POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Acoustic Identification of Wood-Boring Insects with TinyML

Laurea Magistrale in Computer Science and Engineering - Ingegneria Informatica

Author: Lorenzo Mainetti

Advisor: Prof. Manuel Roveri

Co-advisors: Prof. Paul G. Flikkema, Massimo Pavan

Academic year: 2022-2023

1. Introduction

Wood-boring insects are a potential threat to several forest ecosystems around the world. An unstable infestation might result in heavy forest losses and subsequent future environmental impacts. The climate warming of the last decades has allowed these insects to reproduce faster and expand in previously unsuitable habitats. Infestation prevention is nowadays a relevant issue, and acoustic monitoring of the area at risk is one of the most effective prevention techniques. For an efficient monitoring system, it is necessary to pervasively deploy multiple devices across a forest. However, constant data transmission to the cloud makes it impractical for these devices to maintain prolonged battery life. Hence, this thesis aims to devise a solution capable of functioning on-device, thereby making the system practically feasible. The thesis introduces an innovative and efficient method for identifying and categorizing wood-boring insects within trees through the analysis of their acoustic behavior with deep learning. The auditory data is gathered in real-time by an embedded device, which also carries out the data processing and inferencing, directly on edge. In this regard, the work is within the scope of TinyML, a field of study at the intersection of Machine Learning and Embedded Systems that focuses on developing models that can run on small, low-powered devices. This poses significant challenges due to severe constraints in terms of memory, computation, and power consumption of the devices. To cope with this complexity, the core of the proposed solution comprises a multi-task convolutional neural network characterized by the use of 1D convolutions to handle the raw audio data and quantization mechanisms to reduce the computational and memory demands.

2. Background

2.1. Tiny Machine Learning

TinyML is an interdisciplinary field that combines Embedded Systems and Machine Learning, with a specific focus on deploying ML models on small, low-power devices [5]. The primary objective of this field is designing approximated deep learning solutions suitable for resource-constrained environments. To achieve this, various techniques have been devised, including weight quantization, pruning, knowledge distillation, and Gate-Classification CNNs. These methods facilitate the adaptation of complex algorithms for deployment on such devices. In recent years, several applications of TinyML have garnered significant attention. Notably, the analysis of audio and visual data for predictive maintenance in industrial settings has gained traction. Additionally, audio and visual wake word detection has emerged as a popular application. Typically, these solutions are limited to binary tasks, discerning only the presence or absence of a specific target. Even if less common, some studies focus on traditional machine learning algorithms, such as k-nearest neighbors, decision trees, or even new ad hoc approaches.

2.2. Multi-task learning

Multi-Task Learning (MTL) is an active research field in machine learning. It is a learning paradigm that aims to jointly learn several related tasks to improve their generalization performance by leveraging common knowledge among them [2]. The majority of the literature is focused on MTL applied to the deep learning domain. In this regard, any MTL algorithm shares some characteristics:

- The model usually has some layers that are shared across all tasks, and other layers that are specific to each task.
- The model is trained on all tasks simultaneously. During each step of training, the model might be updated based on a single task, or on a mixture of tasks.
- The loss function that the model is trained to minimize is usually a weighted sum of the loss functions for each task. The weights determine the importance of each task in the overall learning objective.

Thereby the recent studies of MTL mainly focus on two perspectives, network architecture design, and weighting strategies. In the design of network architectures, hard parameter sharing (HPS) is a popular network design method where a shared trunk and task-specific heads are used. Soft parameter sharing (SPS) differs by providing each task with a unique trunk and encouraging parameter similarities between them. More complex methods, like cross-stitch networks (CSN), allow more flexible representation sharing [3]. Balancing multiple losses corresponding to multiple tasks is another way to deal with task relationships since the shared parameters are updated by all the task losses. Thus, different weighting strategies have been proposed to balance losses or gradients.

To our best knowledge, MTL was never applied to the tinyML domain before.

