IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 29, No. 5, October, 2024, pp. 1755-1762
Journal homepage: http://iieta.org/journals/isi

Recognition of the Sound of the Lonchura Maja Bird and the Threat of House Sparrows Using Edge Impulses Based on a Custom Deep Neural Network to Protect Rice Plants

Aeri Rachmad[1*], Eko Setiawan[2], Abdul Wahib Hasbullah[3]

[1] Department of Informatics, Faculty of Engineering, University of Trunojoyo Madura, Bangkalan 69162, Indonesia
[2] Department of Natural Resources Management, Faculty of Agriculture, University of Trunojoyo Madura, Bangkalan 69162, Indonesia
[3] Department of Electrical Engineering, Faculty of Engineering, University of Trunojoyo Madura, Bangkalan 69162, Indonesia

Corresponding Author Email: aery_r@trunojoyo.ac.id

https://doi.org/10.18280/isi.290509

Received: 11 October 2023
Revised: 11 June 2024
Accepted: 31 July 2024
Available online: 24 October 2024

Keywords:
audio recognition, bird sound, aves, Lonchura Maja, RP2040 Connect, custom deep neural network, Edge Impulse

ABSTRACT

The presence of birds can be used as a biological indicator related to the quality of environmental health in development. However, the presence of pest birds is a threat to farmers. This paper employs edge machine learning regarding audio recognition of birds Lonchura Maja and the sound of birds of house sparrow, which can be applied to a low-power microcontroller. We also train another nearby bird sound of turtledove, which is often seen around the rice fields on Bangkalan, to act as noise or background sound; we test the reliability of four machine learning (ML) algorithms, then embed them in the microcontroller RP2040 and connect. The first machine learning model is a custom deep neural network (CNN) 1D with two layers, and the second model uses transfer learning-based architecture. The Edge Impulse embedded platform learning machine is used to conduct training and testing. The resulting learning model was then implemented as an Arduino Library, as an Unoptimized float (32-bit) and Optimized integer quantization (8-bit). The estimated values produced by the microcontroller are evaluated in 4 cases, using the EON compiler and Tensor Flow Lite. In this paper, the custom 1D CNN model provides the best accuracy value, with 87.4% accuracy during training and 84.59% accuracy on testing, and it uses very efficient resources, 66.2 Kbyte Flash memory and 11.8 Kbyte Peak RAM.

1. INTRODUCTION

Pest birds are a threat to farmers, because they form colonies and attack in large numbers, ranging from hundreds to thousands. Researchers have developed various tool to scaring bird, including the use of sensors [1], timers, and detection techniques. It is important for these tools to work effectively in rice fields or as bird scarer as this affects battery power consumption and other resources. In previous research, a detection technique using movement and camera monitoring was proposed [2], in other research it has also been made which has not been proven to be effective, while other research uses an image object detection technique that uses two (2) high resolution cameras [3] which is certainly is an option that requires complex computing and of course has implications to power or battery used [4]. And we all know, equipment to repel pest birds sold on the market still operates on 12v voltage, it is very rare to use 5v low voltage and minimum power consumption [1, 3, 5].

Birds serve as critical ecological indicators, reflecting the overall environmental health. Nevertheless, the presence of pest birds poses a threat to farmers. The cacophony of bird sounds in rice fields can be utilized to gauge the population of bird pests such as Lonchura Maja and house sparrows, making it a valuable parameter for audio recognition [6].

In today's life, various applications of artificial intelligence (AI) are known. however, tasks such as taking photos, checking the weather, and so on require expensive AI models and internet connection. Generally, Little ML calculations work similarly to normal AI intelligence, with models prepared on a cloud directory or computer client. Figure 1 and Figure 2 illustrate the technology requirements for cloud ML, edge ML, and Tiny ML [7], including algorithms, hardware, and so on.

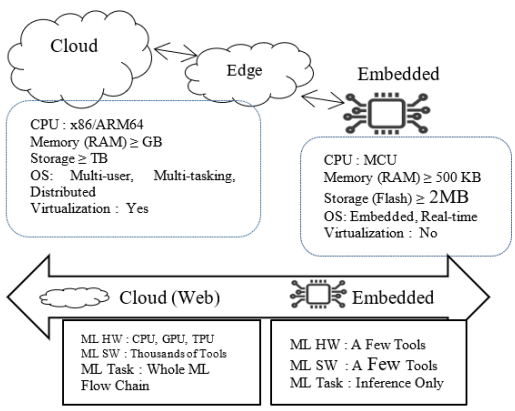

Figure 1. Miniscule of implementing ML [8-12]

Tiny ML technology collects sensor data and processes it into Convolutional Neural Networks at the micro- or nano-level. A microcontroller operates the neural network and may also be equipped with hardware accelerators. Processes are moved to deep neural networks using GPUs, multi-core CPUs, and TPUs for complex cases.

In Artificial Intelligence, models have also been developed directly from raw data with low processing [13], including in the classification of bird sound [14]. From 2016, the reliability of Convolutional Neural Network (CNNs) consistently outperformed other classifications in classifying bird sounds in BirdCLEF [15]. In another study, Okan Kücüktopcu et al. [16], created a bird sound monitoring tool based on sound recognition using a Tiva C microcontroller, where the price of the board is more than 41$, which in our opinion is not yet efficient if used in our place besides that the development board is also not sold here.

