Edge AI System Using a Thermal Camera for Industrial Anomaly Detection

Vítor M. Oliveira and António H. J. Moreira(✉)

2Ai – School of Technology, IPCA, Barcelos, Portugal
a13062@alunos.ipca.pt, amoreira@ipca.pt

Abstract. Predictive maintenance plays an important role in reducing long-term maintenance costs, unplanned downtime, and improving the lifetime of industrial machines. A common trait of machines is that they produce heat while working, resulting in a temperature pattern. Temperature can be a key parameter for monitoring the condition of machines, further aiding the diagnostics of problems. This paper presents an Internet of Things (IoT) system that monitors and detects thermal anomalies in industrial machines using deep neural networks (DNNs). The proposed system enables the DNN to run and make predictions inside a microcontroller, reducing the amount of data that needs to be transmitted to any external server. Furthermore, this system uses a platform that centralizes multiple sensors with the option of communicating with a server that runs two additional neural networks that are specialized in highlighting zones of interest in the thermal image and monitoring the temperature behavior over time. The system was tested in a laboratory and two industrial environments. Overall, the system performed well and can detect machine anomalies while also drastically reducing the amount of data needed to be transmitted. The system also presented high adaptability to different environments.

Keywords: Edge AI · Anomaly detection · Deep neural networks · Thermal camera · Industrial IoT

1 Introduction

Predictive maintenance (PdM) is used to prevent and delay industrial equipment failure. In many industrial areas, it is not economical to shut down machines for maintenance during production, since such interruption will waste time, money, and often the resources being processed [1].

One popular method of PdM is monitoring machines using infrared thermography (IRT) technology. IRT is based on measuring the distribution of radiant thermal energy (heat) emitted from a target surface and converting this to a surface temperature map or thermogram. Thermal energy is present with the operation of all machines. It can be in the form of friction losses, energy losses, or any combination thereof. IRT enables early

S. Paiva et al. (Eds.): SmartCity360° 2021, LNICST 442, pp. 172–187, 2022.
https://doi.org/10.1007/978-3-031-06371-8_12

detection of equipment flaws and faulty industrial processes under normal operating conditions, thereby reducing system downtime, catastrophic breakdown, and maintenance costs [2].

Many companies talk about the Industrial Internet of Things (IIoT) and the potential of connecting multiple physical assets to the Internet. However, expanding the number of sensors means larger amounts of data, and data will be useless unless there is someone or something to interpret it. With near real-time data streaming in, it is very difficult for a person to visualize small changes in images or graphs that may indicate the beginning of anomalies. Machine Learning helps to pinpoint areas for someone to focus their time on. The computer deals with the monitoring and warns the analyst, so he can focus on meaningful anomalies.

The main strategies of PdM are based on three techniques: 1) Regression Models that predict the machine's remaining useful time (RUL); 2) Flagging Anomalous Behaviour; 3) Classification Models to predict failure in a time frame. The description of these techniques is very related to the workflow of popular machine learning methods, and in fact, modern implementations of PdM are focused on using machine learning models like: ARIMA [3], Random Forests [4] or CNN's [5].

The trend towards combining machine learning with PdM raises the question of whether machine failures can be easily predicted. Although technological developments have improved the possibilities for companies to cope with machine breakdowns, their practical application is still a challenging task for many reasons [6].

This paper proposes a small and low-cost device that can silently monitor a machine and send data to the server only when it detects an anomaly. Specifically, it runs a convolutional neural network (CNN) inside an ESP32 microcontroller to detect anomalous patterns in thermal images. When moving large amounts of data across the wide-area network (WAN), monetary costs, transmission delays, and privacy leakage can become a major concern as more devices are added to a company's network. The alternative is using edge devices that process all data locally and extract specific insights, transmitting only the necessary information to the end-user. This reduces the bandwidth consumption, allowing the expansion of IoT devices without the need to exponentially expand the company's resources [7]. The system developed also runs two deep neural networks on a server with the purpose of reinforcing anomaly detection and improving user decision-making.

2 Related Work

The compression of complex mathematical models such as neural networks to fit smaller embedded systems is becoming a field of interest for IoT since it enables these systems to become intelligent and make decisions without user intervention.

F. Funk et al. [8] introduced a system that utilizes a neural network inside a microcontroller to automatically control the speed of a DC motor. The algorithm runs on an Arm Cortex-Mo microcontroller with only 4 kB of RAM and can adapt the motor's desired speed to changes in motor characteristics, e.g., heating or wear-out.

P. Andrade et al. [9] implemented an unsupervised TinyML solution based on the TEDA algorithm inside an Arduino Nano 33 IoT to detect road anomalies and changes

using the accelerometer sensor data embedded in a vehicle. The authors concluded that the experiments could detect potholes, bumps, and obstacles.

Also, G. Cerutti et al. [10] used a CNN model in an ARM-Cortex-M4 based microcontroller to detect people's presence with a low-resolution thermal image acquired by the 8x8 infrared sensor Grid-EYE. The system required minimal power to continuously classify images using only 6kB of RAM.