2.3. Related work

The problem of wood-boring insects' automated acoustic detection is quite common in the literature. Different solutions have been proposed ranging from simpler signal processing techniques to more sophisticated machine learning and deep learning approaches. The wide majority of the related works include a lowpass or bandpass filter as the first preprocessing step. Bedoya et al. [1] extensively studied the stridulatory sounds produced by bark beetles. They propose an energy-based segmenter to extract the stridulations from the recordings; then, several classical machine learning algorithms are employed to discern between species using pre-extracted audio features. Rigakis et al. [4], instead, compared five popular convolutional neural network architectures, feeding audio recordings as spectrograms (computed with the Short-Time Fourier Transform), a 2D visual representation of the spectrum of frequencies of a signal as it varies with time. Numerous other techniques have been proposed, some inspired by anomaly detection for time series such as rolling window approaches, some energy-based such as ultra short time energy detection, and some neural such as autoencoders and recurrent neural networks. To our best knowledge, none of these methods was tailored to a constrained environment and ported to an embedded device. Indeed, the standard workflow described in the literature consists of a device responsible for capturing the recordings through a sensor and applying minor preprocessing to the audio data, before sending it to a remote server, where the actual analysis is carried out. Furthermore, the use of MTL for wood-boring insect detection and classification was never applied before to this domain setting.

3. Problem formulation

In a real application, a piezoelectric sensor will collect signals in real-time from the tree according to a certain duty cycle. After analog-to-digital conversion, the digital audio signal will be fed to the embedded device as a stream of discrete data. The embedded device will chunk the stream in frames, it will process the frames one by one and output a prediction.

The focus of the work is to enable the execution of detection (or binary classification) and multi-class classification of insect sounds in trees on highly constrained embedded devices. The detection task is particularly delicate, since detecting infestation of any sort should be the primal goal of the system. The task consists in distinguishing frames of the input stream that contains some insect sounds from background noise and disturbances (such as environmental sounds, device vibrations, electrical noise, etc.). While the classification task consists in identifying what class of insect is infesting a tree, which can be a useful insight to decide how to act and what pest control method to introduce. The problem has been formalized as a combination of a binary classification task and a multi-class classification task. Specifically, for each input x_t, the aim is to assign two labels: d_t, which can assume values only in $\{0, 1\}$ and y_t as described by the following functions:

$$d_t = f(x_t) = \begin{cases} 1 & \text{if insect activity is present in } x_t \\ 0 & \text{otherwise} \end{cases}$$

$$y_t = g(x_t) = \begin{cases} \alpha & \text{if activity of insects of class } \alpha \text{ present in } x_t \\ 0 & \text{if no activity, or not related to insects} \end{cases}$$

The objective is to achieve the best possible approximation of functions f and g by training a model using the available dataset.
As previously mentioned, the specificity of this problem setting lies in the technological constraints posed, in terms of memory and computational power, by the target embedded device described in *Section 6*. Precisely, the available RAM is 64 kB and the available Flash memory is 256 kB.

4. Proposed solution

Two different and somehow antithetical approaches have been developed to solve the problem at hand. The main proposed solution revolves around a multi-task architecture, which is novel to this domain. It is based on the idea of squeezing the stages, thanks to the usage of 1D convolutions and MTL. This approach represents the major contribution of the work, while the other, a cascade approach made of preprocessing, detection, and classification stage, will serve as a comparison to highlight the various trade-offs.

