Research Article

Open Access

Evaluation of the energy viability of smart IoT sensors using TinyML for computer vision applications: A case study

Abstract

TinyML technology emerges from the intersection of Machine Learning, Embedded Systems, and Internet of Things (IoT), and presents itself as a solution for various IoT fields. For this technology to be successfully applied to embedded devices, it is essential that these devices have adequate energy efficiency. To demonstratethe viability of TinyML technology on embedded devices, field re-search and real experiments were conducted. An embedded systemwas installed in a turnstile of a Federal Institute, in which a TinyMLcomputer vision model for people detection was implemented. Thedevice counts the number of people, analyzes the battery level,and sends data in real-time to the cloud. The prototype showed promising results, and studies were conducted with a lithium battery and three in series. In these experiments, voltage consumptionwas analyzed every hour, and the results were presented through graphs. The camera sensor prototype had a consumption of 1.25 volts/hour, while the prototype without the camera sensor showed a longer-lasting consumption of 0.93 volts/hour. This field researchwill contribute to the advancement of applications and studies related to TinyML in conjunction with IoT and computer vision.

Keywords: TinyML, Iot, embedded systems, computer vision

Volume 9 Issue 2 - 2023

Adriel Monti De Nardi,[1] Maxwell Eduardo Monteiro[2]

[1]Mestrado Profissional em Computação Aplicada (MPCA), Brazil
[2]Mestrado Profissional em Computação Aplicada (MPCA), Brazil

Correspondence: Maxwell Eduardo Monteiro, Mestrado Profissional em Computação Aplicada (MPCA) Serra ES, Brazil, Email maxmont@ifes.edu.br

Received: May 18, 2023 | **Published:** May 26, 2023

Introduction

In the current scenario of the digital world, there is a wide variety of technological tools available, resulting in a growing multitudeof connected devices on a computer network. The Internet is a global network that connects millions of computers,[1] and its technological advancement has transformed society over the years.[2]

It is believed that the Internet is advancing towards an era of pervasive computing, which is driving the potential for growth of the Internet of Things (IoT).[3] Among the various issues involved in implementing IoT solutions, energy efficiency stands out. Ensuring efficient energy consumption in IoT devices is an essential requirement for ensuring the longest possible device lifespan.

With the advent of Tiny Machine Learning (TinyML), this technology becomes a possibility for applications in the field of IoT. The differential of TinyML is the application of machine learning in small embedded devices.[4] These devices are usually equipped with a microcontroller and sensors, which collect data that, along with the microcontroller's processing, can affect the battery life, becoming a challenge for these devices.[5] Thus, the followingresearch question arises:

What is the energy feasibility of TinyML technology applied in embedded devices in an intelligent IoT scenario using computer vision?

To achieve this objective, the purpose of this research is to select a TinyML computer vision solution and conduct a battery life study on microcontrollers in an IoT scenario. Additionally, analyzing the performance of lithium battery consumption may be relevant for future research in IoT projects. Currently, researchers and experts are striving to improve TinyML-based IoT for smart devices.[6]

In this work, Section 2 presents the concepts of IoT and TinyML, the advantages and challenges of embedded devices, the frame works used in the construction of the TinyML model, and the related works to this field research. Section 3 provides the related works of this research. Section 4 describes the research methodology, including the study design, the materials used, and the development of the prototype. Section 5 presents the research results through graphs and discusses the data and the scenario set up. Section 6 presents the conclusions and suggestions for future work. Finally, in Section 7, the studied bibliographic references are listed.

The results obtained in this research were considered satisfactory, as shown in Section 5 of Results and Discussion, since the TinyML application with a computer vision model consumed only 7.5V in six hours of use, and there are possibilities for improvements basedon the future work presented in this study in Section 6.

Theoretical reference

The increase in data production, along with processing and storage of information, has led to a widespread adoption of technological improvements.[4] Among these improvements, the adaptation of Machine Learning (ML) to the IoT field stands out. In Section 2.1, we briefly discuss IoT and TinyML technology. In Section 2.2,we explore the advantages and challenges present in embedded devices. Finally, in Section 2.3, we discuss the frameworks available in the market for computer vision applications.

IoT and Tinyml concept

The field of Machine Learning (ML) can be defined as a subset within Artificial Intelligence (AI). Its function is to build mathematical models based on training data to make predictions and decisions without needing to be explicitly programmed to perform a specific task.[7] On the other hand, the Internet of Things (IoT) consists of interconnected devices with sensors that collect data and share information through public or private networks.[3] When these two technologies are combined, they allow for the adaptation of the ML environment to the IoT field. The increasing production and processing of data have made these technologies more important and contributed to their constant improvement.[4]

The IoT technology is increasingly present in our daily lives, including in Smart City projects.[8] A recent example occurred in the city of Christchurch, New Zealand, after an earthquake in 2011. The city was rebuilt with a Smart City solution, where sensors were installed to collect real-time data on various aspects, such as trafficflow and water quality. This provided greater urban efficiency and productivity for the city, and the cost of implementing these IoT devices was relatively low.[9] This technology can be further enhanced with the incorporation of Machine Learning (ML), which allows devices to perform tasks, collect data, and share information without needing to be explicitly programmed to perform a specific task.[7]