M. Kraft et al. [11] described a system to estimate the density of people in a space, facilitating energy savings in buildings. It uses a U-Net model inside a Raspberry Pi 4 that gathers thermal images using the infrared sensor MLX90640.

Although solutions on the edge are growing, to the best of the author's knowledge, there are still few designed to tackle the recent problems in the industry as discussed earlier.

The rest of this paper will be divided into four sections. Sect. 2 will describe the methods used to develop the system. Sect. 3 will discuss the experiments made in a laboratory setup and in two industrial environments. Sect. 4 will discuss the results obtained for each experiment. The paper ends in Sect. 5 by drawing the conclusions and discussing the implications and future work of this project.

3 Methodology

3.1 System Overview and Prototype

The overall system (Fig. 1) is divided into three parts. The main part represented in Fig. 1 (a), shows the edge system which gathers a thermal image and predicts anomalies inside the microcontroller. The second part defined in Section (b) of Fig. 1 is the communication bridge between the device and the user dashboard. The last part described in Fig. 1 (c). is a server that runs two complex neural network models to give more information to the user. The dashboard and neural networks server were tested on an Intel Core i7-7500U 2.70GHz 12GB RAM laptop.

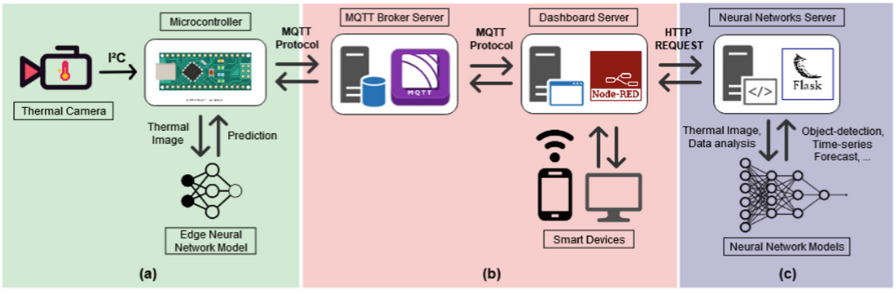

Fig. 1. Overview of the system, (a) Edge System, (b) Dashboard, (c) Neural Networks Server.

3.2 Acquisition Hardware

Thermal Imaging. The system uses the low-cost thermal infrared sensor MLX90640 with 32×24 pixels produced by Melexis. Although it presents a low resolution, it has already been utilized in security smart systems for human localization and action recognition [12], so it has the potential to gather the characteristics of industrial machines. The module can also measure object temperatures between –40 ºC to 300 ºC which is enough range for testing the system in various industrial machines (e.g., many motors or hydraulic presses overheat below 200 ºC).

Edge Processing Unit. The processing unit is the ESP-WROOM-32. This module was selected among other Microcontroller Units (MCUs) for being a cheap and low-power system, having a dual-core microprocessor, and for the built-in Wi-Fi antenna. It also has 520 KB of on-chip SRAM, which is enough memory to gather the thermal sensor data and process it with a quantized feed-forward deep convolutional neural network (CNN). The MCU also includes a 4 MiB flash memory that can store the CNN model architecture and weights [13].

3.3 Software Solution

ThermalCheck Device. As shown in Fig. 1 (a), the system uses the IR camera to collect images. The images are transmitted to the microcontroller via the I^2C protocol and then are sent to an embedded convolutional neural network (CNN) that classifies the thermal images. The predictions are sent to an external server using the MQTT protocol. This protocol is optimized for sending lightweight messages over high-latency or unreliable networks, which is well-suited for IoT devices [14].

Tiny ML Implementation. The CNN running on the edge device was trained on a computer using the Keras (2.2.4) library in Python (3.6.7). First, the images, which are in a float array of 768 positions, are sent to the computer via Wi-Fi or USB port. Then, the array is converted to an image and manually labelled with the corresponding class. In the model's training phase, data augmentation was used to increase the size and variability of the dataset, which overall improves the training process [15]. The model was then exported to the MCU if it performed well on new images.

Using the tinymlgen library from EloquentArduino[1], the finished model is converted to a C file byte array optimized so it can fit on the microcontroller [16]. The model is stored in the flash memory of the microcontroller. The MCU uses the TensorFlow Lite Micro library to read the converted model and make predictions. This library is prepared to receive as input the 768-pixel array directly from the camera.

Back Office. The back office (BO) converts the information processed by the microcontroller to visual feedback for the user. The BO is split into two servers: the first receives the data and presents the information to the user, and the second processes the data further using two neural networks to give more insight to the user. This second server is designed to be mostly inactive and turn on only at a specific time to target a specific machine and help the operator decide if, in fact, it has a fault.

[1] https://github.com/eloquentarduino/tinymlgen.

