Digital Twin for Machine-Learning-based Vehicle CO_2 Emissions Concentration Prediction in Embedded System

David Tena-Gago, Mohammad AlSelek, Jose M. Alcaraz-Calero, Qi Wang
School of Computing, Engineering & Physical Sciences
University of the West of Scotland
Paisley, United Kingdom
{david.tena, mohammad.alselek, jose.alcaraz-calero, qi.wang}@uws.ac.uk

***Abstract*—In this paper, we describe the design, implementation, and installation of a digital twin version of a physical CO_2 monitoring system with the aim of democratizing access to affordable CO_2 emission measuring and enabling the creation of effective pollutant reduction strategies. The presented digital twin acts as a replacement that enables the measuring of CO_2 emissions without the use of a physical sensor. The exhibited work is specifically designed to be installed on a low-powered Micro Controller Unit (MCU), enabling its accessibility to a broader base of users. To this end, an optimized Artificial Neural Network (ANN) model was trained to be capable of predicting CO_2 emission concentrations with 87.15% accuracy when performing on the MCU. The ANN model is the result of a compound optimization technique that enhances the speed and accuracy of the model while reducing its computational complexity. The results outline that the implementation of the digital twin is 86.4% less expensive than its physical CO_2 counterpart, whilst still providing highly accurate and reliable data.**

Keywords*—*ANN, CO_2, Embedded systems, Micro controllers

I. INTRODUCTION

The global challenge of reducing Carbon Dioxide (CO_2) emissions has become an urgent priority for governments and businesses worldwide. Despite the growing awareness of the devastating effects of greenhouse gases on the environment, the implementation of effective CO_2 emission reduction strategies remains a significant challenge. One of the major obstacles is the lack of accessible and affordable CO_2 sensing/monitoring tools, which are essential for implementing and evaluating emission reduction mechanisms and policies.

Traditionally, CO_2 emission measuring equipment has been expensive and invasive, requiring specialized knowledge and expertise to install and operate, with Portable Emissions Monitoring Systems (PEMS) being the preferred choice. As a result, only a limited number of organizations, mainly regulatory institutions, have access to accurate monitoring tools, hindering the widespread knowledge of emissions levels.

As an alternative, a digital twin version of a CO_2 monitoring system can provide a cost-efficient, feasible solution that enables permanent measurement and monitoring of CO_2 emissions without the need for installing any additional sensors. A digital twin is a virtual representation of a physical object or system by replicating its behaviour in a virtual environment. By creating a digital twin of a vehicle CO_2 monitoring system, it is possible to continuously measure CO_2 exhausts and analyse the impact of emission reduction strategies without the need for physical sensors.

Internet of Things (IoT) devices are characterized by their small size, low power consumption, and adaptability, making them an ideal choice for distributed monitoring systems. However, IoT devices are recognized for their limited hardware capabilities in comparison with workstations, which typically renders them inadequate for executing large and complex tasks, such as Artificial Neural Network (ANN) model executions. Scholarly literature indicates that research efforts have been directed towards compressing ANN models through various techniques, including quantization, pruning, fusion, and other methods to convert a model into a simpler version.

Consequently, considerable research attention has been focused on executing pre-trained converted ANN models on edge devices. Nonetheless, most of this research is oriented towards the use of resource-limited conventional computers, such as Raspberry Pi or Nvidia Jetson [1]. Nevertheless, the implementation of ANN models in Micro Controller Unit (MCU)s is still a shallowly explored topic, given the challenges posed by its reduced computational capabilities [2]. Normally, the original model has to undergo several network modifications to fit the requirements demanded by micro controllers in terms of float precision, model size and ANN layers compatibility, usually having a negative impact in the model accuracy.