The AI embedded in the field of IoT allows data to be collected through sensors (such as temperature, camera, and audio) powered by battery and operating at low power (1mW or less). With the trend of small machine learning, TinyML emerged in 2019. This feature uses limited processing, memory, and energy, with the aim of targeting a wide variety of microcontroller types and offering various applications and solutions in the world.[4]

Advantages and challenges of embedded devices

TinyML is used in embedded devices, as highlighted by Soro 2021,[4] which emphasizes the advantages and challenges of these devices in small machine learning. Among the advantages, we can mention latency, energy efficiency, low cost, privacy, connectivity, and autonomy. Regarding latency, the application runs within the device itself, reducing the need for communication with remote servers and improving real-time response. In addition, energy efficiency allows these devices to operate for long periods of time without the need for battery replacement, which is crucial in many IoT applications.

Energy efficiency in embedded devices is a critical factor, as these devices are designed to perform calculations with low power consumption, aiming to operate for a prolonged period with a limited energy budget. Additionally, the low cost of embedded devicesallows for the expansion of IoT applications in various segments, making them a viable option for many solutions. Privacy is ensured, as the code is executed directly on the embedded device, without the need to share sensitive data with external servers. Finally, the small size of embedded devices contributes to their autonomy, allowing them to operate independently and with greater durability. However, despite these advantages, embedded devices face significant challenges, such as limitations in hardware resources, such as memory, storage, and processing power, which make the execution of complex ML models based on neural network systems a challenge.

One of the main challenges of embedded devices is the execution of complex ML models based on neural network systems, due to hardware constraints in terms of memory, storage, and processing. It is difficult to deploy neural networks on microcontrollers with such restrictions. To address this challenge, ML models must be small enough to fit onto embedded devices, requiring the use of frameworks to design the compression of learning onto devices through a neural network-based model.

Another limitation to be considered is battery life, as energy efficiency depends on the computational cost, ML algorithm, and processing cycle, which can vary according to the model's power on the embedded device. Additionally, heterogeneous devices may have different energy and consumption measures, making comparison between them difficult. Another relevant factor is the variationin the lifespan of the lithium batteries used, making it difficult to compare battery lifetimes. Furthermore, each device may use a different way of processing data, with different additional peripherals, making it challenging to measure energy consumption accurately. Therefore, energy consumption measurements can vary significantly from device to device, according to their development and quantity of peripherals used.

Tinyml framework for computer vision application

To build TinyML models and accelerate application deployment, there are several available frameworks such as TensorFlow Lite, ELL, and ARM-NN. All of these frameworks use a similar workflow, which involves model conversion and optimization for deployment on a specific hardware platform. This optimization process is important to ensure that the model can be efficiently executed on deviceswith limited memory and processing resources. By converting and optimizing the model, its size and complexity can be reduced, making it suitable for deployment on embedded devices. Moreover, these frameworks also support multiple programming languages and hardware architectures, allowing developers to create TinyML applications for a wide variety of devices and platforms.

Due to the limited resource constraints of MCU devices, they do not have the capacity to support Machine Learning model training. Instead, the model is trained in the cloud or on more powerful com puting devices, and subsequently deployed to the embedded device. [10]Among the most widely used frameworks for implementing TinyML models, TensorFlow Lite stands out, developed by Googleand featuring a specialized ML library for small mobile devices.[11] Currently, the TinyML field is primarily focused on applications involving computer vision.

Related works

According to Kravari and Chatzimisios,[12] billions of devices are currently connected to the Internet of Things (IoT), most of them being battery-powered. IoT is applied in various areassuch as Smart Cities, Smart Energy, Smart Environment, among others, each of which relies on battery-powered devices, where optimizing their lifespan provides a significant improvement in the IoT scenario. However, accurately measuring the battery chargeon such devices becomes a challenge. Therefore, Jolly and Gupta[13] assert that it is crucial to understand in detail howthe device consumes energy and adjust its charge according to the consumption to ensure a longer battery life.

According to Banbury et al. 2020,[14] in Figure 1, it is possible to compare active energy consumption between traditional ML devices and those supported by TinyML. On the horizontal axis, the NDP100 device (150uW consumption) and Cortex-M7 (23.5mW consumption) belong to the TinyML system, while the RasPi 4 (3.4Wconsumption) and 6049GP-TRT (2000W consumption) belong to the MLPerf systems. Through the vertical axis, it is possible to see that TinyML systems can save up to four times more energy compared to state-of-the-art energy consumption management systems. This shows that using TinyML-supported devices can bring significant advantages in terms of energy efficiency when compared to traditional ML devices.

The TinyML is capable of running computer vision models and can be applied in various research areas, as highlighted by the study conducted by Khan et al.[15] The areas were divided into several fields such as Weather Forecasting, Agriculture, Medical Sciences, Professional Sports, and Facial Recognition, each accounting for 6%; Industries accounting for 12%; Traffic Levels in Cities and Professional Sports each accounting for 13%; and finally Biological Sciences and Human Activities representing the majority of the division of fields, each accounting for 19%.