Dashboard Server. In Fig. 1 (b), the link between the edge system and the interface is an MQTT Broker, which routes all messages from one client to another. The system used both a local broker installed on a laptop, and a public Eclipse Mosquitto broker to transmit data over the Internet. The dashboard was made in Node-RED, which is a flow-based platform to connect hardware devices with online services.

The MQTT topics transmitted from Node-Red to the MCU are string sequences that control the microcontroller behavior. Conversely, the MCU transmits the thermal image and the predictions as float arrays and the interface extracts and displays relevant information for the user, e.g., the ratio of anomalies for all predictions and temperature statistics (average, maximum and minimum values).

Since some images received on the dashboard had imperfections related to interferences, a function was developed to ignore images with unrealistic temperature values. This function was implemented to compute the average temperature of any location in the image and is defined by the following equation:

$$t_c = \frac{\sum_{i=a-k}^{a+k} \sum_{j=b-k}^{b+k} p_{i,j}}{(k*2+1)^2} \tag{1}$$

where a, b is the image starting position at the central pixel of the kernel, k is the width of the kernel in relation to the central pixel, and $p_{i,j}$ is the pixel value at position i, j. This function is also useful to monitor areas of interest in the thermal image.

Neural Networks Server. The neural networks server shown in Fig. 1 (c) is made using FLASK, which is a Python framework to build web applications. It receives the thermal image and temperature statistics from Node-RED via an HTTP Request and uses them as input to a temporal anomaly detection model and a pattern detection model. The output of these networks is then sent back to Node-RED. The time of prediction for these models is dependent on the delay placed on the MCU image gathering process, which by default is 5 s.

1. Temporal Anomaly Detection
   The first server-based model is based on temporal anomaly detection. Whenever a temperature pattern changes in a working industrial machine, it may be a sign that the machine may soon come to a fault. It was studied two different approaches to analyze the temperature of these machines over time.

   The first was using a neural network model approach with a Long-Short Term Memory (LSTM). LSTMs are useful for learning sequences containing longer-term patterns of unknown length, due to their ability to maintain long-term memory [17].

   The LSTM will predict the next temperature value and the standard deviation according to the previous values seen. The standard deviation will act as the confidence level of the model. If there is an unexpected pattern, then the real value will be outside the range of the predicted value deviation, resulting in an anomaly.

   There were four different LSTM methods considered and are explained below:

- LSTM_2MODELS: This configuration uses one LSTM to predict the next temperature value and another LSTM to predict the next standard deviation.

- LSTM_2OUTS: This configuration only has one LSTM to forecast both temperature and standard deviation values.
- LSTM_DROPOUT: The model uses dropout layers to make the output always different. The standard deviation is calculated with multiple predictions on the same temperature forecast [18].
- LSTM_ROLLING_AFTER: This model predicts only the next temperature value. It then uses a rolling window function which will be detailed in Sect. 4.1 to calculate the deviation using the predicted temperatures

The second approach was using the seasonal stochastic model SARIMA. This model uses autoregression and moving windows to forecast the next value and the confidence intervals. It needs cyclical data arranged in seasonal patterns [19].

After analysing the two approaches, it was found that SARIMA had better performance with a small dataset, but it failed to fit a larger dataset with increased pattern variation resulting in unstable predictions. On the contrary, this does not occur with LSTMs, which train better and overfit less with bigger datasets and variation [20]. For this reason, it was chosen to focus on LSTMs for temporal anomaly detection.

2. Pattern Detection
The second server-based model locates and classifies anomalous patterns in thermal images. Currently, the YOLO network is one of the most used models for predictions on thermal images. According to M. Kristo et al. [21], using the YOLOv3 model achieved excellent results in person detection in difficult weather conditions for a sophisticated video vigilance system. It was compared to other modern models and achieved similar results while being faster. Also, A. Banuls et al. [22] utilized the YOLOv3 architecture for detecting objects in search and rescue scenes and reported a good performance of the model in both RGB and thermal images.

The YOLOv3 model is made for complex object detection tasks (e.g., human tracking, vehicle detection). Since the thermal camera resolution is very low, the level of detail and variation is also lacking, which makes training with the standard model inefficient since most of the layers are not necessary since deeper convolutions try to find details in images. Instead, it was used the YOLOv3-Tiny which is a smaller version using only a 24 layers model compared to the standard, which has 106 layers. The tiny version has increased detection speed and works well in smaller datasets while ensuring good detection accuracy [23].

4 Experiments

The validation of the system was split into two phases: first, a laboratory test to check the performance of the system; and second, deploying the system in an industrial environment to validate it with working machinery as seen in Fig. 2.

4.1 Laboratory Setup

The laboratory experiment seen in Fig. 2 (a), uses a controlled setup that can be manipulated to test many aspects of the system's capabilities.