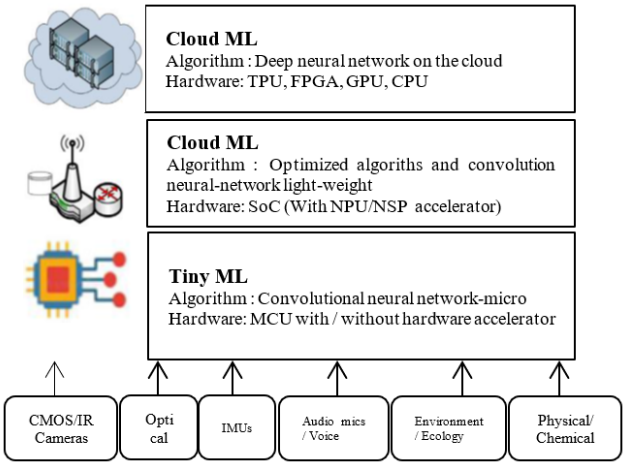

Figure 2. Tiny ML with layered approach

In this study a feasibility test was carried out on the application of ML audio recognition of the house sparrow sounds and Lonchura Maja pest birds. This would be the first step for a low -power sound detection system, certainly can be applied in remote settings. The methodology used the bird audio recognition model is described in section 2. in section 3, the performance of each model will be tested using both 1D CNN and transfer learning, while the comparison of resource use and model reliability is discussed in section 4, and the analysis of various performances is presented in section 5.

2. MATERIAl AND METHOD

It is now possible to incorporate machine learning algorithms into limited microcontroller resources using software tools and using frameworks. Therefore, we use Edge-Impulse studio on this paper [17].

2.1 Edge Impulse studio

Edge Impulse software give an easier AI developers on data collecting or data importing, train and testing machine learning models, and implement it to the embedded device. Edge Impulse Studio uses the TensorFlow Lite interpreter and Keras to train neural network models, and scikit-learn for classical machine learning models. The studio provides 2 options for designing models, 1. Simple GUI interface with visual layer mode, and 2. Expert display which is intended for those who require advanced optimization in Keras expert mode.

Additionally, in terms of model optimization, Edge Impulse Studio provides the Edge Optimized Neural (EON) feature which is a powerful tool for optimizing and effectively running NN by reducing RAM and Flash usage. When embedded in a microcontroller, TensorFlow Lite was proven to have no effect on accuracy values and inference time. The EON compiler includes its own compiler to compile neural networks to C++ code, and if we use TensorFlowLite (TFlite), we need an interpreter, but the weakness of EON is that it is not easy to update instantly [18].

In this paper, five classes of audio classification models were created, and then trained on a 1D CNN model and two transfer learning models. These model were then embedded in an Arduino RP2040 Connect microcontroller which has an internal PDM microphone. The reliability of the four models is evaluated and compared using the TFlite and EON deployment option using both representation of quantized and floating point.

2.2 Data set

The hardware used for collecting sampling data is the Arduino Nano RP2040 Connect which was specially developed for building Artificial Intelligence (AI) models. It contains a 32-bit ARM Cortex M0+ processor that operates at 133MHz. This module has equipped by 264KB SRAM, and a 16MB Flash memory chip for extra storage. The RP2040 Connect is also equipped with an internal microphone for developing audio recognition or sound activation, as shown in Figure 3.

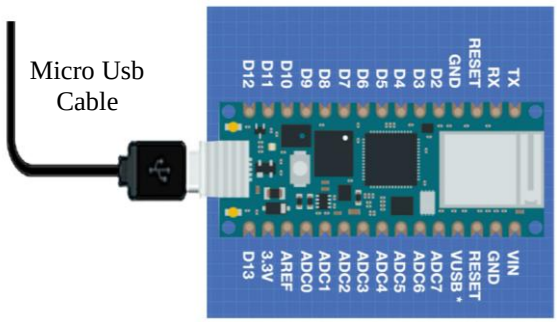

Figure 3. Arduino RP2040 Connect

When taking samples, this device is connected to a computer using a Micro USB cable. The computer must has connected to the edge-impulse CLI and edge-impulse studio software, as shown in Figure 4.

The sound samples recorded according to the Edge Impulse GUI image above consist of 2 sources, namely the first is a sound sample in the rice fields and the second is a sound sample in a room or studio by placing sparrows and doves in a cage, while sparrows were recorded when they Enter the house area, semi-open building and even the mosque.

Figure 5 and Figure 6 illustrate that samples are taken in .wav audio format with a length of 3 seconds, with an average sample size of 46.9 KB, so the total dataset occupying the Edge Impulse cloud drive is 56.28 Mbytes.

Once connected, samples can be taken, by selecting the sensor to be used. In this study, samples were taken using the Arduino RP2040 Connect internal PDM microphone, with a rate of 8Ksa/second for the 1D CNN Model and with a rate of 16Ksa/second for MobileNet data processing.

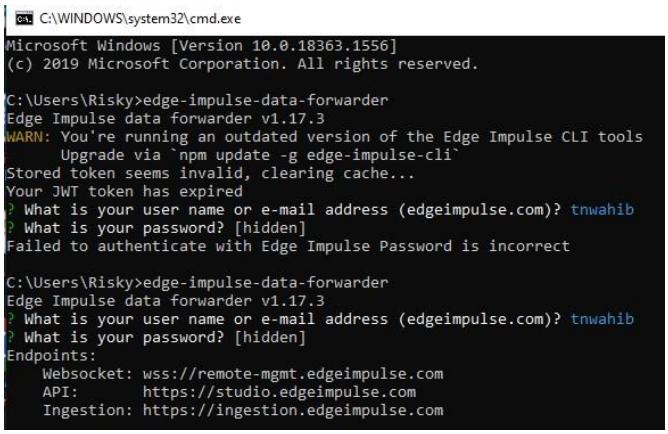

Figure 4. The process of connecting the Edge Impulse CLI with the device with a USB cable