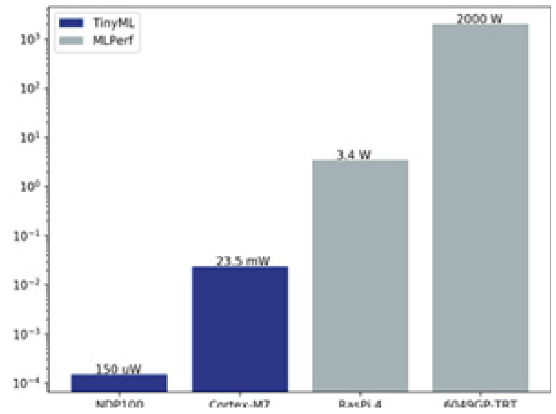

Figure 1 A comparison of active power consumption between ML systems and those supported by TinyML by Banbury et al. 2020.[14]

Regarding the use of Machine Learning (ML) in computer vision applications, it has been successful in several areas such as weather forecasting, biological sciences, facial expression reading, food safety, species classification, sports, traffic flow monitoring, and predictive maintenance in industries.

Although areas such as biological sciences, human activity interpretation, traffic management, and professional sports are still emerging, object detection, classification, and prediction are the most frequent uses of ML in computer vision, as concluded in a recent study.

It is important to highlight that all the areas mentioned in the research can be applied in the context of the Internet of Things (IoT), reinforcing the importance of studies to improve the energy efficiency of these applications.

An example of work that addresses energy management and conservation in IoT devices is the study by Srinivasan et al. 2019.[16] In this study, intelligent and sustainable technology is implemented in IoT using the methodology of Smart Plugs. These are devices that connect to the internet through Wi-Fi networks, allowing the remote control of some home appliances.

The system is designed to capture data that is transferred through wireless gateways to a central database. Additionally, the methodology allows the Smart Plugs to be remotely turned on or off when not in use at the server center. This approach offers an efficient way to manage the energy consumption of IoT devices, making them more sustainable and economical.

Furthermore, the system provides an analysis of the amount of energy saved compared to the total energy consumption before the implementation of the system. This analysis allows the verification of the impact of the technology on energy savings. Based on the data collected by the Smart Plugs, it is possible to forecast the total energy consumption based on consumer behavior.

A web application was developed so that users can control these intelligent devices. The application offers a user-friendly interface to manage connected devices and monitor real-time energy consumption. This approach makes energy management more convenient for users, allowing them to control their devices and save energy efficiently.

The objective of this work is to explore the use of Smart Plugsin an office environment for energy conservation and load management. To collect data, a REST API web service is used. Sensor datais sent to a server using a wireless communication gateway and stored in a database.

In real-time, data is monitored using a wireless-smart plug, a wireless gateway, a server installed in the environment, and a dis- play. This approach allows for real-time energy consumption management, which is essential to ensure that energy conservation strategies are effective. Additionally, the use of Smart Plugs pro- vides an easy and efficient way to manage plug loads, allowing users to control their devices and save energy intelligently.

The methods were experimented and tested in an office environ ment, using a minimum amount of devices. Prior to implementing the energy conservation system, a six-month calculation period was established. To estimate the energy conservation system's efficiency in future time series, a time series analysis involving the use of statistics was performed.

The energy conservation system used two tests: scheduling programming and a chatbot system. The total energy consumption result was excellent. This solution suggests a scalable approach for various IoT application environments, where the objective is energy management and savings. The use of Smart Plugs and similar technologies can be applied in various settings to optimize energy efficiency and reduce unnecessary energy consumption.

The study presented in[13] addresses the behavior of the battery in IoT devices, showing the charge levels on a vertical axis from 0 to 100 percent. The time and charge spent by the device at a certain level are presented, highlighting that 5 percent of the time and 8 percent of the charge are consumed when the level is above 900A. As the voltage supplied by the battery decreases, the device consumes the charge differently, which is crucial to understand the end of the battery life. The study used the battery ofKeysight X8712A IoT device and a sensor operating at 3.3V, taking measurements every 1,023 seconds at intervals of 0.2V, from 3.3V to 1.9V. The charge consumption varied slightly between 3.3V and 2.3V, increasing below 2.3V and more significantly at 1.8V, with an increase of 1Ah.

This study can be used by device designers to make important decisions in optimizing IoT devices, improving customer satisfaction. Some ways to extend the battery life include turning off the real-time clock and using the RC circuit as an alarm, among other features. Knowing how the device operates at different voltage levels near the end of the battery life is essential for increasing the efficiency and durability of the device.

IoT devices often have sensors, and firmware programmers can extend battery life by disabling or reducing the frequency of less important measurements. Another way to extend battery life is to reduce data transmission. IoT devices consume battery power to transmit and receive data, so reducing the transmission frequency can be a good choice, such as sending data every hour instead of every minute. Additionally, reducing the amount of data transmitted when the battery is low can alert users that it's time to replace the battery.

