POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Enhancing Drone Spectra Classification: A Study on Data-Adaptive Pre-processing and Efficient Hardware Deployment

Tesi di Laurea Magistrale in
Computer Engineering - Ingegneria Informatica

Author: **Dario Del Gaizo**

Student ID: 996790
Advisor: Prof. Davide Martinenghi
Co-advisors: Prof. Sarunas Girdzijauskas
Academic Year: 2022-23

Abstract

Focusing on the problem of Drone vs. Unknown classification based on radar frequency-amplitude spectra using Deep Learning (DL), especially 1-Dimensional Convolutional Neural Networks (1D-CNNs), this thesis aims at reducing the current gap in the research related to adequate pre-processing techniques for hardware deployment. The primary challenge tackled in this work is determining a pipeline that facilitates industrial deployment while maintaining high classification metrics. After presenting a comprehensive review of existing research on radar signal classification and the application of DL techniques in this domain, the technical background of signal processing is described to provide a practical scenario where the solutions could be implemented. A thorough description of technical constraints, such as Field Programmable Gate Array (FPGA) data type requirements, follows the entire project justifying the necessity of a learning-based pre-processing technique for highly skewed distributions. The results demonstrate that data-adaptive pre-processing eases hardware deployment and maintains high classification metrics, while other techniques contribute to noise and information loss. In conclusion, this thesis contributes to the field of radar frequency-amplitude spectra classification by identifying effective methods to support efficient hardware deployment of 1D-CNNs, without sacrificing performance. This work lays the foundation for future studies in the field of DL for real-world signal processing applications.

Keywords: Deep Learning, Adaptive Pre-processing, 1D-CNN, Radar, Spectrum, micro-Doppler, Signal Processing, Hardware Deployment, Drone, Unmanned Aerial Vehicle, FPGA

Abstract in lingua italiana

Mirando a ridurre l'attuale gap di ricerca relativo alle tecniche di pre-processing per la distribuzione hardware di reti neurali, questa tesi si concentra in particolare sul problema della classificazione di "Drone vs. Unknown" basata sugli spettri radar (frequenza-ampiezza) utilizzando una tecnica di Deep Learning (DL), in particolare le Reti Neurali Convoluzionali 1-Dimensionali (1D-CNN). La principale sfida affrontata consiste nella determinazione di una pipeline che faciliti l'implementazione industriale, mantenendo alte le metriche di classificazione. Dopo aver presentato una rassegna completa delle ricerche esistenti sulla classificazione dei segnali radar e sull'applicazione delle tecniche di DL in questo dominio, segue il contesto tecnico dell'elaborazione dei segnali (Signal Processing) per fornire uno scenario pratico in cui le soluzioni indagate potrebbero essere implementate. Una descrizione approfondita dei vincoli tecnici, come ad esempio i requisiti del tipo di dato richiesti da Field Programmable Gate Array (FPGA) segue l'intero progetto, giustificando la necessità di una tecnica di pre-processing che si adatti alle distribuzioni altamente asimmetriche. I risultati mostrano come il pre-processing adattivo faciliti l'implementazione su hardware, mantenendo elevate le metriche di classificazione, mentre altre tecniche contribuiscono alla creazione di rumore e alla perdita di informazione. In conclusione, la tesi contribuisce al campo della classificazione di spettri radar identificando metodi efficaci per supportare un'efficiente implementazione hardware delle reti neurali convoluzionali, senza sacrificare le prestazioni. Pone le basi per studi futuri in ambito DL sulla classificazione di spettri radar 1D.

Parole chiave: Deep Learning, Pre-processing adattivo, Reti Neurali Convoluzionali 1D, Radar, Spettro, micro-Doppler, Signal Processing, Implementazione Hardware, Drone, Aeromobile a pilotaggio remoto, FPGA

Acronyms

AI Artificial Intelligence

AUC Area Under the Curve

CFAR Constant False Alarm Rate

DL Deep Learning

DFT Discrete Fourier Transform

DOA Direction of Arrival

DPU Deep learning Processing Unit

DSP Digital Signal Processing

CNN Convolutional Neural Network

MD micro-Doppler

ML Machine Learning

RF Radio Frequency

SNR Signal-to-Noise Ratio

SP Signal Processing

SVM Support Vector Machines

ECG Electro Cardiogram

IF Intermediate Frequency

FC Fully Connected

A/D Analog-to-Digital

LSS Low-Small-Slow

LSTM Long-Short Term Memory

FPGA Field Programmable Gate Array

TWS Track While Scan

TPR True Positive Rate

FPR False Positive Rate

UAV Unmanned Air Vehicle

STE Straight-Through Estimator

RCS Radar Cross-Section

ROC Receiver Operating Characteristic

RhI Rheinmetall Italia

Contents

1 | Introduction

The development of radar systems has consistently followed technological advancements in pursuit of enhanced performance relative to existing state-of-the-art systems. These advancements typically focus on hardware design and signal processing techniques. The rapid proliferation of Artificial Intelligence (AI) and Deep Learning (DL) algorithms has ushered in a new age of exceptional performance in areas such as Computer Vision. Alongside these applications, the radar domain is also benefiting from the innovative DL approaches, as researchers seek to develop signal processing solutions that increase sensing accuracy by capitalizing on novel signatures present in the backscattered signals. The focus has been mainly set on unconventional features extraction, thanks to Convolutional Neural Networks (CNNs), and adequate pre-processing techniques that require signal processing expertise. Within this research field, hardware deployment is hardly considered upstream.

First, there is a lack of work focusing on radar targets' 1D spectra classification, so this project aims at creating a baseline that can be used for future comparison. Then, the problems arising from hardware deployment are listed, including the type and distribution of the data: highly skewed distributions, which are the result of high amplitude spikes, are hard to rescale without losing information if they exceed the targeted 8-bit integer dynamic. This work shows that static approaches like scaling using the maximum value or clipping to an arbitrary value fail at retaining post-quantization accuracy, if compared to methods that adapt to the input distribution during the learning phase. It is proven by setting up pipelines that follow different data pre-processing methods. They are fed with 221k frequency-amplitude spectra which values occupy a 16 bit dynamic. These are labelled in two classes, Drone and Unknown, forming a binary classification task. Results show how with the same data and similar CNN models for feature extraction, the adaptive pre-processing techniques highly contrast post-quantization accuracy drop compared to static techniques, reaching an accuracy of above 98% on the final quantized model. The final discussion interprets the results and points at how this technique could benefit problems with similar input type, datasets and requirements.

1.1. Background

The aerial landscape scanned by radars is becoming increasingly complex, dynamic, and diverse due to a range of new targets, including commercial drones and new types of Unmanned Air Vehicles (UAVs). Detection and classification need to advance in order to effectively address the newly emerging threats that are on the horizon.

Within the defence context, enhancing radar spectra classification adds useful information to the situational awareness and makes the subsequent engagement actions more effective. UAVs classification relies on the ability to extract key features from the target that uniquely distinguish it among the others. The frequency domain brings the opportunity to separate target's moving parts from its steady body, providing a great advantage in rotary-wings drone classification.

Drones represents today one of the most challenging radar targets because their low electromagnetic signature comparable, for example, to that of birds which must be treated as non-dangerous targets. There are several classification tasks, that can be divided by the different nature of the input. Traditional images are the latest trend, as several CNNs like ResNet [1], Alexnet [2] and VGG-16 [3] are ready to use with very little effort and show outstanding results as shown in [4], where all the models scored above 90% in classification accuracy, with Alexnet winning the comparison at 98.53%. Another popular approach is the classification of micro-Doppler (MD) spectrograms: it involves the frequency-amplitude spectrum of the target and its time evolution, resulting in a 2D time-frequency image where amplitude is represented by the intensity of the colour, which is used for feature extraction and classification. In [5], pre-processing and data augmentation techniques are presented for human vs. vehicles targets, as well as a comparison of CNNs architectures and feature extraction methods for micro-Doppler classification with exhaustive results. It shows the remarkable results obtained by employing neural networks in the frequency domain, hinting already to the necessity of an appropriate pre-processing technique. The last and least common approach is time-invariant frequency-amplitude spectra classification, which aligns with the scope of this project. It is equivalent as taking one column (fixed time) from a MD image, better represented with a spectrum. For this reason, similar feature extraction and pre-processing techniques made for the micro-Doppler dimension, can be applied to the 1D frequency spectra as well. There are already algorithms for automatic radar target classification that work with 1D spectra as shown in [6], which features are not extracted with artificial intelligence methods, but require a certain level of domain specific knowledge.

These different techniques presented in table 1.1, change in the input shape, which is

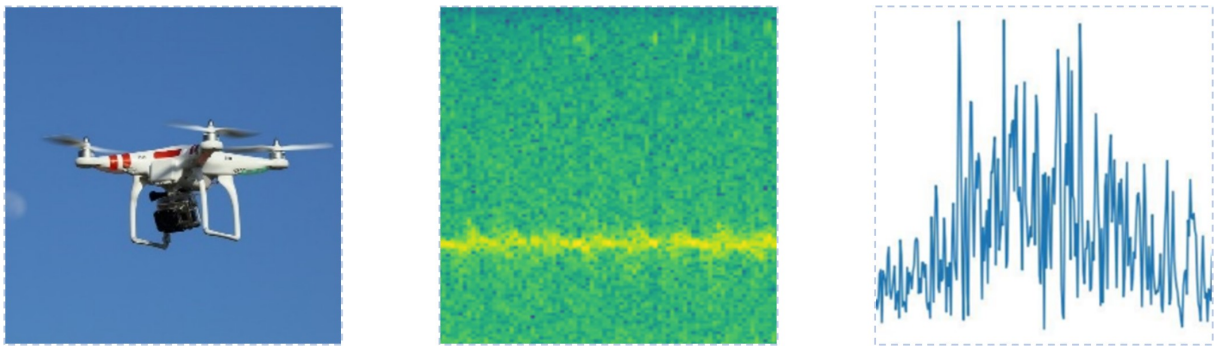

Figure 1.1: From left to right: a drone image, a general target's micro-Doppler and a general spectrum.

strictly related to the complexity of the final model and number of operations required, as well as the domains in which they exist. This project is going to focus on 1D spectrum classification, envisioning an industrial development of the final solution. This considers several constraints that are rarely encountered in pure deep learning projects.

Table 1.1: Different inputs and techniques for classification.

Input	Shape	Domain	Channels	Lightweight
Traditional Image	2D	Object shapes	3	No
Micro-Doppler	2D	Frequency, amplitude, time	1	No
Spectrum	1D	Frequency, amplitude	1	Yes

1.2. Problem

Hardware deployment and the problems arising from it do not strictly come from the research field itself, but from an industrial necessity instead. It is in fact common for a scientific work regarding classification to either focus on the pre-processing and the model or just the quantization strategy (mostly with 8bit int compliant input images), but not both. The research related to MD and spectral classification strongly focuses on feature extraction via CNNs and Long-Short Term Memory (LSTM) networks as shown in [7] with state-of-the-art results on multi-class classification, with standard precision (float32) models. In order to deploy on hardware, there are several constraints that need to be considered: these constraints define the problem that this project aims at resolving. The most significant one is related to the data types required by the Field Programmable Gate Array (FPGA), which are, at the time of writing, 8 bit signed integers for both the input data and the neural network's weights and biases. To match this requirement, the

network must be quantized [8], a procedure that adapts the parameters to a lower dynamic resolution (typically from float32 to int8). The input data suffers from this problem as well, two different scenarios can be identified:

1. Fixed range inputs like traditional images as RGB channels are already described by 8 bits, there is no loss in the dynamic as only rescaling dictated by the quantization process and shifting are required.

2. Wide range amplitude values like MD and 1D spectra, the ones treated in this work. Rescaling such inputs to fit into an 8 bits dynamic is harder, especially when the distribution is far from being normal and the effect of large values is heavier on information loss when applying max value-dependant pre-processing and rounding.

The information loss problem can be extended to any scenario where the input occupies a wide range of values, not only in the radar Signal Processing (SP) field where the high amplitude values are spikes in the radar recording. The same problems could be faced in the medical and financial fields.

1.3. Purpose

The thesis discusses a set of pre-processing techniques for spectral classification pipelines, in the context of SP and aims to demonstrate the effective application of CNNs for 1D spectra and data-adaptive techniques to mitigate the distribution's skewness. It presents the technical constraints and sets them as leading forces for the adoption of certain solutions, highlighting the strength of adaptive pre-processing to retain quantization accuracy. These research questions have been identified:

1. Can a lightweight 1D-CNN achieve good performance on a spectral classification task?

2. How can pre-processing learn from and adapt to the classification task?

3. Can data-adaptive pre-processing mitigate the post-training quantization accuracy drop?

The first question aims at creating a baseline for future work, the second and third aim at proving the effectiveness of techniques that adapt to the data distribution.