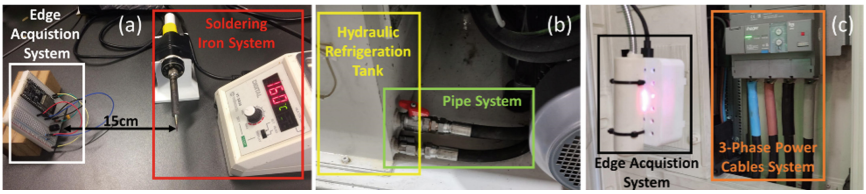

Fig. 2. Experiments Overview: (a) Laboratorial Setup, (b) Hydraulic Refrigeration Setup, (c) 3-Phase Power Cable Setup.

Hardware Setup. The setup consisted of a soldering iron with an adjusted temperature. The iron's normal operation was set to 160 ºC. It was placed in the center of the thermal camera lens from 15 cm to 25 cm. The close-up images are shown in Fig. 3.

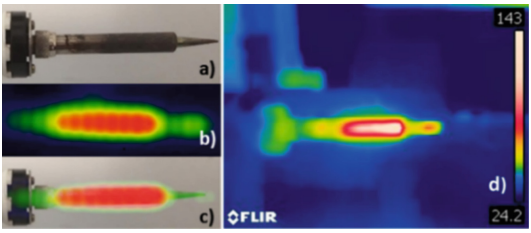

Fig. 3. Laboratorial Experiment: (a) soldering iron, (b) thermal image, (c) merge between image (a) and (b), (d) Ground-Truth thermal image using the FLIR C3 thermal camera.

Edge Neural Network. For the edge model, the dataset was split in four classes of 500 images each. The classes are: "Initial" (iron is at 40 ºC–100 ºC), "Nominal" (iron is at 100º–160 ºC), "Nothing" (iron is at 35ºC or less.), and "Anomaly". In the Anomaly class, the images were created synthetically by obstructing the iron with different objects changing the thermal pattern. Although, this class does not represent a scenario of a real fault in a soldering iron, it could show if the model is able to distinguish patterns changes on a low-res thermal image.

The dataset was split into an 80/20 ratio in a train and validation set since it is one of the most common ratios with good results for learning [24]. The data passes through an augmentation function which randomly tweaks the images with horizontal flips, rotations of 0–10º, width/height shifts of 0–10%, axis shears and zooms of 0–10%. The data was also normalized by dividing all pixels by 300 ºC (Sect. 3.2).

The CNN model was trained for 300 epochs with a batch-size of 32 samples and using the Adam optimizer with a learning rate of 1e-03.

After optimizing the network to a C file, the model size was 761 KB. Keeping in mind that the model needs to fit into a microcontroller with limited resources, the network has 30.484 parameters with the following architecture (Table 1):

Table 1. Edge CNN model.

Layer	Filters / Neurons	Kernel Size	Activation Function
Convolution 2D	4	3×3	RELU
Max Pooling 2D	–	2×2	–
Convolution 2D	6	3×3	RELU
Convolution 2D	6	3×3	RELU
Flatten	–		
Dense	64		RELU
Dense	4		SOFTMAX

Temporal Anomaly Detection. The temperature forecast dataset was created using a 5 V relay module controlled by an Arduino that switched on and off the soldering iron with a two-minute delay in between. Using (1) on the iron core, the images collected are converted in a time-series format by calculating the average temperature of the iron with a kernel of 5×5 pixels.

The standard deviation dataset was created by passing the temperature dataset through a moving standard deviation function defined as:

$$z_k = \sqrt{\frac{\sum_{i=k-m+1}^{k} (y_i - \overline{y})^2}{m-1}} \forall k = 1, \ldots, n \tag{2}$$

where m is the window size, n is the number of metric values, y_i is the metric value at the element i^{th}, and $\overline{y}$ is the average of the metric.

After testing multiple windows, the best fit was using the five previous samples.

Both datasets have 464 samples split in a 70/30 ratio between the train and validation sets and normalized by 300 ºC. To choose the best LSTM configuration, the four models were trained for 300 epochs with a batch-size of 8, using the Adam optimizer (learning rate of 1e-03) and with mean squared error as the loss function.

The architecture was equal for all configurations using two LSTM layers with 50 neurons each followed by a dense layer with one neuron. In the network with dropout, those layers were added after each LSTM layer and the scenario with the least error found was using 10% dropout. They were evaluated using the following four metrics and the results are presented in Table 2:

- Loss: the minimum loss value of the model.
- Test RMSE: the root mean squared error (RMSE) of the temperature predictions compared to the validation data.
- Deviation RMSE: The RMSE of the predicted standard deviation compared to the validated standard deviation created using (2).
- Processing Time: The average model processing time based on 100 predictions.

Table 2. LSTM configuration results. Test and Deviation RMSE are in Celsius (ºC).

	Loss	Test RMSE	Deviation RMSE	Processing time (s)
LSTM_2MODELS	2e-05[2], 0.0678[3]	1.8689	**1.5003**	0.0868
LSTM_2OUTS	0.0323	*2.7110*	1.7115	0.0570
LSTM_DROPOUT	2.1e-05	1.6993	1.6035	*1.1114*
LSTM_ROLLING_AFTER	1.9e-05	1.5953	1.8702	**0.0457**

Based on the table, the model selected was the first configuration which had the lowest total RMSE. The second configuration had the worst Test RMSE, and the third had a very slow processing time. The fourth configuration's drawback, albeit displaying good results, is that the standard deviation is fully dependent on the predicted temperature, which always comes with an error. A visual representation of the selected configuration can be seen in Fig. 4.

