Automatic Classification of Unmanned Aerial Vehicles with Radars On-The-Move

Hani Haifawi

Delft University of Technology

TU Delft

robin

Automatic Classification of Unmanned Aerial Vehicles with Radars On-The-Move

by

Hani Haifawi

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday August 22nd, 2022 at 02.00 P.M.

Student number:	5271924	
Project duration:	February, 2022 - August, 2022	
Thesis committee:	Prof. DSc. Alexander Yarovoy,	TU Delft, advisor
	Dr. Francesco Fioranelli,	TU Delft, supervisor
	Dr. Julian F. P. Kooij	TU Delft
	MSc. Rob van der Meer,	Robin Radar Systems, supervisor

This thesis is confidential and cannot be made public until August 22nd, 2024.

TU Delft robin

Preface

Success consists of going from one failure
to another without loss of enthusiasm.

Winston Churchill

In my very first semester of higher education, the lecturer told us: ”*Being a professor is not a job. It is a vocation.*”.

Reaching the ending of this journey, I wholeheartedly began to understand those words. After having met my supervisors and having followed their courses, I also found my vocation.

I shall forever be grateful to the people I have had the opportunity to follow, and who took us on their shoulders, and helped us broaden our view.

The world needs such people, to ensure the future becomes a better place.

Therefore, I would like to express, from the bottom of my heart, my most sincere gratitude and appreciation to Dr. Francesco Fioranelli, Assistant Professor at the Microwave Sensing, Signals and Systems research group within TU Delft. His highly valued help, advice, and directions had a significant impact to achieving the proposed milestones, and motivated and supported me to go above and beyond during my project. Moreover, he considerably helped me develop my confidence and excitement in conducting my research and showing the achieved results. And perhaps amongst the most important, I am happy to have gained from his dedication to science, coupled with an extraordinary empathy and spirit of understanding, which greatly contributed to my personal development. I have learned so much from him, which will help both in professional and personal lives. I am really thankful for having had such a role-model at this point in life.

I would also like to express, in all sincerity and without any equivocation, my most profound admiration to Prof. DSc. Alexander Yarovoy, the chairman of the Microwave Sensing, Signals and Systems group within TU Delft. His leadership is inspiring, alongside his passion for science and the drive to achieve the epitome of veracity. Seeing a person with such a track-record that every young and aspiring student dreams of, significantly contributed and nourished my ambition. The way a person with so many incredible achievements can connect with his team and share his incredible knowledge to young and naive students, not only catalyzed my professional development and broadened my scientific knowledge, but also taught valuable lessons about leaders, openness, and modesty. And in the even greater scheme of life, I learned about the very nature of helping each other, to grow and to become better versions of ourselves.

Furthermore, I want to show my uttermost sincere gratitude and recognition to Ir. Rob van der Meer, the System Architect at Robin Radar Systems. To me, he was my very first supervisor, but more importantly, he was the first person in the industry to give me a chance. More specifically, the chance I have been preparing for, and the chance I have been doing my very best to try to obtain. I was from the very beginning impressed by his formidable knowledge of the entire systems, alongside his openness, patience and trust shown towards the most inexperienced person in the building, but also his incredible professionalism towards the most experienced ones. I always felt I can ask him the most simple and intuitive questions without feeling lost nor unaware, but also the most challenging and difficult questions that do not have a clear answer. And moreover, I am grateful for all the guidance and opportunities extended, that significantly contributed to achieving many professional milestones, catalyze results, and the chance to reach even further thanks to his support and confidence.

As a young engineer, having the space and assurance that I can count upon such persons greatly facilitated my development journey, and I am today able to write these words thanks to their expertise and trust confined in me, as well as offering me the tools required to succeed.

One could not ask for better people to look up to, nor better instruments to succeed. It is privilege that I truly value. In my turn, the very best I can do to put in practice the above words is to continue to propagate these distinctive features, and do my very best to raise to the heights of my mentors, and further build upon these values.

Additionally, I find it highly important to offer the well deserved credits and acknowledgments to the team of professors and engineers at TU Delft as well as Robin Radar Systems. None of this would be possible without their great influence in this journey.

I want to extend the most sincere appreciation to Gerben Pakkert, the Director of R&D at Robin Radar Systems. The way he connects to his team and understands every person's perspective is inspiring. Moreover, his business knowledge integrated with technology vision and team leadership, all while showing true empathy and synergy with his team showed me what a true business leader is like, and set a high standard.

Moreover, I want to thank Bart Portegijs, the Data Scientist at Robin Radar for all his help and advice offered through my very first real deep learning project. He helped me achieve one of my highest career ambitions of building and understanding data-driven models, and shared his incredible expertise with me which enabled me to conduct my research.

I would also like to thank Wouter Keijer, the System Specialist at Robin Radar for taking the time to explain software engineering concepts and phenomena encountered with radars in real environments. Many times I found myself not understanding the data or the system, and having Wouter's ear was an enormous help and time saver.

I want to thank Vitor Pinheiro, the Software Specialist from Robin Radar, for being the person I always counted on to ask questions and bounced ideas with since the very first days, who helped me in building my reasoning and comprehension of the software development and implementation process.

I would also like to thank Jurjen Westra, the Software Architect of Robin Radar, for his empathy and always sharing his experience with me. As a young adult with many doubts and questions about the future as well as software engineering, hearing his advices shared from his journey greatly helped me find mine.

I wish to thank Dennis Hoveling for always helping me obtain the very essence of my project, for going to the field together to gather data and help me see radars in action, for his patience and team-spirit, as well as being an amazing person and colleague.

I also want to express my recognition to Michiel de Leeuw, the Product Manager at Robin Radar Systems, for his remarkable openness. Having been here in the very first days when I began my initial internship, his transparency and understanding helped me become more at ease in the new, professional environment.

I want to thank Jelle Bout, the System Developer at Robin Radar, for having helped and guided me when I felt stuck in different phases of my project, for having shared his experience and information with me, and being a reliable colleague that we can count upon.

In the end, I want to thank Marta Buenaventura Camps, the System Developer at Robin Radar, for the very nice discussions and pool plays we had partook in, for being a friend at the office, and sharing the passion for Mediterranean and French cuisines.

Hani Haifawi
Den Haag, August 2022

Abstract

Drone detection and tracking systems are nowadays a requirement in most public, private and political events, because of the increasing risk of unintentional or malicious misuse of these platforms. Moreover, in order to ensure adequate protection, full spatial coverage is a must for every such system. However, the research literature focuses on staring radars that have a limited field of view, but which yield rich target information via time-frequency distributions that facilitate the target recognition task. In this thesis, surveillance radars that offer full spatial coverage are presented, albeit their usage for classification is made more complex because of the rotating nature of their antennas which limits the dwell time on targets.

Additionally, due to the incredible fast growth of the drone market, novel counter-drone radars that are able to jointly localize and classify small targets while on-the-move now represent a highly in-demand remote sensing system. Nonetheless, surveillance sensors anchored on moving vehicles are a brand-new technology that is currently being developed. This work therefore investigates surveillance systems in a novel scenario, and presents the technological challenges alongside the proposed solutions to achieve reliable object detection via grounded counter-drone radars on-the-move. Specifically, the required pre-processing steps to remove the clutter from the data while the radar is rotating and moving on the ground are developed and discussed.

In the end, the joint detection and classification problem is traditionally solved separately by different algorithms due to the computational complexity of the task. This thesis project presents a novel framework that localizes and labels drones in an unified pipeline under the umbrella of object detection via computer vision, and that is able to operate while being static or on-the-move. Thus, an end-to-end radar data processing architecture that is robust against homogeneity constraints and based on You Only Look Once (YOLO) model is used to perform object detection in real-time. In brief, this work opens new avenues towards multi-class and multi-instance plot-based target detection and classification by transferring cross-disciplinary algorithms from computer vision into remote sensing.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
C-UAS	Counter-Unmanned Aerial Systems
OTM	On-The-Move
UAV	Unmanned Aerial Vehicle
YOLO	You Only Look Once
NATO	The North Atlantic Treaty Organization
RCS	Radar Cross Section
ATC	Automatic Target Classification
DL	Deep Learning
CV	Computer Vision
STFT	Short-Time Fourier Transform
CPI	Coherent Processing Interval
FFT	Fast Fourier Transform
ISAR	Inverse Synthetic Aperture Radar
JEM	Jet Engine Modulations
HERM	Helicopter Rotational Modulation
CW	Continuous-Wave
FMCW	Frequency-Modulated Continuous-Wave
FFT	Fast Fourier Transform
MVDR	Minimum Variance Distortionless Response
MUSIC	Multiple Signal Classification
AI	Artificial Intelligence
ML	Machine Learning
SVM	Support Vector Machine
DNN	Deep Neural Networks
MLP	Multilayer Perceptron
CNNs	Convolutional Neural Networks
RoI	Region of Interest
R-CNN	Region-based Convolutional Neural Network
IoU	Intersection over Union
AE	Autoencoder
STAP	Space-Time Adaptive Processing

Abbreviation	Definition
ULA	Uniform Linear Array
DoA	Direction of Arrival
PPI	Plan Position Indicator
RGB	Red Green Blue
mAP	Mean Average Precision
BCE	Binary Cross Entropy
HSV	Hue, Saturation and Value
SGD	Stochastic Gradient Descent
ARMA	Autoregresive Moving Average
CFAR	Constant False Alarm Rate
NA	Not Available

Symbols

Symbol	Definition	Unit
v	Velocity	[m/s]
f_D	Doppler frequency	[Hz]

List of Figures

List of Tables

1 Introduction and Overview

In this chapter, the background and motivation to partake in the research and development of Counter-Unmanned Aerial Systems (C-UAS) with surveillance radars On-The-Move (OTM) are discussed in detail. Once the need of counter-drone systems has been established, the problem statement alongside the research objectives are presented. Furthermore, the novel elements brought by this project are emphasized, juxtaposed with the research assumptions. In the end, the outline of the thesis project is provided.

1.1. Background and motivation

The drone market has been gradually expanding in the previous years, entering markets such as surveillance & monitoring, filming & photography, mapping & surveying, inspection & maintenance, precision agriculture, and many others. Furthermore, this niche industry is predicted to grow exponentially in the near future from a market value of approximately 20 billion US$ in 2021, to a staggering market cap of 50 billion US$ in 2028, as more and more fields recognize the revolutionary benefits brought by Unmanned Aerial Vehicles (UAVs) [1].

However, this phenomenal growth leads to an increased risk of misuse. NATO has defined in 2009 the UAV categories and classes [2] based on their weight, as seen in Tab. 1.1. In this project, only the first class is relevant, given the UAVs used are the micro-drone Autel Robotics Evo II weighting $1,15$ kg, and the mini-drone DJI Inspire 2 weighting $4,25$ kg. There are also Class 2 drones that weight over 150 kg, and Class 3 drones weighting over 300 kg, but these are military level drones which are labeled for tactical and strategic operations and are beyond the scope of this project.

Class	Category	Weight	Operational altitude	Mission radius	Payload
1 (less than 150 kg)	Micro	<2 kg	90 m	5 km	<0.5 kg
	Mini	2 - 20 kg	1000 m	25 km	0.5 - 10 kg
	Small	20 - 150 kg	1500 m	50-100 km	5 - 50 kg

Table 1.1: Drone categories created by NATO [2].

Furthermore, the U.S. Federal Aviation Administration highlights that in November 2021, close to 1 million drones were registered on American soil while the number of certified pilots is only a quarter of that value [3]. Thus, despite legislation stepping up to ensure adequate safety, there is always a risk of users not adhering to the code of conduct, either maliciously or unknowingly.

A prime example of drone enthusiasts not being aware of the rules and risks is The Gatwick Airport incident from December 2018 seen in Fig. 1.1, where users flew drones into restricted areas close to

the landing and take-off lanes of aircraft. As a consequence, flights were suspended for 30 hours in the holiday season and over 140,000 passengers had their flight plans disturbed, as well as huge financial costs [4]. However, drone threats do not stop at economic burdens and common nuisances. They may constitute a real danger to the population in the hands of terrorists and criminals. Such individuals may modify UAVs to accomplish malicious activities and try to bypass security measures, such as drug smuggling from distance, or even equip drones to hold & use weapons.