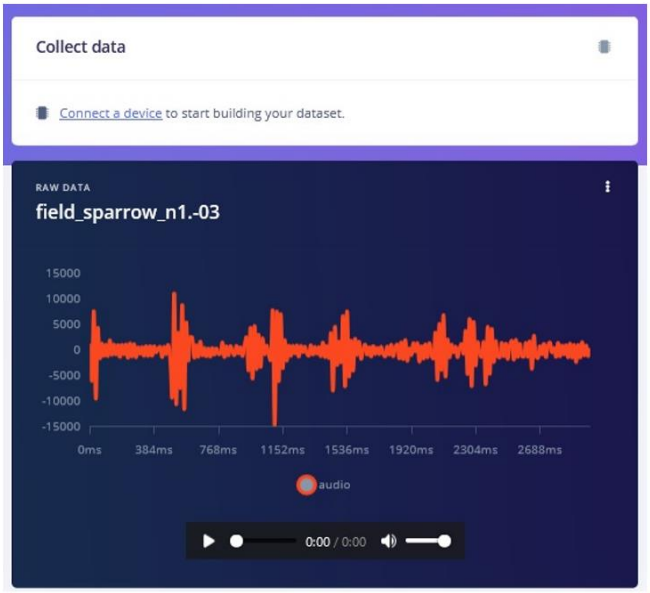

Figure 5. Graphical user interface display of Edge-Impulse when sampling audio sound

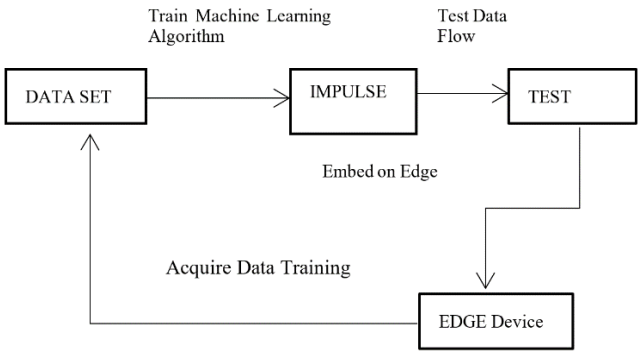

Figure 6. Block diagram of Edge Impulse

2.3 Models

Once the necessary information in the form of bird sounds and noise sounds is gathered, the next step is to prepare a suitable model. The Edge-Impulse studio, with its user-friendly interface, allows for easy handling of blocks graphically and also facilitates group Brain Keras Block learning. The results of the design are conveniently stored on the edge drive. The model is designed by starting to set up scale, channel, and FFT settings, making the process straightforward and manageable.

In the Edge-Impulse tool, the processes start from creating a training data set; the training data set is generated using a split process. Train data and test data are provided in five classes. Training data and test data will be in the ratio of 80/20. Data pre-processing was carried out using spectral analysis. The following steps extracted features on the training dataset, followed by classification. Then, classification will be carried out using the NN classifier and the Keras method. The final step is to test the model (if the results are not satisfactory, retrain the model), followed by creating a model to be embedded in hardware (microcontroller RP2040 Connect).

2.4 Running inferences

One way to run impulse is to use the Arduino IDE. Impulse can be run on the Arduino library, a package file containing all the signal processing blocks, configuration, and learning blocks in one package. Deployment using Raspberry RP2040 Connect is used with an Arm-based Arduino Development Board with 264 kB RAM and low power consumption; this device is also equipped with onboard Bluetooth, WIFI, 6-axis IMU accelerometer, gyroscope, temperature sensor, RGB LED, and microphone.

3. MAIN RESULTS

3.1 Pre-process and data acquisition

The result of the audio signal segmentation stage is a crucial step in our work, yielding 735 sound samples in .wav format, with rates of 8Ksa/s and 16Ksa/s, each audio split every 3 seconds. These results are then divided into two groups, the data training group and the data testing group. This division is pivotal in determining which data will be used as training data for the Tiny ML neural network model and which data will be used for testing, underscoring the importance of your role in this process.

The data set consists of 1,200 audio data resulting from sound segmentation. 960 sound data will be taken as data training, and the remaining 240 data will be used as test data. The composition of the distribution of training data and test data can be seen in Table 1.

Table 1. Subgroup number of audio data training and testing

Class Audio Category	Training	Testing
Lonchura Maja	192	48
House Sparrow t1	192	48
House Sparrow t2	192	48
Turtledove	192	48
Quiet	192	48

3.2 Feature extraction

Figure 7 shows the feature extraction process aims to obtain a vector that characterizes an object. The basic principle is to reduce the size of the data, without losing important information that characterizes an object. In this research, the MFCC coefficient 15 will be used, so that characteristic vectors with dimensions of 15×240 will be produced for each category.

On MFCC, in addition to our number of coefficients, the

number of filters needs to be set at 32 and the FFT length 256.

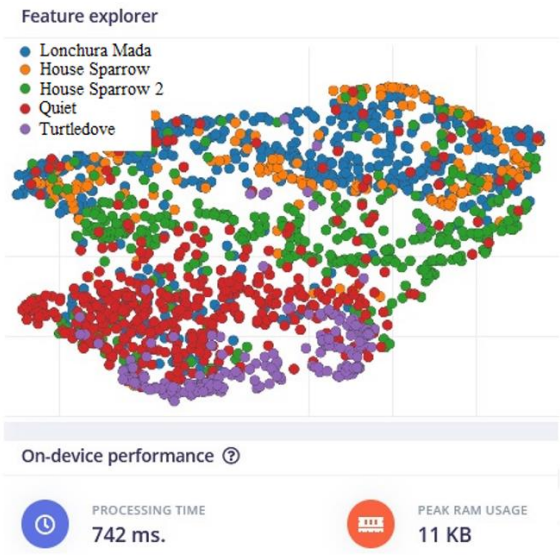

Figure 7. Feature extraction on GUI result

3.3 Model performance

Bird sound classification models using Convolutional Neural Networks have been used in previous research and proven to be reliable [11, 12], so in this paper model design in a custom CNN 1D 2 Layer, given dense 10 Neuron design was used, which is shown in the Figure 8.