This work proposes a new digital twin CO_2 monitoring system and its real-world use case in a commercial vehicle. By integrating the proposed digital twin in a MCU, a continuous monitorization and prediction of the CO_2 emission concentrations can be conducted cost-effectively in a resource-constrained environment. The main contributions of this paper are listed below:

- Presentation of a new CO_2 emissions monitoring system.
- ANN model optimisation for micro controller-compatible execution.
- Deployment of an ANN model in micro controller installed within a vehicle.
- Comparative analysis of performance indicators between sensor-based measurement system and its digital twin.

979-8-3503-3554-5/23/$31.00

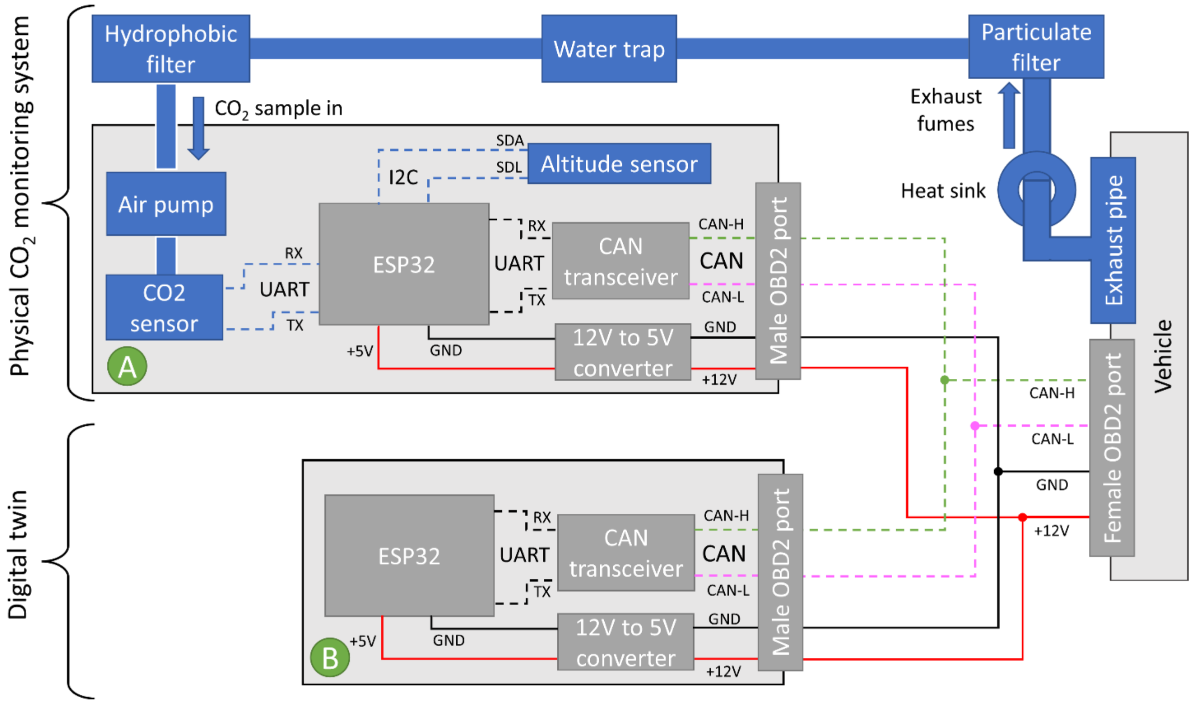


Fig. 1. Components diagram showing the connections made from the physical monitoring system (A) and the digital twin (B). Note that the elements in blue indicate that their use is exclusively made by the physical monitoring system only.

The remainder of this document is structured as follows. Section II introduces and explores the implementation of the reference physical CO_2 monitoring system. Section III details the approaches taken for the design, implementation and validation of the developed digital twin. Section IV discusses some of the insights obtained during this research, and Section V concludes this paper.