Figure 1.1: The Gatwick Airport was intensively covered by the news [5], as even the military had to intervene. It is seen as one of the precursors for civil counter-drone systems.

Thus, counter-drone systems are a must nowadays for any public or private event, ranging from political summits between world leaders such as NATO or G7, to entertainment events such as Formula 1 or parades, to civil or military aviation safety and many others [6]. Moreover, a high demand for C-UAS products that are capable of being anchored on moving platforms has emerged. Such systems may be used for escorting political figures from one destination to another, or even military convoys, essentially being able to move to the most suitable positions to provide optimised performances. Given the very high demand, it is of uttermost importance to create innovative and robust solutions that perform well in these novel scenarios.

With the need of C-UAS established, the first choice is to select the type of sensor to be used to detect and classify drones. Each technology presents multiple trade-offs in terms of range, identification

accuracy, capability of dealing with multiple targets, visibility in sub-optimal conditions and low light, data processing complexity and manufacturing costs. A comparison of multiple remote sensing technologies tailored specifically for counter-drone solutions was realised by P. Wellig et al. [7], which highlighted radars to offer the best overall performances, at the cost of an increased price and complexity, as seen in Tab. 1.2.

Sensor	Range	Position accuracy	Identification	Multiple targets	Low visibility	Price
Visual	++	++	++++	++	-	++
Infrared	++	++	++++	++	-	+
Acoustic	-	-	+++	++	++++	++++
Radar	++++	++++	++	++++	++++	-
Human	+	+	++++	-	-	++++

Table 1.2: Sensor comparison specifically for UAV detection, tracking and classification, from [7].

Thus, the radar scientific community has extensively researched the topic of automatic UAV detection & classification to answer the C-UAS needs. Nonetheless, there are many challenges that need to be solved to catch a rogue drone, amongst the most notable being:

- The overlapping characteristics of UAVs and birds, such as similar sizes and Radar Cross Section (RCS). This may lead to numerous false alarms and focus on unwanted targets.
- Targets flying at low altitudes or in very dense urban environments may be drowned in strong clutter, making detection next to impossible.

With the distinction between multiple types of sensors and targets covered, the next concern is how to perform Automatic Target Classification (ATC). There is a plethora of information available focusing on automatic classification of small UAVs based on their micro-Doppler signature, as this subject was extensively researched in the previous years. However, there are two main categories using either static radar with high dwell-time, or surveillance radars. In brief, the main differences are:

- The state-of-the-art heavily revolves around staring, non-rotating radars with high time-on-target. These benefit of a great number of continuous samples, which in turn yield a great amount of details that can be inferred from the micro-Doppler profiles, such as the number of blades, the frequency of the blade rotations, the potential release of payloads, and many others UAV-specific characteristics that are used as classification features.
- With surveillance radars, i.e. radars with a rotating antenna to cover a broader surveillance space, most of these modulations can not be seen in the micro-Doppler profile and the classification of different flying objects such as birds and drones becomes more challenging and prone to false alarms. However, distinction between drone vs. bio-target is still possible even when most of the specific UAVs micro-Doppler modulations are not recorded. This is thanks to the very high velocity of the propeller blades compared to the birds' wings beat, that can still be visible in the Doppler domain despite the low dwell time on target.

Albeit surveillance radars being around for a long time, very few attention was attributed to the detection and classification of UAVs using mechanically rotating antennas. The very few papers addressing these challenges [8, 9, 10] further emphasize the scarcity of resources and the gap in research.

1.2. Problem statement

This thesis project aims to integrate Deep Learning (DL) and Computer Vision (CV) pipelines within remote sensing solutions, used to reliably detect and classify UAVs using jointly rotating and moving grounded radars.

However, CV was initially developed for optical cameras which use much shorter wavelengths compared to radars. This yields sharp boundaries and clear contours of objects, allowing to quickly distinguish their shapes and therefore facilitate the detection & classification task. Radars on the other hand use longer wavelengths based on electromagnetic waves that are highly accurate at detecting targets without being influenced by low-light conditions, glare effects, or unfavorable atmospheric phenomena such as fog, snow or rain. Nonetheless, this in turn results in low-definition modeling that looses the precise shape of objects in the resulting radar data representations. While the underlying structure of radars and optical images are based on 3D matrices, the information encoded is entirely different. Optical images aim to have a high density of pixels to obtain high fidelity image reconstructions. Conversely, radar outputs are not images, but rather range-Doppler-angle data structures where Fourier transforms are used as means to interpret the data. Visualizing these data structures as images can be leveraged to decipher the embedded kinematic information of targets, revealing the range, velocity and angle of the targets. However, pixels in radar images encode entirely different information meant to characterize the targets dynamics, and are fundamentally distinct from those in optical images which aim to obtain high-definition photography.

Thus, one of the main objectives of this thesis project is to assess to what extent video processing algorithms used on optical cameras, such as You Only Look Once (YOLO), can be transferred to radar engineering to classify targets based on range-Doppler images.

Additionally, the initially static radar is anchored on a moving vehicle to create a new product that is capable of detecting drones while on-the-move, with the additional complexity of clutter removal that the combined rotational and translational motion of the radar will cause. Due to the competitive market and novelty of the system, to the best knowledge of the author, no open source information, solutions, analysis or datasets to solve the problem.

On one side this thesis offers unique insights into the challenges of fusing techniques from the computer vision community into remote sensing technologies and investigates their potential & performances on real radar systems. Secondly, it analyzes the degradation induced by moving nature of the radar in terms of micro-Doppler visibility and object detection performances.

1.3. Research objectives

Most research done on drone classification is centered around non-rotating, staring radars with high time-on-target [11, 12, 13]. This allows to capture a plethora of micro-Doppler modulations of targets, which are leveraged to perform classification and distinguish drones from other potential flying targets, i.e. birds [14]. However, in this project we focus on surveillance radars with reduced time-on-target.

Thus, the first objective is to explore the interaction with the environment of counter-drone remote sensing systems using mechanically rotating antennas while static and on-the-move. The resulting low dwell time yields an inferior, low-resolution micro-Doppler profile of the target that does not benefit of all the possible features used in the literature [15, 16, 17] for classification. While there are different methods to classify UAVs, for example based on their tracks and kinematics obtained via surveillance radars [18], or even based on sound detection [19], such methods rely heavily on constant tracking and may not work in urban and clutter rich environments. To this end, the research community investigated thoroughly the micro-Doppler signature of UAVs, which have the great advantage of being inherently different from other types of objects. Therefore, an analysis of micro-Doppler modulations obtained with surveillance radars is presented, and a clear differentiation between drones and other targets is discussed based on micro-Doppler profiles.

In the second phase, the obtained data needs to be labeled. In this thesis project two classes are considered: UAVs and clutter[1], and the data is labeled by adding bounding boxes around the drones and giving each box a label. Thus, the selection of the adequate automatic target detection & classifi-

[1] The term *clutter* refers to every target that's not of interest, such as birds, wind-turbines, ships, cyclists, etc., alongside the ground reflection.

cation architecture begins. Multiple possible algorithms are analyzed [20], and the state-of-the-art from DL is adopted to perform UAVs classification [21]. Moreover, given the data is processed such that the detector inputs are strictly images, CV pipelines based on Convolutional Neural Networks (CNNs) are emphasized. While the research community focused on ML and DL techniques, to the best of the author's knowledge, no specific CV methods were directly used on the surveillance radar data. Thus, CV architectures are integrated within radar engineering in a novel framework, and their performances to jointly detect and label drones based on range-Doppler images are analyzed.

Nonetheless, CV was initially developed for optical cameras that use significantly shorter wavelengths compared to radars, which in turn yields sharp boundaries and clear contours of the objects. Moreover, cameras are capable of recognizing 2D shapes and with higher resolutions, even fading lines can be read. Radars however use longer wavelengths and the data matrix is inherently different. The radar data structure embeds the kinematic information of targets, which allows to estimate the location, velocity and angle of the targets. Moreover, the performances of radars are more robust against environmental conditions compared to cameras. Nonetheless, radars are relatively weak at modeling the precise boundaries and contours of objects, making target identification based on low-definition shapes more challenging. Hence, one of the main research objectives of this thesis is to assess how well can CV methods be transferred to the radar domain, and what are their performances when the sharpness and clarity of images are reduced due to wavelength differences.

The final research objective of this thesis project consists of assessing the changes induced by the radar anchored on a moving platform. This is a novel counter-drone system that emerges from an in-demand product, and this thesis project aims to build frameworks that are able to answer this need. The scenario of a rotating antenna anchored on a moving platform leads to even lower-resolution and degraded micro-Doppler profile of targets, as well as increasing the background noise. Specifically, this thesis project aims to develop a signal processing chain for clutter cancellation that is robust against homogeneity and statistical assumptions which do not hold due to the translational and rotational movement of the antennas. Therefore, a clear differentiation between static and moving radar images is presented, alongside an analysis of the updated processing chain and the performances changes of the CV model.

1.4. Novelties and contributions

To achieve the research objectives of this thesis, the following novelty elements and contributions in the radar research community are brought:

- Develop a new adaptive processing chain for clutter removal for a novel C-UAS, namely grounded surveillance radars that are simultaneously moving and rotating;
- Integrate computer vision pipelines within radar engineering to jointly solve the detection and classification problem, and assess the performances of drone detection via range-Doppler plots;
- Validate the proposed algorithm on experimental data in real scenarios with challenging objects that yield potential false alarms, such as wind-turbines;

Moreover, given the promising results alongside the novelty and potential of the proposed method, a paper based on this work was written for the IEEE Radar Conference, San Antonio, Texas, Spring 2023. The research paper can be found in Appendix A.

1.5. Thesis roadmap

Chapter 2 introduces the evolution of the state-of-the-art of UAVs signature analysis based on time-frequency distributions and range-Doppler snapshots, alongside methods for automatic classification based on ML and DL.

Chapter 3 covers the preliminary knowledge for DL required to build comprehensible AI, and break down the black boxes by explaining how CV pipelines function, juxtaposed by their development history.

Chapter 4 highlights the radar system used, as well as the analysis of real recorded data. Moreover, an in-depth analysis of the signal & image processing chain is discussed, and clear examples of the targets' micro-Doppler modulations, as well as the limitations of this method are highlighted.

Chapter 5 analyzes the performances of the proposed methods for UAVs detection using static radars. Given CV is traditionally used with high-resolution cameras, a clear assessment on how such pipelines perform on radar images is analyzed, and improvements to adapt the method to this novel scenario are proposed.

Chapter 6 highlights the new performances of YOLO for the novel C-UAS consisting of the same radar anchored on a moving platform. Moreover, the required improvements to the adaptive processing chain are shown, juxtaposed by further improvements proposed to the object detection pipeline.

Finally, the main conclusion is drawn and the recommendations for future research are proposed in **Chapter 7**.

2 Literature Review

This chapter covers how the state-of-the-art tackles the challenges of automatic classification of UAVs with radars. After a brief introduction in how the micro-Doppler is visualized, the main characteristic of multiple types of drones and birds are highlighted. The analysis is carried with staring radars as well as surveillance radars. In the end, AI techniques used to perform ATC are discussed, and a summary encapsulating the main methods used for classification based on the micro-Doppler is presented.

2.1. The Doppler and Micro-Doppler effect

Radars transmit electromagnetic signals with pre-defined waveforms that are reflected from the illuminated environment and captured by the receivers. The changes in the characteristics of the received signals are used to determine the properties of the scanned objects. For example, when an illuminated target is moving with a constant velocity, the returned signal will present an extra modulation, known as the Doppler shift [22]. Moreover, radars have the advantage of not being greatly influenced by poor environmental settings such as rain, snow or low light, compared to other types of sensors which are severely affected by such conditions.