The model contains 2 layers of CNN 1D, the 1st layer is 1D Convolutional NN 16 neuron, then followed the 2nd layers of 1D CNN 32 neuron, given dropout 0.25, then flattening, a dense layer 10 neuron, and final layer of 5 neurons (5 classes). The model was iterated by 100 epochs and given learning rate of 0.008. CNN models were trained using a cross-entropy loss and the Adam optimizer, but on MobileNet model by default use the Nadam optimizer and also cross-entropy loss.

From the Confusion Matrix (Table 2), the highest inferencing accuracy value is obtained in the Audio Bird of Turtledove, which is worth 88.9%; the 2nd highest accuracy value is the result of the sound sample of the Lonchura Mada audio class, which is with an accuracy value of 84.3%. The value the 3rd highest sound class accuracy is the sound of the sparrow type_tone 1 83.3%, followed by the 4th best accuracy value is the sound class house sparrow bird of the type_tone 2 (Sparrow Sound During Calls) 82,8%, and Quiet Environment (No Sound Sparrow Detected) class audio accuracy 72,2%. The accuracy value of all the experiment results meets the standard Deep Learning requirements, where the F1 Score number that is a reference of good value is 0.7 or the accuracy value of at least 70%

In other Confusion Matrix (Table 3), the same model implemented with the 10 Neuron Dense Layer a little impact on the addition of the accuracy of the bird house sparrow Type_sound 1 and the audio house sparrow Type_Sound 2 sound class, but a little reduces the accuracy level in Sound class of Bird Lonchura Mada, this reduction is not too significant. Because it is still above the Robotics Standard Insurance threshold, especially in general, precisely with with the addition of a dense layer improves the entire inference value.

MobileNet has also been widely used in bird sound applications in various studies such as Incze et al. [19], also Saad et al. [14], This model was chosen because its size is relatively small, making it possible to embed it in edge devices. In this paper, the MobileNet V1 0.1 and MobileNet V2 0.35 models were used using input of 16Ksa/second. We implement a dense layer of 128 neurons in MobileNet experiment, the final output will be 5 neurons. Additionally, the model was iterated over 100 epochs, and the learning speed was 0.008 seconds for the CNN model and 0.01 seconds for MobileNet. The accuracy value during validation was 72.7% and during testing 62.43% on MobileNet V1, while the accuracy value of the MobileNet V2 model was 82.2% when testing 80.79%. The classification results are given in Table 4 and Table 5.

We can see from the four models that the uncertain classification values tend to be high (CNN 1D 13.9% and 18.1%, MobileNet V1 48.7%, MobileNet V2 20.8%). This problem can be overcome by providing more training data. However, due to the limitations of Edge Impulse's free license, the accuracy value in this research is not the primary objective. Models that have not yet been optimized will be considered against adequate resource capabilities.

Figure 8. Model preset on Edge Impulse

Table 2. Confusion Matrix for classification on testing using deep custom CNN 1D

	Lonchura Mada	House Sparrow t1	House Sparrow t2	Quiet	Turtle-dove	Uncertain
Lonchura Mada	84.3%	3.7%	0.9%	3.7%	0.9%	6.5%
House Sparrow t1	0%	83.3%	0%	0%	0%	16.7%
House Sparrow t2	0%	1.0%	82.8%	5.1%	0%	11.1%
Quiet	0%	6.9%	0%	72.2%	2.8%	18.1%
Turtledove	0%	0%	0%	8.3%	88.9%	2.8%
F1 Score	0.91	0.80	0.90	0.76	0.90	

Table 3. Confusion Matrix for classification on testing using CNN 1D 10 Neuron

	Lonchura Mada	House Sparrow t1	House Sparrow t2	Quiet	Turtle-dove	Uncertain
Lonchura Mada	79.6%	6.5%	0.9%	9.3%	0%	3.7%
House Sparrow t1	0%	100%	0%	0%	0%	0%
House Sparrow t2	0%	2.0%	85.9%	7.1%	0%	5.1%
Quiet	0%	5.6%	1.4%	77.8%	1.4%	13.9%
Turtledove	0%	0%	0%	8.3%	88.9%	2.8%
F1 Score	0.89	0.87	0.91	0.76	0.93	

Table 4. Confusion Matrix for classification on testing using MobileNet V1

	Lonchura Mada	House Sparrow t1	House Sparrow t2	Quiet	Turtle-dove	Uncertain
Lonchura Mada	52.8%	5.6%	2.8%	5.6%	0%	33.3%
House Sparrow t1	15.4%	35.9%	0%	0%	0%	48.7%
House Sparrow t2	3.0%	0%	76.8%	1.0%	0%	19.2%
Quiet	1.4%	2.8%	0%	65.3%	1.4%	29.2%
Turtledove	0%	0%	0%	0%	75%	25%
F1 Score	0.65	0.46	0.85	0.75	0.84	

Table 5. Confusion Matrix for classification on testing using MobileNet V2

	Lonchura Mada	House Sparrow t1	House Sparrow t2	Quiet	Turtle-dove	Uncertain
Lonchura Mada	82.4%	5.6%	0.9%	2.8%	0%	8.3%
House Sparrow t1	0%	100%	0%	0%	0%	0%
House Sparrow t2	0%	0%	84.8%	3.0%	0%	12.1%
Quiet	2.8%	6.9%	0%	69.4%	0%	20.8%
Turtledove	0%	0%	0%	8.3%	66.7%	25%
F1 Score	0.89	0.88	0.91	0.76	0.80	

4. ANALYSIS

When using Edge Impulse, there are two options: the first is to create a code library with various programming language options, and the second is to generate binary code that functions as firmware. In this research, the Arduino Library was chosen to be embedded into the Arduino RP2040 Connect board. Arduino RP2040 Connect board has an ARM Cortex M0+ clock speed of up to 133MHz 32-bit, 264 kB SRAM, 16 MB Flash memory, and high-performance energy efficiency. Additionally, it is also equipped with a 6-axis Inertial Measurement Unit (IMU), gyroscope, PDM, temperature sensor, WIFI, and Bluetooth 5.0.