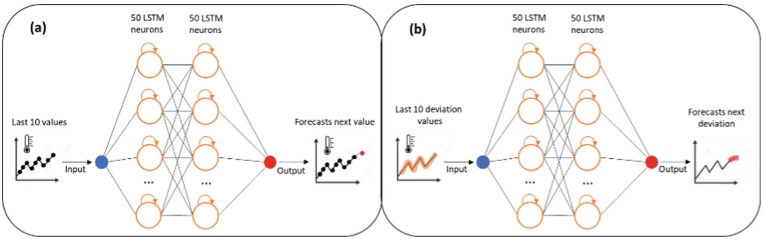

Fig. 4. LSTM_2MODELS configuration. (a) LSTM for temperature forecast. (b) LSTM for temperature deviation forecast.

The best parameters for the temperature forecast model were two LSTM layers with 127 and 24 neurons each, the learning rate was 0.0133, and the best optimizer was Adam. As for the standard deviation forecast, the best model was three LSTM layers with 47, 39, and 63 neurons, the learning rate of 9.09e-04, and the Adam optimizer. To decrease the false anomalies detected, the standard deviation was multiplied by three (3-sigma deviation) to increase the tolerance of the model.

Pattern Detection. The dataset for the pattern detection model is made of images from the anomaly class of the edge model. Since the images acquired have small resolution, they are first pre-processed by increasing the resolution ten-fold and passing a median filter to remove noise while preserving sharp edges. This makes the zones of interest in the image stand out, which can facilitate the model learning process.

[2] LSTM for temperature forecast.
[3] LSTM for standard deviation forecast.

The dataset was created by manually drawing bounding boxes on the gaps made by obstructions on the laboratory iron. The TinyYOLO model was then trained for 100 epochs with 500 images in an 80/20 split.

4.2 Industrial Environment

The system was then tested in a real scenario with a hydraulic refrigeration system and a 3-Phase cable system (Fig. 5).

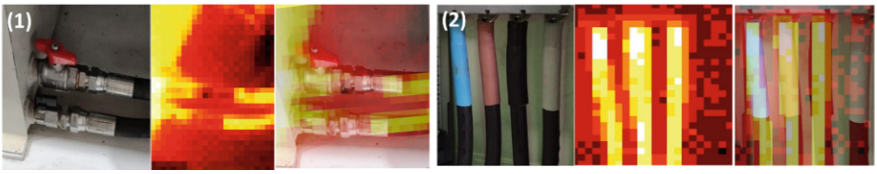

Fig. 5. On the left (1) is the Hydraulic Refrigeration system, the thermal image captured by the MLX90640 sensor, and the merge between images. On the right (2) is the 3-Phase Power Cable system, the thermal image, and the merge between images.

Hydraulic Refrigeration System. The first experiment is monitoring a refrigeration system for a hydraulic press. The setup is shown in Fig. 2 (b) and the close-up in Fig. 5 (1). This system has a refrigeration tank and two pipes. The edge device gathered images every five seconds for four days. It was extracted from every image the mean temperature of the tank and the two pipes using a 3×3 kernel using (1).

The dataset was split into four classes like the previous experiment. To train the edge model, it was necessary to create anomalies synthetically, since in the limited time that the camera gathered images, there were no reports of them. The anomalies were created by changing the pixels of the thermal images from 10–20 ºC specifically on the tank and the two pipes. The model architecture was changed to meet the pattern complexity by adding a Max Pooling Layer 2D with a kernel of 2×2 between the last two convolution layers and increasing those layer's filters to 8. The model was trained in an 80/20 split with 1000 images in each class for 100 epochs with the rest of the parameters equal to the previous experiment.

For the Temporal Anomaly Detection, the model was trained to forecast the temperature of the refrigeration tank in the same way as in the laboratory experiment with the difference of using a batch-size of 32 to meet a larger dataset of 7714 samples.

The pattern detection model was not evaluated due to the limited time of the experiment. However, the model would be useful to extract the positions of the tank and pipes to calculate more accurate temperatures of the refrigeration components and use that data in the Temporal Anomaly Detection model.

3-Phase Power Cable. The final experiment was monitoring a 3-phase power cable as shown in Fig. 2 (c). The camera was focused on the electric board, which had four cables seen on Fig. 5 (2). The images were gathered every five seconds for two days. The same logic from the previous experiment was also applied for this one. The edge model dataset had 900 images for each of the four classes. As for the Temporal Anomaly Detection, the model was trained with samples of the red wire. The dataset had a total of 5882 samples.