1.4. Goal

The goals have been set according to Rheinmetall Italia (RhI)'s AI roadmap. For this project, they are articulated as:

1. Identify and implement a CNN architecture able to perform binary classification with 1D drone spectra.
   (a) A classification accuracy of at least 90% has been set as threshold to consider the result satisfying.
2. Quantize the neural network and pre-process the input in order to run the inference on FPGA, according to constraints. It can be divided into:
   (a) Find a suitable pre-processing technique.
   (b) Retain accuracy after quantization. Minimize the quantization accuracy drop.

1.5. Benefits, Ethics and Sustainability

Apart from the benefit related to RhI advancing in their technology roadmap, the entire Signal Processing community will benefit from the findings of this thesis, as well as every industry that wants to employ DL to work on one dimensional signals, waves or series like the medical and financial fields, especially for hardware deployment of the final models.

When it comes to ethics, deep learning can often be a black box that doesn't necessarily show at first glance what is the algorithm learning in order to classify, in this case. A small step towards explainability is done by showing the results of one of the two proposed pre-processing layers. This helps identifying which frequencies are taken more into considerations by the network.

Sustainability issues are not directly raised from the start and are not at the centre of the discussion, but it's worth noting that compared to other techniques in the field [4] [5], 1D convolution is much lighter in terms of computational resources and model size. This project could help shifting the attention from 2D micro-Doppler images to 1D spectra if state-of-the-art accuracy are possible.

1.6. Research Methodology

The methodologies that can be employed in a research project follow a larger concept, the Research Design, as [9] describes. The design influences assumptions, data collec-

tion, analysis and how decisions are taken. It composed of three elements: Philosophical worldviews, Strategies of Inquiry and, as anticipated, Research Methods.

Framing the worldview, which is "a basic set of beliefs that guide action", the Postpositive view certainly defines the most this type of research. This is mainly because of one of the key assumptions of this position states that "Data, evidence, and rational considerations shape knowledge". It is clear that the adoption of the Postpositive worldview is adequate for Quantitative methods, which set data and results as the main source of comparison. As a Data Science project, it strongly relies on quantitative measures, so an Experimental research method is going to validate decisions.

When it comes to strategies, a broader consideration must be made on the project. Strategy provides a specific direction for procedures in a research design. Even if the final success of a solution is evaluated on quantitative and experimental methods, the necessities of the industry and the company must be taken into account to create a successful research project. This introduces some qualitative considerations, directions suggested by necessities and field expertise.

Moving downwards in the portal of research methods [10], the Deductive approach is certainly the most suitable, as the theories behind the assumptions are tested by using quantitative methods. Applied to this project, the initial pipeline is designed according to RhI's necessities, using some qualitative measures, like implementability. Then, the building blocks that compose the pipeline will be chosen and fine-tuned according to quantitative measures, comparing the effectiveness of each solution. The pipeline is presented in Chapter 3, where the focus is set also on data collection, data analysis and quality assurance (validation).

1.7. Stakeholders

The project has been issued by Rheinmetall Italia S.p.A., more specifically the company's Research, Innovation Technology (RT) department. The findings are presented in a Master's Thesis published by KTH Royal Institute of Technology, following EIT Digital's Data Science track requirements. The people directly involved in the thesis are the academic supervisor Dr. Niharika Gauraha, the examiner Professor Sarunas Girdzijauskas and the company supervisor Dr. Francesco De Palo.

1.8. Delimitations

Delimitations go on par with the problems described in 1.2. It could be argued why are some solutions adopted, and it will be hinted referencing the delimitations listed here. The objective is to overcome such delimitations with the introduction of smart solutions along the pipeline.

The first constraint is the FPGA's required data type: signed int8. It implies that both the neural network model weights and the inputs must be quantized. The complexity of the model is fixed, and so is the complexity of pre-processing. For instance, Histogram Equalization has been considered to fit the input dynamic into 8 bits, but it would require a secondary structure on unseen data, if performed globally. The ideal pre-processing is kept as simple as possible.

1.9. Outline

In Chapter 2, the technical background of Radar Theory is presented to provide an overall view of the context around the thesis. Target classification is then described, focusing on the targets considered in this work, including a review of the existing solutions for classification.

Chapter 3 focuses on the methods, describing the pipeline and the recorded data, while Chapter 4 presents the experiments conducted emphasizing some solutions. The latter are compared in Chapter 5, which elects the best one with the evaluation after the training of the model.

A discussion based on the initial research questions is held in Chapter 6, ending with the conclusion and future work.

2 | Technical Background

A detailed description about the background of the degree project is presented together with related work. First, the Radar Theory is introduced to grasp the environment around the project, with particular attention to Signal Processing as the classifier will be implemented in its domain. Then, drone targets are described from a classification point of view highlighting their characteristics, together with a horizontal view of the classification methods in 2.2. The focus is then shifted vertically on deep learning classification methods in section 2.3. Some attention will be given also to the medical field, where the use of 1D CNNs is wider [11], for early diagnosis, structural health monitoring and anomaly detection [12]. While analysing the current classification methods, reasons about the applicability of such methods is provided, together with a brief discussion about the uniqueness of this research project.

2.1. Radar Theory

Radar, which stands for "Radio Detection and Ranging," is a remote sensing device that detects and locates objects in the environment by using electromagnetic waves. In this subchapter, the high-level functionalities and design of radar systems is covered together with the Signal Processing of radar data up to the Discrete Fourier Transform (DFT) and spectrogram representation. Most of the information about this topic is extracted from Barton's "Radar System Analysis and Modeling" [13], and Skolnik's "Radar Handbook" [14], milestones and reference points in the radar world.

2.1.1. High level Architecture and Functionality

When it comes to high-level functionality, radar systems operate by sending electromagnetic waves into the atmosphere, where they reflect off objects in their path. The radar antenna then receives the reflected waves, or echoes, and the system analyses the incoming signals to calculate the range, direction, and relative velocity of the detected objects.

As anticipated, there are several components in the architecture of a radar system. The

transmitter, antenna, receiver, signal processing, and controller are the five main radar components. A high-frequency electromagnetic wave is produced by the transmitter, and the antenna is in charge of broadcasting the wave and catching any echoes that are reflected back to it. The signal processor inside the receiver retrieves the pertinent information for further analysis after the receiver amplifies and filters the incoming signals. It is worth providing an overview of what happens in the receiver, as it generates the input learned by the neural network.

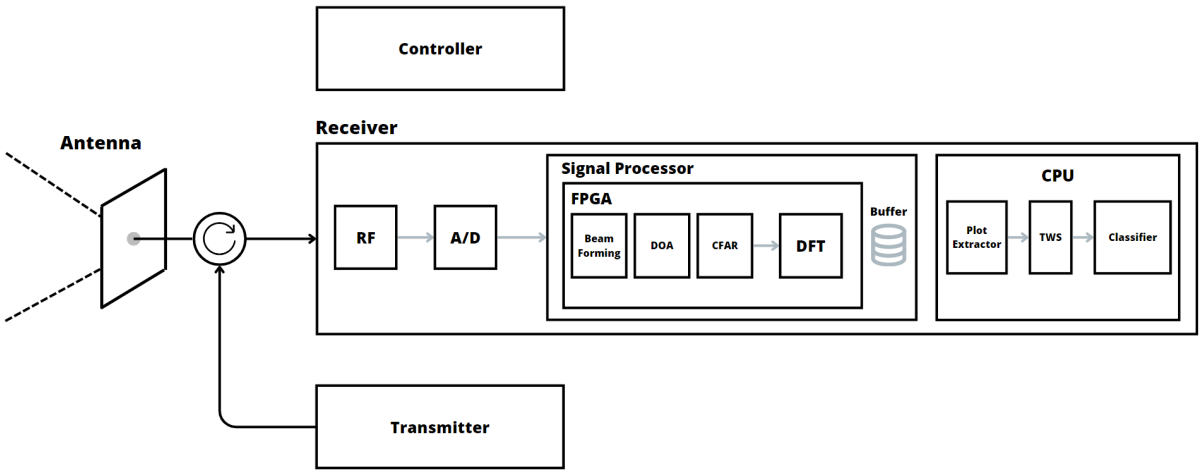

Figure 2.1: Simplified generic radar system and its internal components with focus on the Receiver.

First in a radar receiver, the Radio Frequency (RF) component is in charge of detecting reflected radar signals (echoes) and converting them into an Intermediate Frequency (IF) or baseband signal for further processing. The Analog-to-Digital (A/D) converter follows the RF and intermediate frequency signal processing stages, but before the Digital Signal Processing (DSP). Its principal role is to convert a continuous-time analog signal to a discrete-time digital signal that can be handled by DSP algorithms.

After converting the signal to a digital one it is passed through a Beamforming layer, to focus the receiver signals in a particular direction. By raising the gain, boosting the Signal-to-Noise Ratio (SNR), and enhancing target recognition capabilities, this strategy improves the performance of the radar system. The Direction of Arrival (DOA) estimation is the technique of detecting the direction of origin of a received radar signal. It is important to determine the target's angle. Constant False Alarm Rate (CFAR) detection is a technique used in radar systems to set the detection threshold adaptively in order to maintain a constant false alarm rate. False alarms arise when the radar system misiden-

tifies noise or clutter as a target. CFAR detection can lower the risk of false alarms while retaining the required probability of detection.

Finally, the Discrete Fourier Transform turns the discrete-time sequence of data points into a frequency-domain representation, which is the one used in this project. Given a sequence of complex numbers x_k of length N its DFT X_k can be computed as follows:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{2\pi i}{N} kn} \tag{2.1}$$

The result of the DFT is a frequency-domain spectrum that can be directly fed to an algorithm. Traditionally, the detection made of target information including range and velocity (Doppler frequency) are fed through a Plot Extractor. The extractor provides representation of the target data from different detections after filtering, peak detection and parameter extraction. The Track While Scan (TWS) mode is responsible for building up the track after various plots. It is commonly represented as the target with a vector that describes the direction and velocity.

Traditional classifiers consider the kinematic information. The aim of the project is to classify the target solely looking at the DFT.

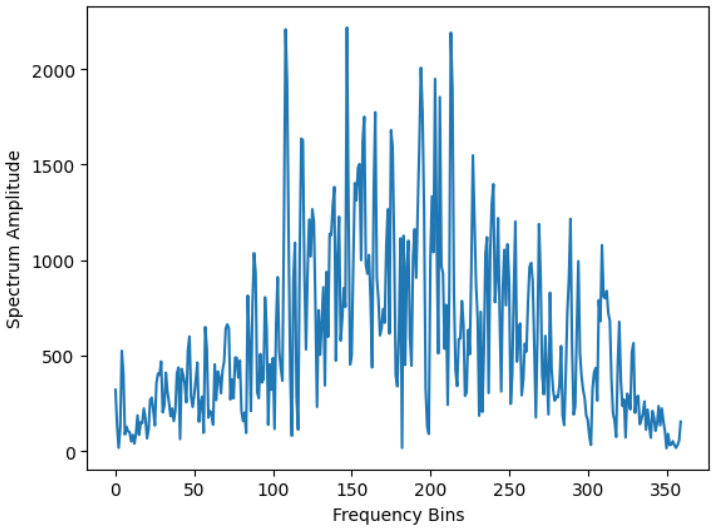

Figure 2.2: A generic 1D spectrum.

2.1.2. Micro-Doppler Image

The Doppler frequency modulation generated by the target's internal motion, such as the rotation of a drone's propellers or the flapping of a bird's wings, is referred to as the

micro-Doppler effect [15]. The leading factors for this effect are rotations, vibrations and oscillations of some of the parts of the target. A micro-Doppler image represents time on the horizontal axis, and the Doppler frequency shift on the vertical axis. The intensity of the image is higher as the power amplitude at a given frequency in the radar response increases. From a spectrum perspective, each column of the micro-Doppler image has a fixed time. It is understandable that a MD image is just the representation of a frequency spectrum evolving in time. This consideration will lead to the assumption that deep learning algorithms are suitable for the spectral classification, as explained in Chapter 3.

2.2. Target Classification

Radar classification involves categorizing detected objects according to their characteristics, such as size, shape, speed, or Radar Cross-Section (RCS). For applications like surveillance, air traffic control, and military operations, where it's critical to discriminate between different sorts of targets like airplanes, vehicles, or even tiny drones, accurate classification is vital.

UAVs, or drones, are aircraft systems piloted remotely or autonomously by onboard computers. These adaptable machines have grown in popularity in recent years due to their wide variety of applications in both the civilian and military sectors. Ranging from small consumer versions to large-scale military UAVs, they have transformed several sectors and have become indispensable instruments in modern life. Equipped with cameras, thermal imaging, LiDAR, or radar systems, they can be used for aerial photography, environmental monitoring and especially the military field.

Figure 2.3: Two UAVs with different propellers and body.

Low-Small-Slow (LSS) targets are challenging for radar systems because, as suggested by

the name, they have small radar cross-sections, slow speeds, and may travel close to the ground or in cluttered settings. Small drones, birds, and even intruders are examples of LSS targets.