Arduino libraries are built for each model (1D CNN and MobileNet) using the EON or TFLite in quantized (8-bit) and floating-point (32-bit) representations. The research results are plotted using four different values per model.

To ensure that the architectural model we use can work well on the targeted embedded chip, namely the RP2040 connect, it is necessary to compare the inference of each architectural model, especially how much the model uses memory (SRAM) and how much Storage utilization (Flash), for that we have previously also conducted similar research using other types of Tiny ML with slightly smaller SRAM and flash capacities, namely the Seeeduino NRF52840 Sense [20], so based on the results of this research, we then made it a reference again on the ML RP2040 Connect board, of course with adjustments to the calculation of larger memory and RAM capacity, although this has a small effect, but the ML RP2040 Connect Board here is easier to get than the Seeeduino Board, let alone the Tiva C.

Deployment to Edge Impulse calculates memory usage and time per inference for each model based on deployment board options. This software provides both unoptimized float models (F) and quantized models (Q) using EON or TensorFlowLite interpreters. The Arduino libraries for the four models can be downloaded and embedded separately to measure the board's reliability. Table 6 shows the results of the inference values for the RP2040 Connect microcontroller.

Figure 9 shows the results of the classification accuracy of training process on each models architecture where all model representing the accuracy in the training process moves up at each epoch.

Figure 10 shows loss chart of training process on each model's architecture where all model representing the loss in the training process decrease at each epoch.

Table 6. The deployed classification model result inference on Raspberry RP2040 Connect

	Quantized (EON)	Float (EON)	Quantized (TFLite)	Float (TFlite)
CNN 1D				
RAM (KB)	11.8 K	11.8 K	11.8 K	11.8 K
Flash (KB)	41.6 K	64.4 K	62.5 K	91.3 K
Accuration(%)	82.07%	81.79%	82.07%	81.79%
Time (ms)	841 ms	1004 ms	841 ms	1004 ms
CNN 1D 10 Neuron				
RAM (KB)	11.8 K	11.8 K	11.8 K	16,2 K
Flash (KB)	45.0 K	77.3 K	66.2 K	104.4 K
Accuration(%)	84.59%	84.31%	84.59%	84.31%
Time (ms)	844 ms	1004 ms	844 ms	1004 ms
MobileNet V1 0.1 128 Neuron				
RAM (KB)	37.0 K	54.9K	37.8K	70.8K
Flash (KB)	117.8K	230.9K	159.4K	275.4K
Accuration(%)	61.86%	62.43%	61.86%	62.43%
Time (ms)	489 ms	1607 ms	489 ms	2189 ms
MobileNet V2 0.35 128 Neuron				
RAM (KB)	216.3K	399.9K	205.3K	488.5K
Flash (KB)	730.2K	2.2M	833.4K	2.2 M
Accuration(%)	77.97%	80.79%	77.97%	80.79%
Time (ms)	1457 ms	9633 ms	1457 ms	9633 ms

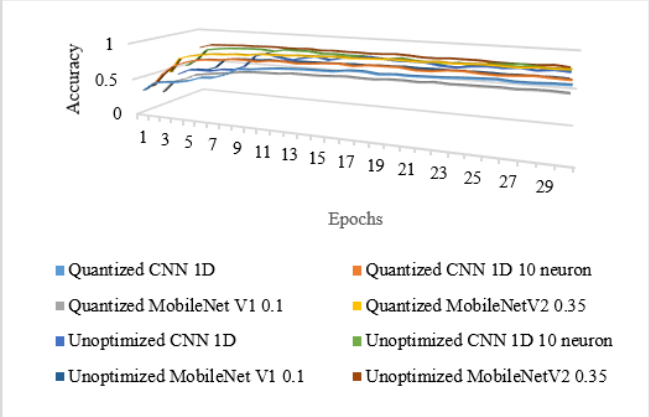

Figure 9. Accuracy chart on training process

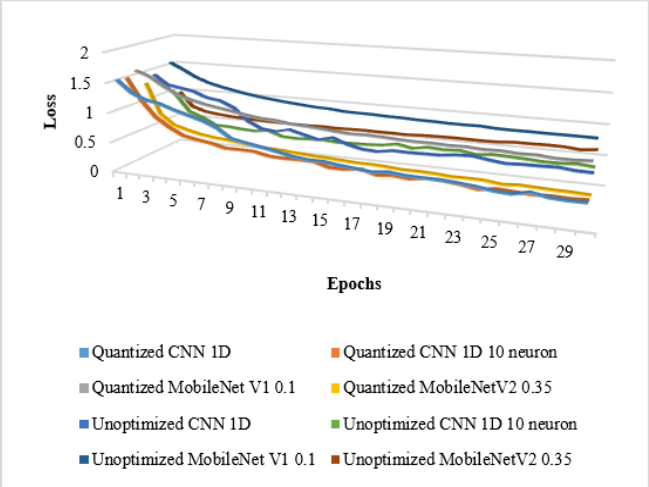

Figure 10. Loss chart on training process

5. DISCUSSION

System performance using embedded machine learning and the experiments that have been carried out are influenced by different factors. The accuracy value of the custom 1D CNN model is 3.52% higher than MobileNet V2 in unoptimized 32-bit float and 6.6% high in 8-bit quantized. This difference in accuracy usually occupies more memory and takes longer to infer. However, in this research, the two models use different sampling frequencies (where 1D CNN allows using 8Ksa/sec and MobileNet with it is MFE feature extraction requires a minimum of 16Ksa/sec) so that better accuracy values are obtained with the 1D CNN model without sacrificing memory and inference time.