If the battery is running low, it's also advisable to disable functions such as cloud data sending and Wi-Fi to preserve the remaining charge. As the number of battery-powered devices grows, the need for insights to improve battery life becomes increasingly important. By using event-based analytics, engineers can obtain valuable information about battery end-of-life and device behavior with the aim of adding value to end-users. In summary, optimizing battery usage is crucial to ensuring device performance and user satisfaction.

Method

In this section, we will present the methodology used in our field study. In Section 3.1, we will describe the design of the proposed study, while in Section 3.2, we will discuss the hardware used for the development of the prototype. In Section 3.3, we will address the software used to develop the model and the entire mechanism that allowed for the implementation of the embedded system and data transmission. Finally, in Section 3.4, we will detail how the prototypes were assembled, highlighting their differences.

Study design

In the first prototype, we used an Esp32cam with computer vision application. When the camera sensor of Esp32 detects a person, an LED is triggered. Additionally, we incorporated resistors and an Esp8266 microcontroller designed to measure the charge level of a single lithium battery that powers the system. This prototype was deployed at the designated location.

In the second prototype, in addition to Esp32cam and Esp8266, we also used the same LEDs and resistors. However, we added three lithium batteries to extend the application's usage duration. Since the lithium batteries were connected in series, it was necessary to use a voltage regulator to reduce the supplied voltage to the circuit. More details about the prototypes are presented in detail in Chapter 4.4.

To obtain the results, we selected the Federal Institute of Espírito Santo (IFES) as the location for the field test due to high foot traffic. The prototype was installed at the entrance, between the turnstiles, to allow access to Campus Serra.[17]

Through an experiment, steps were performed in the Machine Learning model development flow to enable the prototype to run its experiment in the case study. To address each step of this ML model development flow until its implementation on an embedded device, the following steps were carried out:

Selection of a computer vision model trained by the Tensor-Flow framework; Conversion of the model, leveraging an available pre-trained TensorFlow Lite model for this specific computational version already available on the Espressif GitHub; Execution of the model inference on the microcontroller using the Espressif Framework, ESP-IDF, responsible for bringing intelligence to the embedded system; Optionally, an optimization step was performed on the model, customizing it to count people and send data to the cloud by activating an ESP-CAM output port. The C++ language was used in this optimization step. Subsequently, the model inference was executed on the microcontroller via a Linux terminal.[11]

Hardware

In the development of the prototype, two microcontrollers were used: the ESP-32 with a camera sensor module and the ESP8266. In addition to the microcontrollers, the assembly of the prototype involved the use of other components and materials, such as a proto-board to interconnect the microcontrollers, a multi-meter to evaluate the voltage of the batteries, 18650 lithium batteries to power the system, a lithium battery charger module, a voltage regulator to keep the voltage below 5V, a LED to indicate the detection of people, resistors, and jumpers. To support the hardware, the entire project was mounted on a wooden structure. Additionally, a case was used to protect the batteries during use.

ESP32-CAM: The ESP32 CAM board is a low-cost microcontroller equipped with Wi-Fi and camera sensor, ideal for projects involving computer vision, IP cameras, and video streaming. Sup- porting both OV2640 and OV7640 cameras, along with built-in flash, it has a low-power 32-bit CPU and a clock speed of 160MHz. It has various built-in sleep modes and a CAM SRAM board, enabling easy implementation of low-power projects. Its Wi-Fi connectivity al- lows for live data visualization with high-quality, real-time images. Encoding can be done using the FTDI cable or the Module B (also called shield), which provides a micro USB port for easier micro- controller programming, as well as additional hardware protection. With all these features, the ESP32 CAM board is an excellent option for IoT projects and others that require a microcontroller with computer vision capabilities.[18]

ESP8266: In the proposed project, the NodeMCU is used to capture the battery level at the time of person detection, total people count, and the runtime of the application. The NodeMCU is a low-cost microcontroller that runs firmware on the ESP8266 WiFi SOC, developed by Espressif Systems. This hardware is an open- source platform and can be worked with IoT due to its features. The NodeMCU is based on the ESP-12 module and has 16 digital I/O pins and one analog pin. Additionally, it has Wi-Fi connectivity features and operates at a voltage of 3.3V. Using the Wi-Fi connection, it sends this data through a gateway to the network, where the database receives them in real-time, allowing them to be viewed through a computer or smartphone.[18]

Battery charger 18650: The loading module used is ideal for ESP projects, allowing the recharge of a rechargeable 18650 battery through a micro USB or type C input. In addition, the module has 3v and 5v outputs to power external projects and a 5v female USB output. It is worth noting that the model used only accommodates one battery, however, there are options from manufacturers that support two batteries.

The product specifications include: Shield model, micro USB input voltage from 5V to 8Vdc, charging current of 0.5A (500mA), output 5V up to 4A, output 3V up to 1A, and charging indicator LEDs that blink in red during charging and remain on when charging is complete. The module is compatible with 18650 type batteries andhas an on/off switch for the USB type A output.

However, it is important to be careful with the positive and negative polarities of the product, as incorrect assembly may cause the components on the charger board to burn out. The module dimensions are 107x30x20mm and its weight is 23g.