Modern methods for LSS target classification in radar systems frequently include the combination of advanced signal processing techniques and machine learning algorithms. Among these approaches are:

1. Micro-Doppler Analysis: by evaluating the observed objects' micro-Doppler signatures, features indicative of their motion characteristics can be extracted and used for classification. Typically, 2D CNNs are employed for the task, in order to consider both vertical time invariant and horizontal time dependent features.

2. Kinematic Features Analysis: as anticipated, a traditional approach consists in the classification of targets based on kinematic features extracted during the radar recording. These features including RCS, Doppler frequency, range, azimuth and elevation are perfect candidates for traditional machine learning algorithms like Support Vector Machines (SVM) [16] and Random Forests [17].

3. Traditional Images: another way of extracting features from a target is by taking a picture and learning its components with a CNN. It strongly relies on the scenario and use case. Images must have a certain quality in order to be effective, and environmental conditions strongly influence the ability to detect targets. Nigh time, rain, clouds and buildings can all negatively affect the use of images for detection and consequently, classification.

The next section 2.3 expands what is discussed in Chapter 1, providing a complete view of the current target classification field, focusing on DL models and drone targets, where possible.

2.3. Literature Review

This review aims at showing the great results obtained by employing deep learning algorithms in the context of drone classification, with a focus on spectral targets, as well as the success of 1D-CNNs in the medical field because of the similar input to the spectra in this project.

Drone images are the perfect target for perfect conditions. Distinguishing a drone from a bird or generic background is an easy task for a deep (large number of convolutional layers) CNNs. The paper [4] shows it by training several CNN models on an image

dataset, and comparing the three-class accuracies, as well as the processing time and size of the networks (in terms of megabytes). Results over 97% of accuracy have been reached by three different architectures, with AlexNet [2] claiming the number one spot at 98.53%. It shows the robustness and stability of deep learning methods for this task. As highlighted by [18], the main drawbacks of visual-automated techniques are related to disguised drones or odd-looking targets, which may be the consequence of not ideal conditions like fog or night time. Limited visual range is also keeping visual approaches away from some sensible military domains.

Closing the gap towards the spectral target treated in this work, there are micro-Doppler images. From a DL point of view, the technique consists in combining image features extraction with the time-variant frequency domain. In [19] an extensive review is presented, together with a comprehensive table of micro-Doppler based target recognition and classification with both ML and DL algorithms and their accuracy. In particular, [20] achieves over 90% of accuracy using a deep CNN for human activity classification, while [21] utilise merged Doppler images to classify drones with pre-trained GoogLeNet [22], obtaining the outstanding result of perfect classification. With the same architecture, [23] reached 98.5% of accuracy on a 4-class task, including drones, birds, clutter and noise. It is important to focus on drone and clutter (alias unknown) as they mimic the same binary classification task proposed in this work. In [5] both micro-Doppler noise reducing pre-processing and data augmentations techniques are presented. Among a wide comparison of previous research, it highlights the importance of adequate pre-processing.

Because of hardware implementation, model complexity has a high level of priority. Light CNN [24] is a lightweight model for drone vs. noise multiclass classification. It achieves 97.14% of accuracy after a stationary point concentration method is applied to the input. The technique is designed to improve the signal-to-noise ratio in frequency-modulated continuous-wave radar systems, particularly for detecting small drones [25]. For the scope of this project, the relevant result is the high classification accuracy using a fairly simple architecture. Light CNN's main components are taken and reworked to fit the 1D input task, explained in Chapter 4.

This thesis could be interesting for the medical field, as one-dimensional inputs are frequent compared to the radar field (especially after the recent success of micro-Doppler based approaches) and because of the nature of most hardware-implementation aimed prototypes. It is in fact common in this field to merge Machine Learning (ML) and DL models with sensors to detect, predict and classify problems or diseases. Electro Cardiogram (ECG) classification is one of the most popular tasks, [26] shows its effectiveness, as well as proposing a low-cost hardware CNN architecture for inference on FPGA. The ac-

curacy drop in percentage is only 0.07 between the CPU and the target hardware device, achieving a final 98.84%.

As outstanding as it seems full hardware implementation requires strong domain knowledge. For this reason Xilinx's Vitis AI [27] , a comprehensive AI inference development solution is used to move from a floating point model to the final quantized model, compiled for Deep learning Processing Unit (DPU) on Xilinx's target board. The framework and methods are described in the next Chapter 3.

3 | Artificial Intelligence Methodology

This chapter presents the data used for the project starting from the collection to the statistical analysis of the dataset. Then, the model and its architecture are described justifying the choice, and the final pipeline that each data point follows from collection to model evaluation on hardware is depicted. The pipeline has been designed following company necessities and qualitative measures, like implementability, but each component of the pipeline has been then evaluated with a comparison among selected solutions with quantitative metrics.

3.1. Data Collection Scenario

The collection activity is not strictly related to the thesis work, as it has been performed by RhI in previous projects. An attempt to describe the scenario is still done, to show the high level of representativity of the dataset in real situations. An S-Band Pulsed Radar has been deployed on RhI's building, with the target identifying drones' landing point. The drone is then maneuvered along a specified path in the scope of the specific radar's scan range. The distance between the target and the radar is about 1.7 km. In order to obtain a clean result and avoid most of the clutter, the antenna has been properly tuned and positioned.

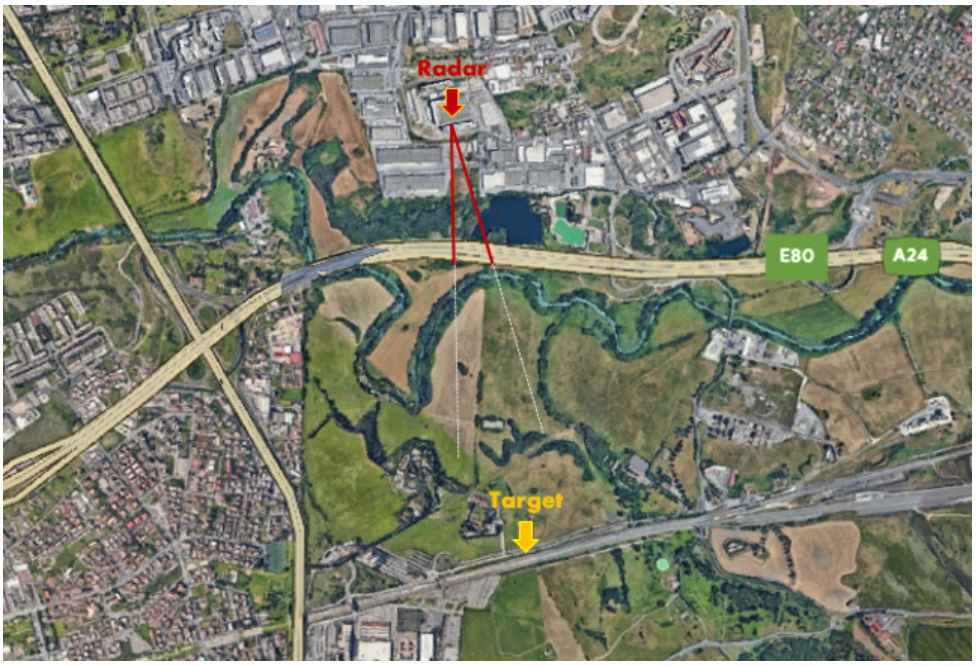

Figure 3.1: Geographical representation of the data collection scenario.

3.2. Dataset

Emerging from the measurements is a dataset composed of 221k samples belonging to two classes: Drone (class 1) and Unknown (class 0). Samples are slightly unbalanced towards drones, which are around 60% of the total dataset. Each sample is a spectrum with 360 values (representing the frequency bins), and the value of each bin is the power amplitude of such frequency bin. The spectrum is the direct result of the DFT performed by the radar, described previously in subsection 2.1.1. From a data science perspective, samples are just vectors of amplitudes, having length 360. Amplitudes are therefore the most important measure worth examining further. The values of amplitudes range in the natural numbers set $\mathbf{N}$, from zero to a maximum around 24k. The distribution is highly skewed positively, similar to a Rayleigh distribution with a small value of the scale parameter σ.

This characteristic may seem irrelevant, but it creates problems when applying normalization or scaling the values to fit inside a specified dynamic. Traditional images are already in the 8-bit integer range, to fit them in a 8 bit signed data type, they just need to be shifted by a value of -128. The same doesn't hold for data that highly surpasses the maximum 8-bit value of 255. Applying a famous and widely used normalization technique as min-max scaling would bring most of the values down to small values leaving most of the dynamic empty.

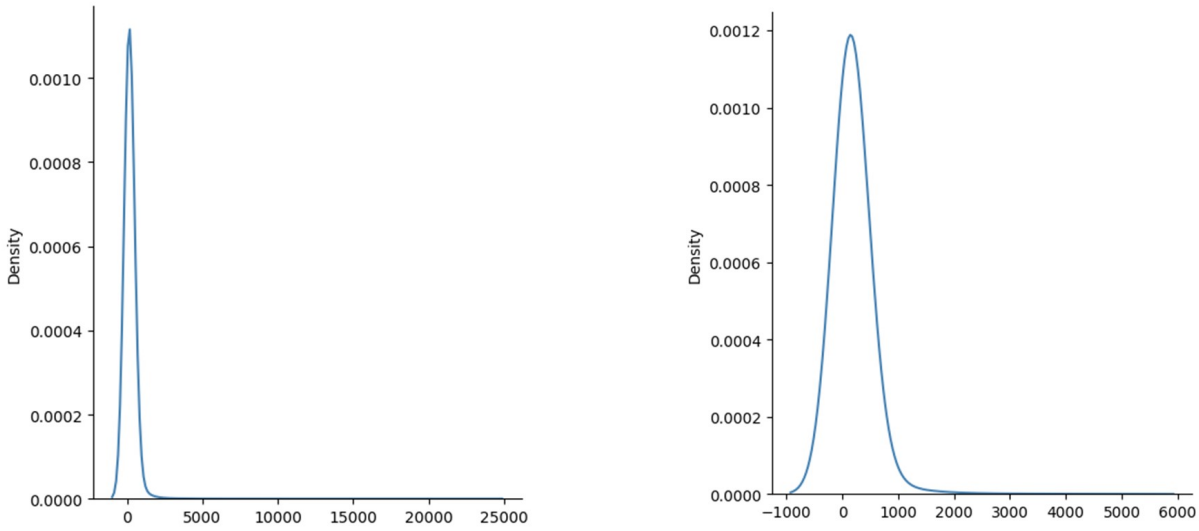

Figure 3.2: Distribution of the data on the left, distribution of clipped data on the right.

The rounding then nullifies most of the information retained by the data, being these the wanted target's response representation. The problem of outliers strongly influences the project and the use of an adaptive pre-processing solution, which solves the problem keeping all the outliers untouched.

$$x' = round\left(\frac{x - \min}{\max - \min} \cdot 255\right) - 128 \tag{3.1}$$

3.3. Convolutional Neural Networks for Signal Processing

The effectiveness of CNNs for classification tasks has already been discussed in the Introduction 1 and Literature Review 2.3, here a brief description of the convolution operation and the overall architecture aims at providing an answer to their eligibility in the signal processing scenario.

Convolution is a mathematical operation between two functions that produces a resulting function that describes how the shape of one is influenced by the other. In the discrete scenario it involves a filter, which is typically a N-dimensional matrix, that slides across the input, performing element wise multiplications and summations that result in a single value. Taking every small region once at a time, it creates a spatially aware feature map that represents the presence of a feature in the original input matrix. Convolutional Neural Networks extend this procedure. Each convolutional layer has multiple filters initialized

at random, so that different features can be learned. The result is then activated via a function to introduce a degree of non-linearity and subsampled with pooling layers for dimensionality reduction. These filters slide across the input just like convolutional filters, but apply a function aimed at obtaining a smaller output with respect to the input size, like *max* or *avg*. It doesn't only help quicker computation but also the generalization capabilities of the network, as the features appear to be more abstract.

Just like other fields, stacked convolutional filters are useful in signal processing for local feature learning, detecting recurrent and highly significant patterns from the input signal. If the filters are not manually initialized and the only supervised aspect is labelling of the target, it is the case of automatic feature extraction as it doesn't require domain expertise. Features are then used as signatures for discrimination tasks, like the binary classification that this project aims at. The input spectra are passed through many convolutional filters, which goal is to learn local features that are significant enough to understand if the target is a drone or an unknown object.

3.4. Straight-Through Estimator

The concept of Straight-Through Estimator (STE) [28]is described in this section to lay down the thoretical tools that will help explain what has been done in the Development Chapter 4. It is a trick often employed to enable the training of neural networks with non-differentiable functions, such as the rounding function which is used in this work, or the sign function.

The main constraint that STE is designed to address arises from the fact that neural networks, built on backpropagation, require differentiable activation functions. When a function is non-differentiable, its gradient is undefined, stopping backpropagation. STE acts as a workaround by providing an approximate gradient for the non-differentiable function.