In some cases of unoptimized float, it seems not possible to implement the MobileNet-V2 model to Raspberry RP2040 Connect chip without quantization because it requires 1.85 times and 1.51 times the available RAM for the TensorFlowLite models and EON, Even though the Flash memory size is available 86.25% of the total memory in floating point on both TensorFlowLite and EON. Meanwhile, with its unoptimized float, even the 1D CNN model still occupies not more than half of the available space (RAM or Flash) for both the Edge Optimized NN model and the TensorFlowLite model. However, for optimal embedding, microcontroller machine learning algorithms focus on the 8-bit optimized model.

After optimization, the MobileNet V2 model size reduces Flash by a factor of 1.89, and The MobileNet V1 compressed by a factor of 1.48 for TensorFlowLite and EON. RAM usage consumes more than 50% of resources while using MobileNetV2, with a decrease after optimation of a factor of 1.84 for the Edge Optimized NN and 2.38 for the TensorFlowLite. However, if using the MobileNet V1 model, it is still possible to do inference on board using both quantized and float-32 options. There is no memory footprint reduction value on the 1D CNN model after optimation (Q). Still, it reduces Flash memory usage by 36.59% for TensorFlowLite and reduces Flash memory usage by 41.79% for EON.

The optimation of the CNN model in the research did not reduce the memory footprint but reduced the use of Flash; however, in the experiments we conducted, there was an increase in accuracy after quantization; 1D CNN after quantization, we found accuracy increased by 0.28%, MobileNet on the other hand reduced accuracy by 0.57% after quantization due to a decrease in the required memory footprint. Memory footprint requirements in the ratio of Flash usage and available RAM are depicted in sixteen different cases visualized in Figure 11 and Figure 12.

In this paper, the accuracy values with various combinations of learning rates on Figure 13, the accuracy values of the MobileNet V1 model tend to be lower than other models. Even when compared with MobileNet V2, as the comparison carried out by Tan et al., the difference in accuracy values between the two versions of the transfer learning model can be 4% to 21.9% [21]. We also found a difference in accuracy of around 16.11% to 18.36% in this study where the Mobilenet V2 model had accuracy values that tended to be better, although not as good as the 1D CNN model. The custom 1D CNN model provides the best accuracy result value of 87.4% accuracy during training, and 84.59% on testing. As seen in Figure 13 and Figure 14, 1D CNN shows the highest accuracy results when the learning rate value is 0.008 second.

According to the result of TensorFlowlite or EON (Edge Optimized NN) on model deployment and time inference, we know clearly that TensorFlowLite requires more resources than Edge Optimized NN. However, the disadvantage of using Edge Optimized NN is that it is difficult to change the model, contrary to TensorFlowLite. Edge Optimized NN does not impact the accuracy or inference time because it still runs TensorFlowLite in the background [22]. The accuracy test results for both models show that the deep 1D CNN model has a generally higher accuracy value; however, the inference time for the Edge Optimized NN and TensorFlowLite 8-bit quantized(Q) models appears to be alike. The MobileNetV1 model has a time difference of 355 ms, with MobileNetV1 optimized through quantization having a faster inference time than 1D CNN. In the MobileNetV2 model, there is a difference of 613 ms, using MbileNetV2 after quantizing having a lagged inference time compared to CNN 1D.

From all the experiments and running inferences carried out, it can be concluded that using 8 Khz audio 1D CNN 2 layers with an Edge Impulse framework and then embedding it on the low-power Arduino RP2040 Connect board is more than enough to detect the sound of Lonchura Mada birds and house sparrow birds to support the work of automatic bird repellents in rice fields. With this optimization, the use of excess resources, such as the use of batteries or excess power, can be reduced. Farmers' losses due to bird attacks are also expected to impact Rp significantly. 2040-connect is also relatively easy to buy here; if we want to make a cheaper bird-

repellent tool with an internal PDM microphone, we can also embed the results of this inference model on the super cheap Seeed Studio NRF52840 Sense microcontroller, which in terms of specifications is only slightly different from the RP2040-Connect. However, the weakness of this embedded system is that it takes work to update instantly.

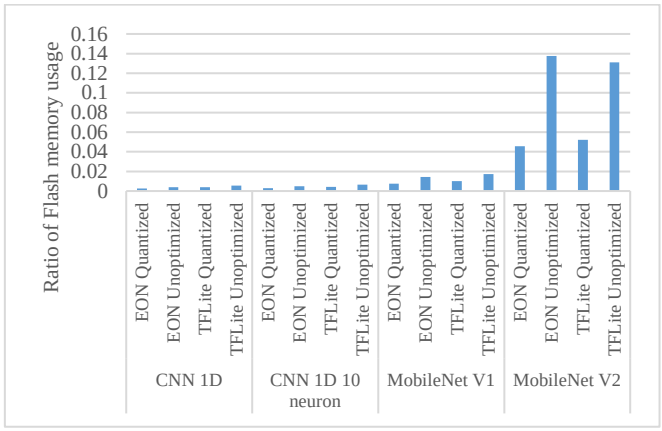

Figure 11. Ratio of RP2040 Connect memory usage for CNN 1D and MobileNet models using Edge Optimized NN and TensorFlowlite interpreter in unoptimized float-32 (F) and quantized(Q)

