32]. All FUOTA tests have been made with only one device. The system behaves as expected and implements the new firmware after reboot meaning there is no particular issue for an update due to the nature of the TinyML compared to a classic firmware update. However, the duration of the update is more important with the TinyML firmware update (2m23s average of 10 tests) compared to the example update (32.3s average of 10 tests) provided in the OS. This is explained by the update size (83.6 kb for the tinyML firmware vs. 5.2 kb for the example update). You can find all the sources of our prototype with the adapted Mbed OS and further explanation on our GitHub [44].

B. *Energy consumption evaluation*

Modeling energy consumption is a complex process in the IoT domain as it is highly dependent on the hardware specificities and various other parameters, as discussed in [21]. For our experiment, we used the simple energy evaluation model from the INET framework of the Omnetpp simulator [45] on a PC running Ubuntu 20.04 with 16GB RAM and an Intel I7 8565U. The hardware power consumption values for different running modes are gathered from the datasheet of the Arduino Portenta and are presented in table I.

TABLE I
POWER CONSUMPTION OF THE SYSTEM

Mode	Current Consumption
Standby	2.95 μA
Run	121 mA
Transmission Lora	21.5 mA
Transmission WiFi	310 mA

To show energy consumption, we considered two scenarios. In each scenario, the device is in standby mode the majority of the time and only wakes up once a day to perform one TinyML inference and transmits a small telemetry message to the server. Moreover, the device wakes up once a week to be updated with a new firmware of 100kb size. In the first scenario, all communication are performed with LoRa, and in scenario 2 we use WiFi as a point of comparison. The simulator then evaluates the lifetime expectancy of the battery and run until both scenarios are out of power for a 2000 mA battery. Results are presented in Figure 6 and 7. It appears that using LoRa in such application could represent a significant energy saving compared to WiFi usage. However, improved optimization should be proposed in future work.

Furthermore, we propose an evaluation of the battery lifetime for various sizes of firmware updates. Results are presented in Figure 8 and 9. As expected, the update's size negatively impacts energy consumption, and the system runs out of battery quicker with large update files. However, the energy consumption is not proportional to the firmware size.

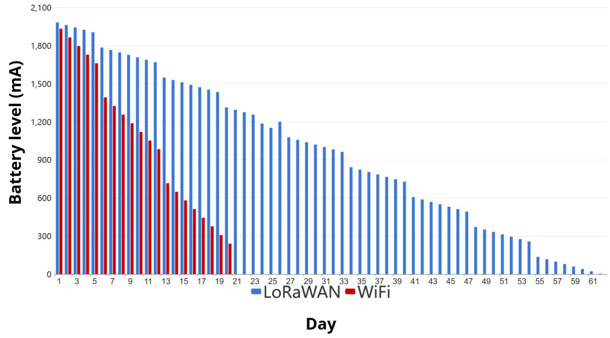

Fig. 6. Evolution of Battery level over time

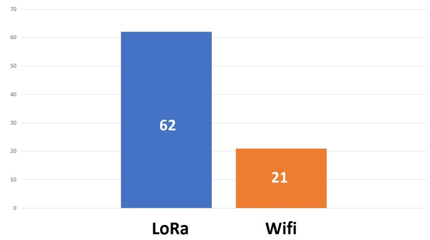

Fig. 7. Battery lifetime of the system in days

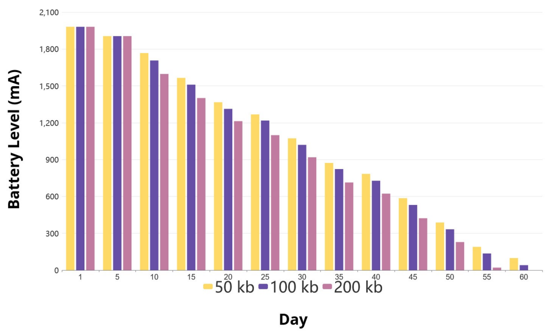

Fig. 8. Evolution of Battery level over time for various size of firmware

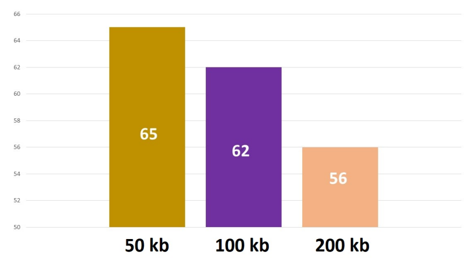

Fig. 9. Battery lifetime of the system in days for various size of firmware

C. *Packet delivery ratio*

We use the Flora framework from Omnetpp [46] on the same previous hardware to characterize the packet loss ratio of the network regarding the number of devices being updated at the same time and the size of the firmware. The results are presented in Figure 10. We notice that the packet delivery ratio of the network decrease as the number of device on the network increase. This is a result of higher interference. We also emphasize the effect of the firmware size on the network. As it increases, the packet delivery ratio drops down faster as the number of nodes increases.