The STE operates as so: during the forward pass, it applies the non-differentiable function, such as rounding in this work. During the backward pass, it approximates the gradient of the non-differentiable function instead, as a constant - usually as the identity function. Essentially, it "pretends" that the function is differentiable and that the derivative is a constant. This allows the backpropagation process to proceed as if the function was differentiable, enabling the training of the network which is the main goal.

One common application of the STE [29] [30] is in the training of networks with quantized weights or activations, such as binary or ternary networks, where the activation values are

constrained to a set of discrete values. The STE allows these networks to be trained effectively using gradient-based optimization methods by providing a means to backpropagate through the quantization function.

It is important to note that while STE allows us to train networks with non-differentiable functions, it does not provide a perfect solution. The use of STE introduces a discrepancy between the forward pass and the backward pass, which could potentially lead to suboptimal training results. Moreover, the theoretical underpinnings of STE are not yet fully understood, and it remains an active area of research.

3.5. Pipeline

This subsection presents the pipeline that each sample follows after the collection described in 3.1. First, an overall view is provided, then it is articulated in two configurations that change in the way pre-processing is performed on data.

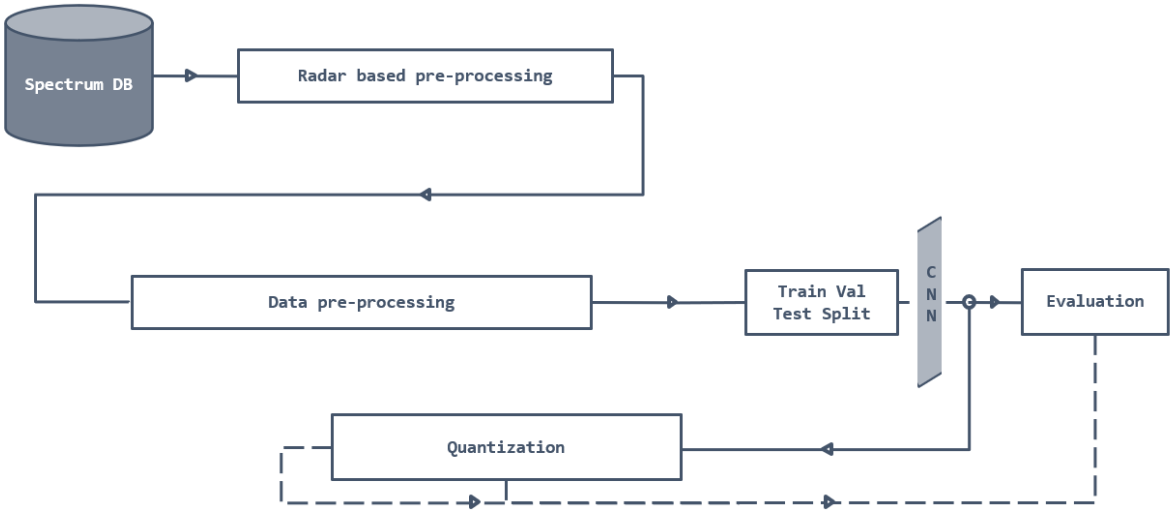

Figure 3.3: The overall proposed pipeline from the database to the quantization and final evaluation.

The pipeline 3.3 starts from the database where spectra are saved. After being analysed, they pass through a layer of radar-based pre-processing where the focus is set on a certain subrange of frequencies from the original spectrum. Then according to the nature of the pre-processing technique, either three different paths are followed for traditional methods or two paths for learning-based methods. The split into training, validation and test set is performed with percentages of respectively 64%, 16% and 20%. The test set is saved

until the very end of the model selection procedure while the validation set is used to tune hyperparameters. This phase is anticipated when adaptive pre-processing is applied, as it will automatically learn the data scaling parameters and apply the transformation to the test set before evaluation.

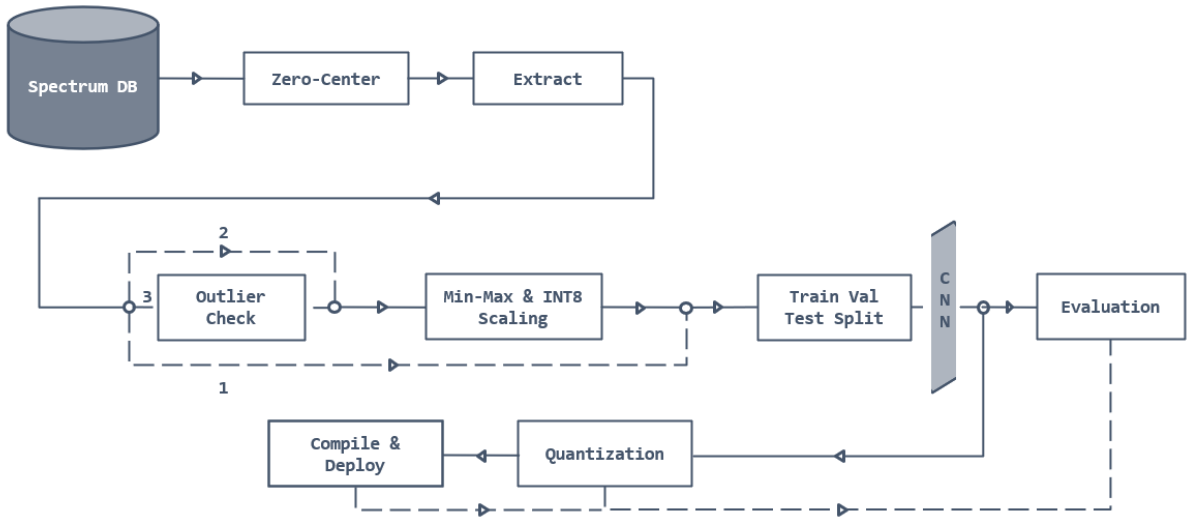

Figure 3.4: The pipeline for starting pre-processing methods.

After the training phase, the model performance is evaluated on the unseen test set, both before the quantization with different metrics and after quantization with the classification accuracy. The last step includes both turning the CNN into its 8-bit counterpart and compiling the model to enable the inference on hardware, thanks to Xilinx-provided Vitis AI. Evaluating the quantized model, the accuracy drop is calculated to measure the effectiveness of each solution. More on the evaluation metrics in the next section 3.6.

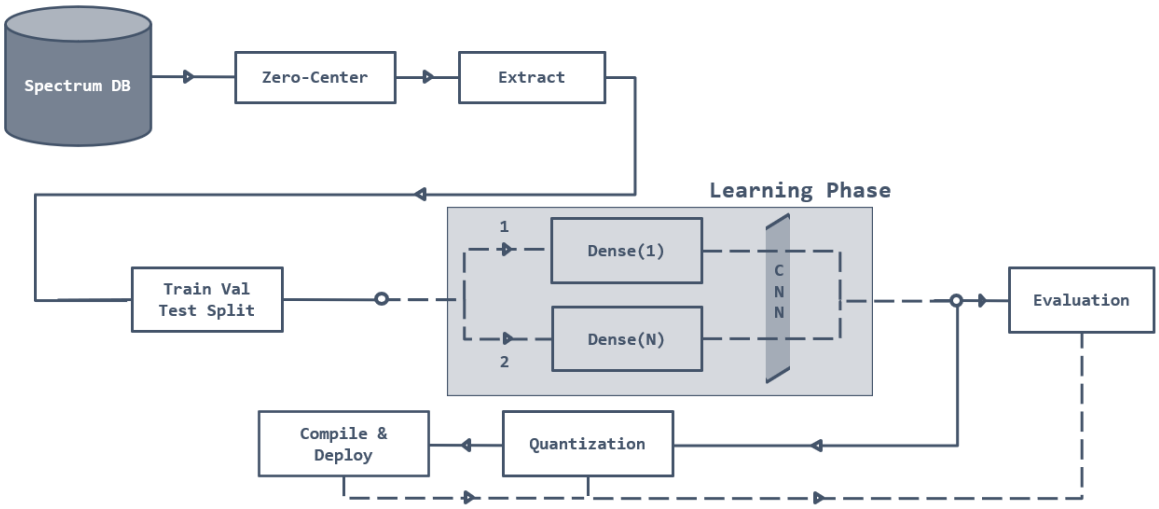

Figure 3.5: The pipeline for adaptive pre-processing methods.

3.6. Evaluation

The evaluation methodology is purely quantitative, as the only qualitative choice has been limited to the definition of the pipeline and to the size of the initial model, which wants to be "relatively small". Here, the objective is to understand which pipeline and which technique is the best one based on evaluation metrics that are presented in order of importance. First, it is worth mentioning that drones are going to be considered as the positive class and unknowns as the negative class. With this decision the Confusion Matrix labels are defined as follows:

Table 3.1: Prediction Results

Label	Prediction	Result
Drone (1)	1	True Positive
Drone (1)	0	False Negative
Unknown (0)	0	True Negative
Unknown (0)	1	False Positive

Accuracy: in the context of a classification task, the first metric that has been considered essential is the accuracy with which the model is able to classify spectra. This metric is

defined as the number of samples correctly classified (sum of true positives and true negatives) over the total number of samples.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.2}$$

Recall: choosing to assign the drones as the positive class, one of the most significant metrics is recall. It is defined as the number of elements of the positive class correctly classified over the total positive class. In the defence scenario, sending an alert for a target that might be dangerous is key. It has been preferred to keep track of the times that a real drone target is correctly classified over the number of drones, in order to understand (and minimize) the times that a dangerous target bypasses the classifier. It is clear from the formula below that the maximization of recall relies on the minimization of false negatives.

$$Rec = \frac{TP}{TP + FN} \tag{3.3}$$

Post-Quantization Accuracy Drop: this metric is calculated as the difference between post-quantization accuracy and the original accuracy. It is important to define which pre-processing method, under the same quantization settings, results in the lowest accuracy drop.

$$Drop \;=\; P.Q.Acc \breve{}\, Acc \tag{3.4}$$

Thresholding: the defence scenario requires additional certainty, especially when automations are embedded in systems that act on the external environment. Presenting the radar pipeline in Chapter 2, it was mentioned that a track on the operator's screen is made of several detections. For safety reasons it has always been discussed to mitigate the uncertainty of the network. One way to do it is to higher the probability threshold above 0.5 and create an uncertainty range in which samples are not notified as drones or unknown but as "probable targets", to be further investigated in a subsequent classification stage. For this work the threshold has been set at 0.75, an analysis of the accuracy removing the samples falling in this region is performed, counting the percentage of the test set that is "probably a drone" to provide a measure of model uncertainty.

AUC-ROC: the Receiver Operating Characteristic (ROC) curve is a plot that shows the binary classification ability of a model as the threshold used for discrimination is changed. In the ROC space, the y-axis describes the True Positive Rate (TPR), that above is defined as recall. On the x-axis there is the False Positive Rate (FPR). Each

point of the curve expresses how, given a certain threshold for binary classification, the model is able to correctly classify the positive class with the y-value and how poorly it is able to classify the negative class with the x-value. Given this definition, a perfect classifier has a ROC point at $(0, 1)$ with an outstanding ability to classify the positive class and the worst possible result in badly classifying the negative class.

Decreasing the threshold from left to right, the ROC curve summarizes all the confusion matrices (hence TPR and FPR) that each threshold produces. The performance of a random classifier lies on the quadrant bisector, while good classifiers tend to have a curve that moves towards the perfect classification point.

The Area Under the Curve (AUC) of the ROC curve, is a great measure of a model threshold-independent performance, as it sums up to 1 for perfect classifiers.

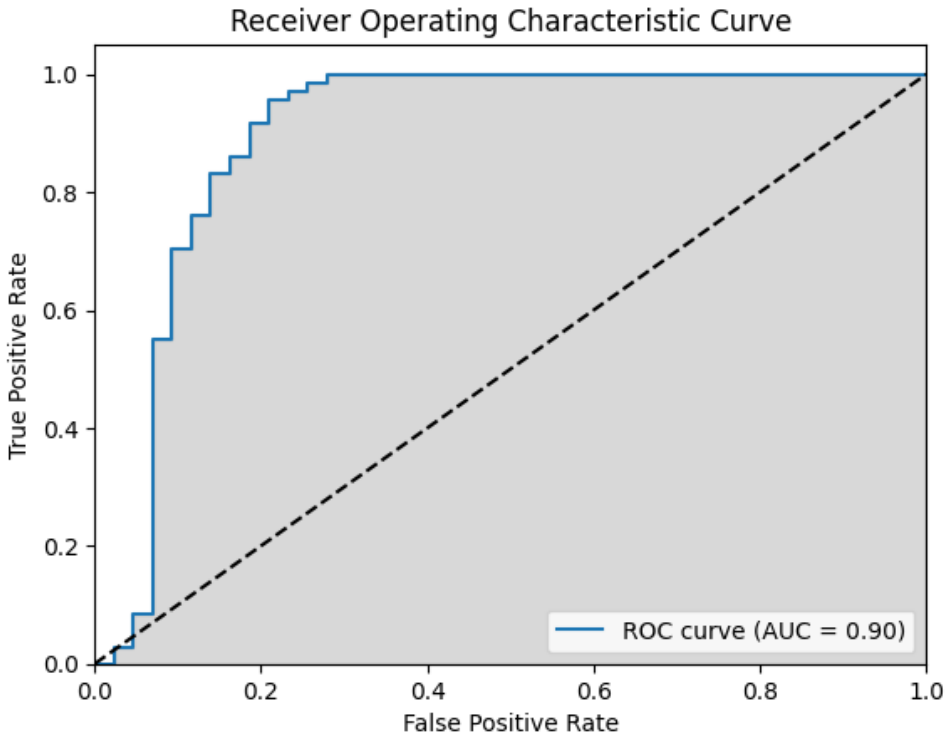

Figure 3.6: Example of a ROC curve and its AUC score.