Furthermore, radars are capable of capturing targets' micro-motion dynamics such as mechanical vibrations, tumbling or rotations. Therefore, above the well known bulk Doppler effect, these additional micro-dynamics induce an extra-modulation in the reflected signal, known as the micro-Doppler [23]. This effect was initially investigated in laser systems [24], known as ladars, and the amplitudes analysis of micro-vibrations was patented by K.J. Parker et al. since 1992 [25]. In radar applications, the micro-Doppler signature of targets has been widely analysed as it shows great potential in the classification of targets based on their unique micro-motion signatures. In order to visualize these micro-Doppler modulations, high-resolution time-frequency distributions are being usually used, such as spectrograms or cepstrograms, amongst others.

Albeit multiple joint time-frequency distributions have been extensively analysed [26, 27], i.e. linear distributions such as the STFT, Gabor Expansion, the continuous wavelet transform, or quadratic such as the Cohen's class distributions including the Wigner–Ville distribution and its variations, the most widely such distribution used in the remote sensing literature is the STFT which yields the spectrogram. The main reasons spectrograms are preferred in C-UAS are:

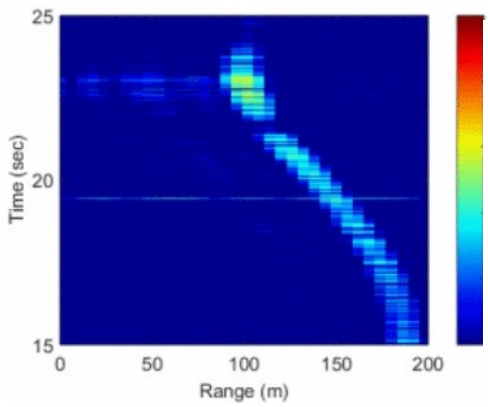
- Mathematically simple;
- The short windows allows to visualize very fast micro-dynamics, such as drones' propellers;
- While it can not maximize both the time and frequency resolutions simultaneously, it is computationally cheaper and can be tailored for the desired application.

Other fancier distributions posses different trade-offs, such as an increased complexity cost or the emergence of unwanted artifacts. All these transformations are used in a wide area of research topics, such as speech processing, wave propagation, bio-medicine, optics etc., and have also been broadly used in the radar literature to capture the micro-Doppler signature of targets [28, 29].

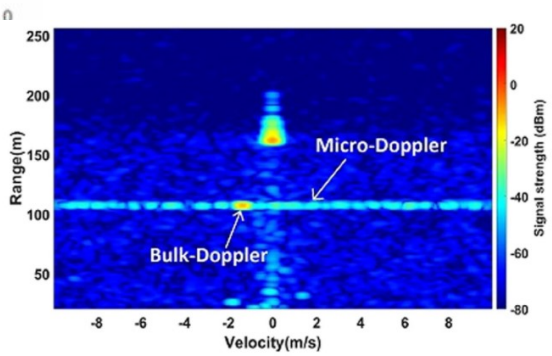
In order to obtain the spectrogram used to visualize the micro-Doppler modulations of targets, a STFT is applied across the summed range bins obtained from multiple sweeps, which represent the Coherent Processing Interval (CPI). The STFT is preferred rather than a typical Fast Fourier Transform (FFT) because it bypasses the abrupt frequency changes in the signal which may destroy the result. It achieves this by sliding a window of predefined length across the entire signal, and applies an FFT on each segment of the signal that has an almost constant frequency. Depending on the length of the window, we may improve the resolution either in the temporal or in the frequency domain. Unfortunately, one can not improve both resolutions simultaneously, and a trade-off decision must be made: either use a short window to improve the temporal resolution to catch fast modulations in time at the price of a poor frequency resolution (the spectrogram), or the opposite (the cepstrogram).

2.2. Doppler and RCS characterisation of UAVs

Before looking the the micro-dynamics of drones, it is important to highlight why the research chose to focus on this visualization method. Traditionally, to obtain the profile of the target, a 3D Fourier Transform (or a 3D matched filter) is applied across the data to obtain the Range-Time, the Range-Doppler, and the Range/Doppler-Angle profiles. However, only the first two are of interest in this case. The direction of arrival is not investigated in this project because the elevation is easily obtained via multiple beamforming at the receiver channels via traditional signal processing. Meanwhile, given the antennas are rotating, the target azimuth is estimated via the classification pipeline. Once the drone is classified based on the spectrogram, the location of the drone in the picture also yields the azimuth, given it is known where the first sweeps begin with respect to the true North.



(a) Range-Time profile of a drone moving away from the radar from [14].



(b) Range-Doppler profile of a drone moving away from the radar from [30], where we can see the bulk Doppler, and the propellers being extended across the entire spectrum due to their very fast rotation speed.

Figure 2.1: The Range-Time and Range-Doppler profiles mostly capture the main movement of the main body of the drone, i.e. either moving away from or towards the radar.

In Fig. 2.1a we can see the bulk kinematics of a drone. Unfortunately, by using solely this visualization method we capture almost no specific UAVs modulations and the target recognition is very difficult to achieve. Nevertheless, in Fig. 2.1b we can clearly see both the bulk Doppler and micro-Doppler created by the main body of the DJI Phantom 3 drone and its propeller blades.

Another way of looking at the consumer UAVs was presented by C.J. Lin et al. in [31], who present multiple 2D Inverse Synthetic Aperture Radar (ISAR) images in optimal, laboratory conditions. In Fig. 2.2 it is highlighted that a clear visualization of larger drones is possible, although the RCS level being relatively low. Unfortunately, while this method shows great potential in recognizing the type of target,

the performances in real-life scenarios were not analysed. The measurements were taken only in an anechoic chamber at UAV Direct, Liberty Hill, TX, USA.

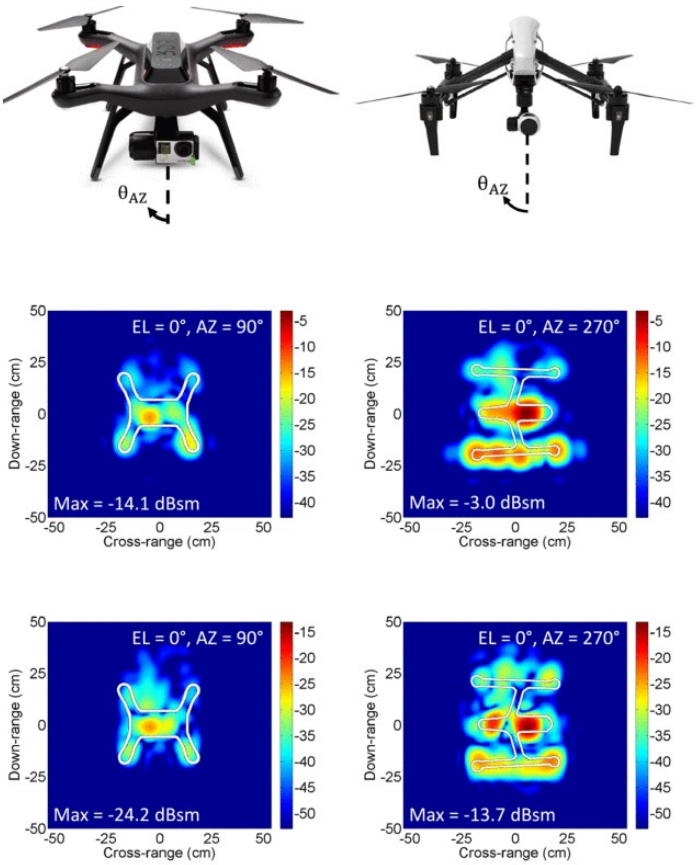


Figure 2.2: Drone RCS analysis based on ISAR images from [31], performed in laboratory conditions, i.e. an isolated environment without clutter.

Thus, while the range-time profile offer valuable insights of the drone main dynamics and the range-Doppler plot begins to show specific characteristics of UAVs, the research community looked further into how to analyse the micro-dynamics of drones, namely more specific micro-Doppler modulations. Moreover, despite ISAR images offering better insights of the geometry of drones, their reliability in experimental real-life situations may be drastically affected and degraded. Furthermore, another limitation of using this technique is that the radar used in this project is continuously rotating.

2.3. Micro-Doppler analysis of UAVs and feature extraction

2.3.1. Radar with high dwell time

The micro-Doppler signature of UAVs was thoroughly analysed in the previous years, however the state-of-the-art revolves around staring pulse, Continuous Wave (CW) or Frequency-Modulated Continuous-Wave (FMCW) radars with high illuminating time and non-rotating antennas. This approach yields a fine Doppler resolution thanks to the high time-on-target, up to the point that many features such as the rotating blades periodicity can be almost perfectly seen and further leveraged for classification.

J.J.M de Wit from TNO alongside R.I.A. Harmanny & G. Prémel-Cabic from Thales Nederland have extensively investigated the micro-Doppler signatures of drones and birds since as early as 2012 [16, 32]. They showed that the very fast rotation of the propeller blades induce specific micro-Doppler

modulations that appear in the spectrogram as spectral lines named Jet Engine Modulations (JEM) or Helicopter Rotational Modulation lines (HERM), depending on their orthogonality. Such JEM can be seen in Fig. 2.3, where the CW radar had a total time-on-target of approximately 25 seconds.

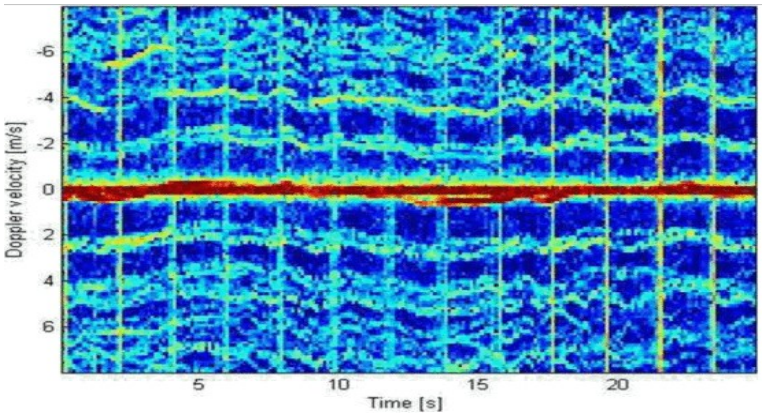
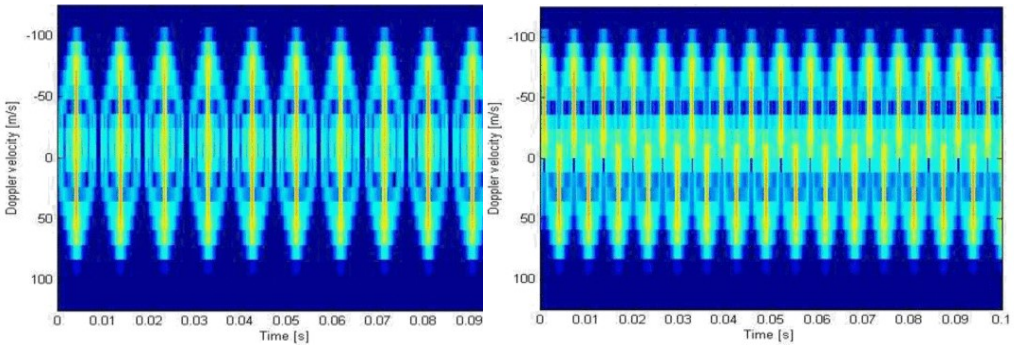


Figure 2.3: Spectrogram of a hovering rotary-winged drone from [32] using a X-band CW radar. The main component centered around 0 Hz represents the main body of the static drone, while the pattern of horizontal lines spread across the spectrum represent the jet engine modulations induced by the propeller blades.

Additionally, it is possible to see the HERM modulations induced by the propellers flashes in more detail by zooming in across the time axis and by using a short window. In this case, a better temporal resolution is obtained, as seen in Fig. 2.4 and Fig. 2.5, given we can clearly see the temporal pattern of the rotating blades. Based on this approach, another possible feature used classifying the type of drone is the symmetry of the micro-Doppler profile, which is related to the number of propeller blades. To facilitate the visualization and distinction between odd or even number of blades and not be affected by real-world clutter, simulated spectrogams may be obtained by mathematically modeling of the rotary-wings, as seen in Fig. 2.4.