Voltage regulator: The LM2596 voltage regulator module is a Step-Down DA-DC converter with excellent efficiency and additional features such as a voltmeter display. With a wide input range of 4 to 40V, this module can provide an adjustable output voltage between 1.25 and 37V, making it an extremely useful component in electronic projects that require high-efficiency voltage regulation. Additionally, the presence of the voltmeter display allows for easy real-time monitoring of the output voltage, ensuring regulation accuracy.

Software

The project utilizes a combination of software to achieve its objectives. TensorFlow Lite is one of the main components as it isresponsible for converting Machine Learning models into TinyML, allowing for their implementation on low-power devices such as the ESP32-CAM used in this project. Additionally, the ESP-IDF framework is used to bring the technology to the ESP32-CAM. The collected data is stored on Firebase, an app development platform that offers real-time cloud storage and analysis. Github was usedto access the source code of the proposed Computer Vision model. Visual Studio Code, with support for the programming language used in the project, was used for programming and code modification. It is worth noting that the operating system used for the project development was Linux Ubuntu.

Tensor flow lite: TensorFlow Lite is an open-source Deep Learning framework developed by Google with the aim of enablingthe execution of ML models on embedded devices, such as micro- controllers with limited storage capacity. It has been tested on the ESP32 processor.[4] This tool provides a set of machine learning resources for developers who want to run models on mobile, embedded, and IoT devices. TensorFlow Lite is compatible with various microcontroller platforms and software, including Android, iOS, and Linux, and can be used in several languages, such as Java, Swift, Objective-C, C++, and Python.[11]

Framework ESP-IDF: According to Espressif's official documentation, the Espressif IoT Development Framework (ESP-IDF) is an application development framework intended for the System-on-Chips (SoCs) of the ESP32, ESP32-S, and ESP32-C series. The use of ESP-IDF allows the configuration of the software development environment for hardware based on the ESP32 chip, as well as enabling the modification of the board on which the application will be used. It is possible to customize the menu of sensors andapplications, as well as build and update the firmware on an ESP32 board according to the specific needs of the project.[18]

Firebase realtime database: The Firebase Realtime Database is a cloud-hosted database developed by Google. It stores data inJSON format and enables real-time synchronization. The data isavailable for access both online and offline through different platforms, such as Arduino, Apple, Android, and JavaScript. All clientsshare an instance of the Realtime Database, receiving automatic updates with the latest data.

This database is based on a NoSQL, non-relational model. This means that there is no need to establish a relationship between classes and the database.[19] The Firebase Realtime Database is a powerful option for developers who want to create multi-platform applications and keep data synchronized in real-time across different devices and platforms.

Prototype

For the computer vision application, a system capable of recognizing people through a camera sensor was chosen. The code was modified to count the number of people detected and activate an output door indicating that the person was identified. This data is then sent to a control system that combines it with the current battery level and sends it in real-time to a database. The TinyML system and thecontrol system are interconnected through a protoboard, while the lithium battery provides all the necessary power for the system.

The prototype development was divided into two steps. Firstly, the computer vision application was developed using TinyML, which detects and counts people through a camera sensor using artificial intelligence. Then, a controller was developed that sends the data over the network and saves it in a non-relational database. With this system, it is possible to monitor the presence of people in a given space and collect valuable information for analysis and decision making.[4]

As shown in Figure 2, we can see how the first prototype was assembled, using a single battery module. From there, improvements were made for a second prototype, as shown in Figure 3, where a case with three batteries in series and a voltage regulator was used to power the circuit at 5v.

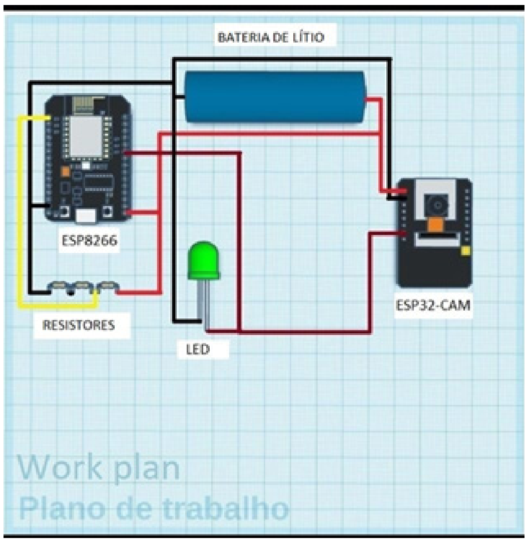

Figure 2 Prototype work plan 1.

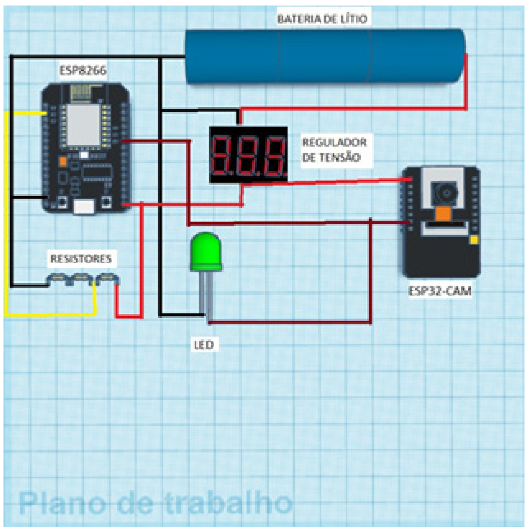

Figure 3 Prototype work plan 2.