5 Results

The main goal of this work was to evaluate the viability of the edge system The system was tested in a laboratory, and it was further validated in two industrial environments. The implementation of the three models in the interface is displayed in Fig. 6.

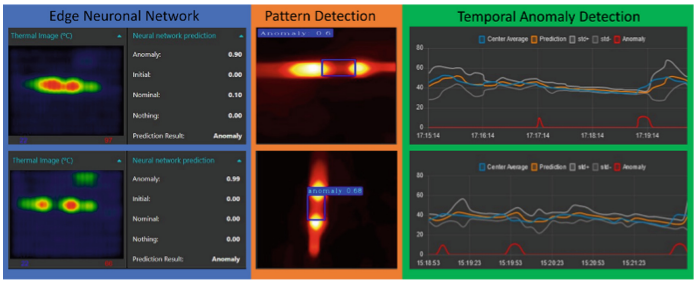

Fig. 6. Interface Models Result. In the graphs the x-axis is time and y-axis is in degrees (ºC).

5.1 Laboratory Evaluation

The edge model predicted most images correctly having a validation accuracy of 94.37% with a f1-score of 92.25%. In Fig. 7 (a), are displayed five predictions of this model for all classes.

As for the Temporal Anomaly Detection (Fig. 8 (a)), the model did not correctly fit some parts of the dataset, resulting in false anomalies. The main reason for this is that the training samples were too low for the model to learn the pattern correctly.

Despite that, the model predicted the rest of the dataset correctly, having a Test RMSE of 1.4931 ºC and a Deviation RMSE of 0.3535 ºC.

The pattern detection model had problems fitting the data, resulting in underwhelming results. Even though the model predicted some of the anomalies, the score was always less than 70%. The model validation loss was 4.11.

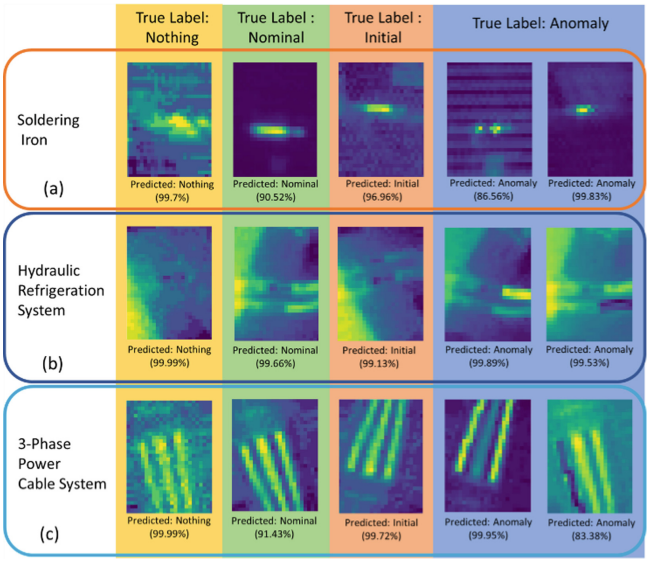

Fig. 7. Edge Neural Network Model Predictions. (a) Soldering Iron, (b) Hydraulic Refrigeration System and (c) 3-Phase Power Cable System.

5.2 Industrial Evaluation

In the industrial environment, the evaluations were focused on the Edge Neural Network and the Temporal Anomaly Detection. These models performed better than the previous results, mainly because the dataset had more samples.

The edge model in the hydraulic refrigeration system, as seen in Fig. 7 (b), performed well with a validation accuracy of 99.73% and an f1-score of 99.5%. The model identified all the anomalies, having only a small difficulty distinguishing the "Initial" from the "Nominal" class. The Temporal Anomaly Detection model identified an abnormal peak in the dataset (Fig. 8 (b)). This peak could have happened because of transmission errors, interferences, or obstructions between the camera and the tank, but although this was not a real anomaly, the potential of the network can be seen. The model had a Test RMSE of 0.182 ºC, and a Deviation RMSE of 0.07 ºC.

As for the 3-phase power cable system, the edge model had a validation accuracy of 98.95% and an f1-score of 98.5%. The predictions are seen in Fig. 7 (c).

The Temporal Anomaly Detection model had a Test RMSE of 0.258 ºC and a Deviation RMSE of 0.04 ºC. A small part of the test dataset had missing data due to connection issues, which resulted in a strong transition of temperature values which the model picked up as an anomaly (Fig. 8 (c)).

In all experiments, the edge model performed well, with an overall accuracy above 90%, which means that this approach can in fact help companies detect machine anomalies while also cutting the data transmission to minimal levels. The temporal anomaly detection model served as a good support model that could help in identifying different types of anomalies that may escape the edge model. The only step back is that it requires a considerable amount of data to be useful. The pattern detection model did not perform well in finding the anomalies in the dataset, which, for the most part, is due to the smaller resolution of the thermal camera and the quality of the dataset acquired. This model, however, still presents great potential for improving the accuracy of the temporal anomaly detection model.