II. Proposed Physical and Digital Twin CO_2 Monitoring Systems

In this research, firstly a physical CO_2 monitoring system, previously presented in [3], is defined. This system is composed of an ESP32 micro controller mounted on a Pycom PySense board to ease the communication with the peripherals through the GPIO ports. Furthermore, the micro controller is powered by a 5V line from a transformed vehicle's +12V power supply that the OBDII port features. In order to allow bidirectional CAN bus communication, a CAN transceiver was connected via serial port to the micro controller. The reason why the physical sensor is directly connected to the vehicle's CAN bus is because it was originally intended to serve as a CO_2 emissions recorder that could report CO_2 measurements through the CAN bus to collect vehicle on-board metrics [4]. As for the CO_2 sensor, a SprintIR-R 20 sensor was used for measuring and recording the vehicle's exhaust CO_2 concentration, expressed in parts per million (ppm). The Sprint-IR sensor is able to report CO_2 concentration values of up to 20% with an accuracy of ±70 ppm and a resolution of 10 ppm at a maximum sampling frequency of 50 Hz via non-dispersive infrared (NDIR) technology.

Fig. 1 illustrates the components involved in the use of the physical and digital twin CO_2 monitoring systems, labelled with A and B respectively, to measure the CO_2 concentration from the vehicle's exhaust gases. The blue boxes are the exclusively used by the physical monitoring system only and not by its digital twin. Therefore, the digital twin CO_2 monitoring system does not need any physical CO_2 sensors.

For the exhausts gases collection, a set of tubes and filtering/cleaning components were installed in the vehicle's exhaust pipe in order to separate impurities and water vapour from the CO_2 analyte, whose presence in the sensor would cause inaccuracies (shown in Fig. 2 and Fig. 6). The set of filtering tools included a hydrophobic filter, a water trap, and a particulate filter. Furthermore, an electric pump helped the system to have a constant 0.5 l/min sample flow rate through the sensor's measurement chamber. Ultimately, a set of heat sinks were installed in the metal tubes to prevent the PVC tubes from melting due to the exhaust gasses high temperatures, which can easily reach 500 °C.

$$C_A = \frac{C}{1 + F(1013 - P)} \tag{1}$$

The board in which the ESP32 is mounted features an altitude sensor that helps correct the CO_2 measurement based on the present atmospheric pressure. To this end, the physical CO_2 sensor is commanded to update the atmospheric pressure to adjust the measurement every 850 iterations. Ultimately, the system corrects the obtained CO_2 concentration by applying equation 1, where C_A represents the adjusted concentration value in ppm, C represents the reported concentration by the sensor in ppm, F represents the compensation factor and P represents the atmospheric pressure (mbar) obtained from a MPL3115A2 barometric pressure/altitude sensor embedded in the PySense board. More information regarding the sensor measurement process can be found in [5]. The complete sequence diagram that illustrates the process of CO_2 collection by means of the physical monitoring system is depicted in Fig. 3a.

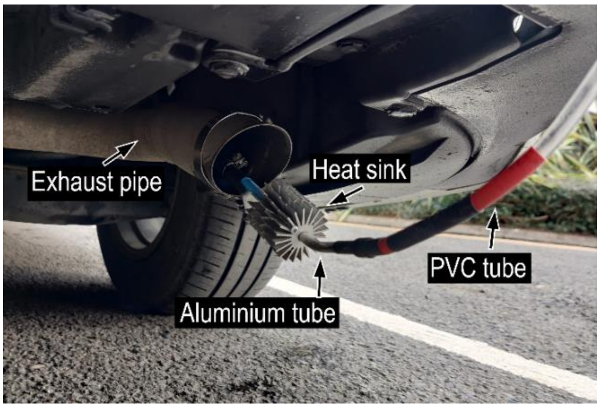


Fig. 2. Physical sensor installation setup in the vehicle's exhaust pipe.