TinyML Model for Computer Vision: Espressif offers resources for IoT application development, which made it a suitable choice for this work in conjunction with the TinyML model for computer vision. To begin development, it is necessary to install and con- figure the ESP-IDF framework, following the programming guide available on the Espressif website.[20] Before that, however, it is important to install some compilation tools, such as CMake and Ninja, as well as Git to clone the chosen model. The ESP-IDF framework, version v4.1 release, is run through the command line of the Linux Ubuntu 20.04 LTS operating system, which is the most recent stable version at the time of development.[21] After incorporating the code through ESP-IDF, the camera components and the TensorFlow Lite library are already configured.[22]

The TinyML model for computer vision was chosen to detect people through a camera sensor. The author of this work implemented a counter and a digital port activation to communicate withthe ESP8266 board during people detection. After optimizing the code, the project is built and can be monitored directly through theterminal.[23]

Data controller: The Arduino IDE was used to develop the code responsible for controlling and sending system data. When the computer vision application detects a person and performs the counting, the ESP32-CAM pin output is triggered and an LED is turned on to indicate that detection has occurred. Then, the ESP8266 microcontroller receives the information and stores the count of detected people, in addition to having a voltage divider to calculate the level of the lithium battery. The battery level value is obtained through an analog pin, and to monitor the prototype's operating time, a seconds count was implemented, which starts as soon as thesystem is executed. Finally, the ESP8266 microcontroller sends thedata through its WiFi connection to a gateway that forwards them to the network, allowing real-time access to the collected data.[24]

Results and discussion

This section presents the results obtained with the proposed study. To analyze the data collected in the non-relational database, the high-level programming language Python[25] and the Google Co-laboratory(Colab) tool[26] were used. Based on this data, graphs were plotted for a better visualization of the results. It is worth noting that the 18650 model batteries used in the prototype have a capacity of 3000 mAh and a maximum current of 30 A.

First prototype: In this prototype, we used a single 18650 battery connected to the lithium charger module. Since the circuit requires 5V, the battery has 4.2V, but the module circuit has a 5V output, which can powerthe prototype circuit this way. As can be seen in Figure 4, 281 people were detected on the x-axis, and on the y-axis, the battery level consumed. The cycle detected a total of 281 people in a time of 2 hours and 21 minutes. In the next prototype, we will use a multi-meter to obtain more research data on battery power consumption during circuit use.

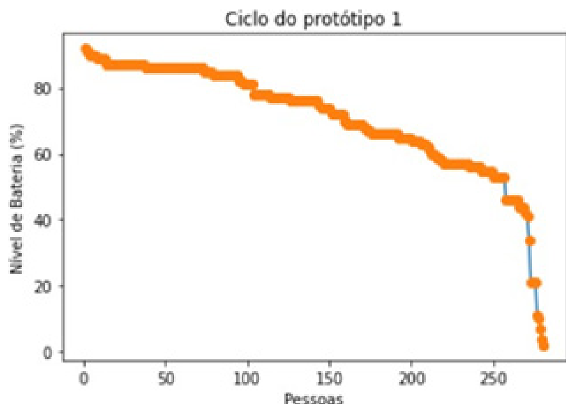

Figure 4 Prototype Cycle 1 - Axis x (People) / Axis y (BatteryLevel).

Second prototype: In this prototype, three 18650 battery models were used, connected in series through a battery case, allowing an output of up to 12V. To power the circuit, a voltage regulator was used to convert the 12V voltage to 5V. Two tests were performed, one with the camera sensor module connected to detect people, and another without the camera sensor module to compare battery consumption. A graphwas also generated to analyze the consumption of the three batteriesover a six-hour interval, which was the maximum time the circuit with the camera sensor module was able to support, generating significant results.

With camera sensor module: From the analysis of the graph presented in Figure 5, it is possible to observe that on the horizontal axis we have the number of people detected, while on the vertical axis we have the level of battery usage. The graph shows that the prototype detected 337 people over a period of 6 hours. In Figure 6, it is possible to observe the time interval on the horizontal axis and, on the vertical axis, the initial voltage of the 3 batteries. Analyzing the behavior of the batteries, it is possible to verify that battery 3 had its consumption drastically altered between the 2nd and 3rd hour of prototype use. To obtain these results, all batteries were measured every hour using a multi-meter. Initially, when the circuit began to be powered, the three batteries in series summed 11.81 V. At the end of the test, the batteries totaled 4.31 V.