6 Conclusions

This work presented the design and evaluation of a low-cost edge system that monitors thermal patterns of industrial machines and detects anomalies. The proposed system presented good results and showed high adaptability to different setups.

Overall, the edge system took less than a second to gather and process the thermal image, feed it to the CNN and have a prediction ready to send to the external server. This system can send the predictions and the thermal image to the server if there needs to be a deeper analysis on the machine, but the major benefit is that it can process everything inside and only when an anomaly occurs, transmit data to the server.

As for future work, further investigation of the pattern detection model is needed by testing other pre-processing options, improving the dataset quality, and verifying if it improves the detection score. It is also considered a strong option to test and compare it with other state-of-the-art object detection models.

Acknowledgment. This project was funded by national funds (PIDDAC), through the FCT – Fundação para a Ciência e Tecnologia and FCT/MCTES under the scope of the project UIDB/05549/2020 and through the special FCT program "Verão Com Ciência" with the project 2Ai Summer School (process 77, 20/229). Moreover, it was also funded by SmartHealth "NORTE-01–0145-FEDER-000045", supported by Northern Portugal Regional Operational Programme (Norte2020), under the Portugal 2020 Partnership Agreement, through the European Regional Development Fund (FEDER).

Appendix

Temporal Anomaly Detection Results

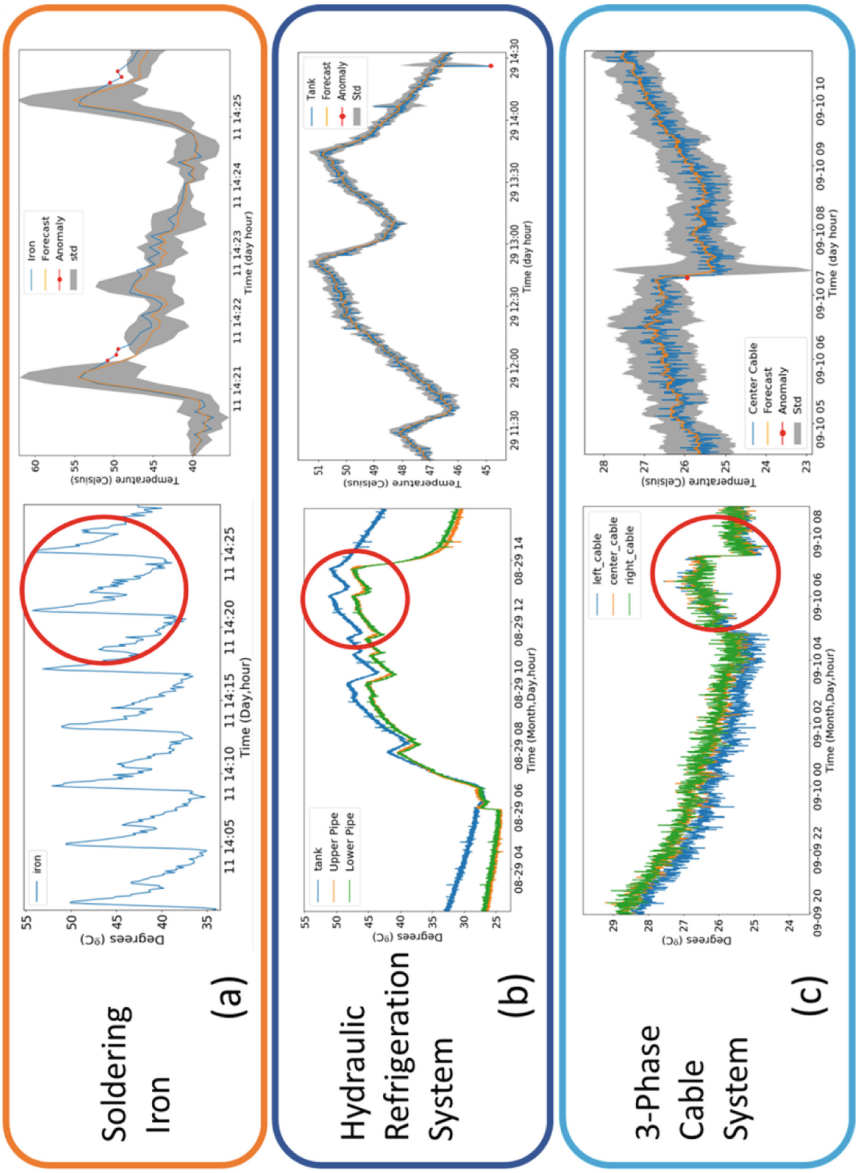

Fig. 8. Temporal Anomaly Detection Model results. (a) Soldering Iron, (b) Hydraulic Refrigeration System and (c) 3-Phase Power Cable System.