4.1. Multi-task approach

The proposed architecture presents a small preprocessing stage, which sequentially applies a Butterworth bandpass filter to isolate the frequency band of interest (i.e., 200 Hz - 20 KHz according to the literature) and a z-score normalization.
The inference stage implements an HPS multitask algorithm, which simultaneously optimizes the detection and classification tasks. The algorithm is trained in a supervised manner on the dataset described in *Section 5.1.2*. The loss function to be minimized is:

$$Loss = \lambda L_1 + (1 - \lambda) L_2$$

where L_1 is the Binary Cross-Entropy loss for the detection task, L_2 is the Sparse Categorical Cross-Entropy loss for the classification task, and λ is a weighting term. Post-training integer-only (int8) quantization is applied to the network to cope with the device constraints.
An overview of the proposed neural network, named MAINet (Multi-task Audio Insect Net), is given in *Figure 1*. More specifically, MAINet receives in input a raw audio time-series x_t of size $M \times 1$, and produces as output $d_t \in \{0, 1\}$ representing respectively the "clean" and the "infested" class, and $y_t \in \{0, 1, 2\}$ representing respectively the classes "background", "big mandibles", and "small mandibles", in accordance with the generated dataset. In detail, the proposed MAINet comprises:

- a shared 1D convolutional feature extractor composed of a sequence of K 1D convolutional blocks. Each block is organized into a 1D convolutional layer (characterized by a number f of filters with a kernel size of r) to extract the main features of the audio, a ReLU activation function, and a Max Pooling 1D layer, used to reduce the dimension of the activation map;
- a Global Average Pooling 1D layer, which takes the average of all the values for each feature map generated by the preceding convolutional layer, reducing the output to a 1D tensor while maintaining the depth;
- a fully connected part, made of a shared dense layer and two task-specific heads. Each head is composed of a dense layer and an output

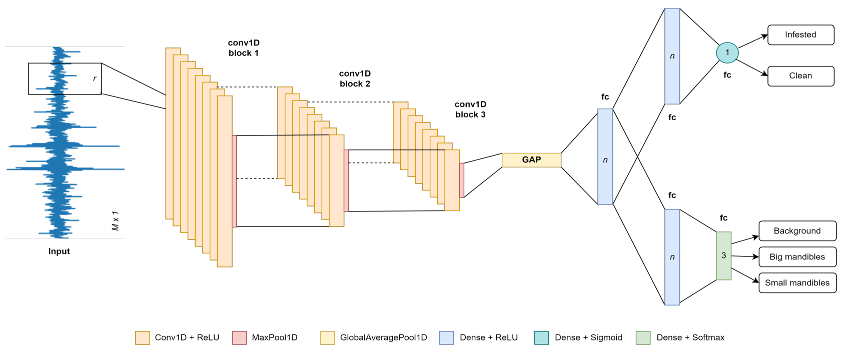

Figure 1: An overview of the proposed MAINet.

layer. The dense layers are characterized by a number n of neurons and a ReLU activation. The output layers respectively use a Sigmoid activation for the detection task and a Softmax activation for the classification task.

Conv1D and MaxPool1D are the key layers behind the described solution as they enable a purely time-domain approach on the raw digital signal, which provides a fast inference avoiding time-consuming feature extraction. The multi-task approach produces a faster and more compact model than using two separate networks. Furthermore, simultaneously optimizing the two tasks reduces overfitting, by learning a shared underlying structure across them. Another relevant advantage is related to MAINet's tiny architecture, which guarantees small energy and memory requirements.

5. Experiments

5.1. Dataset

No dataset for wood-boring insect sounds classification is publicly available. Among the few accessible datasets, TreeVibes [4] is the most relevant, but the data is annotated only with binary ground truth labels since its aim is detection. For this reason, a first attempt to collect a multi-class dataset was carried out, but, due to issues in gathering different insect sources in a controlled lab environment, the actual dataset used in this work for training and evaluation has been generated. We have tested the solution also on the collected dataset.