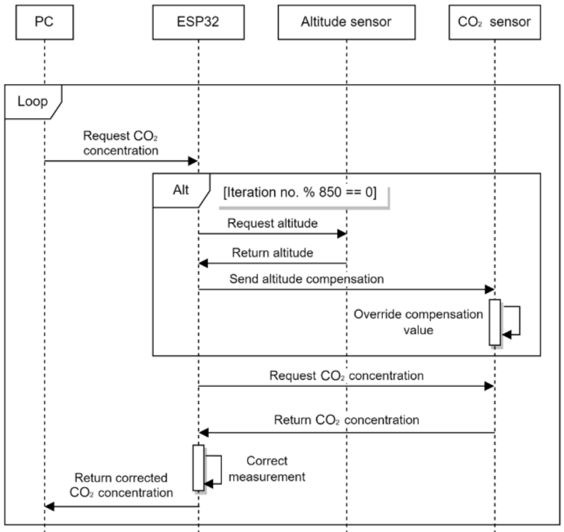
III. Digital Twin Model Development

This section details the process carried out for the development and installation of the digital twin for the prediction of vehicle CO_2 emission concentration. Aiming to replicate as much as possible the characteristics of the physical sensor, the MCU chosen for the development of the digital twin was also a Pycom FiPy board. As in the physical monitoring system, it features an ESP32 microprocessor with 4MB of RAM and 8 MB of flash memory. The limited hardware specifications were crucial for the design and adaptation of the ANN model. Fig. 4 summarizes the different steps taken for its implementation. In order to optimize the model to ease its execution on the MCU, a compound optimization strategy including three optimization techniques was carried out: 1) ANN model structure redefinition, 2) model pruning and 3) model conversion. The network modifications and optimizations were using TensorFlow (TF) and TF Lite framework, which is noted to be one of the most prominent artificial intelligence development frameworks.

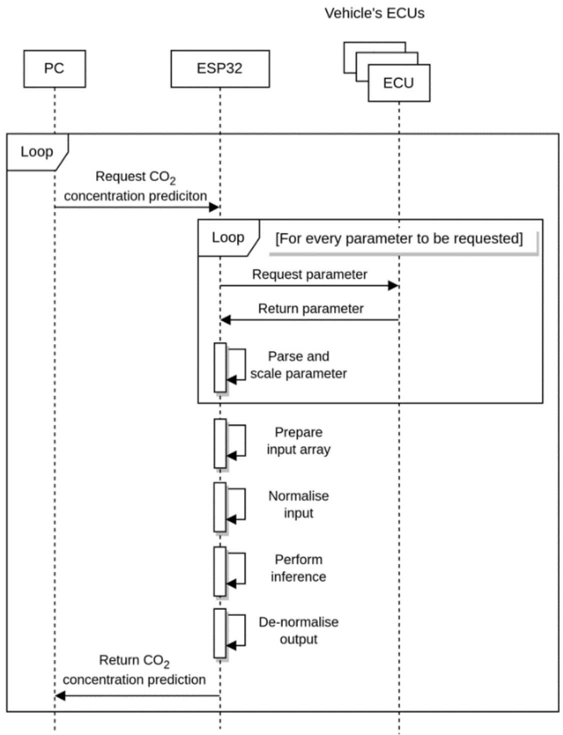
A. Proposed ANN Model Structure

The network structure employed was a modified version of the model shown in [4], where a Long Short-Term Memory (LSTM)-based ANN model with 8 different vehicle operational parameters was presented. In this study, the same input parameters were used to train, evaluate and execute the developed ANN model: vehicle acceleration (m/s^2), engine exhaust flow rate (kg/h), engine mode, engine speed (rpm), vehicle speed (MPH), mass air flow (g/min), engine coolant temperature (°C) and hybrid battery SOC (%). Despite the adequate results exhibited in [4] (97.5%), the model size was 7.7 MB, which made it unsuitable for our use-case given the MCU memory constraints.

As a result, the network size was reduced by decreasing the number of neurons of each layer, resulting in a network with an input layer of dimensions 32x8, where two stacked LSTM layers featured 128 and 64 neurons, respectively. In addition, the second LSTM layer had a dropout layer with a dropout rate of 0.2. As opposed to the original model, the modified model was provided with a flatten layer after the second LSTM layer, which proved to improve drastically the accuracy when implemented in the MCU. Subsequently, another dropout layer with a dropout rate of 0.3 precedes two dense layers with 8 and 1 neurons, respectively, being the latter the model output layer.