Other useful quantitative metrics are used in Chapter 5, even if they are not at the centre of discussion. They help to provide additional context to what could be an industrial product based on the classifier.

3.7. Development Framework

The project has been developed on two Docker containers, one dedicated to the design, training and evaluation of the models and one for model inspection, quantization and additional evaluation.

The first one is Tensorflow's official Docker image [31], with Tensorflow 2.10 and all the necessary libraries for data loading, multi-dimensional array and plot creation like *pandas*, *numpy* and *matplotlib*. It eased the development incredibly by providing an all inclusive solution that would avoid conflicts with the versions of the libraries in the quantization Docker.

Xilinx Vitis AI's [27] Docker image is an excellent way to have a complete toolbox for model inspection, quantization and evaluation. The first step is crucial to understand which layers are accepted for quantization, e.g. 1D convolutional layers are not, forcing the use of 2D layers. Quantization is eased incredibly with the dedicated functions, which are highly customizable.

The hardware board for deployment is the Xilinx evaluation board, Zynq UltraScale ZCU102. The board gives the possibility to deploy a DPU exclusively designed to accelerate Deep Learning operations.

To train the neural network, the workstation has been equipped with a Ryzen 9 3.4 GHz 16 Core processor, 64GB RAM and a RTX3080Ti Gaming 8GB GPU. It enables a much quicker training phase, resulting in a better fine tuning capability.

Figure 3.7: The Zynq UltraScale ZCU102 board.

4 | Development

This chapter is dedicated to the work itself. The pipeline presented in Chapter 3 is expanded approaching different solutions, which are described now and then evaluated in Chapter 5. After choosing the blocks in a semi-qualitative way according to RhI's necessities, quantitative measures elect the best solution that contributes to the highest accuracy and recall and lowest drop in these metrics from the original model to the quantized correspondent.

4.1. Model Choice and Resizing

The first step is the model selection, which has been narrowed down to a convolutional neural network, because of the great results obtained by 2D CNNs for the micro-Doppler classification task, as depicted in the Literature Review 2.3.

It may be argued why is a CNN adequate for the task. Being the convolutional layers' task to automatically extract spatial features from the input as described in section 3.3, 2D convolutional layers perform such feature extraction both time wise (horizontally) and frequency wise (vertically) in a micro-Doppler image. It is not wrong to assume that a 1D convolutional layer that acts only on frequencies could extract spatial features, as it is a subspace of a MD spectrogram.

For this reason, the previously mentioned (Chapter 2) Light CNN has been taken and reworked to be adapted to the input. The input layer has been rescaled calculating the ratio r between the original network's input (128) and this work's input size (N). The ratio has then been used to rescale the convolutional kernels, rounding them to either an even or odd value, respecting the order of the original network.

Then, the Fully Connected (FC) layer's size is set to 2, for binary categorical classification, and a Batch Normalization layer has been added before every activation following [32]. Finally, Dropout is implemented before the FC layer to introduce some regularization and mitigate overfitting [33].

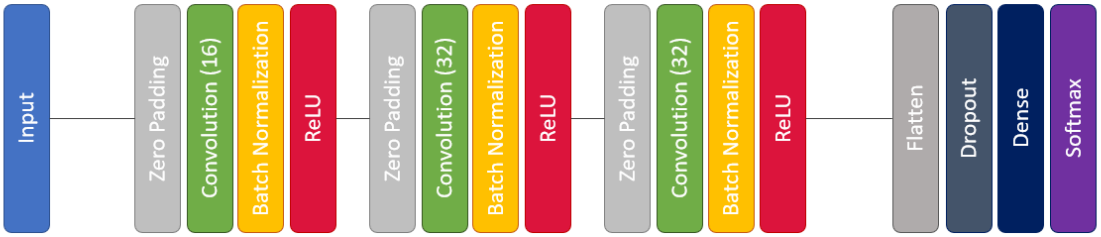

Figure 4.1: High level architecture of the employed CNN.

4.2. Radar-specific Pre-processing

The first pre-processing applied to the spectra comes from a radar-specific assumption. Knowing that targets velocities fall in a certain range, and frequencies are synonym of velocity via the wavelength λ. First, the input is FFT-shifted of half the range, then a specific subrange of the shifted input is extracted, reducing dimensionality and noise. This subrange is where the targets main frequencies are expected to be found.

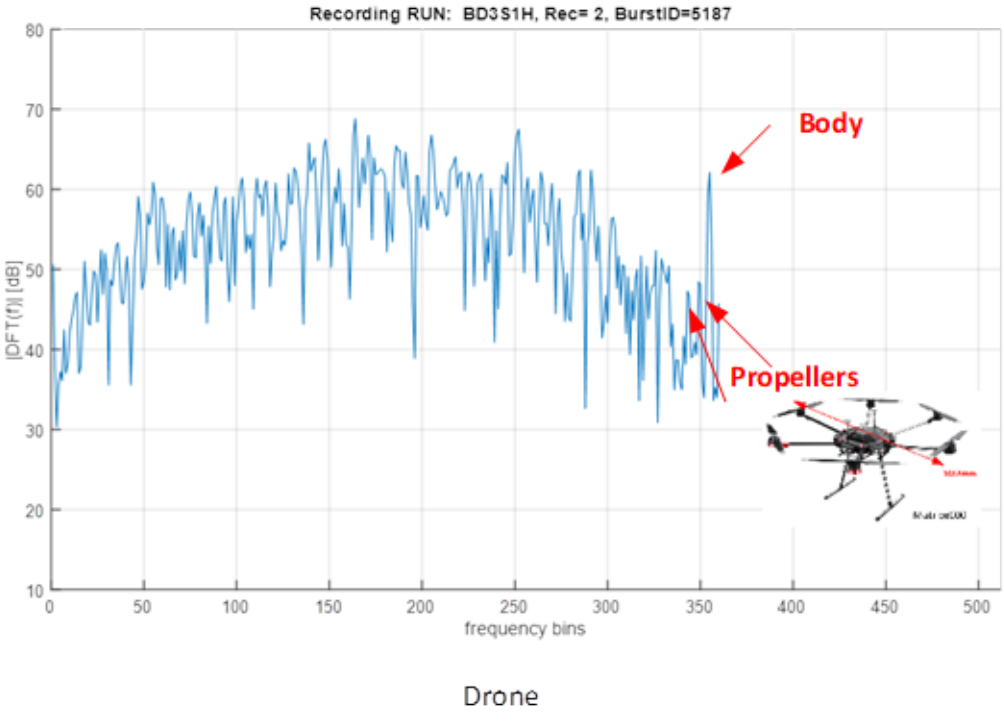

Figure 4.2: Position of drone's body and propellers.

4.3. Data Pre-processing

The last block to be described before training, that holds also the highest level of importance, is the pre-processing block. Now, action is taken directly on the samples rather than using radar-related assumptions. Every consideration made from now on is solely related to the data science part of the project.

4.3.1. No outliers, min-max, int8 scaling

The first, and most intuitive approach addresses the problem of high amplitudes. As earlier described in the Dataset section 3.2, the amplitude values fall in a 16-bit dynamic ($log_2(24K) \sim 14.55$). Being skewly distributed, applying min-max scaling $x' = \frac{x-min}{max-min}$ to such a high value of maximum would shrink all the significant values to low values, which must be rounded to be accepted by the hardware, operation that will destroy almost the entire information held by the input.

For this reason, a threshold for maximum value has been set according to statistical considerations. The objective was set to retain 99% of the original amplitude values and clip all the values above the threshold. Clipping is a numerical operation directly provided by NumPy; given an upper bound, it sets all the values above it to the bound value. The same holds for the lower bound which has not been set in this case (same as setting it to 0). The output value is then min-max scaled with the new N_{max} value from the clipping step, and rescaled in the int8 dynamic. The complete transformation is defined in the following equation 4.1.

$$x' = round\left(\frac{x - min}{N_{max} - min} \cdot 255\right) - 128 \tag{4.1}$$

4.3.2. Bit Split

This naive bit split approach is presented, even if not described in the pipeline. The results are presented in Chapter 5 even if massively underperforming compared to the others, in order to avoid replicating the attempt in other scenarios. It comes from the idea that each amplitude could be divided into its most significant bits and least significant bits. After the transformation and a -128 shift, the CNN input layer would just require an additional channel, as a traditional image requires 3 for the red, green and blue representations. Doing so, the convolutional layers are looking for features with a divide-et-impera approach, discriminating between the high values and low values (< 256).

A discussion about why this method shall be avoided is carried out in Chapter 6.

4.3.3. Adaptive Pre-processing

While developing the pipeline, a key concept started getting relevance: if the hardware device requires a signed 8-bit input and signed 8-bit quantized model (which is not a problem given quantization), where does the model struggle to retain the key metrics used for evaluation?

If trained on data that has been statically pre-processed, the loss due to quantization has no ability to decrease under a certain value, related to the optimal clipping parameter used in the "No outliers" approach. To overcome this problem, an attempt to shift the pre-processing inside the learning phase has been considered. Can the Convolutional Neural Network's loss positively influence the optimal action performed on data, effectively "learning" how to scale and shift the input?

Test-bed

Before explaining the solution, here is a small description of the initial idea of transforming radar spectra into their optimal 8-bit representation.

The very first brute force approach was to place a $N \times 8$ matrix in front of the neural network, as a result of the product between the input vector $N \times 1$ and a 1×8 layer with eight units. It would then be activated with a *sigmoid* function $\sigma(x) = \frac{1}{1+e^{-x}}$ and rounded to the nearest integer value. This way, each row of the matrix would be cast to an 8-bit integer and learn an alternative representation of the input. The idea was discarded because it would require an explicit cast of the data and it wouldn't have great library support. The objective is to use Tensorflow's layers, or fairly simple custom layers given the constraint of Vitis AI supported operations. This approach has been implemented with a *Dense(N)* layer, exlplained later.

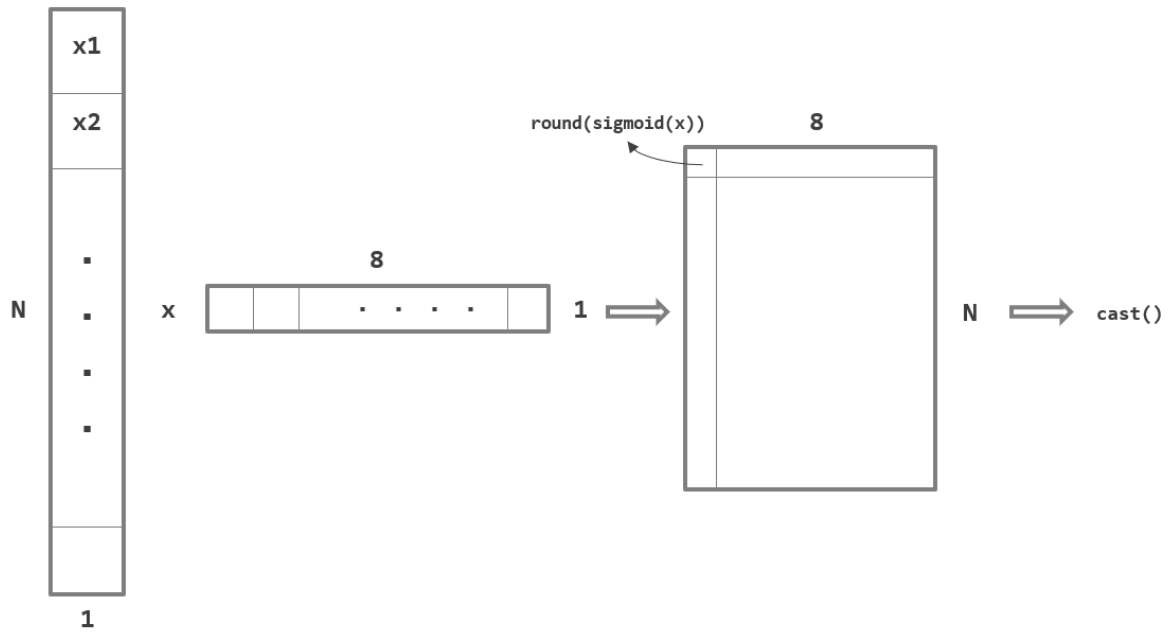

Figure 4.3: Naive ideated matrix.