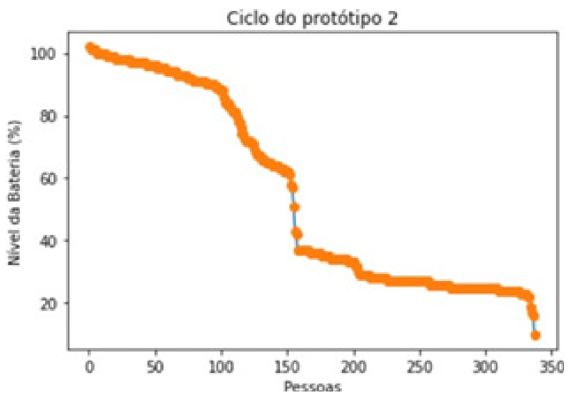

Figure 5 Prototype Cycle 2 - Axis x (People) / Axis y (BatteryLevel).

Without camera sensor module: In this test, no people weredetected, so it was not necessary to generate a counting graph. However, in Figure 7, a graph of the voltage consumption of thebatteries in series without the camera sensor module was presented. On the x-axis, we have a time interval of up to 6 hours, and on the y-axis, the initial voltage of the 3 batteries, allowing for visualization of their behavior. Battery 3 had its consumption changed between 4and 5 hours of prototype use, a longer time compared to the graph in Figure 6. All batteries were measured every hour using a multi-meter. Initially, when the circuit began to be powered, the three batteries in series totaled 11.86 V. In the end, the batteries presented a total voltage of 5.58 V.

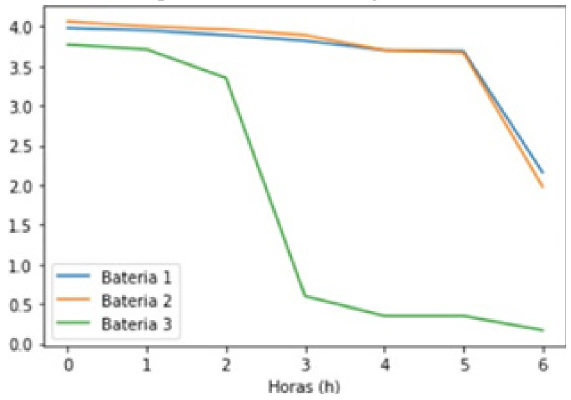

Figure 6 Batteries with camera sensor - X-axis(Hours) / Y- axis(Battery voltage).

Final considerations: Firstly, it is important to highlight that accurately measuring the consumption of a lithium battery is a challenge that has been addressed by various researchers, such as Kravari et al. 2017[12] and Jolly 2019.[13] To overcome this difficulty, we used a multimeter to obtain more precise measurements from prototype 2, allowing for a more accurate analysis of the consumption of the TinyML modelin conjunction with the circuit and microcontrollers. In the first test with prototype 1, no significant changes were observed in itsbehavior, as shown in the graph of Figure 4.

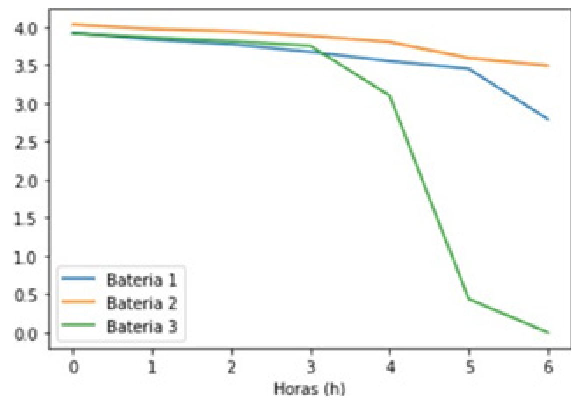

Figure 7 Batteries without camera sensor - X axis(Hours) / Y axis(Battery voltage).

The proposed TinyML model has a relevant accuracy for people detection. However, the objective of this work was not to precisely measure people detection, but rather to evaluate how TinyML, in conjunction with a computer vision application, behaves in the context of IoT in terms of battery consumption. Thus, our focus was to present data on the lifespan of the lithium battery that powers the entire embedded device.

Considering that a 3000mAh, 30A model 18650 lithium battery has a maximum voltage of 4.2V and a minimum of 3.7V, we analyzedtwo different cases with prototype 2. In the first case, where the camera sensor was used, the batteries in series started with 11.81Vand ended with 4.31V after 6 hours of use, totaling a consumption of 7.5V, or 1.25 volts/hour. In the second case, where the camera sensor was not used, the batteries started with a total of 11.86V andended with 6.28V after 6 hours, totaling a consumption of 5.58V, or 0.93 volts/hour. These data indicate that the camera sensor has a significant energy expenditure in a computer vision application with machine learning.

Based on the results obtained, we can conclude, according to Banbury et al. 2020,[14] that TinyML presents much higher energy efficiency in microcontrollers with low processing, memory, and storage. Compared to large-scale machine learning models, this experiment would be unfeasible. Although the application serves its functions for several hours, improvements can be proposed to further increase the battery life in an IoT scenario. An interesting solution to this problem was presented by Srinivasan et al. 2019,[16] who propose energy management and conservation strategies. Thus, we suggest some improvements in future works for thisstudy.