(a) Physical monitoring system's CO_2 emission concentration collection sequence diagram.



(b) Digital twin's CO_2 emission concentration prediction sequence diagram.

Fig. 3. Sequence diagrams.

Thanks to these modifications, the model size was decreased from 7.7 MB to 202.1 KB (97.37% lighter).

$$\tilde{x}_i = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{2}$$

As for the training, the model utilized the training, validation and test dataset described in [4]. In order to facilitate the model convergence, the dataset was re-scaled between 0 and 1 by applying normalization,

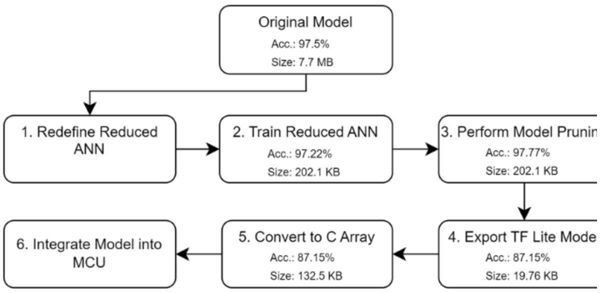


Fig. 4. Digital twin development flowchart.

defined by equation 2. In this equation, $\tilde{x}_i$ represent the normalized i^{th} value, x_i is the original i^{th} value, and x_{min} and x_{max} are the minimum and maximum value of a feature in the dataset, respectively. Furthermore, the model training was performed using batch size value of 32 for 50 epochs using the mean squared error as the loss function. During the training a changing learning rate scheduler was implemented to help reduce the over-fitting. Thus, the learning rate parameter was chosen for every epoch following equation 3, where *Lr* is the computed learning rate at epoch *t*. During the evaluation tests, the model exhibited an accuracy of 97.22% (R2 score), closely comparable to the 97.5% accuracy of the original model, which proves an outstanding trade-off between accuracy drop and model size.

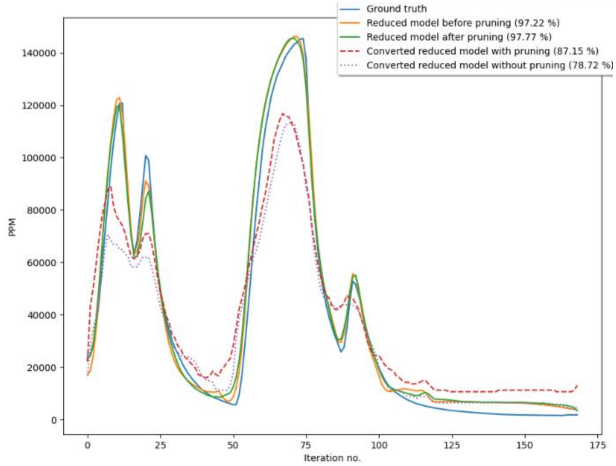
$$Lr_t = \begin{cases} 2 \times 10^{-5}, & \text{if } t < 10 \\ Lr_{t-1} \cdot e^{-0.1}, & \text{if } t \geq 10 \end{cases} \tag{3}$$