Another idea was to tune the clipping parameter N_{max} given the network loss. If done in a naive way, it requires processing the data at each step, that is why a single unit FC layer has been employed instead. It doesn't explicitly tune the clipping parameter, but it sill performs the scaling required to fit the data to the network in the best possible way.

Dense(1)

The first approach uses a single unit (neuron) to scale the input, inserting a *Dense(1)* layer (FC) right after the input layer. It holds only two parameters: a weight w and a bias b. These parameters create a response of the type $X' = Xw + b$. Where X is a $N \times 1$ vector and w, b are 1×1 vectors. It generates a $N \times 1$ output, the same size as the original vector.

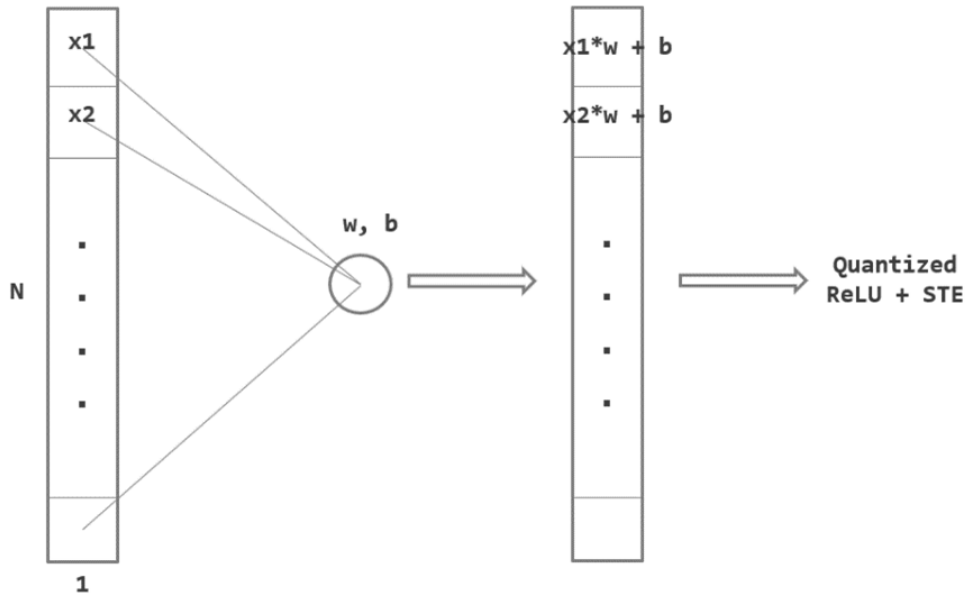

Figure 4.4: From left to right: the input vector, the *Dense(1)* layer and the results that is then activated.

The output of this layer is then activated with a custom version of a Quantized ReLU $q(x)$ (the generic function is described in the appendix of [29]), which in this case is not clipped at 0, but operates between the values -128 and 127 instead, the ones required by the signed int8 DPU.

As anticipated in section 3.4, the problem arises because of the non-differentiable nature of the activation function (because rounded). While training the model, the backpropagated gradient of the layer would be null. To overcome this, the STE is employed. A work on the application of STE to train quantized neural networks [29] poses the question "How to estimate the gradient of a loss function with respect to the input of such stochastic or non-smooth neurons? i.e., can we "back-propagate" through these stochastic neurons?".

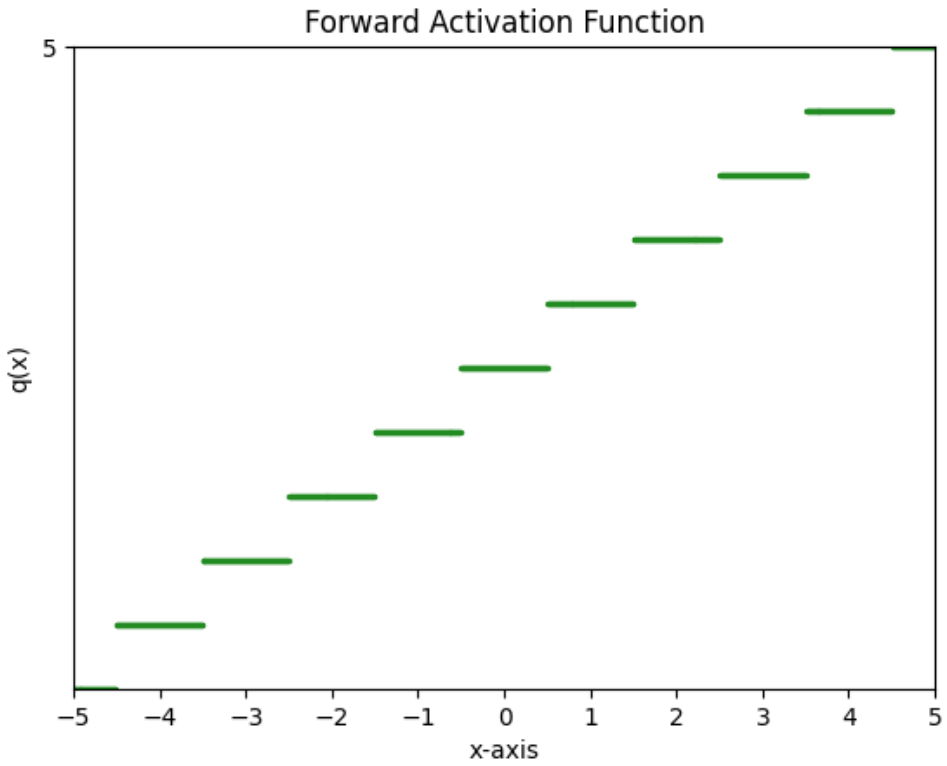

Figure 4.5: The activation function is a rounded ReLU, clipped at -128 and 127.

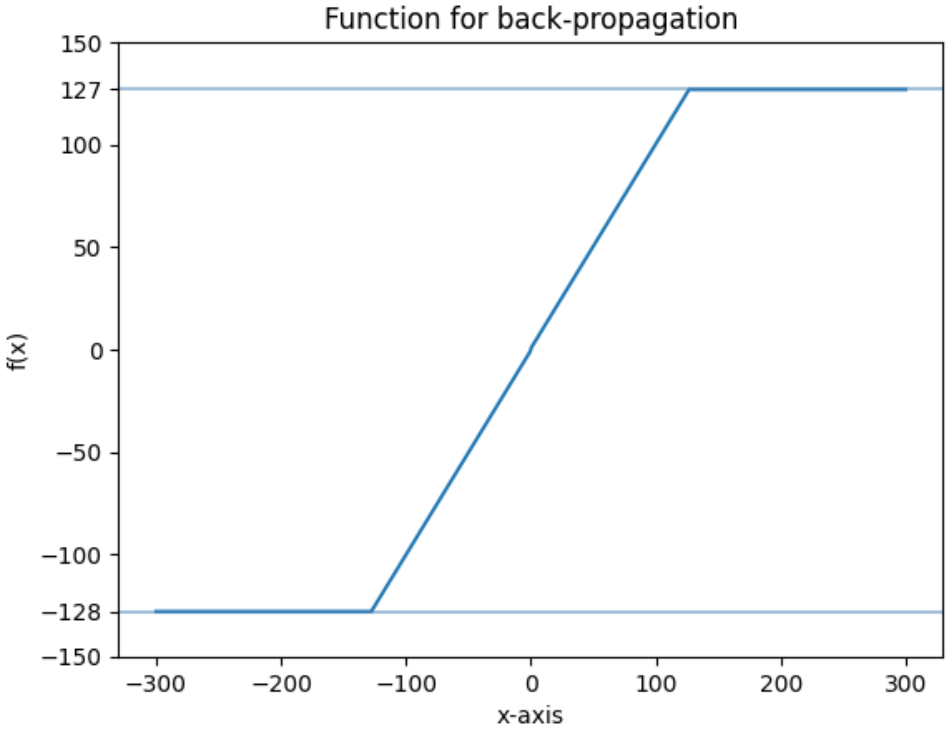

Figure 4.6: Backpropagation function.

Here there are no stochastic neurons, but the problem still holds for non-differentiable functions, like the Quantized ReLU. Backpropagating, the original ReLU operating between -128 and 127 is used instead as a replacement. Following this approach, the CNN is fed with signed 8-bit activated inputs and the gradient is not null to update the weight and bias of the *Dense(1)* layer.

Dense(N)

What happens if instead of one single unit the FC layer has the same size as the input, *Dense(N)*?

The parameters create a response of the type $X' = Xw + b$. Where X is a $N \times 1$ vector and w, b are $1 \times N$ vectors. It generates a $N \times N$ matrix output. To mimic a dot product between the two vectors, a custom layer takes the diagonal, resulting in a $N \times 1$ output as the original amplitudes vector. It is worth mentioning that a custom layer that performs the dot product (and bias sum) between N weights and the input vector would have the same result.

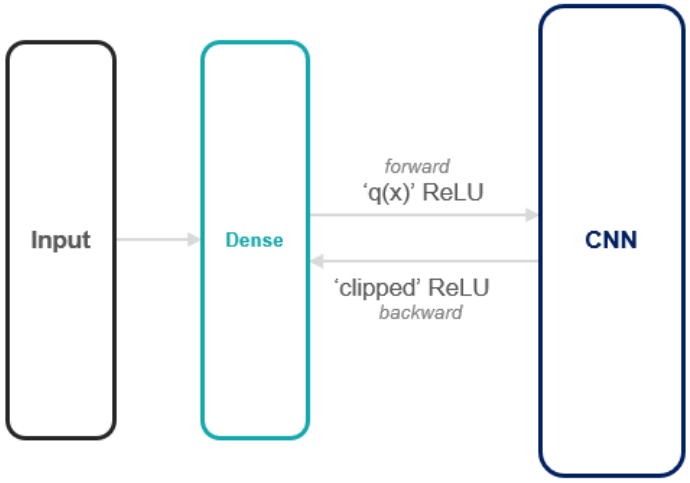

Figure 4.7: High level design of the adaptive pre-processing.

4.4. Training Phase

The training (learning) phase is one of the most sensible components of the pipeline, as it influences the stability of the model especially if quantization is required after the

training. It has been empirically noticed in this project that unstable trainings result in higher post-quantization accuracy drops, leading to the choice to present the settings used for this phase.

4.4.1. Exponential Decay

Before introducing the decay, it is worth mentioning that learning rate α is a hyperparameter that regulates the strength of the weight update at each iteration. The weights (and biases) of each layer are updated as follows (L is the loss function):

$$W' = W - \alpha \cdot \frac{\partial L}{\partial W} \tag{4.2}$$

Exponential Decay is a learning rate scheduling method, where the α hyperparameter is reduced over time following an exponential curve. It helps the starting phase by setting a higher value of the learning rate, leading to quick initial convergence and then the parameter is gradually decreased over time to perform smaller steps and finer adjustments. It is not decreased every epoch, but only after a certain number of epochs as it increases stability in almost all the configurations that have been tried in the degree project. This is obtained by setting *'staircase=True'* in Keras' ExponentialDecay class. The obtained schedule follows the graph below:

In this case, the decay has been set every 50 epochs, with an upper bound of 500 epochs. It provides a quick learning in the first epochs and a later smaller tuning when approaching the end. In the next tables are condensed the training and pre-processing techniques tested in the project.

Train ID	Initial LR	Final LR	Early Stopping	LR Decay	Epochs (N)	Early Stopping Patience
1	10e-4	10e-5	Yes	Exponential every N/10	500	N/10
2	10e-5	10e-5	Yes	No	500	N/10
3	10e-3	10e-4	Yes	Exponential every N/10	500	N/10

Figure 4.8: Training strategies.

Pre-processing ID	Description
1	None
2	Apply round(n/max * 255) - 128
3	Apply clip(N_max) and round(x / N_max* 255) - 128
4	Split MSB and LSB on two channels and - 128
5	Dense(1)
6	Dense(N)

Figure 4.9: Pre-processing strategies.

5 | Results

This chapter is purely dedicated to the presentation of the results obtained by training the CNN model described in Chapter 4, following the different settings of the pipelines presented in the AI Methodology Chapter 3. The results want to prepare a response to whether the research question of adaptive pre-processing's success turns out to be positive or the technique is not able to achieve better performance, compared to traditional methods.

For the first model tested no action has been performed on data to set a baseline for other tries. Of course, this is the only model that cannot be deployed as the CNN is not trained on data of a fixed 8-bit dynamic. This problem is going to be stated again to ease understanding.

Although Xilinx's Vitis AI quantization procedure extracts a scaling factor that can be applied to data before hardware inference, the same data has to be rounded to the nearest integer anyway after the scaling, losing information that translates in an uncontrolled accuracy loss. To control accuracy and limit the accuracy drop upfront several methods have been tested.