References

1. Zhu, Z., et al.: Preventive maintenance subject to equipment unavailability. IEEE Trans. Reliab. (2019). https://doi.org/10.1109/tr.2019.2913331
2. Bagavathiappan, S., Lahiri, B.B., Saravanan, T., Philip, J., Jayakumar, T.: Infrared thermography for condition monitoring - a review, Infrared Phys. Technol. **60**, 35-55 (2013) https://doi.org/10.1016/j.infrared.2013.03.006
3. Kanawaday, A., Sane, A.: Machine Learning for Predictive Maintenance of Industrial Machines using IoT Sensor Data (2018) https://doi.org/10.1109/ICSESS.2017.8342870
4. Paolanti, M., et al.: Machine Learning approach for Predictive Maintenance in Industry 4.0. (2018) https://doi.org/10.1109/MESA.2018.8449150
5. Sun, W., Liu, J., Yue, Y.: AI-enhanced offloading in edge computing: when machine learning meets industrial IoT. IEEE Netw. **33**(5), 68–74 (2019). https://doi.org/10.1109/MNET.001.1800510
6. Kammerer, K., et al.: Anomaly detections for manufacturing systems based on sensor data—insights into two challenging real-world production settings. Sensors (Switzerland) **19**(24), 5370 (2019). https://doi.org/10.3390/s19245370
7. Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., Zhang, J.: Edge intelligence: paving the last mile of artificial intelligence with edge computing. Proc. IEEE **107**(8), 1738–1762 (2019). https://doi.org/10.1109/JPROC.2019.2918951
8. Funk, F., Bucksch, T., Mueller-Gritschneder, D.: ML training on a tiny microcontroller for a self-adaptive neural network-based DC motor speed controller. In: Gama, J., et al. (eds.) ITEM/IoT Streams -2020. CCIS, vol. 1325, pp. 268–279. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-66770-2_20
9. Andrade, P., et al.: An Unsupervised TinyML Approach Applied for Pavement Anomalies Detection Under the Internet of Intelligent Vehicles (2021) https://doi.org/10.1109/MetroInd4.0IoT51437.2021.9488546
10. Cerutti, G., Prasad, R., Farella, E.: Convolutional Neural Network on Embedded Platform for People Presence Detection in Low Resolution Thermal Images (2019) https://doi.org/10.1109/ICASSP.2019.8682998
11. Kraft, M., Aszkowski, P., Pieczyński, D., Fularz, M.: Low-cost thermal camera-based counting occupancy meter facilitating energy saving in smart buildings. Energies **14**(15), 4542 (2021). https://doi.org/10.3390/en14154542
12. Spasov, G., et al.: Using IR Array MLX90640 to Build an IoT Solution for ALL and Security Smart Systems. (2019) https://doi.org/10.1109/ET.2019.8878637
13. Maier, A., Sharp, A., Vagapov, Y.: Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things. (2017) https://doi.org/10.1109/ITECHA.2017.8101926
14. Masdani, M.V., Darlis, D.: A Comprehensive Study on MQTT as a Low Power Protocol for Internet of Things Application. (2018) https://doi.org/10.1088/1757-899X/434/1/012274
15. Mikołajczyk, A., Grochowski, M.: Data augmentation for improving deep learning in image classification problem. (2018) https://doi.org/10.1109/IIPHDW.2018.8388338
16. Dokic, K.: Microcontrollers on the edge – is ESP32 with camera ready for machine learning? In: El Moataz, A., Mammass, D., Mansouri, A., Nouboud, F. (eds.) ICISP 2020. LNCS, vol. 12119, pp. 213–220. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51935-3_23
17. Malhotra, P., Vig, L., Shroff, G., Agarwal, P.: Long Short Term Memory Networks for Anomaly Detection in Time Series. (2015)
18. Gal, Y., Ghahramani, Z.: Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. (2016)

19. Chen, P., et al.: Time Series Forecasting of Temperatures using SARIMA: An Example from Nanjing. (2018) https://doi.org/10.1088/1757-899X/394/5/052024
20. Ying, X.: An Overview of Overfitting and its Solutions. (2019) https://doi.org/10.1088/1742-6596/1168/2/022022
21. Kristo, M., et al.: Thermal object detection in difficult weather conditions using YOLO. IEEE Access**8**, 125459-125476 (2020), https://doi.org/10.1109/ACCESS.2020.3007481
22. Banuls, A., et al.: Object Detection from Thermal Infrared and Visible Light Cameras in Search and Rescue Scenes. (2020) https://doi.org/10.1109/SSRR50563.2020.9292593
23. Adarsh, P., Rathi, P., Kumar, M.: YOLO v3-Tiny: Object Detection and Recognition using one stage improved model. (2020) https://doi.org/10.1109/ICACCS48705.2020.9074315
24. Gholamy, A., Kreinovich, V., Kosheleva, O.: Why 70/30 or 80/20 relation between training and testing sets : a pedagogical explanation. Dep. Tech. Reports (2018)
25. Huda, A.S.N., Taib, S.: Application of infrared thermography for predictive/preventive maintenance of thermal defect in electrical equipment. Appl. Therm. Eng. **61**(2), 220–227 (2013). https://doi.org/10.1016/j.applthermaleng.2013.07.028