B. Model Pruning

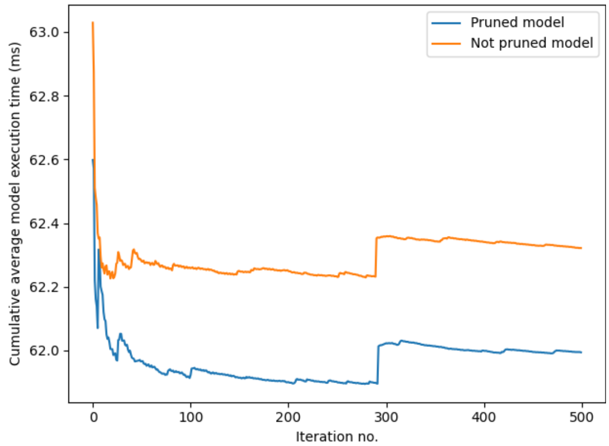
During this process, a model pruning technique was first performed to augment the sparsity of the model, which is defined as the proportion of weights or connections between neurons that are zero or near-zero. Pruning has proved to reduce the complexity of neural networks by zeroing the less useful weights of a network, speeding up its execution and enhancing its generalization by reducing the over-fitting [6] [7]. The pruning was performed for 13 epochs, where an initial and final sparsity of 0 and 75% were configured, respectively. The pruning process increased the model accuracy from 97.22% to 97.77%.

C. Model Conversion

After the optimization, the model was compressed and converted to a TensorFlow Lite compatible version. When converting a model to TensorFlow Lite flat buffer, some operations of the original model are discarded due to limitations in the operations catalogue. When this happens, these operations can often be replaced with approximations, which may cause an accuracy drop in comparison with the original model. Although quantization can significantly decrease the model size and improve the inference time, no quantization option was chosen for the conversion since previous attempts with float-16 and int-8 led to a drastic accuracy drops. Thus, the decimal representation of the converted model was float-32. After the conversion, the size of the TF Lite model was 19.76 KB. Lastly, since the target MCU did



(a) Accuracy evaluation among the presented ANN models.



(b) Pruned and not pruned model execution speed comparison.

Fig. 5. Performance comparison

not have a native file system, the compressed TF Lite model had to be converted into a C array to be compiled by the microprocessor at run time. After this conversion, the final model size was 132.5 KB.

D. Model Evaluation

1) CO_2 Emissions Percentage Calculation Accuracy

In order to assess the ability of the developed model to predict the CO_2 concentrations from the vehicle's exhausts, a comparative analysis was conducted, where the accuracy of each of the models generated in the stages of this research was contrasted. The results are summarised in Fig. 5a, where 4 different model performances are shown alongside the real ground truth data captured with the physical sensor. It is noted that the data present in the validation ground truth dataset used for this evaluation has not been used for any of the model trainings. In favour of the chart readability, the graph represents just a reduced part of the validation dataset used to measure the accuracy. Nonetheless, the R2 accuracy scores shown on the legend refer to the complete validation dataset, composed of over 14,000 data samples. As visible, the reduced models before and after pruning show the best accuracy of over 97%, closely comparable to the ground truth. However, having achieved an enormous decrease in the model size after the network modifications and optimisations yet preserving an adequate model accuracy, the models had to necessarily undergo a conversion that reduced the accuracy to 87.15%, which is still a decent result for a highly usable solution. Moreover, thanks to the pruning process, the pruned model outperformed the non-pruned model (78.72%), reducing the impact of the model conversion.

2) *Inference Time*

The model inference time was also evaluated to validate the suitability of the optimised model when executed in the MCU. To this end, the inference time of the pruned and not pruned model were assessed. Fig. 5b illustrates the cumulative average execution time (ms) recorded by both models, calculated after the application of equation 4. In this equation, x is the inference time at a time step i and t is the current time step. The inference time was measured since the input is fed to the model until the results are provided in the output.

As can be seen, the pruned model shows a faster inference time, averaging 61.99 ms per model execution, whereas the not pruned model manifests a 0.52% slower execution, averaging 62.32 ms. To this respect the benefits of pruning the model are visible in both accuracy and execution speed.