Last, at this point in the project the validation set has been merged with the training set, as the adequate hyperparameters have been found in the development stage, so the models are trained with more data to hold as much knowledge as possible. The test set evaluation is the real method to present the unbiased performances. It must be noted that model size is increased only at the end, when the best pipeline path is assessed.

ID	Name	Model	Train - Test Accuracy	Recall	PT-Quantized Test Accuracy	Accuracy > 75%	% removed test set > 75%	Parameter Number	Deployable	Train ID	Pre-processing ID
1	Base	50x1x1 + no pre-processing	94.4 - 94.5	95	92.4	97	8.33%	6498	No	1	1
2	Pre_process	50x1x1 + pre-processing	90 - 89	88.9	85.1	94.7	18.8%	6498	Yes	2	2
3	Pre_process_clip	50x1x1 + pre-processing + no outliers	94.5 - 94.5	95	91.1	97.1	8.5%	6498	Yes	1	3
4	Bit_split	50x1x2 + bit split	93.3 - 92	94.6	77.8	96	12%	6642	Yes	1	4
5	Adaptive_1	50x1x1 + Dense(1)	96.2 - 95.7	96.3	95.4	97.7	5.8%	6500	Yes	1	5
6	Adaptive_N	50x1x1 + Dense(N)	96.2 - 95.9	96.6	95.7	97.8	5.5%	6598	Yes	3	6
7	Final	50x1x1 + Dense(N)	99.5 - 98.8	99	98.6	99.1	0.8%	24358	Yes	3	6

Figure 5.1: Results.

5.1. Model comparison: accuracy, recall and % drop

The baseline model (ID=1) achieves a good 94.4% on the training set and a 94.5% on the test set. Recall stands at 95%. It can be seen from the result table 5.1 that increasing the discrimination threshold to 0.75, the accuracy stands at 97% with 8.33% of test samples in the uncertainty region of [0.5, 0.75]. The Post Training (PT) quantization accuracy on the test set is 92.4%, even if this result is not informative as this network is not deployable, the baseline drop in accuracy is -2.1%.

When introducing unclipped pre-processing (ID=2) the train and test accuracies fall to 90% and 89% respectively, with similar recall, a higher level of uncertain samples that fall in the [0.5, 0.75] range and an accuracy drop of -3.9%. The results were expected to improve with the introduction of clipping (ID=3), that again, was set to retain 99% of the original values. The table clearly shows the competitive results of the third strategy with 94.5% of accuracy and 95% of recall, reaching the performance of the baseline model. This approach doesn't only saturate the input using a method that is between quantitative and qualitative, but also fails to match the quantized accuracy of the baseline model, as it stops at 91.1%. The number of samples in the uncertain region remains similar, but the accuracy drop stands at -3.4%, quite worse than the baseline. The bit-split model (ID=4) underperforms after quantization, as it only reaches 77.8% of quantized accuracy, despite showing results that are in line with other methods at the training and test phases, 93.3% and 92% respectively. This drop shows that the method causes some noise in the data, discussed in 6.

The adaptive pre-processing layers step up both in terms of overall performance, achieving 96.2% and 95.7% on training and test accuracies and 96.3% recall for the *Dense(1)* layer (ID=5), 96.2% and 95.9% accuracies and 96.6% recall for the *Dense(N)* layer (ID=6). The number of samples in the uncertain region is decreased to 5.8% and 5.5%. It doesn't only

increase general accuracy but after dividing the layer from the network and quantizing the CNN, both methods hold the original accuracy with drops of only -0.3% and -0.2%. These methods do not significantly differ between them when analysing results but outperform all the previous methods keeping the dataset intact with no semi-supervised clipping, but with a learned rescaling.

As the difference between training and test accuracies is small for most models, it is possible that the complexity of tested models is not enough to separate the dataset. This is why after obtaining the results, the filters number of each convolutional layer of the CNN has been doubled, creating a model (ID=7) with 24358 parameters, to which the *Dense(N)* pre-processing is applied as it has shown the best result. The final model scores 99.5% and 98.8% of accuracy on the training and test sets and 99% recall, with the same quantization accuracy drop of -0.2% and less than 1% of samples in the uncertain region below the 0.75 threshold. The procedure was tested to show additional adequacy of the model for this task.

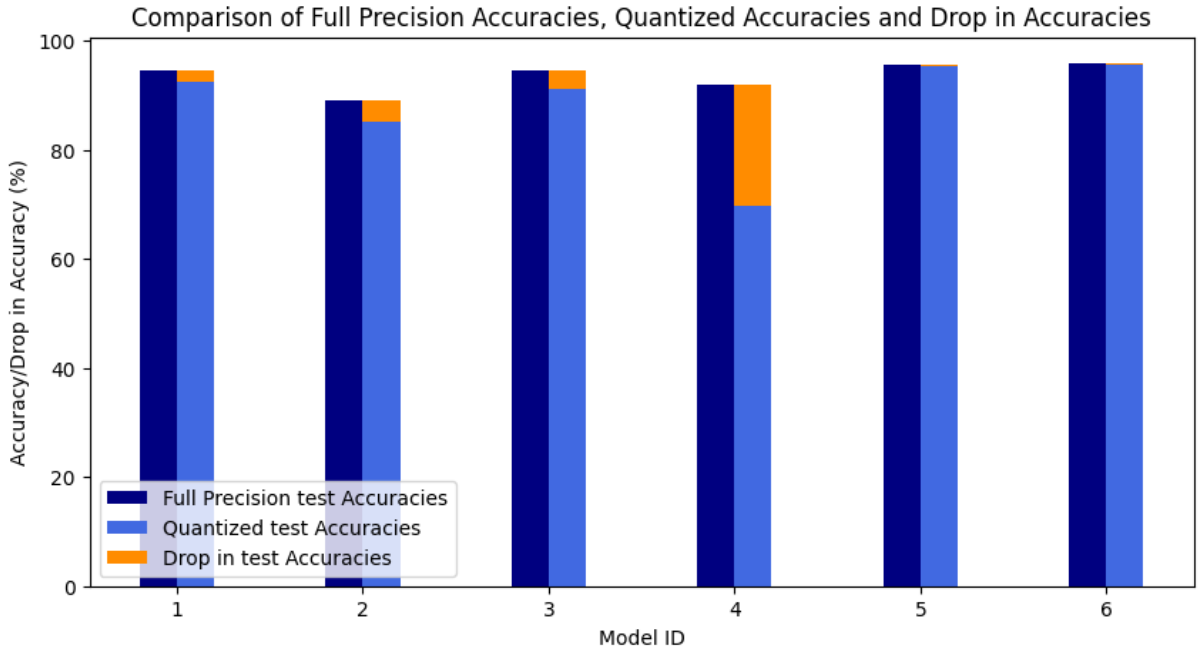

Figure 5.2: This bar plot shows how models fed with different pre-processed data perform in different way in retaining accuracy after quantization.

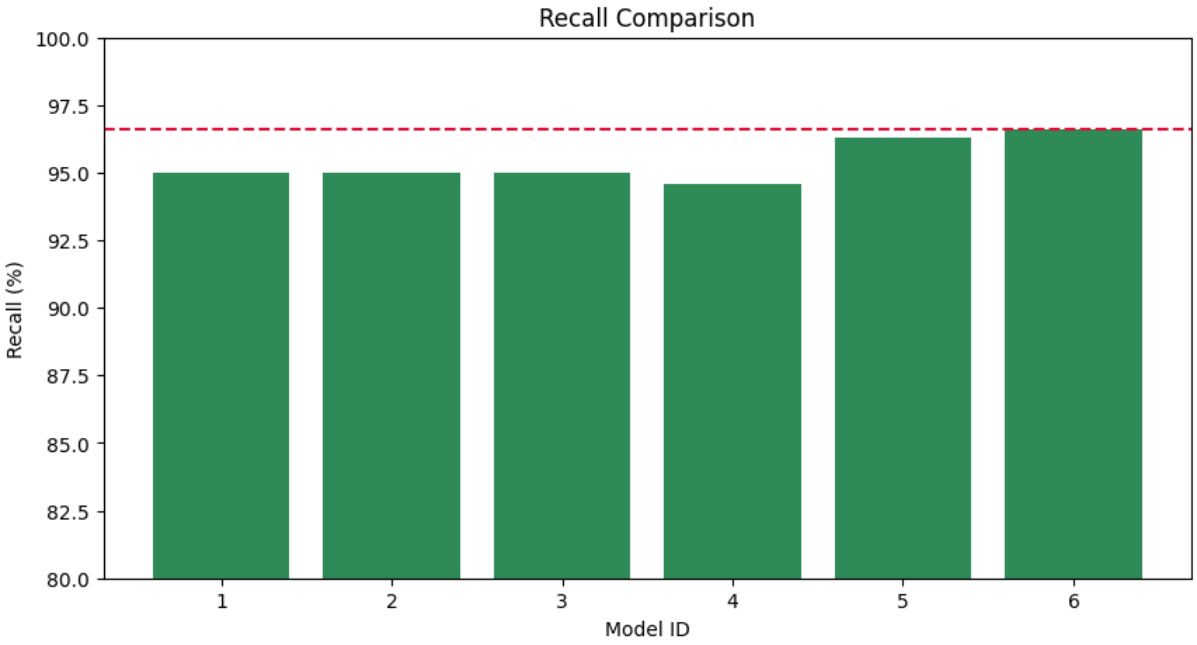

Figure 5.3: This bar plot shows different Full Precision test Recall values for the applied pre-processing methods.

5.2. ROC-AUC score

The area under the ROC curve importance was stated in the Evaluation section 3.6. Only the most significant models have been tested to provide a fair comparison, leaving out the baseline as it lacks deployment capabilities, the bit-split method as it strongly underperforms and the final enlarged model because its results are clearly biased because of the higher complexity (it is presented on its own).

The rescaled input pre-processing obtains an AUC score of 0.9551, which is immediately surpassed by introducing the input clipping with 0.9839. The two adaptive pre-processing layers improve again the model capabilities calculated at various thresholds, reaching 0.9904 for the *Dense(1)* layer that surprisingly outperforms (slightly) the *Dense(N)* layer at 0.9901 for this metric. Although the two methods are similar, the discussion section is going to highlight major conceptual differences between them.

Below, the comparison of the ROC curves and, on its own, the ROC curve of the expanded model.

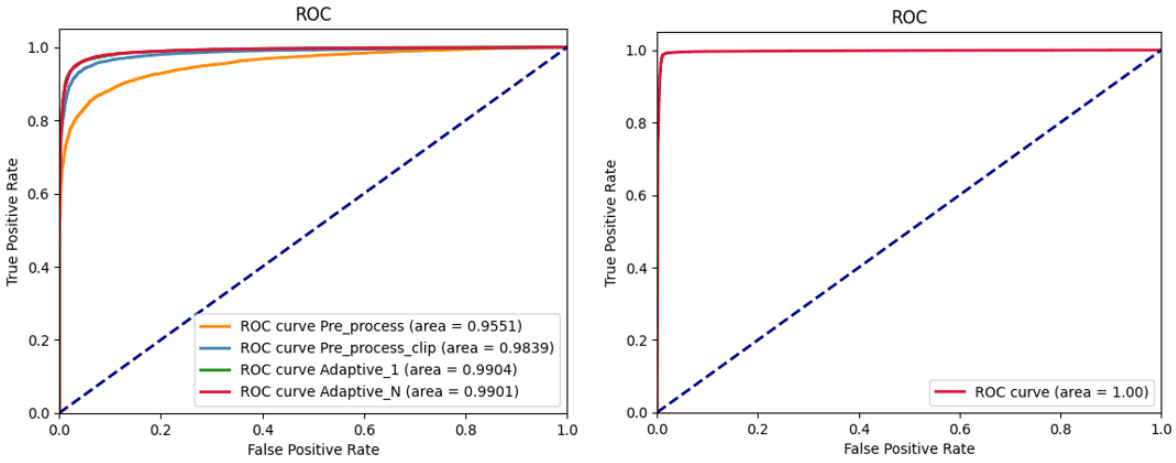

Figure 5.4: On the left, the ROC-AUC scores of the four compared pre-processing methods. On the right, the score of the final enlarged model.

5.3. Dense(N) Weights

This section provides a visual representation of the difference between the *Dense(1)* layer and the *Dense(N)* layer with one weight and one bias dedicated to each frequency bin, printing the weights with a colormap. Considerations about the effectiveness of one solution compared to the other are carried out in the Discussion section 6.1.

For this result, the Viridis colormap embedded in matplotlib is used to show the weights with higher value.

Figure 5.5: Weights of the *Dense(N)* layer plotted with a Viridis colormap. The descending order of value is: yellow, green, blue, purple.

It can be clearly noticed by Figure 5.5 that some frequencies (in yellow) have a higher weight, while other parts seem to not respond to the activation of the layer (the purple areas). Especially, the central part is highlighted: the meaning could hide behind the fact that in the second step of radar-based pre-processing, explained in section 4.2, the input is FFT-shifted of half the range. This way, the edge frequencies of the spectrum, which have a higher chance of containing the drone body and propellers, are in the central part. They of course would have a higher weight compared to other frequencies. This moves a step towards explainability as radar experts would expect such behaviour from the first

layer, that again, has the goal of scaling the input and activating it in the defined (8-bit) dynamic.