5.1.1 Data collection

A preliminary dataset was collected from October to December 2022 in Flagstaff, AZ using various pieces of equipment, including a TASCAM DR-40 Linear PCM Recorder, a Mackie 402VLZ4 mixer with an amplifier, and a custom-made piezoelectric sensor. The sensor was created by attaching a metal pin to a piezoelectric transducer, which was then inserted between the outer and inner bark layers (phloem) of a tree log. The sensor was linked to the mixer using a Female RCA to Male XLR adapter. The recorder was also connected to the mixer. All data were collected in a laboratory setting to minimize external disturbances, such as weather conditions and wildlife sounds. Two Ponderosa Pine logs were sourced from a local forest, one of which was infested with Longhorn beetles, while the other was not. Recordings were subsequently made at varying times throughout the day and under diverse noise conditions over several months. Recordings were captured at a sampling frequency of 44.1 KHz to ensure the potential detection of stridulation sounds, which can reach up to 20 KHz. Post-collection, the data was divided into 10-second .wav files, each receiving a label. Files originating from the uninfested log were marked as "clean", while those from the infested log underwent individual examination. Only files with evidence of insect sound activity received an "infested" label. Ultimately, the dataset comprised 425 clean and 89 infested .wav files.

5.1.2 Data generation

The methodology for generating the dataset relies on the understanding and information gleaned from academic literature, the data we have gathered, and other existing data sources. The idea revolves around overlaying insect pulses on a clean background and incorporating data augmentation techniques to enhance diversity. The three main required elements are:

- **Background**: comprises randomly chosen samples from a subset of the clean files in our collected dataset.
- **Acoustic activity pulses**: which include insect sounds and disturbances, previously isolated from different data sources, namely the Bedoya dataset [1], part of our collected dataset (*Section 5.1.1*) that was excluded from the evaluation set, and other available files collected by an entomologist.
- **Data augmentation**: multiple signal augmentation techniques (such as noise injection, pitch shifting, time shifting, fading) are applied to the pulses both before and after overlaying them onto the background.

For each data instance, we introduce a number of pulses that is randomly selected from a uniform distribution between 1 and 8. The first pulse is inserted randomly, and any subsequent pulses follow the first one in accordance with a pulse silence duration, which is randomly chosen from a uniform distribution ranging between 3 and 15 ms. The ranges are based on the acquired domain knowledge. As a result, a synthetic multi-class dataset has been generated following this described method. The generated files have a duration of 50 ms with a sampling frequency of 44.1 KHz, translating to 2205 data points per file, which corresponds to the window over which the device is expected to work. The choice for this length was guided by a literature analysis on the duration of insect sounds, as well as the limitations set by the embedded device (2205 float32 values require 8.8 kB of RAM). For the labeling criteria, we have opted for a classification specificity based on insect mandible size rather than species, leading to the formation of three classes: "background" (which encompasses both noise and disturbances), "big mandibles" (such as longhorn beetles), and "small mandibles" (like bark beetles). The dataset consists of one thousand files per class. The data generation process is flexible and could be adapted even to other problem settings, changing the basic required elements according to the domain of interest. It exposes a method that lets users choose the duration of the generated audio files, their number, and the required sampling frequency.

5.2. Experimental results

5.2.1 The proposed MAINet

The structure of the proposed MAINet comprises $K = 3$ convolutional blocks, $f = 8$ filters per block, and a kernel size $r = 80$ in the first block and $r = 3$ in the latter two. Each dense layer has $n = 32$ neurons. The network has been trained for 100 epochs with $\lambda = 0.5$, early stopping (patience of 10 epochs) and Adam optimizer. The inputs have $M = 2205$ corresponding to 50 ms of audio data.

5.2.2 Comparison with MTL algorithms

Two other MTL methodologies (i.e., SPS and CSN) have been tried and implemented, and their results, obtained with the same hyperparameters of the proposed solution, are compared in *tab. 1* and *tab. 2*. The two tasks are evaluated separately. For the detection, false alarm rate (FPR), miss detection rate (FNR), F1 score, and binary accuracy are reported. For the classification task, multi-class accuracy is reported. The listed results are obtained by making predictions on 20% of the synthetic dataset, which has been held out. The remaining 80% has undergone another 80:20 split between training and validation. Note that all the results have been obtained after converting the Tensorflow model to Tensorflow Lite and applying int8 quantization.