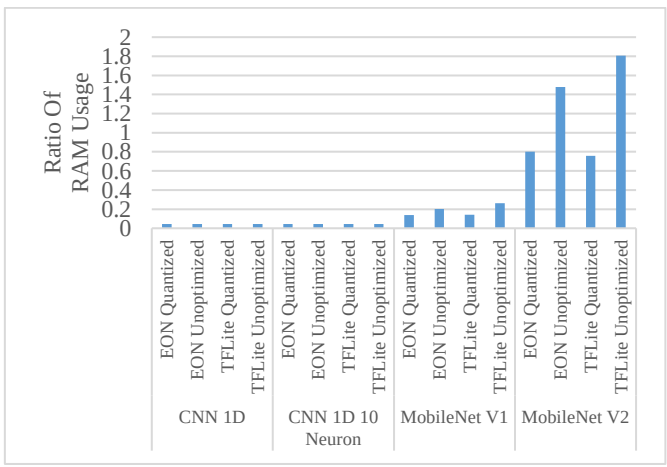

Figure 12. Ratio of RP2040 Connect RAM usage for CNN 1D and MobileNet models using Edge Optimized NN and TensorFlowLite interpreter in unoptimized float-32(F) and quantized(Q)

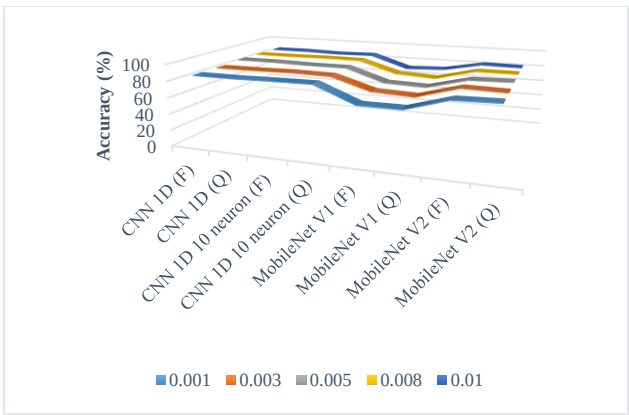

Figure 13. Accuracy result chart based on given learning rate on training process

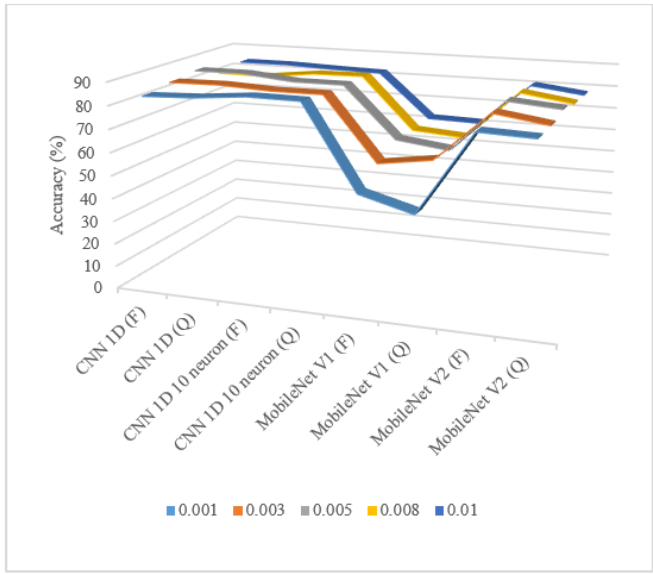

Figure 14. Accuracy result chart based on given learning rate on testing process

6. CONCLUSIONS

This series of experiments shows that a custom 1D CNN model has been created using Edge Impulse with a sample audio frequency of 8Ksa/second with MFCC feature extraction. The classification of the sound of the rice pest birds Lonchura Maja and sound of the rice pest bird house sparrow, which is carried out using a custom method using embedded machine learning, produces sound recognition that allows it to be embedded in the small microcontroller RP2040 Connect. This microcontroller is very efficient in power consumption. The 1D two-layer CNN method provides voice recognition accuracy of 87.4% on training and 84.59% on testing using 66.2 Kbyte Flash memory with 11.8 Kbyte Peak RAM. Suppose you use the MobileNet V2 models for almost the same accuracy value. In that case, you need a Flash memory size that is 13 times larger and a RAM size that is around 17 times larger due to the MFE feature extraction requiring a significant sample frequency of 16Ksa/second. Therefore, it is only possible to embed it on this RP2040 Connect board using the MobileNet V2 Model with the quantized option.

ACKNOWLEDGMENT

We extend our heartfelt thanks to the Faculty of Agriculture and the Faculty of Engineering at the University of Trunojoyo Madura - Indonesia for enabling scientists to complete this research. We also express our gratitude to the investigative team at the Multimedia and Networks Laboratory for supporting the relevance and impact of this research.