(a) Spectrogram of a drone with an even number of blades resulting in a symmetric pattern.

(b) Spectrogram of a drone with an odd number of blades resulting in an asymmetric pattern.

Figure 2.4: It is possible to classify the type of drones by their number of propeller blades. Depending on their parity and analysis of the symmetry of the chosen time-frequency distribution, it is possible to gain more insights about the type of UAV. The data was simulated by J.J.M de Wit et al. in [32].

Nonetheless, simulated data does not contain real-word artifacts such as the presence of noise across the entire spectrum, or the clutter centered around 0 Hz, as seen in Fig. 2.5. This may detrimentally affect the classification performances due to a poorer plot intelligibility, or even lead to potential false alarms. However, ideal condition are rarely, if ever, encountered in real-life scenarios. Thus, it is quintessential to have all these natural artifacts present in the training data so that the classifiers may properly perform in sub-optimal scenarios. Moreover, due to costs and the huge memory nature of radar data, there are very limited available datasets that contain natural noise or clutter, such as rain, snow, fog etc., that may boost classification performances by expanding the diversity of training images.

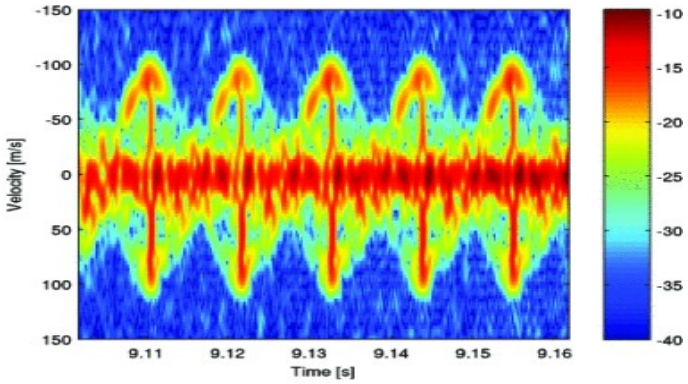


Figure 2.5: Spectrogram of a hovering rotary-winged drone from [15] using a X-band CW radar. Here we can clearly observe the rotation speed of the blades, which are either moving towards or away from the radar. Additionally to simulated spectrograms, the presence of static clutter centered at 0 Hz is clear and makes the ATC problem more challenging.

With the HERM and JEM lines, the number of propeller blades, and the benefit and challenges of natural artifacts covered, a more in-depth analysis of drones' micro-Doppler signature can now be covered. In Fig. 2.6, B-S. Oh et al. [17] highlight more in-depth the features used to effectively classify the type of drone. They show how the symmetry of the spectrogram, the rotational speed, as well as the propellers' cycle can be mathematically analyzed and used to model the target geometry.

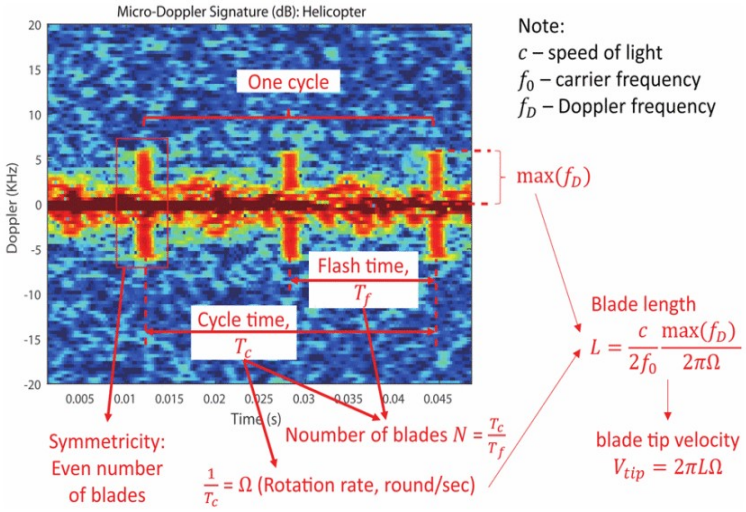


Figure 2.6: A very short spectrogram of a hovering drone from [17] using a X-band CW radar. A clear analysis of drones' micro-Doppler modulations is shown, focusing on the flashes induced by the fast rotation of the propeller blades. These features may even be used to distinguish between different kind of drones.

Moreover, there are multiple types of UAVs besides the helicopter design. A. Huizing et al. [33] conducted multiple measurements and computer simulations to visualize the micro-Doppler modulations for different kind of drones, including fixed wing aircraft, alongside helicopter, quadcopter and octocopter designs as seen in Fig. 2.7. It is clearly seen that the size of the propeller greatly influences the specificity of the micro-Doppler modulations. As a rule of thumb, the more blades the UAV has, the smaller they will be, which in turn are less reflective and thus are more difficulty to visualize in the presence of strong noise. On the other side, in the case of fixed winged drones, the modulations appear from the single propeller which is located either in the front or the back of the aircraft. Depending on the direction of motion, this may represent an advantage in capturing possible JEMs or HERMs, or may be hidden by the main body of the drone and lead to missed classifications. Furthermore, in the case of a helicopter design, the main rotor will lead to a stronger reflections, given the transmit electromagnetic waves will reach it easier, compared to the smaller propeller located at the back of the aircraft.

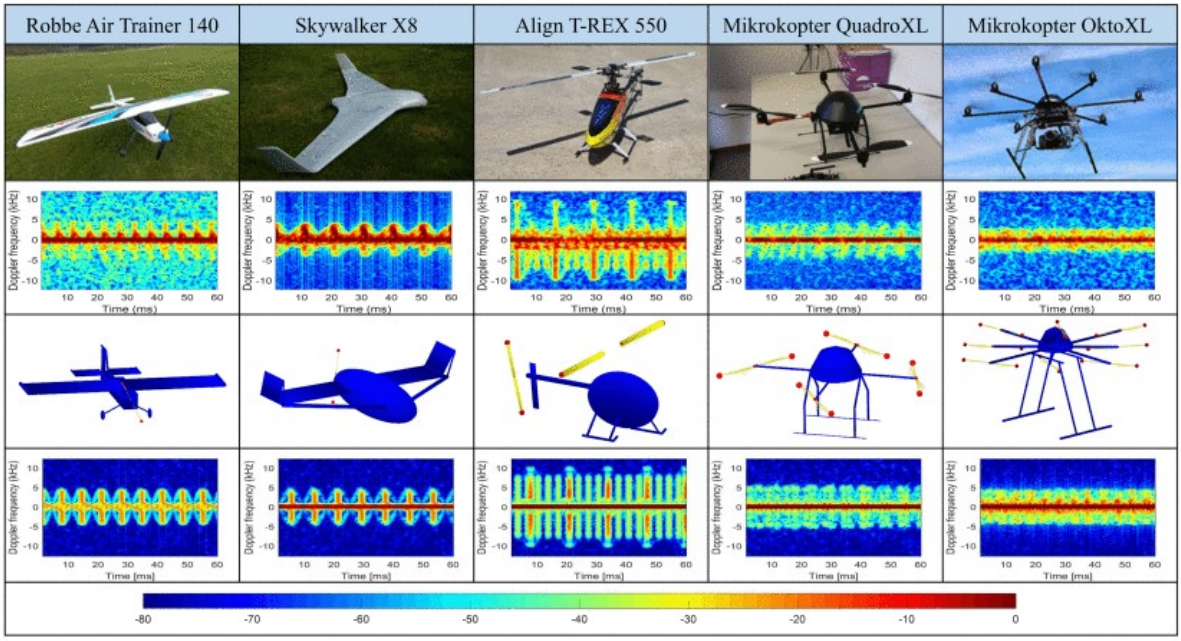
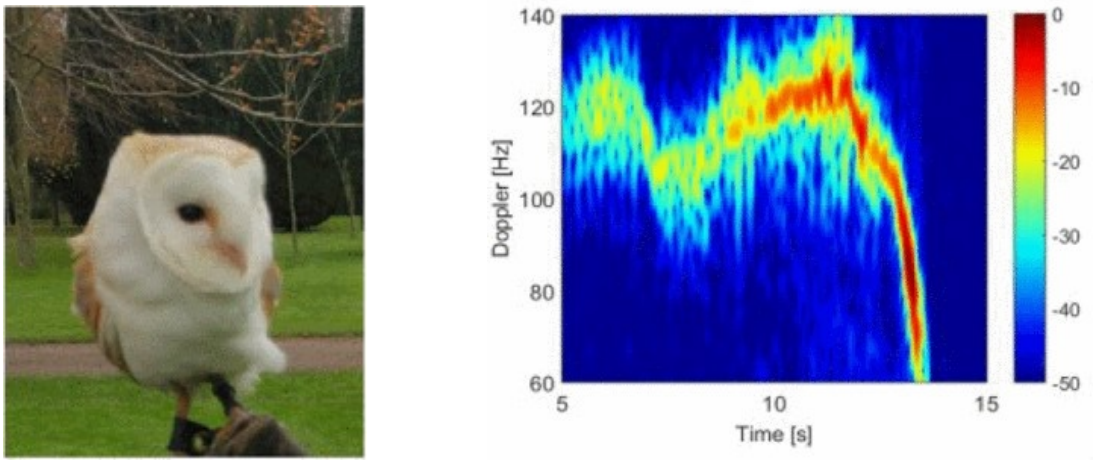


Figure 2.7: Multiple spectrograms of different types drones from [33], each presenting different micro-Doppler modulations. The top line represent experimental measurements containing natural noise and clutter, while the second row spectrograms were obtained via computer simulations. Based on the geometry of the target, the micro-Doppler modulations greatly differ.

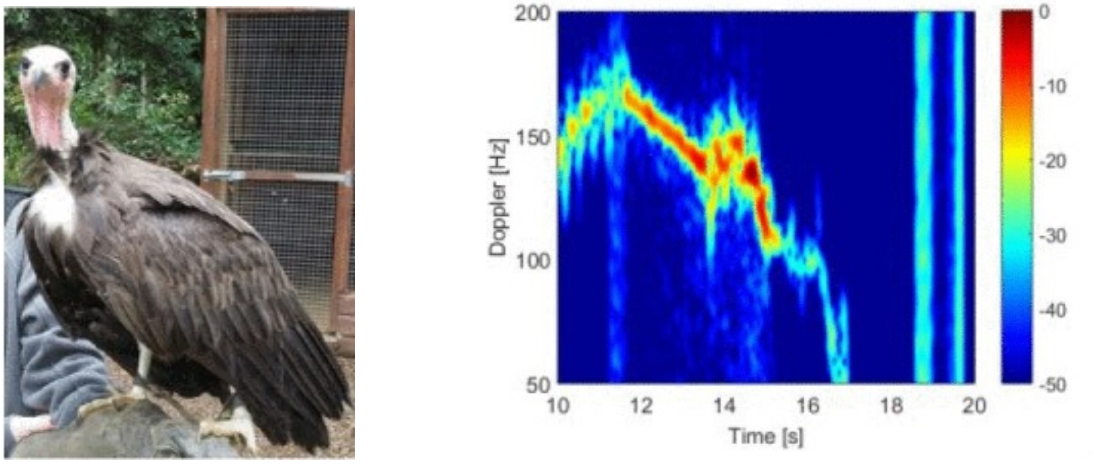
Regarding the false alarms, the prime type of aerial target leading to UAVs miss-classification and unwanted detections are biological flying targets, namely birds. Given their similar sizes and RCS, radars using solely range-Doppler images may easily misinterpret drones and birds. F. Fioranelli et al. [14] analyzed real-life data gathered with a X-Band radar by comparing the micro-Doppler modulations of a DJI Phantom 2 drone and different kind of birds, such as barn owls weighting approximately 300g seen in Fig. 2.8, or hooded vultures who can weight approximately 2 kg seen Fig. 2.9. Note, once again, the very long time-on-target.



(a) Small to medium sized bird, namely a barn owl.