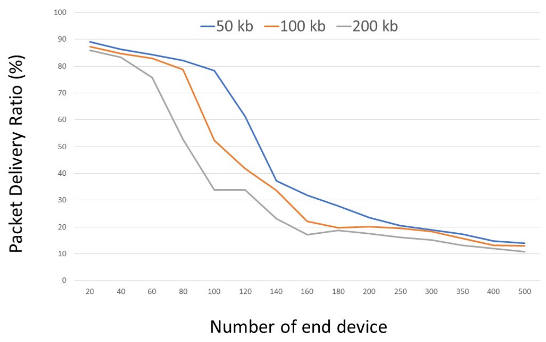

Fig. 10. Packet Delivery Ratio per node number

V. Conclusion

We propose in this paper an analysis of the FUOTA process for LoRaWAN with a specific application for TinyML model in an Agricultural Scenario. We then perform real-life experimentation of a FUOTA process to prove the feasibility of the solution while determining its energy efficiency through different scenarios. Despite promising results, it seems that remotely updating large-size firmware with LoRaWAN is costly in terms of energy consumption, especially if multiple devices are to be updated simultaneously, creating interference. However, It is our assumption that such methods could, if better optimized, be used to deploy Smart devices performing AI in farms. Future work should be oriented toward optimizing the FUOTA process for the TinyML model in LoRaWAN. It should also be essential to study the offload of images through the LoRaWAN network to collect data to facilitate the model re-training after its deployment in a new environment. Finally An interesting approach could be to use different Radio Access Technologies on the same device, such as LoRaWAN for standard data transmission and LTE for Firmware Update and Data offload.

References

[1] Ranganathan, Janet, et al. "How to sustainably feed 10 billion people by 2050, in 21 charts." (2018).
[2] Tilman, David, et al. "Global food demand and the sustainable intensification of agriculture." Proceedings of the national academy of sciences 108.50 (2011)
[3] Koli, Pushpendra, Nitish Rattan Bhardwaj, and Sonu Kumar Mahawer. "Agrochemicals: harmful and beneficial effects of climate changing scenarios." Climate change and agricultural ecosystems. Woodhead Publishing, 2019. 65-94.
[4] Hamza, M. A., and Walter K. Anderson. "Soil compaction in cropping systems: A review of the nature, causes and possible solutions." Soil and tillage research 82.2 (2005): 121-145.
[5] Madramootoo, Chandra A. "Sustainable groundwater use in agriculture." Irrigation and Drainage 61 (2012
[6] Elijah, Olakunle, et al. "An overview of Internet of Things (IoT) and data analytics in agriculture: Benefits and challenges." IEEE Internet of things Journal 5.5 (2018)
[7] Gomes, Juliana Freitas Santos, and Fabiana Rodrigues Leta. "Applications of computer vision techniques in the agriculture and food industry: a review." European Food Research and Technology 235.6 (2012)
[8] Wolfert, Sjaak, et al. "Big data in smart farming–a review." Agricultural systems 153 (2017)
[9] Liya, M. L., and D. Arjun. "A survey of LPWAN technology in agricultural field." 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC). IEEE, 2020.
[10] Chaparro B, Fabián, Manuel Pérez, and Diego Mendez. "A Communication Framework for Image Transmission through LPWAN Technology." Electronics 11.11 (2022): 1764.
[11] Ray, Partha Pratim. "A review on TinyML: State-of-the-art and prospects." Journal of King Saud University-Computer and Information Sciences (2021)
[12] Nicolas, Chollet, Bouchemal Naila, and Ramdane-Cherif Amar. "TinyML Smart Sensor for Energy Saving in Internet of Things Precision Agriculture platform." 2022 Thirteenth International Conference on Ubiquitous and Future Networks (ICUFN). IEEE, 2022.
[13] Sanchez-Iborra, Ramon, and Antonio F. Skarmeta."Tinyml-enabled frugal smart objects: Challenges and opportunities." IEEE Circuits and Systems Magazine 20.3 (2020): 4-18.
[14] Zandberg, Koen, et al. "Secure firmware updates for constrained iot devices using open standards: A reality check." IEEE Access 7 (2019): 71907-71920.
[15] Jongboom, Jan, and Johan Stokking. "Enabling firmware updates over LPWANs." Embed. World Conf. 2018.
[16] Davcev, Danco, et al. "IoT agriculture system based on LoRaWAN." 2018 14th IEEE International Workshop on Factory Communication Systems (WFCS). IEEE,
[17] Shafique, Kinza, et al. "Internet of things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios." Ieee Access 8 (2020): 23022-23040.
[18] Arakadakis, Konstantinos, et al. "Firmware over-the-air programming techniques for IoT networks-A survey." ACM Computing Surveys (CSUR) 54.9 (2021): 1-36.
[19] Bauwens, Jan, et al. "Over-the-air software updates in the internet of things: An overview of key principles." IEEE Communications Magazine 58.2 (2020): 35-41.
[20] Zandberg, Koen, et al. "Secure firmware updates for constrained iot devices using open standards: A reality check." IEEE Access 7 (2019): 71907-71920.
[21] Ruckebusch, Peter, et al. "Modelling the energy con-