	Detection				Classification
	FPR	FNR	F1	ACC	ACC
MAINet	0.15	0.13	0.90	0.86	0.80
SPS	0.25	0.11	0.89	0.85	0.76
CSN	0.17	0.15	0.88	0.84	0.79

Table 1: MTL algorithms performances.

The superiority of MAINet is demonstrated on both tasks, and it is strengthened by better memory usage due to fewer parameters compared to the other more sophisticated solutions, which also require the introduction of custom layers.

	Params	tflite model size
MAINet	3,580	15.0 kB
SPS	4,916	25.1 kB
CSN	4,928	31.3 kB

Table 2: MTL algorithms memory requirements.

	Performance			Ex time	Memory		Energy
	ACC-D	F1-D	ACC-C	Avg (ms)	Ram (kB)	Flash (kB)	Estimate (μA)
Multi-task approach	0.86	0.90	0.80	104.7 ± 1.2	45.0	76.18	~ 630
Cascade approach	0.72	0.76	0.81	B: 80, W: 235	23.22	153.1	

Table 3: Cascade vs multi-task. D: detector, C: classifier, B: best, W: worst.

5.2.3 Cascade vs multi-task approach

As anticipated in *Section 4*, the proposed multi-task approach has been extensively compared with the cascade approach. Four metrics have been considered: model performance, execution time, memory footprint, and energy consumption. In *tab. 3* are listed the results of the comparison, which are obtained after porting the proposed solutions to the target device, as described in *Section 6*.
The main advantage of the cascade approach is inherent in its design; in the detection stage, an energy-based filter prevents noise-only frames to reach the next stage. Thus, postponing the most expensive part of the architecture and activating it only a limited number of times will for sure enable major energy savings once the device is deployed in a real-world setting. On the other hand, the multi-task approach has its set of advantages as well. Firstly, working directly on the raw audio removes the need for expensive spectrogram extraction, and its impact can be seen in the lower execution time on average. Comparing the performance, the multi-task approach turns out to be a much stronger binary classifier compared to the energy-based filter of the cascade approach. Specifically, the energy detector places a threshold on the moving standard deviation (rolling window of 50 data samples) of the analyzed frame, which is set based on the desired false alarm rate. This is a crude filter where the trade-off between false alarms and missed detections is highly pronounced, thus significantly impacting overall performance. The classification performance is much closer, being both neural classifiers. The cascade approach follows the state of the art for audio classification (Conv2D and spectrograms) and produced slightly better results. It is important to note that the classification stage is affected in the performance by the goodness of the detection stage, indeed miss detections never go through the classifier.
The multi-task approach requires a higher RAM consumption, given by the more complex network and/or library (generated by X-CUBE-AI), but halves the Flash consumption, eliminating the need for the FFT-related files. The energy consumption has been estimated with the power consumption tool (PCC) provided by STM32CubeIDE assuming the device is active for 5 minutes every hour. The PCC tool provides a static measure of the energy based on measurements taken from the MCU datasheet and the active peripherals; a more in-depth analysis will be carried out in the future.

5.3. Evaluation on real data

To prove the validity and the coherence of the developed solution, an evaluation on the real collected dataset has been carried out. The MAINet model already trained on the synthetic dataset is used to make predictions on the real data. The 10-second audio files are cut into windows of 2205 samples to generate the predictions for each window. The predictions over the windows are then re-grouped by setting the final prediction to "infested" according to a heuristic on the number of windows predicted as "infested" by the model. To avoid the introduction of bias when evaluating on the collected dataset, a subset of the collected data (211 clean, 44 infested) was not employed in the generation process. The results listed in *tab. 4* are obtained on that part of the dataset and using different values for the heuristic (column H).

H	FPR (%)	FNR (%)	F1 (%)	ACC (%)
1	34.12	18.18	47.37	68.63
2	13.74	56.82	41.30	78.82
3	9.18	63.64	40.51	81.57
4	7.11	68.18	38.36	82.35
5	4.27	70.45	39.39	84.31

Table 4: Multi-task approach, detection evaluation on real data.