(b) Spectrogram of the barn owl.

Figure 2.8: Barn owl micro-Doppler analysis from [14].



(a) Large bird, namely a hooded vulture. **(b)** Spectrogram of the hooded vulture.

Figure 2.9: Hooded vulture micro-Doppler analysis from [14].

Based on Figures 2.8 & 2.9, we can notice that the size of birds play an important role in the micro-Doppler modulations they create, besides the differences in RCS. Due to the differences in their aerodynamics, each specie will have a more-or-less unique wing beat pattern. For example, large birds such vultures or condors can glide with a higher efficiency, thus their wing beats frequency is a lot lower. On the other side of the spectrum, a small sized bird such as a hummingbird will have a wing beat with a very high frequency, without ever stopping. However, compared to UAVs, birds create much narrower micro-Doppler modulations no matter their sizes. Therefore, despite the very similar sizes and RCS, the micro-Doppler provides a great method to differentiate between organic and inorganic aerial point targets.

2.3.2. Radars with reduced dwell time

In many practical applications such as ground-based surveillance, radars transmits a wide beam in elevation and a narrow beam in azimuth, but is rotating in azimuth at a fixed rate to ensure 360° spatial coverage. Thus, the received signals carry limited information due to the significantly reduced dwell time. This in turn yields a very poor micro-Doppler resolution, leading to a lack of details and features that may be used in automatic classification. This problem is even more pronounced in the discrimination of similar point targets, such as drone vs. birds. B-S. Oh et al. [8] and H. Sun et al. [9] further emphasize the scarcity of literature addressing the classification problem of UAVs using grounded FMCW surveillance radars.

Furthermore, most of the aforementioned drone micro-Doppler modulations were recorded using static CW Doppler radars. This is because a signal rich in information with high time-on-target will yield close to ideal classification features. However, such non-modulated waveforms will lack the temporal information used to estimate range. Moreover, such radar systems may require human intervention to steer the antenna towards the angle of interest, which is highly unfeasible for small and agile targets such as drones. One solution would be to deploy 4 antennas in a squared geometry to ensure 360° azimuth coverage [12, 13], but the cost of such a design will quickly grow. Another option would be to select a-priori a region of interest where the radar should look, instead of continuously scanning in every directions.

However, in the case we want to ensure full spatial coverage in azimuth and also keep the costs relatively low, rotating FMCW radars were developed as an option for counter-drone surveillance systems. Thanks to the modulated waveform, the time delay (and thus the range) can be estimated, alongside the Doppler and micro-Doppler profiles. Moreover, compared to traditional pulse radars, FMCW radars do

not suffer of eclipsing zones induced by the duty cycle between transmitting and receiving signals. For this reason, FMCW radars became a very popular solution adopted in both surveillance and automotive applications.

In [8], B-S. Oh et al. investigated the micro-Doppler signature of drones without looking at the spectrograms, but rather only the range-Doppler plots obtained via FFT using an X-band FMCW surveillance radar. In Fig. 2.10, we can see multiple range-Doppler profiles of three types of drones at close and long range. Firstly, it can be seen that the main body of the drone is clearly visible as a more pronounced bulk Doppler, and that the propellers are vertically extended across the entire frequency spectrum surrounding the drone. Secondly, it is noticed that the propellers modulations are stronger in the figures on the left column, given the target is closer to the radar. In the case the target is twice as far, as shown in the right column, the micro-Doppler modulations of the propellers are barely visible and only the bulk Doppler can be seen, which is similar to the signature of birds.

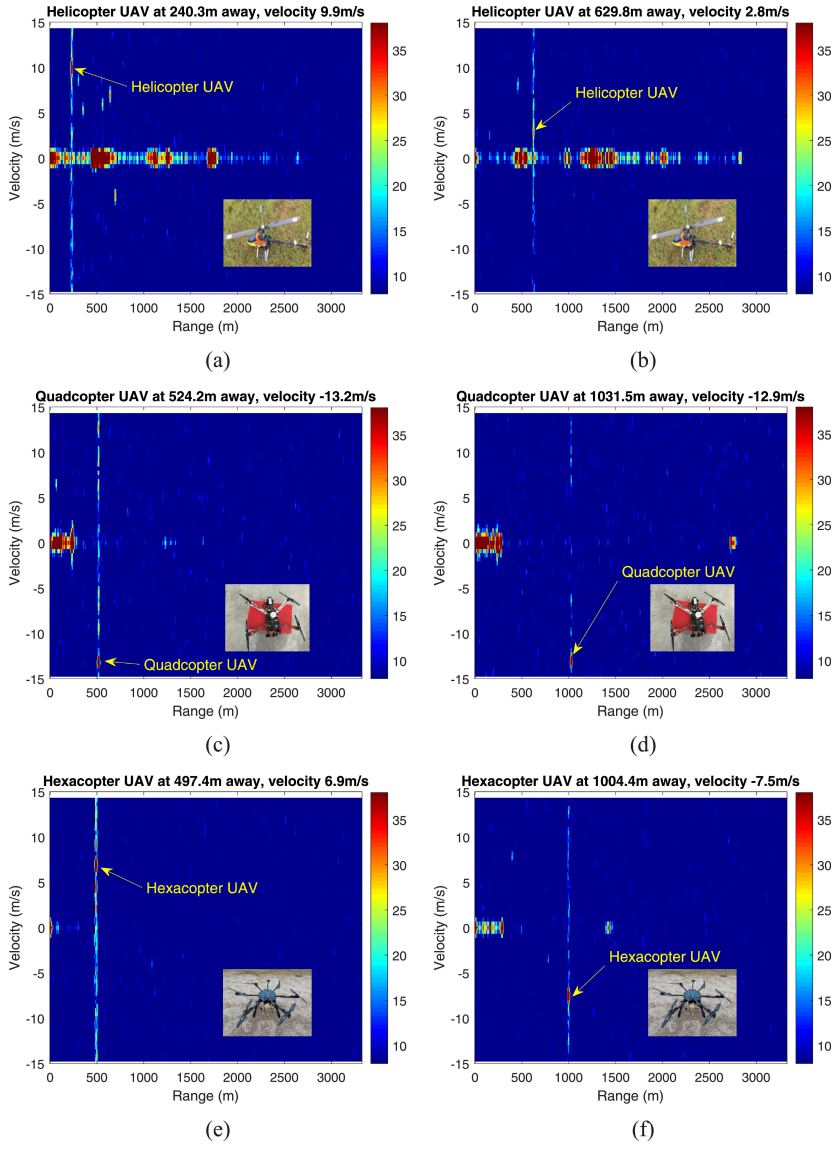


Figure 2.10: Multiple range-Doppler obtained via 2D FFT of different types of drones, from [8]. In the left column, the drone is located relatively close to the radar, while in the right column, the drone is twice as far. The visibility of the drone and its propellers degrades proportionally with distance.

Furthermore, the range-Doppler plots of different targets were also analysed [8] to compare their modulations. In Fig. 2.11, the most important feature used to lead to a binary classification decision (drone vs. non-drone) is the length of the micro-Doppler modulation across the spectrum. In the aforementioned

case, the drone's propellers yield wide Doppler modulations due to the very fast propeller rotations, while every other type of object reveal a relatively narrow Doppler shift. This difference in the velocities of the micro-dynamics of targets represent the most important classification feature that can be leveraged to properly distinguish UAVs from other targets, albeit the low dwell time. However, this differentiation may not work if the drone is too far and the rotary blades can not be seen by the radar.

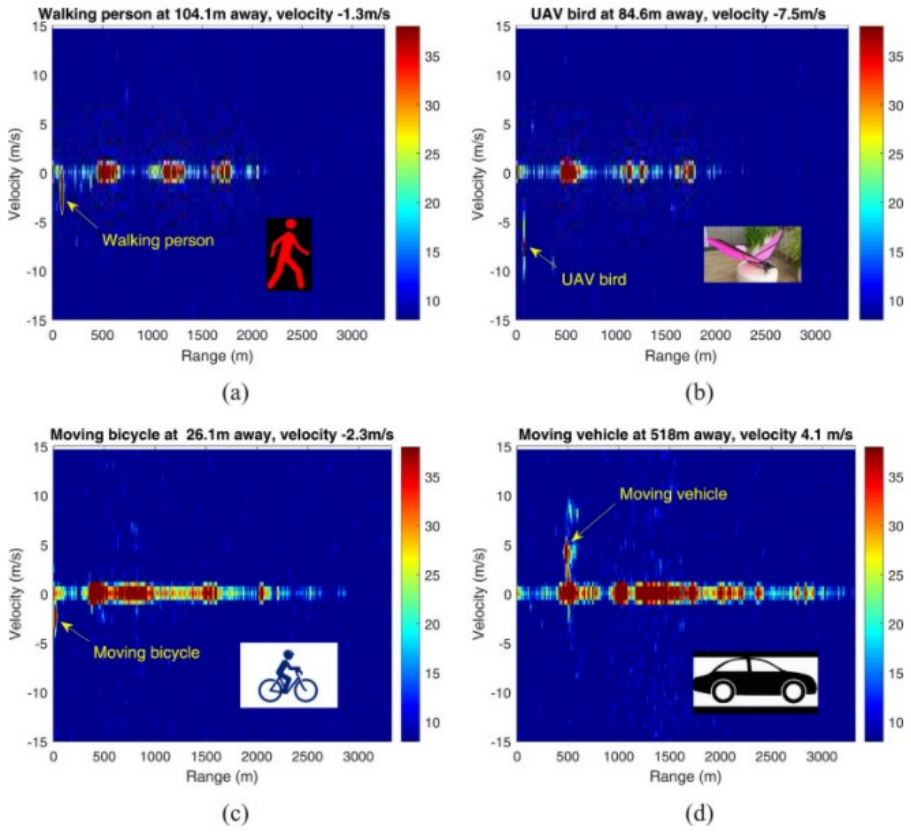
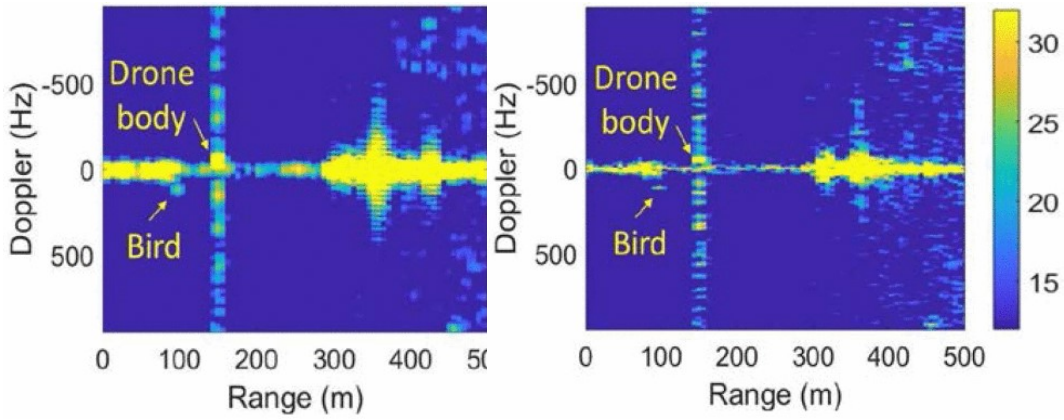


Figure 2.11: Multiple range-Doppler obtained via 2D FFT of different targets, from [8].

Given the time-on-target with this type of sensors is drastically reduced, multiple techniques to improve the resolution were investigated. For example, temporal processing using super-resolution algorithms from array processing such as Minimum Variance Distortionless Response (MVDR) and Multiple Signal Classification (MUSIC) [34] were initially proposed to detect & classify large aircraft. However, these methods require multiple snapshots to estimate the covariance matrix from multiple correlated sweeps. Thus, such classical algorithms may not be successfully applied to mechanically rotating antennas due to the relatively low CPI [9].