$$T_t = \frac{\sum_{i=1}^{t} x_i}{t} \quad (4)$$

E. *CO_2 Emissions Concentration Prediction Process*

As opposed to the physical sensor implementation, this digital twin version is able to calculate and report CO_2 concentrations out of some vehicle's performance parameters without using any physical CO_2 sensor. The complete iteration process that the digital twin follows to infer a measurement is depicted in Fig. 3b. Thus, by removing the physical CO_2 sensor, the digital twin eliminates the process of having to calibrate the sensor every 850 iterations. On the contrary, the ESP32 starts by querying the vehicle's necessary Electronic Control Units (ECU) by means of the Unified Diagnosis Services (UDS) protocol, defined in ISO 14229 [8]. The correspondent messages are sent by the pertinent ECUs and received by the ESP32 via the CAN bus, accessed through the vehicle's OBDII port with almost no hardware invasion within the

TABLE I. PHYSICAL SENSOR AND DIGITAL TWIN EXPENSES BREAK-DOWN COMPARISON.

	Physical sensor	**Digital twin**
Microprocessor (FiPy)	72 USD	72 USD
Development board (PySense)	30 USD	30 USD
CAN transceiver	14.4 USD	14.4 USD
OBDII cable	10.8 USD	10.8 USD
CO_2 sensor	429 USD	N/A
Heat sink	9 USD	N/A
Pump, tubing and filtering kit	199 USD	N/A
Total	**737.2 USD**	**100.2 USD**

vehicle premises. When each of the 8 model inputs parameters are received by the ESP32, these have to be

Fig. 6. Physical sensor (A) and digital twin (B) installed inside the vehicle. Labels A and B refer to the same labels as in Fig. 1.

parsed and converted to its original measurement scale and unit to afterwards be fed into the model. Once all the parameters have been collected from the vehicle's CAN bus and parsed, the ESP32 normalises the obtained parameters to scale them in using equation 2. By this step, the model will be able to recognise the input parameters on the same scale as the model was trained with. Furthermore, the ESP32 performs the model inference, and the resulting CO_2 concentration is de-normalise by applying the inverse formula of equation 2. Ultimately, the ESP32 yields the re-scaled value to the PC.

IV. DISCUSSION

In this study, the development, modification and optimisation of a previously unveiled ANN model able to predict vehicle exhausts CO_2 concentration is presented. One of the main motivations of this work was to democratise the access of vehicle CO_2 emissions concentrations by reducing capital costs derived from the commercial off-the-shelf measuring equipment. Table I reflects the expenses breakdown derived from the development of the physical measuring system and its digital twin. As can be seen, by solely utilising a reduced set of affordable and widely accessible electronic components, the total price of the designed system was drastically lowered, resulting in a 86.4% cheaper total cost.

Apart from the cost's reduction, another benefit of the digital twin is the independence from the real vehicle exhaust gasses, and therefore, from the need to install any physical measuring tools, such as the tubes, pump, etc. Fig. 1 depicts a comparison of the components needed per each approach. The elements coloured in blue indicate that its use is exclusively carried out by the physical measuring system and not by its digital twin.

Another aspect taken into consideration for the design of the digital twin is the elimination of the altitude sensor present in the physical monitoring system. Given the fact that the digital twin does not need to update its atmospheric pressure-based compensation value to perform the inference, it was no longer required. However, the reader the reader may interpret that its absence might damage the ANN accuracy, not being able to interpret the altitude at which the vehicle is located, and leading to a less accurate measurement. Nonetheless, the reader must take into account that the CO_2 correction done by the physical sensor is a post-processing practice made after the sample measurement, and therefore does not affect the isolated measuring process. As for the possibility to include the altitude or pressure as part of the training dataset parameters, this option was considered to be counterproductive in [4] as it increased the model over fitting, which would cause the model to perform poorer upon altitudes/pressure values unseen during the training.

Furthermore, in terms of memory usage, despite the use of the necessary TF-related libraries, both the physical sensor and the digital twin showed very similar levels of allocated RAM memory in use, with the former averaging 3.19 MB and the latter averaging 3.25 MB. Lastly, the digital twin is more power-efficient than the physical version since the elimination of the air pump and the CO_2 sensor provokes a reduction in the power consumption from 2.02 W to 1.74 W (13.8% less).