sumption for over-the-air software updates in LPWAN networks: SigFox, LoRa and IEEE 802.15. 4g." Internet of Things 3 (2018): 104-119.
[22] Jongboom, Jan, and Johan Stokking. "Enabling firmware updates over LPWANs." Embed. World Conf. 2018.
[23] Pule, Mompoloki, and Adnan M. Abu-Mahfouz. "Firmware updates over the air mechanisms for low power wide area networks: A review." 2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC). IEEE, 2019.
[24] Chaudhari, Bharat S., Marco Zennaro, and Suresh Borkar. "LPWAN technologies: Emerging application characteristics, requirements, and design considerations." Future Internet 12.3 (2020): 46.
[25] Davcev, Danco, et al. "IoT agriculture system based on LoRaWAN." 2018 14th IEEE International Workshop on Factory Communication Systems (WFCS). IEEE, 2018.
[26] LoRa Alliance : FUOTA Working Group of the LoRa Alliance Technical Committee, 2019. FUOTA Process Summary - Technical Recommendation. [online] LoRa Alliance.
[27] Catalano, Julien. "LoRaWAN Firmware Update Over-The-Air (FUOTA)." Journal of ICT Standardization (2021): 21-34.
[28] Abdelfadeel, Khaled, et al. "How to make firmware updates over lorawan possible." 2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM). IEEE, 2020.
[29] Elshabrawy, Tallal, and Joerg Robert. "The impact of ism interference on lora ber performance." 2018 IEEE Global Conference on Internet of Things (GCIoT). IEEE, 2018.
[30] "Chirpstack Open-Source Lorawan Network Server". Chirpstack.Io, 2022, https://www.chirpstack.io/.
[31] "Mbed OS — Mbed". Os.Mbed.Com, 2022, https://os.mbed.com/mbed-os/.
[32] Jongboom, Jan. "Github - Armmbed/Mbed-Os-Example-Lorawan-Fuota: Mbed OS 5 Firmware Update Over Lorawan Example Application". Github, 2022, https://github.com/ARMmbed/mbed-os-example-lorawan-fuota.
[33] "Lora Gateway — Packet Forwarder Lora — Multitech Conduit®". Multitech.Com, 2022, https://www.multitech.com/brands/multiconnect-conduit
[34] "Pycom - Next Generation Internet Of Things Platform". Pycom, 2022, https://pycom.io/.
[35] Charilaou, Christia, et al. "Firmware update using multiple gateways in LoRaWAN networks." Sensors 21.19 (2021): 6488.
[36] Heeger, Derek, et al. "Secure LoRa firmware update with adaptive data rate techniques." Sensors 21.7 (2021): 2384.
[37] Sharf, Samy H., et al. "An Efficient OTA firmware updating Architecture based on LoRa suitable for agricultural IoT Applications." 2021 International Conference on Microelectronics (ICM). IEEE, 2021.
[38] Ren, Haoyu, Darko Anicic, and Thomas A. Runkler. "Tinyol: Tinyml with online-learning on microcontrollers." 2021 International Joint Conference on Neural Networks (IJCNN). IEEE, 2021.
[39] Kwon, Jisu, and Daejin Park. "Hardware/Software Co-Design for TinyML Voice-Recognition Application on Resource Frugal Edge Devices." Applied Sciences 11.22 (2021): 11073.
[40] Sudharsan, Bharath, et al. "Ota-tinyml: over the air deployment of tinyml models and execution on iot devices." IEEE Internet Computing 26.3 (2022): 69-78.
[41] Moreau, L. and Kelcey, M., 2022. Announcing FOMO (Faster Objects, More Objects). [online] Edge Impulse. Available at: https://www.edgeimpulse.com/blog/announcing-fomo-faster-objects-more-objects
[42] Aadithya, V., et al. "Comparative Study Between MobilNet Face-Mask Detector and YOLOv3 Face-Mask Detector." Sustainable Communication Networks and Application. Springer, Singapore, 2022. 801-809.
[43] Devalal, Shilpa, and A. Karthikeyan. "LoRa technology-an overview." 2018 second international conference on electronics, communication and aerospace technology (ICECA). IEEE, 2018.
[44] Chollet, N., 2022. GitHub - LoRaWAN-FUOTA-TinyML . [online] GitHub. Available at: https://github.com/NicolasChollet51/LoRaWAN-FUOTA-TinyML
[45] Mészáros, Levente, Andras Varga, and Michael Kirsche. "Inet framework." Recent Advances in Network Simulation. Springer, Cham, 2019. 55-106.
[46] Slabicki, Mariusz, Gopika Premsankar, and Mario Di Francesco. "Adaptive configuration of LoRa networks for dense IoT deployments." NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2018.