H. Sun et al. [9] proposed an iterative statistical method, named Iterative Adaptive Approach that may achieve similar performances to super-resolutions algorithms and requires only a single temporal snapshot. This is a great achievement as it bypasses all the well-known issues of array processing algorithms for mechanically rotating antennas. However, this technique still uses a similar framework as MVDR or MUSIC, and requires the estimation of the covariance matrix via an iterative method, as well as solving multiple sets of linear equations for each possible target steering vector. Thus, the main question is whether it is computationally worth to choose this technique over the traditional FFT. In Fig. 2.12, experimental measurements were taken using an X-band rotating surveillance radar and the results show that while smaller sidelobes and a finer resolution are indeed achieved, the increase in complexity may not be justifiable for the slight increase in plot accuracy.



(a) 2D FFT which is traditionally used to obtain the range-Doppler profile of the target.

(b) Iterative Adaptive Approach used to yield a finer Doppler resolution in the plots.

Figure 2.12: Range-Doppler profiles of a DJI Phantom 3 quadcopter obtained using a X-band surveillance radar. The plots are used to compare traditional FFT with the more computationally expensive iterative approach used to improve Doppler resolution, from [9].

Automotive radars present similar low time-on-target constraints due to their moving nature. However, compared to drones that posses three-dimension kinematics, the automotive industry is mostly interested in grounded targets. This in turn offers advantages, as even if the radar is rotating, all the information is focused in the azimuth, while the elevation can be neglected. Furthermore, this allows for a simpler array geometry that maximizes the resolution in a single dimension and does not have to make compromises in another. Moreover, the detection & classification ranges do not need to be extended over a couple kilometers as in the case of surveillance radars.

S. Gupta et al. [35] investigated ATC using Machine Learning algorithms via range-angle images obtained with very high frequency FMCW radars. In Fig. 2.13, they showed how a rotating moving radar with high angular resolution can be used to properly classify other grounded moving targets. In this case, the classification is done based on the RCS of the objects rather than the micro-Doppler, as vehicles are far more reflective than pedestrians. However, it is easily seen that discrimination between humans and drones is not possible via this method without a-priori information.

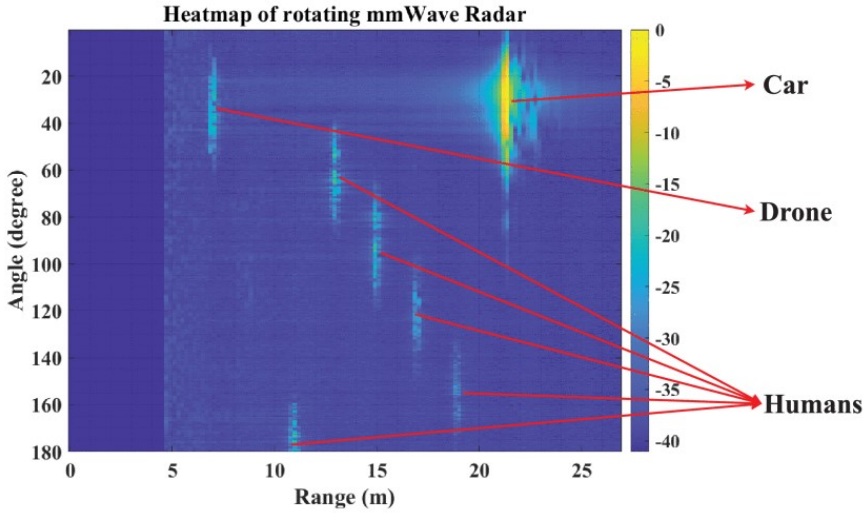


Figure 2.13: Range-angle plot obtained via an automotive rotating moving radar, from [35]. Other vehicles are easily recognized by their strong RCS, while smaller targets such as pedestrians or drones can not be differentiated.

2.4. Radars On-The-Move

Detection via moving radars has been well investigated before, and represented a very hot researched topic in the past. However, this was traditionally done via airborne or even spaceborne moving radars as many intelligence, surveillance, and reconnaissance missions recognized the growing need of performing reliable detection of grounded targets from a safe distance. The main technique used in such scenarios is Space-Time Adaptive Processing (STAP) [36, 37]. However, this technique is one of the most complex processing chains in radar literature, and has very high computational costs.

Besides the military and surveillance markets, automotive radars are a fast-growing niche that face similar constraints. However, they are more focused on extended and very reflective targets such as trucks and cars, or even pedestrians. Furthermore, given such targets are always grounded, all the processing is done in a single dimension, namely in azimuth. Thus, the array geometry allows to obtain a very fine angular resolution and thus most research is centered around angle-Doppler images [35].

Unfortunately, no methods have been proposed for small aerial target detection & classification using grounded moving surveillance radars, as seen in Fig. 2.14. Nonetheless, the DL & CV methods mentioned in Subsection 2.5 are adopted in this project, and their performances in these novel, sub-optimal scenarios are assessed.

Figure 2.14: Robin Radar IRIS Drone Radar during the field tests to gather data for the project. The radar is anchored on a moving vehicle in a real-life and clutter rich environment, containing wind turbines, other vehicles, bids, cyclists and so on.

2.5. Architectures for Automatic Target Classification

There are multiple machine and deep learning methods used to perform ATC that were investigated in the remote sensing literature. However, compared to other fields that leverage techniques from Artificial Intelligence (AI), the radar community does not benefit of large, diversified and labeled datasets. Operating radars requires trained engineers or technicians, and companies tend not to share personal datasets as they contain sensitive information. Furthermore, obtaining diversified data containing natural clutter such as rain, fog, snow etc., is hard to obtain due to the difficulties of going on the field to record data in these harsh conditions.

Alas, ML classification algorithms that scale well with relatively small training data sets are preferred.

The most widely used such classifiers are the K-Nearest-Neighbours [38], Support Vector Machines (SVM) [9, 15, 16, 39] and Naive Bayes [16, 32, 34, 39], which are encountered, or used to compare results of different approaches, in most C-UAS research.

Nevertheless, all these methods require hand-picked feature selection used to train the classifier on. Feature engineering is then an essential but very challenging step, as it requires great expertise [20]. This is one of the reasons why the specific signature of UAVs were extensively analysed by experts, as presented in Section 2.3. Moreover, ML techniques require thorough pre-processing before they are fed into the classifier, otherwise the performances may decrease significantly.

Thus, Deep Learning has became more and more popular as an alternative for ATC. Neural networks have the great advantage of automatically extracting features from the input data. This greatly facilitates the training and classification process with less human intervention, at the cost of an increased computational complexity. Furthermore, DL can work directly with images as inputs, which allows to maintain the spatial correlation between neighboring pixels, compared to ML techniques that require flattening the input image into vectors. Alas, the only requirements in DL is to have enough diversified labeled data to train the networks on, and to make the processing real-time.

Hence, given the computational complexity of DL and its requirement to use bigger labeled datasets, the research community initially focused more on ML techniques in the earlier years, as seen in Table 2.2. However, DL pipelines can be pre-trained on large, uncorrelated datasets from different fields, and then fine-tuned on radar data to perform reliable drone classification [21, 40]. Thus, N. Mohajerin et al. [18] have shown that by using Artificial Neural Networks (ANNs) on the tracks of UAVs, reliable classification with 99% accuracy is possible. Moreover, Kim et al. [40] used Convolutional Neural Networks (CNNs) to perform image classification based on drones' micro-Doppler signature and obtained accuracy scores of almost 100% by using the pre-trained GooLeNet architecture [41]. In [42], D.A. Brooks et al. have tested CNNs, Multilayer Perceptrons (MLP) and Recurrent Neural Networks (RNNs) to test the temporal DL classification capabilities. Given CNNs input are directly the non-flattened spectrograms, they showed the best results as they preserve the spatial information of images. However, MLPs also performed well and given their shallow architecture, they may easily be trained on small datasets since they do not have many parameters to learn. In the end, RNNs are widely used in temporal sequences as they have the ability to use prior information to improve the prediction output. However, the input time-frequency distributions have to be flattened into a 1D series of vectors as input, but showed higher accuracy compared to MLP, given their more complex structure. In the end, J. d. Wit et al. [33] showed that classification of different kind of UAVs is possible via CNNs and RNNs, but the accuracy may vary significantly depending on the geometry of the done, i.e. helicopter versus fixed wing drones.

The main compromises between ML and DL are summarized in Table 2.1. A more in-depth analysis of the DL and CV techniques is given in Chapter 3.

	Machine Learning	Deep Learning
Feature engineering	Handcrafted features	Automatic extraction
Labeled dataset size	Scales well with small datasets	Performs better with larger datasets
Computing power	Relatively low	Relatively high
Human intervention	Relatively high	Relatively low

Table 2.1: Key-list of trade-offs between ML and DL.

Moreover, Computer Vision is a sub-field of DL. It is based on image processing via CNNs and real-time object detection, but was mostly used with high-resolution cameras. The most widely used CV pipelines for real-time small objects classification to date are Region-based Convolutional Neural Networks (R-CNN) and You Only Look Once (YOLO) pipelines, and their more recent versions. These techniques were successfully used to detect and classify UAVs using cameras in [43, 44, 45].

2.6. Summary

This chapter presented how the research literature tackles the difficulties brought by UAVs classification using radar systems. Firstly, the challenges in obtaining clear plots of UAVs using staring and surveillance radars were covered. Moreover, depending on the dwell time, it is possible to obtain either a high or low-resolution micro-Doppler profile of the target, which represents the input to the classifier. Depending on the richness of the obtained spectrograms, different features may be used to perform accurate classification, but all of them are typically based on the modulations induced by the propellers.

Moreover, multiple AI-based methods to perform ATC were discussed, based on ML and DL. While ML was initially more popular thanks to the lower computational complexities and better scaling with small datasets, the main drawback is the requirement of handcrafted features. To bypass this challenge, DL has been proposed and utilised, despite its computational and scaling difficulties. Solutions to bypass these issues were proposed, and their feasibility and performances are to be assessed in the following sections.

Table 2.2 summarizes the most used techniques in terms of radar technologies and micro-Doppler visualization techniques used, alongside the chosen classifiers. The main conclusion is that while both ML and DL algorithms have been successfully investigated to perform automatic drone classification, to the best knowledge of the author, no computer vision pipelines were tested on any kind of range-Doppler snapshots obtained with radars, even though they show great potential in real-time object detection and localization. Thus, one of the main objectives of this thesis is to integrate computer vision methods within the radar signal processing chain, and to assess their performances.

Objective/application	**Visualization**	**Radar**	**Classifier**
Aircraft classification (2002) [34]	MUSIC	CW	Bayes classifier
UAVs micro-Doppler analysis (2012) [32]	Spectrogram	CW	Bayes classifier
UAVs micro-Doppler feature extraction (2014) [16]	Spectrogram, Cepstrogram	CW	SVM (linear and non-linear), Naive Bayes
Small UAVs and birds micro-Doppler classification (2014) [39]	Spectrograms	CW	SVM (linear and non-linear), Naive Bayes
UAVs detection and classification (2014) [18]	Tracks	Pulse	ANN
Micro-UAVs and birds micro-Doppler measurements (2016) [14]	Spectrogram	Pulse	NA
Polarimetry classification of UAVs and birds (2016) [38]	Tracks	Pulse	K Nearest Neighbors
Drone classification (2016) [40]	Cadence velocity diagram	FMCW	CNN
Mini-UAV blade flash reconstruction (2017) [17]	Spectrogram, Empirical mode decomposition	CW	NA
UAVs and birds' micro-Doppler signatures (2018) [30]	Range-Doppler, Spectrogram	CW	NA
Drone micro-Doppler detection & classification (2018) [15]	Spectrogram	CW	Boosting, SVM
Drone micro-Doppler classification (2018) [42]	Spectrogram	Pulse	CNNs, MLP, RNNs
UAVs detection (2019) [9]	Iterative adaptive approach	FMCW	SVM
Mini-UAVs classification (2019) [33]	Spectrogram	CW	CNN, RNN
UAVs Characterization (2021) [46]	Spectrogram	CW FMCW	NA
Drone detection & classification via radars on-the-move (proposed method, 2022)	Range-Doppler	FMCW	YOLO

Table 2.2: Summary of the proposed UAVs micro-Doppler visualization techniques, as well as the type of radar used and the classification method.