As for the digital twin implementation, Fig. 6 displays the final installation of both systems side by side. As observed, the digital twin (B) just needs to be connected to the vehicle's CAN bua via the OBDII port, whereas the physical sensor needs to be attached to the rest of the tubing and gas filtering system. The accuracy comparison of the digital twin and the physical sensor shown in fig. 5a indicate that, despite the slight measurement accuracy loss, the presented digital twin can be conceived as a promising solution to enable a widespread access of vehicle CO_2 emission monitoring that can operate under the same conditions as a real physical sensor.

V. Conclusions

In this paper we have presented a practical and cost-effective solution for democratizing access to affordable CO_2 emission monitoring. Through the design, implementation, and installation of a digital twin version of a physical CO_2 monitoring system, we have demonstrated that it is possible to create an accurate and reliable CO_2 emission monitoring system using low-powered micro controllers with limited computational resources. To this end, we proposed a modified ANN model to optimize it for its deployment on a microcontroller, resulting in a significant reduction in model size from 7.7 MB to 132.5 KB, whilst still maintaining a very usable accuracy of 87.15%. This proves the effectiveness of the optimization techniques used to create a more efficient and accurate model. Moreover, the elimination of some hardware components has made the digital twin version 13.8% more power-efficient, averaging only 1.74 W of power consumption. Furthermore, thanks to the removal of these physical components, the derived capital costs from its implementation were reduced by 86.4%, resulting in a total summation of costs of 100.2 USD. Although the accuracy achieved with our digital twin is adequate for the hardware constraints, we recognize the need for further research towards more refined model optimization and conversion to help mitigate the accuracy drop and enable faster inference time.

Acknowledgment

This work is funded in part by the EU H2020 project ”5G INDUCE: Open Cooperative 5G Experimentation Platforms for The Industrial Sector NetApps” (Grant Agreement Number 101016941), EU H2020 project “ARCADIAN-IoT: Autonomous Trust, Security and Privacy Management Framework for IoT” (Grant Agreement Number 101020259), and EU HE project “INCODE: Programming platform for intelligent collaborative deployments over heterogeneous edge-IoT environments” (Grant Agreement Number 101093069).

References

[1] R. Hadidi, J. Cao, Y. Xie, B. Asgari, T. Krishna, and H. Kim, “Characterizing the deployment of deep neural networks on commercial edge devices,” in 2019 IEEE International Symposium on Workload Characterization (IISWC). IEEE, 2019, pp. 35–48.

[2] B. Sudharsan, D. Sundaram, P. Patel, J. G. Breslin, M. I. Ali, S. Dustdar, A. Zomaya, and R. Ranjan, “Multi-component optimization and efficient deployment of neural-networks on resource-constrained iot hardware,” arXiv preprint arXiv:2204.10183, 2022.

[3] D. Tena-Gago, Q. Wang, and J. M. Alcaraz-Calero, “Non-invasive, plug-and-play pollution detector for vehicle on-board instantaneous co2 emission monitoring,” Internet of Things, vol. 22, p. 100755, 2023.

[4] D. Tena-Gago, G. Golcarenarenji, I. Martinez-Alpiste, Q. Wang, and J. M. Alcaraz-Calero, “Machine-learning-based carbon dioxide concentration prediction for hybrid vehicles,” Sensors, vol. 23, no. 3, p. 1350, 2023.

[5] “Sprintir-r manual,”https://cdn.shopify.com/s/files/1/0019/5952/files/SprintIR-R_Data_Sheet_Rev_4.12_3.pdf, accessed: 2023-03-02.

[6] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding,” arXiv preprint arXiv:1510.00149, 2015.

[7] S. Narang, E. Elsen, G. Diamos, and S. Sengupta, “Exploring sparsity in recurrent neural networks,” arXiv preprint arXiv:1704.05119, 2017.

[8] ISO Central Secretary, “Iso 14229: Road vehicles - unified diagnostic services (uds),” International Organization for Standardization, Standard ISO 14229, 2020.