Conclusion and future work

Although existing literature on TinyML points to the possibility of its use in IoT devices, no studies were found that prove its energy viability as an IoT device. Given this gap, this study was designed to evaluate the energy viability of Smart IoT sensors using TinyML for computer vision applications in a real-world setting. TinyML tech-nology emerges as a promising solution to reduce memory, storage, and computational processing in IoT devices. Additionally, com- puter vision applications in IoT environments have great potential in various areas, as previous studies have pointed out. Field research conducted at a Federal Institute presented the voltage expenditures of prototype 2 and allowed the identification of improvements to be implemented to advance the use of TinyML technology. To increase battery life in the use of the application, it is suggested to add an Arduino Uno that receives 12 V and sends 5 V to the microcontroller, along with the data that should be sent every 10 minutes. During that time, the data controller goes into a deep sleep state. Another suggestion is to power the circuit with a solar kit. These ideas of turning off resources in the microcontroller during a time interval, such as Wi-Fi, which has high power consumption, are suggested inprevious studies. With these improvements, a longer battery life is expected, contributing to the advancement of TinyML technology in the IoT field.

Acknowledgments

I would like to express my sincere gratitude for the incredible teach ing I received during my academic journey. I know that everything I have learned would not have been possible without the commitment, dedication, and passion that you put into your classes every day. The knowledge I have gained thanks to you is something I willcarry with me forever.

Additionally, I would like to thank FAPES for granting me the scholarship that made it possible for me to pursue my master's degree at IFES. Without this opportunity, I would not have had accessto all the learning and academic community I have encountered.

Finally, I would like to thank IFIP-IOT for giving me the chanceto publish my research work. It is an honor to be able to contributeto the academic community and I hope that my work can be usefulto those who read it.

Conflicts of interest

The author declares that there was no conflict of interest.

References

1. Vangie B. *What is the internet?*. Webopedia; 2021.
2. Elisabete Zimmer F, Adriane Maria NO, Silvana Possani M, et al. A influência da internet na saúde biopsicossocial do adolescente: revisão integrativa. *Revista Brasileira de Enfermagem*.2020;73(2):e20180766.
3. Minhaj Ahmad K, Khaled S. IoT security: Review, blockchain solutions, and open challenges. *Future Generation Computer Systems.* 2018;82:395–411.
4. Stanislava S. *TinyML for Ubiquitous Edge AI.* arXiv:2102.01255 [cs.LG]; 2021.
5. Chandrasekar V, Anitha I, Sharat K, et al. *Democratization of AI, albeit constrained IoT Devices & Tiny ML, for creating a sustainable food future*. In 2020 3rd International Conference on Information and Computer Technologies (ICICT); IEEE. 2020;525–530 p.
6. Hiroshi D, Roberto M, Jan H. Bringing machine learn ing to the deepest IoT edge with TinyML as-a-Service; 2020.
7. Xian DZ. Machine learning. *In A Matrix Algebra Approach to Artificial Intelligence*: Springer. 2020;223–440.
8. Felix W, Kristina F. Internet of things. *Business & information systems engineering*. 2015;57(3):221–224.
9. Sarah. Lamppost shines a light on smart cities; 2015.
10. Partha PR. A review on TinyML: State-of-the-art and prospects. *Journal of King Saud University-Computer and Information Sciences*. 2021;34(4): 1595–1623.
11. Tensor Flow Google. *TensorFlow Lite*; 2022.
12. Kalliopi K, Theodoros K, Papadimopoulos AN. *Towards an IOT-enabled intelligent energy management system.* In 2017 18th International Symposium on Electromagnetic Fields in Mechatronics, Electrical and Electronic Engineering (ISEF) Book of Abstracts. IEEE. 207;1–2.

13. Brad J. *The last thing IoT device engineers think about: End of battery life behavior for IoT devices*. In 2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS); IEEE. 2019:837–840.
14. Colby RB, Vijay JR, Max L, et al. Benchmarking TinyML systems: Challenges and direction. arXiv preprint arXiv:2003.04821; 2020.
15. Abdullah AK, Asif AL, Shafique AA. Machine learning in computer vision: A review. EAI endorsed transactions on Scalable Information Systems. 2021;8(32):e4–e4.
16. Ashwin S, Krishnamoorthy B, Grynberg Y. *IoT based smart plug-load energy conservation and management system*. In 2019 IEEE 2nd International Conference on Power and Energy Applications (ICPEA); IEEE. 2019:155–158.
17. ifes. Ifes. Ministério da Educação; 2022.
18. Abhinab S, Ritesh D. IOT Based load automation with remote access surveillance using ESP 32 CAMand ESP 8266 module. *Annals of the Romanian Society for Cell Biology.* 2021;25(3):6904–6914.
19. Firebase Google. *Firebase Realtime Database*; 2022.
20. Espressif. Espressif ESP-IDF; 2022.
21. ubuntu. About the ubuntu project; 2022.
22. Chugh S. Github de S. Chugh; 2021.
23. Chugh S. TensorFlow Blog. *Announcing TensorFlow Lite Micro support on the ESP32*; 2021.
24. Ez Contents blog E. *ESP8266 battery level meter*; 2021.
25. Python P. Welcome to Python; 2021.
26. Google. G, Google Colab.