Furthermore, the classification was evaluated as well; indeed, the "infested" files in the collected dataset contain acoustic activity from longhorn beetles belonging to the "big mandibles" class. Using a similar approach of aggregating the predictions, the model was able to correctly predict

36 out of the 44 infested files, while the remaining 8 were predicted as "background". These results demonstrate the goodness of the data generation process and the potential of the approach described, once a real multi-class dataset is available, and the model is retrained on this.

6. Deployment

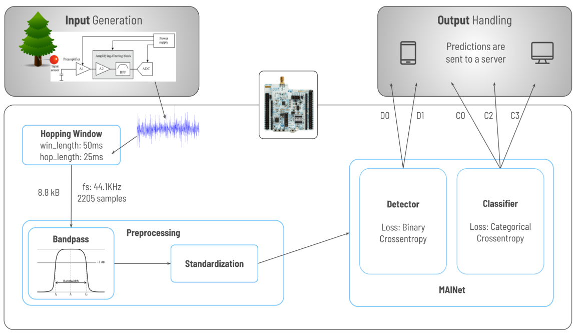

Figure 2: Multi-task approach pipeline.

The proposed solution has been ported to the target embedded device, which is an STM32WL55JC2 of the STM32 NUCLEO-64 development board family. More specifically, the board is equipped with a microcontroller unit (MCU) based on the Arm® Cortex®-M4 core operating at a frequency of up to 48 MHz, and complemented by an Arm® Cortex®-M0+ core. The board embeds high-speed memories (256 kB Flash memory, 64 kB SRAM) and an extensive range of enhanced I/Os and peripherals. A wide software environment is provided through STM32CubeIDE and STM32CubeMX, which support code generation and peripherals setup. *Figure 2* gives a visual representation of the deployed pipeline. The input stream gathered by the piezoelectric transducer is chunked with a hopping window. The bandpass filter and the z-score normalization operations, implemented with the ARM CMSIS-DSP package, are applied to the chunk of data. Afterward, the inference stage, implemented with the help of the X-CUBE-AI library, is activated. Firstly it converts the windowed stream to int8, being the neural network quantized, and then produces the predictions that are sent to the server for visualization.

7. Conclusions

In this thesis, we have proposed a novel approach to solve the relevant real-world problem of wood-boring insects detection and classification. At the time of delivery, this is also the first time this problem has been framed as a TinyML problem, and the proposed solution has been ported to an embedded STM32 development board, with heavy memory constraints. We also proposed and developed an extensible approach to generate synthetic audio files to cope with the data scarcity, and we showed its consistency through the experiments. We compared the proposed multi-task approach with another state-of-the-art inspired solution, highlighting the typical trade-offs considered in the Embedded Systems domain. Measurements have shown that our multi-task solution has a faster execution time (compared to the cascade approach activating the classification stage), better detection performance, and high Flash memory savings.
Future directions involve collecting a real multi-class dataset for wood-boring insects, designing a complete embedded device, and conducting in-depth research on wood properties. Other potential future work may encompass exploring pest control strategies, investigating insect localization, and scaling the system with an LPWAN network. The ultimate goal is to deploy the system in a real forest and evaluate its performance.

References

[1] Carol L Bedoya et al. Experimental characterization and automatic identification of stridulatory sounds inside wood. *Royal Society Open Science*, 9(7):220217, 2022.

[2] Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

[3] Michael Crawshaw. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*, 2020.

[4] Iraklis Rigakis et al. Treevibes: modern tools for global monitoring of trees for borers. *Smart Cities*, 4(1):271–285, 2021.

[5] Pete Warden and Daniel Situnayake. *Tinyml: Machine learning with tensorflow lite on arduino and ultra-low-power microcontrollers.* O'Reilly Media, 2019.