3

Preliminary Knowledge for Computer Vision for this thesis

This chapter covers the principles of deep learning and computer vision that are mostly relevant for this thesis project. The purpose is to break down the black boxes and understand the underlying nature of the chosen pipelines. Based on this understanding, improvements and adaptations to the proposed algorithms are made in the following chapters, in order to harmoniously integrate CV techniques into more conventional radar signal processing approaches.

3.1. Basic principles of deep learning

Deep Neural Networks (DNNs) are a special type of artificial neural networks that contain multiple levels of hidden layers. If the network is strictly feed-forward, i.e. there are no loops, then it is called a Multilayer Perceptron. These in turn are made out of neurons, followed by activation functions such as Sigmoid or ReLu. By training the network on data, the weights & bias of neurons are computed and the DNN learns the intrinsic non-linear pattern. A simple architecture of a DNN can be seen in Fig. 3.1, which takes the input data in the first layer and automatically learns the most prominent features inside the hidden layers, based on which it outputs a classification prediction in the final layer.

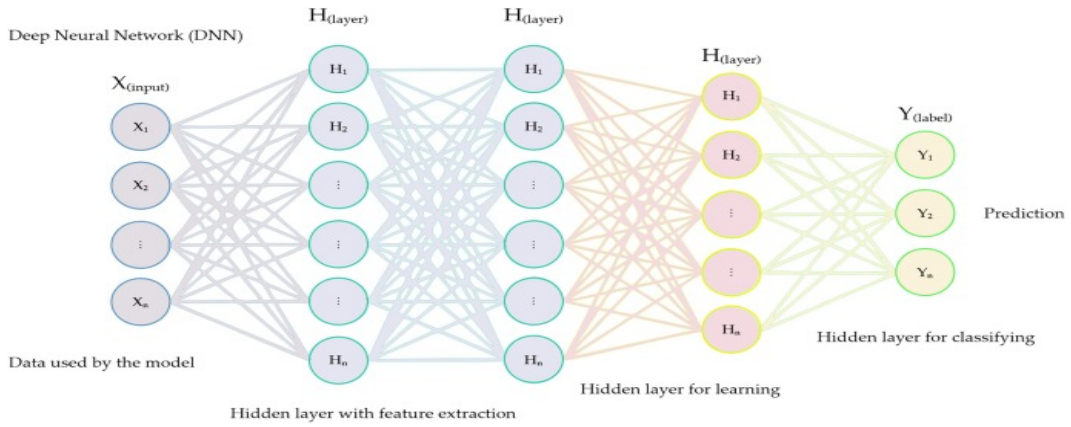


Figure 3.1: A three layer neural network, from [21]. The hidden layers then automatically extract the features from the input X samples, and yields a label prediction at the output Y. The network is trained via backpropagation which aims to minimize a loss function between the prediction and ground-truth. The learning process consists of finding the adequate weights and biases of the neural network.

Unfortunately, albeit the great advantages brought by DL, the real-time processing race still raises problems. The more layers the neural network contains, the slower the computational time will be. While the classification performances improve and the network is able to learn more abstract and complex features, if the algorithm is too slow then it bottlenecks real-time applications. To solve this issue, the development direction splits in two directions: create faster algorithms, and leverage dedicated GPU processors [47].

3.2. Image classification, object localization and object detection

There are 3 main methods from computer vision investigated in this project, which work in an inherently similar manner but yield completely different final outputs, as seen in Fig. 3.2. These techniques are discussed in the following sections, and they are:

1. Image classification
2. Image classification and localization
3. Object detection[1]

The image classification problem can be seen as a binary classification method. A single-class input image is fed into the classifier, and the output will be uniquely the predicted class of the image, for example cat or dog, as seen on the left in Fig. 3.2. There is no extra information available, besides the single label attributed to the unknown picture.

In the classification and localization scenario, after an object was detected in the image, the classifier aims to label and also spatially localize it, as seen in the middle of Fig. 3.2. For example, in the first stage the classifier detects and object and adds a bounding box, then classifies the box. This technique works with both single and multi-class images.

In the last case, the object detection approach is the most complex and works in a similar fashion as image segmentation. Firstly, rather than sliding a window across the picture as above, the entire input picture is broken into a mesh of cells. Thus, each cell detects whether an object is present or not, creates the bounding boxes and labels them accordingly, as seen in the right case of Fig. 3.2.

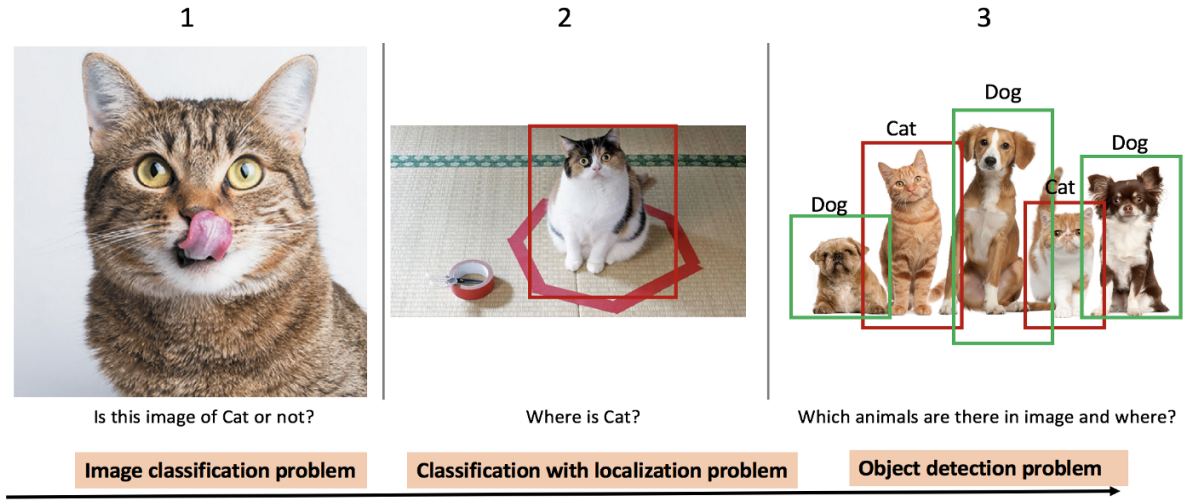


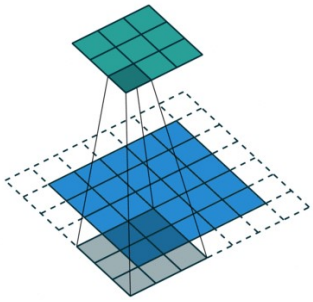
Figure 3.2: Multiple ways of performing image classification, either on single or multi-class images [48]. The first part decides whether an object is present or not in the picture, while in the middle and right cases the objects are also spatially localized.

[1] In computer vision, *detection* means finding all the objects in the picture and label them accordingly.

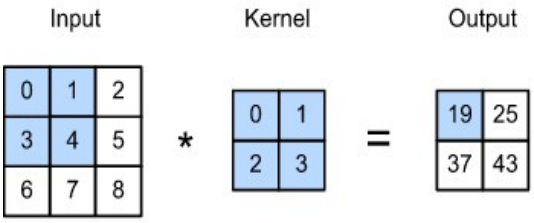
3.3. Convolutional Neural Networks for image classification

Radars' output may be very easily obtained as images that are used to visualize the micro-Doppler signature of targets, as seen in the previous sections. Thus, rather than using artificial neural networks that require flattening the input images into vectors, the DL and remote sensing community moved towards a way to analyze 2D structures without loosing all the prior spatial information. This approach allows to leverage prior knowledge of how neighboring pixels are correlated with each other and build more efficient models. Ergo, the rise of CNNs [49] that were created since as early as 1997 by Yann LeCunn and Y. Bengio. This special type of architecture is able to directly analyze images and represented the foundation of CV, even though they are also applied in speech and time-series applications. However, CNNs are only able to classify single-label images in a binary manner, i.e. is there a drone in this image or not, as seen on the left in Fig. 3.2. They do not work in the case there are input images contain multiple classes, this being one of their main limitation. Nonetheless, they represent the main tool of almost all advanced classification pipelines that are discussed in the following sections.

In DL, the convolution[2] operation means sliding a filter across the input image in order to extract high-level features by calculating the cross-correlation between the pixels from the input and the kernel, as seen in Fig. 3.3. After the image is passed through the convolutional layer, it becomes an abstract representation of the learned features. The activation function is usually added to enforce non-linearity since most real-life models are not linear.



(a) General process of the convolution in DL, where a filter or kernel slides across the entire image to learn its features.



(b) Example the cross-correlation result between an input and the kernel with 0 padding. The highlighted output number comes from the computation: $0 \times 0 + 1 \times 1 + 3 \times 2 + 4 \times 3 = 19$.

Figure 3.3: The convolution operation in DL, where the feature map of an image is obtained by sliding kernel and computing the cross-correlation between the pixel values, followed by feature map reduction via pooling, [50].

Traditionally after the convolution operation, a max or average pooling layer follows to further reduce the size of the feature map. This serves as a mean to reduce the spatial resolution of the hidden representation of the input images, as seen in Fig. 3.4.

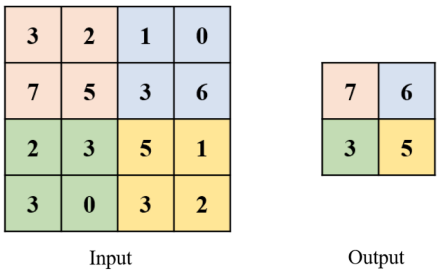


Figure 3.4: Max pooling operation which aims to reduce the dimension of the feature map. In this case, only the maximum value is selected.

These operations are repeated for each layer in the network, until the architecture is completed. In the end, in typical CNNs, the last feature space layer is flattened and then fully connected to the last layer

[2]Not to be confused with the convolution operation from signal processing.

in the neural network, followed by an activation, which servers to classify the input images. A complete architecture from the input image up to the predicted classes is seen in Fig. 3.5.

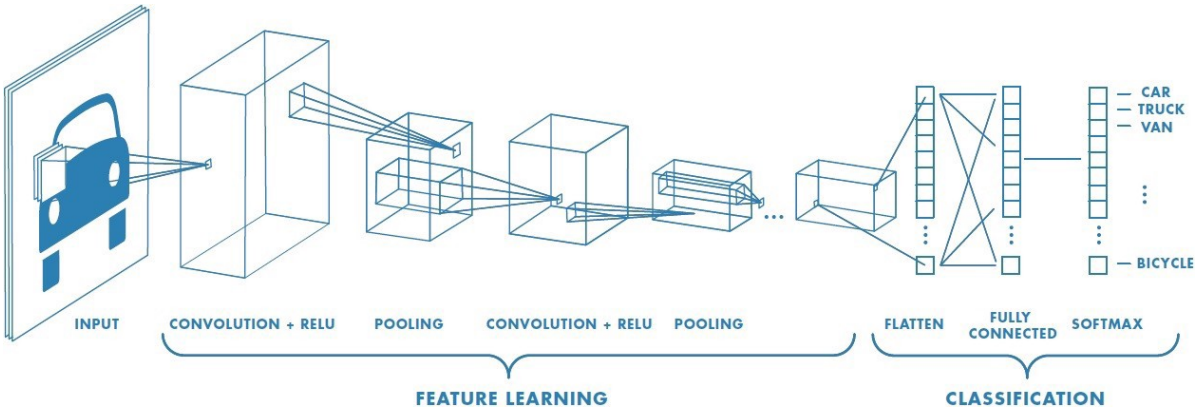


Figure 3.5: Typical CNN where an image is fed as an input. The convolution and activation layers scan the image and extract the high-level features, which are then reduced by the pooling operation. In the end, the last obtained features are flattened and fully connected to a final network layer to yield the classifier output prediction, from [51].