REFERENCES

[1] Bana, M.S., Rahmawati, D., Joni, K., Ulum, M. (2021). Rancang bangun alat pengusir tikus dan burung pada tanaman padi. J-Eltrik, 2(1): 53. https://doi.org/10.30649/j-eltrik.v2i1.53

[2] Abubakar, S.M., Ahmed, S. (2023). A smart and secure

agricultural system using IoT. In 2022 OPJU International Technology Conference on Emerging Technologies for Sustainable Development (OTCON), Raigarh, Chhattisgarh, India, pp. 1-6. https://doi.org/10.1109/OTCON56053.2023.10113901

[3] Muminov, A., Jeon, Y.C., Na, D., Lee, C., Jeon, H.S. (2017). Development of a solar powered bird repeller system with effective bird scarer sounds. In 2017 International Conference on Information Science and Communications Technologies (ICISCT), Tashkent, Uzbekistan, pp. 1-4. https://doi.org/10.1109/ICISCT.2017.8188587

[4] Chéour, R., Khriji, S., Kanoun, O. (2020). Microcontrollers for IoT: Optimizations, computing paradigms, and future directions. In 2020 IEEE 6th World Forum on Internet of Things (WF-IoT), New Orleans, LA, USA, pp. 1-7. https://doi.org/10.1109/WF-IoT48130.2020.9221219

[5] Anil, L., Khandetod, Y.P., Mohod, A.G., Dhande, K.G., Aware, S.V. (2023). Development of solar PV powered and wind operated deterrent system. Pharma Innovation Journal, 12(10): 530-535.

[6] Pérez-Granados, C., Traba, J. (2021). Estimating bird density using passive acoustic monitoring: A review of methods and suggestions for further research. Ibis, 163(3): 765-783. https://doi.org/10.1111/ibi.12944

[7] Viswanatha V., Ramachandra A.C, Prasanna, R., Kakarla, P.C., PJ, V.S., Mohan, N. (2022). Implementation of tiny machine learning models on arduino 33-ble for gesture and speech recognition. Electr. arXiv preprint arXiv: 2207.12866. https://doi.org/10.48550/arXiv.2207.12866

[8] Yu, W., Zhao, C. (2020). Broad Convolutional Neural Network based industrial process fault diagnosis with incremental learning capability. IEEE Transactions on Industrial Electronics, 67(6): 5081-5091. https://doi.org/10.1109/TIE.2019.2931255

[9] Song, J., Cho, Y., Park, J.S., Jang, J.W., Lee, S., Song, J.H., Lee, J.G., Kang, I. (2019). 7.1 An 11.5 TOPS/W 1024-MAC butterfly structure dual-core sparsity-aware neural processing unit in 8nm flagship mobile SoC. In 2019 IEEE international solid-state circuits conference-(ISSCC), San Francisco, CA, USA, pp. 130-132. https://doi.org/10.1109/ISSCC.2019.8662476

[10] Li, B., Najafi, M.H., Yuan, B., Lilja, D.J. (2018). Quantized neural networks with new stochastic multipliers. In 2018 19th International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, USA, pp. 376-382. https://doi.org/10.1109/ISQED.2018.8357316

[11] Najafi, M.H., Lilja, D.J., Riedel, M. (2018). Deterministic methods for stochastic computing using low-discrepancy sequences. In 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), San Diego, CA, USA, pp. 1-8. https://doi.org/10.1145/3240765.3240797

[12] Yang, L., Jiang, W., Liu, W., Edwin, H.M., Shi, Y., Hu, J. (2020). Co-exploring neural architecture and network-on-chip design for real-time artificial intelligence. In 2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC), Beijing, China, pp. 85-90. https://doi.org/10.1109/ASP-DAC47756.2020.9045595

[13] Dai, W., Dai, C., Qu, S., Li, J., Das, S. (2017). Very deep Convolutional Neural Networks for raw waveforms. In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, pp. 421-425. https://doi.org/10.1109/ICASSP.2017.7952190

[14] Saad, A., Ahmed, J., Elaraby, A. (2022). Classification of bird sound using high-and low-complexity Convolutional Neural Networks. Traitement du Signal, 39(1): 187-193. https://doi.org/10.18280/ts.390119

[15] Goëau, H., Glotin, H., Vellinga, W.P., Planqué, R., Joly, A. (2016). Lifeclef bird identification task 2016: The arrival of deep learning. In CLEF: Conference and Labs of the Evaluation Forum, 1609: 440-449. https://hal.science/hal-01373779/file/16090440.pdf.

[16] Küçüktopcu, O., Masazade, E., Ünsalan, C., Varshney, P.K. (2019). A real-time bird sound recognition system using a low-cost microcontroller. Applied Acoustics, 148: 194-201. https://doi.org/10.1016/j.apacoust.2018.12.028

[17] Edge Impulse. https://edgeimpulse.com, accessed on Apr. 7, 2024.

[18] Diab, M.S., Rodriguez-Villegas, E. (2022). Performance evaluation of embedded image classification models using Edge Impulse for application on medical images. In 2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), Glasgow, Scotland, United Kingdom, pp. 2639-2642. https://doi.org/10.1109/EMBC48229.2022.9871108

[19] Incze, A., Jancsó, H.B., Szilágyi, Z., Farkas, A., Sulyok, C. (2018). Bird sound recognition using a Convolutional Neural Network. In 2018 IEEE 16th international symposium on intelligent systems and informatics (SISY), Subotica, Serbia, pp. 000295-000300. https://doi.org/10.1109/SISY.2018.8524677

[20] Hasbullah, A. W. Setiawan, E. Rachmad, A. (2023). Evaluasi Keandalan Model Rekognisi Suara Burung Hama Menggunakan Platform Edge Impulse Pada Mikrokontroller Low Power. Jurnal Teknik Elektro dan Komputer TRIAV, 10(2): 69-75. https://doi.org/10.21107/triac.v10i2.22448

[21] Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., Le, Q.V. (2019). MnasNet: Platform-aware neural architecture search for mobile. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, pp. 2815-2823. https://doi.org/10.1109/CVPR.2019.00293

[22] Jongboom, J. (2020). Introducing EON: Neural networks in up to 55% less RAM and 35% less ROM. https://www.edgeimpulse.com/blog/introducing-eon.