6 | Conclusions

This section presents once again the research questions formulated in Chapter 1 and aims at answering with a discussion of the results presented in Chapter 5. Then, some insights on future work are provided, in order to highlight the open questions that can be answered in projects that start based off this research.

The first research questions asked whether a lightweight Convolutional Neural Network could be used to discriminate between drone and unknown spectra, reducing dimensionally what traditional techniques excellently do with micro-Doppler signatures [19] and RGB images [4]. Results affirmatively answer the question obtaining well over 90% of accuracy with very a lightweight design in almost all the settings. It must be said that binary classification is not the best scenario to demonstrate the assumption, as less classes mean less required complexity to divide the input space, but it is still an excellent starting point to move towards compact and computationally efficient directions.

To assess the effectiveness of adaptive pre-processing compared to other traditional techniques, the discussion section will focus on explaining why some methods work better than others and what are the findings of the project, in general.

6.1. Discussion

The first, simple pre-processing technique, strongly underperforms when compared to other methods. The intention was to fit the data into an 8-bit dynamic to present the same data to the quantized network and enable DPU inference, controlling the accuracy drop upfront. It is clear that, being min-max scaling too sensible to the maximum value and the distribution of the dataset, when rounding the numbers too much information is lost, and the majority of values are shrunk towards zero making feature extraction very difficult.

Introducing a clipping parameter is beneficial, but not optimal as it has been chosen statistically, with no real fit on the data. Another discussed solution was to tune the clipping parameter with the CNN loss, detaching it after the training. It still doesn't

justify the higher drop in quantization accuracy compared to the baseline model, which may derive from the fact that most high peaks are related to just one class of the two, them being drones (they will have high amplitude peaks for certain frequencies). Clipping only one of the two classes could create this type of instability.

The bit-split solution clearly doesn't work as one of the two channels, more specifically the least significant bits channel, creates cycles of values in the rage [0, 255] which confuse the learning phase. To provide an example, the amplitudes 511 and 515 are represented respectively as (1, 255) and (2, 3). While the most significant bit representation remains ordered ($1 < 2$ as $511 < 515$), it is trivial that the same doesn't hold for the least significant bits representation.

Adaptive pre-processing not only increases accuracy and recall compared to the baseline model, but it does so:

- Lowering the post training quantization drop by a factor of 10 (-0,2% and -0,3% compared to $\sim$ -3.5%), which was the main goal of the project.
- Introducing only 2 parameters for the *Dense(1)* layer and $2N$ parameters for the *Dense(N)* layer, very few compared to the size of the network (for the second method it strongly depends on the input size, in this work it was short).
- The *Dense(1)* layer performs the same number of operations that traditional min-max scaling does. A product, a sum and a clipped rounding operation. The number increases by a factor of N for the *Dense(N)* solution, but it guarantees slightly higher accuracy.
- It doesn't require any explicit pre-processing: no outlier is removed from the data.
- As the drop is quite low, it is a way to gain upfront control of the final accuracy, without quantizing models that lead to poor results saving time and computational resources.
- With generalization: the technique is not field-related, and it could be applied to any scenario.

The difference between the two approaches is small especially in terms of results, but an attempt to understand and interpret it must follow. The *Dense(1)* layer's two parameters see pass through the entire input data. Doing so, it rescales and shifts the spectra at each iteration with the same exact values. What it learns is the optimal representation of the input in another domain imposed by the activation function, in this case the natural numbers in [-128, 127], without making any distinction between frequencies. The

Dense(N) layer on the other hand is design in such a manner that each element of an input sample has a weight and a bias dedicated to itself. It results in a distinction between frequencies, being the composing elements of each sample. Having two parameters for each frequency translates in a different rescaling and shifting for each bin. It performs slightly better in term of accuracy as it may be seen as an a-priori feature selection, because a bigger weight in such an early stage could be symptom of more importance for that specific frequency in the context of distinction between the two classes. Initially, it was expected to have an higher impact, but the *Dense(1)* layer performs just fine. The answer could be linked to the feature extraction nature of the CNN: being itself dedicated to the task, a layer that performs an early and premature selection of the most important frequencies could collide with the network learning capabilities, without getting an outstanding increase of performance when compared to a simple transformation learned and applied with a single pair of parameters.

Nonetheless, it is clear that the adaptive method outperforms static pre-processing in this context.

6.1.1. Future Work

When talking about future work, the first natural evolution of this project consists in multi-label classification, expanding the number of targets to different drone types, birds, and other relevant objects that could be found in such a complex defence scenario. Its purpose is to also validate the use of such a simple model, working on the accuracy-complexity trade off.

It would be beneficial to test the adaptive pre-processing with a micro-Doppler input, as well as other research fields where one-dimensional inputs combined with hardware deployment are predominant, such as the medical and the financial field. Certainly, a wider validation would come from an extensive comparison of different traditional pre-processing techniques with the proposed one. In this work, only few worth mentioning methods were compared to the final solution, but quite a few were tried throughout the project. It is no doubt that domain expert could have better solutions to test against this one.

Finally, a computational cost based comparison could focus on electing the best technique in terms of both accuracy/quantization drop and computational/spatial resources required, mainly because in a complex machine as a radar is, every single computation is relevant and lightweight solutions are strongly needed by design.

6.1.2. Final Words

Adaptive pre-processing, a method aimed at inserting data transformations inside the learning phase to extract a more adequate representation, has been proposed to ease hardware deployment, given the constraints imposed by the DPU accepted data types.

It has shown great results in the optimization of the quantization step, as it "listens" to the CNN by updating the scaling factors with the training loss. This way, both the input and quantized weights fall in the lower dynamic and the network doesn't suffer a high loss because it has been trained on 8-bit data. It has enabled hardware inference with high accuracy on 8-bit accepting hardware.

Future work must focus on cross-field validation of this method, as well as a theoretical validation of both the idea and its capabilities.

Bibliography

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Dec. 2015. arXiv:1512.03385 [cs].

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84–90, May 2017.

[3] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Apr. 2015. arXiv:1409.1556 [cs].

[4] H. M. Oh, H. Lee, and M. Y. Kim, "Comparing Convolutional Neural Network(CNN) models for machine learning-based drone and bird classification of anti-drone system," in *2019 19th International Conference on Control, Automation and Systems (ICCAS)*, pp. 87–90, Oct. 2019. ISSN: 2642-3901.

[5] A. Erdoğan and S. Güney, "Object classification on noise-reduced and augmented micro-doppler radar spectrograms," *Neural Computing and Applications*, vol. 35, pp. 429–447, Jan. 2023.

[6] F.-X. Hofele, "A New Algorithm for Automatic Radar Target Classification Using Feature Extraction Having Special Regard to Drones," in *2019 20th International Radar Symposium (IRS)*, pp. 1–10, June 2019. ISSN: 2155-5753.

[7] J. Zhu, H. Chen, and W. Ye, "Classification of Human Activities Based on Radar Signals using 1D-CNN and LSTM," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, Oct. 2020. ISSN: 2158-1525.

[8] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, and T. Blankevoort, "A White Paper on Neural Network Quantization," June 2021. arXiv:2106.08295 [cs].

[9] J. W. Creswell and J. D. Creswell, "Research Design," Apr. 2023.

[10] A. Håkansson, "Portal of Research Methods and Methodologies for Research Projects and Degree Projects," pp. 67–73, CSREA Press U.S.A, 2013.

[11] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1D convolutional neural networks and applications: A survey," *Mechanical Systems and Signal Processing*, vol. 151, p. 107398, Apr. 2021.

[12] G. C. Jana, R. Sharma, and A. Agrawal, "A 1D-CNN-Spectrogram Based Approach for Seizure Detection from EEG Signal," *Procedia Computer Science*, vol. 167, pp. 403–412, Jan. 2020.

[13] D. K. Barton, *Radar System Analysis and Modeling*. Artech House, Oct. 2004. Google-Books-ID: dvlPAwAAQBAJ.

[14] M. I. Skolnik, *Radar Handbook, Third Edition*. McGraw-Hill Education, 2008.

[15] V. Chen, F. Li, S.-S. Ho, and H. Wechsler, "Micro-Doppler effect in radar: phenomenon, model, and simulation study," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, pp. 2–21, Jan. 2006. Conference Name: IEEE Transactions on Aerospace and Electronic Systems.

[16] M. Hearst, S. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications*, vol. 13, pp. 18–28, July 1998. Conference Name: IEEE Intelligent Systems and their Applications.

[17] A. Cutler, D. Cutler, and J. Stevens, "Random Forests," in *Machine Learning - ML*, vol. 45, pp. 157–176, Jan. 2011. Journal Abbreviation: Machine Learning - ML.

[18] "Droning on about RF? - CRFS - Spectrum Monitoring and Geolocation."

[19] A. Hanif, M. Muaz, A. Hasan, and M. Adeel, "Micro-Doppler Based Target Recognition With Radars: A Review," *IEEE Sensors Journal*, vol. 22, pp. 2948–2961, Feb. 2022. Conference Name: IEEE Sensors Journal.

[20] Y. Kim and T. Moon, "Human Detection and Activity Classification Based on Micro-Doppler Signatures Using Deep Convolutional Neural Networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, pp. 8–12, Jan. 2016. Conference Name: IEEE Geoscience and Remote Sensing Letters.

[21] B. K. Kim, H.-S. Kang, and S.-O. Park, "Drone Classification Using Convolutional Neural Networks With Merged Doppler Images," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, pp. 38–42, Jan. 2017. Conference Name: IEEE Geoscience and Remote Sensing Letters.

[22] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan,

V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," Sept. 2014. arXiv:1409.4842 [cs].

[23] S. Rahman and D. Robertson, "Classification of drones and birds using convolutional neural networks applied to radar micro-Doppler spectrogram images," *IET Radar, Sonar & Navigation*, May 2020.

[24] J. Park, J.-S. Park, and S.-O. Park, "Small drone classification with light CNN and new micro-Doppler signature extraction method based on A-SPC technique," *arXiv preprint arXiv:2009.14422*, 2020.

[25] J. Park, J.-S. Park, K.-B. Bae, and S.-O. Park, "Advanced stationary point concentration technique for leakage mitigation and small drone detection with FMCW radar," *IEEE Transactions on Microwave Theory and Techniques*, vol. 69, pp. 1791–1804, mar 2021.

[26] L. Wei, D. Liu, J. Lu, L. Zhu, and X. Cheng, "A low-cost Hardware Architecture of Convolutional Neural Network for ECG Classification," in *2021 9th International Symposium on Next Generation Electronics (ISNE)*, pp. 1–4, July 2021. ISSN: 2378-8607.

[27] "Vitis AI."

[28] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," 2013.

[29] P. Yin, J. Lyu, S. Zhang, S. Osher, Y. Qi, and J. Xin, "Understanding Straight-Through Estimator in Training Activation Quantized Neural Nets," Sept. 2019. arXiv:1903.05662 [cs, math, stat].

[30] L. Kummer, K. Sidak, T. Reichmann, and W. Gansterer, "Adaptive precision training (AdaPT): A dynamic quantized training approach for DNNs," in *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pp. 559–567, Society for Industrial and Applied Mathematics, jan 2023.

[31] "Tensorflow Docker."

[32] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015.

[33] Z. Liu, Z. Xu, J. Jin, Z. Shen, and T. Darrell, "Dropout reduces underfitting," 2023.

List of Figures

List of Tables

Acknowledgements

First, I must thank Professor Sarunas Girdzijauskas and Dr. Niharika Gauraha, who offered to be the Academic Examiner and Supervisor, as they guided me in the writing of the thesis and for the thoughtful insights offered each time that I needed some help. A special thank to Dr. Federico Schiepatti, who has been both a mentor and a friend in the EIT Digital world. It is mainly thank to you that I was able to follow this journey in the first place.

Then, I must massively thank my Company Supervisor and mentor, Dr. Francesco De Palo, for ideating the project and showing me how to manage it in the best possible way, as well as providing an endless source of knowledge that he transferred to me during these months. A big thank to Eng. Fabio Cipriani for leading me to the right path in the hardware world and to Dept. Director Eng. Massimo Sergi for coordinating and believing that I could take on the project. I must thank Dr. Luca Giancane for the research supervision and Dr. Elisa Ricci for helping me every day at the company. I salute all the colleagues of the Research Innovation Technology (RT) Department for the great environment that they created around this project.

I would like to thank my family from the bottom of my hearth, especially my parents and my brother for the endless support shown throughout these years, you made sure that this journey was possible for me. Words are not enough to give a proper weight to the role of my extended family in Rome, Milan and Stockholm. If you're reading this and you feel like we shared amazing memories along the way, know that there will always be a special place for you in my heart.

Thank you,

Dario