Furthermore, thanks to recent computing power advancements, very deep CNNs became reality. This is a big step forward because DL models are subject to hierarchy feature learning, i.e. low-level simple features are learned in the first layers of the network while high-level abstract & complex features are learned in the deep layers [21]. For this reason, having a very deep network may perform better if the dataset is complex, but may not always yield the best results. For example, GoogLeNet [41] uses only 4 million parameters and may outperform AlexNet [52] created by A. Krizhevsky, which uses approximately 60 millions parameters, depending on the dataset.

Moreover, a well known issue in properly training DNNs via backpropagation is the vanishing or exploding gradient. When data is fed into the neural network, the information is propagated forward towards the output through each layer, and predicts a class probability of each sample. Then, the algorithm propagates backwards from the output layer towards the input and calculates the gradients of the cost function. Thus, the parameters, namely the weights and biases, are computed and the neural network is trained. The gradient descent algorithm works by searching for the minima in order to minimize the error between the predicted class and the ground-truth, as seen in Fig. 3.6. In brief, this is mathematically achieved by taking the partial derivatives in each direction and for each layer in network. Then, each derivative is multiplied with the next one. Thus, if one derivative is equal to zero then it will lead to the *vanishing gradient* problem. Contrary to this, if one gradient exponentially increases, all the values after that will be significantly higher leading to *exploding gradient*. More advanced algorithms exist such as Stochastic Gradient Descent, AdaGrad, Adam, RMSprop to name a few.

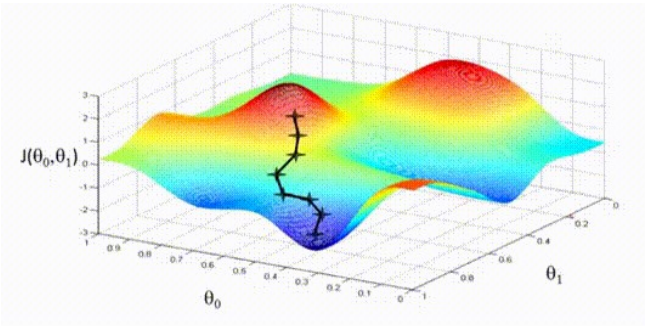


Figure 3.6: Gradient descent algorithm searching for the global minimum point of a function, from gyfcat.com.

Given backpropagation is the core of training neural networks, solutions to bypass these issues were proposed. Reducing the number of layers may be a solution, but it inhibits the ability of CNNs to learn more complex features. Residual Neural Networks (ResNet) bypass the vanishing/exploding gradient

as well as saturation issues by adding skip connections, also known as shortcuts, and jump over some layers [53]. This allows ResNets to build very deep networks, but also speeds up the learning process. Thus, they are today one of the most widely used architectures and the backbone of many CV pipelines. However, there are multiple CNNs architectures that are broadly used for different applications. A table summarizing the most common CNNs is given in Table 3.1.

Model	No. of layers	No. of parameters [Mil.]	Network size [MB]
AlexNet	8	61	227
VGG-16	16	138	515
SqueezeNet	18	1.24	4.6
GoogLeNet	22	7	27
Inception-V3	48	23.9	89
ResNet18	18	25.6	96
ResNet50	50	44.6	167

Table 3.1: Table summarizing the most encountered CNN architectures, with a focus on drone classification using computer vision, from [54].

Even though AlexNet [52] was initially considered a DNN, thanks to all the advancements in computing power and optimization algorithms, much deeper networks emerged, as is the case of VGG-16 [55] which uses the same architecture, but has a deeper design to improve the accuracy. However, the number of layers alone is not a complete measurement of the network complexity. While AlexNet has the lower number of layers, it has one of the largest number of parameters, which yield a substantial network size. This phenomena is even more pronounced in VGG-16. Thus, while it may only contain 5 convolution layers and 3 fully-connected layers, the training may take significantly longer than other architectures with more deep layers but less parameters to train.

GoogLeNet [41] works in a mirrored manner. It uses a relatively high number of layers but has a remarkably lower number of parameters, which results in only a tenth of the network size compared to AlexNet. The architecture is based on the inception module [56], initially developed for Inception-V3. This module aims to improve the accuracy of DNNs by augmenting the number of layers while keeping the number of parameters low. To this end, the network is able to learn more complex features, but the processing time increases proportionally.

3.4. Region-based CNNs for object localization

In many practical cases, classification of a picture alone is not sufficient. Knowing the spatial location of the object is a very important and useful information that can be obtained. Thus, the classification problem can also be extended to also discovering the position of the objects by adding bounding boxes, as seen in the middle of Fig. 3.2.

In order to achieve this, the previously described CNNs were further improved by R. Girshick et al. in 2014 and gave birth to the widely used Region-Based Convolutional Neural Network [57]. This technique follows a region based framework that firstly generates a large number of semi-randomized starting points and then uses a selective search that breaks the main image into many sub-regions based on bottom-up grouping. These regions are examined with neighboring boxes and merged together based on their similarities. Each region is then passed through a SVM algorithm to decide the score whether a specific region is an object or not and the number of regions is gradually reduced. In the end, the remaining regions are cropped and fed to the traditional CNN, which outputs the predicted class of that specific section of the image. Thus, R-CNNs uses the aforementioned CNNs architecture, but it adopts a multi-stage sliding window approach, as seen in Fig. 3.7.

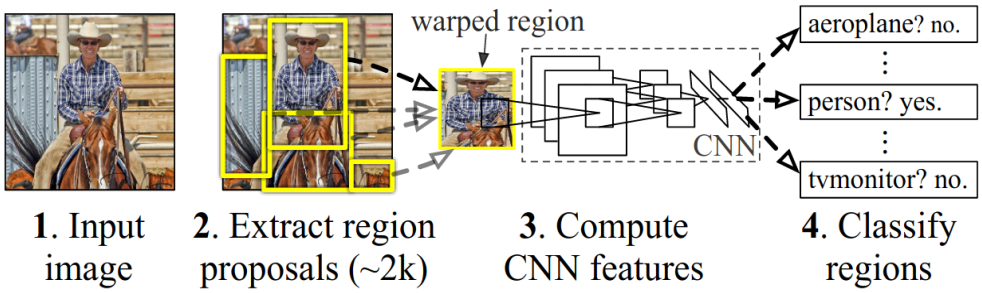


Figure 3.7: R-CNNs shown in a more intuitive fashion, from [57]. Firstly, the bottom-up region boxes are selected. Secondly, the CNNs extract features directly from the bounding boxes. In the end, the boxes are fed into a classifier that yields the predicted label for the box.

However, this method is very slow due to its multi-stage structure. On one side, it needs to create almost 2000 boxes and then gradually correct their sizes. Moreover, it then feeds each region box as input to the CNNs [58], which makes both training and predictions very slow. Thus, R. Girshick further improved his previous work and created Fast R-CNN in 2015 [59] which operates as a singular model rather than a pipeline. This architecture was shown to increase both training and testing speeds, as well as detection accuracy. The main difference between R-CNN and Fast R-CNN is that rather than creating multiple regions and then feed them through a SVM classifier, in Fast R-CNN the entire image is directly fed into the convolutional network that yields a feature map. Then, a Region of Interest (RoI) is predicted for each object and a new feature vector is built based on selected region. Each such vector is then interpreted by a fully connected layer that yields two outputs. One predicts the objects classes via the softmax layer, and the another is a linear output that yields the bounding box. This process is repeated for each RoI. The complete hierarchy of Fast R-CNN is visualized in Fig. 3.8.

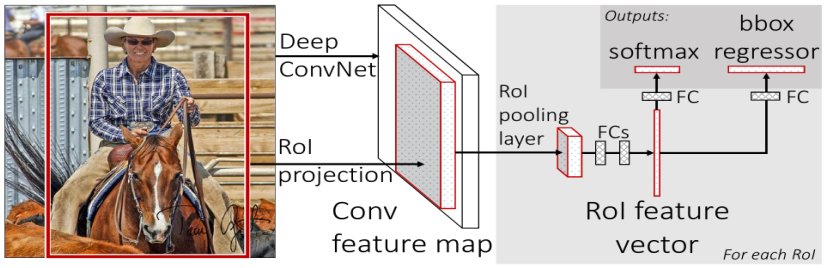


Figure 3.8: Fast R-CNN architecture from [59]. Compared to the previous architecture, the entire image is now scanned directly and the RoI is generated based on the main extracted features, that is then fed into other fully connected layers to yield the classification output.

The last improvement to this architecture was done by Shaoqing Ren et al. in 2016 [60], named Faster R-CNN which further improved the speed of both the training and detection output and aimed at real-time object detection. This architecture uses two modules. The first is a traditional deep convolutional network [54] that yields the feature map and creates the regions, and the latter is the aforementioned Fast R-CNN detector [59]. The most common bottleneck in computational speeds is related to detecting possible object and obtaining the proposed regions. Compared to the above cases that use selective edgebox searching, Faster R-CNN uses a region proposal network that is a very cost effective algorithm [58]. The region proposal network is obtained by sliding a small network across the obtained feature map. The region proposal are the bounding boxes that aim to obtain a binary prediction, namely if there is an object or not inside the box. Once these bounding boxes are created, the second module knows where to look at, rather than scan the entire image, hence the computational improvements. The complete architecture of Faster R-CNN can be seen in Fig. 3.9.

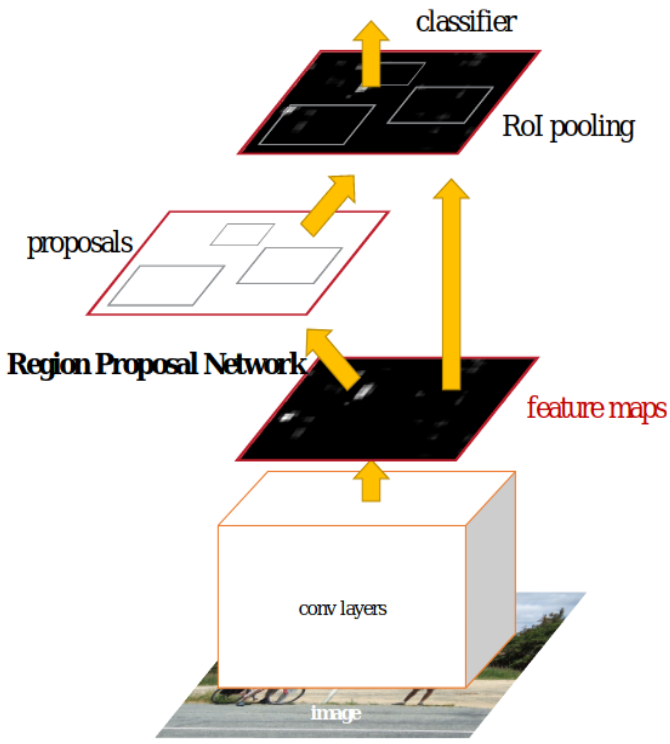


Figure 3.9: Summary of the Faster R-CNN architecture, from [60]. After the entire image is fed into the deep CNN, the obtained feature map is sent in two branches. The first servers to obtain the region proposal network which adds the bounding boxes over the present objects. The second is the Fast R-CNN detector that is only applied to the obtained bounding boxes.

Thus, initially, the R-CNN [57] architecture was done by creating a lot of bounding boxes and gradually merging them together, after which a CNN was applied to each region obtained, which took a huge amount of time to train and yield the classification output.

Later, Fast R-CNN [59] proposed to improve the computational speed by feeding the entire picture inside the CNN and then create RoIs based on the obtained feature map which would then be fed to a classifier. This approach may be very useful if the number of RoIs is low, but it drastically loses performances when there are many proposed regions.

In the end, in order to bypass the last constraint based on the creation of bounding boxes, Faster R-CNN [60] proposed a low-cost algorithm that creates a region proposed network based on a pre-defined number of anchors. This solution saved a significant amount of computational load by making only applying CNNs to the easily selected bounding boxes.

However, while these methods are one of the most widely used today, their focus is anchored on heavy multi-label pictures. In the case of counter-drone surveillance radars, typically there are no more than a only a couple of targets present in a sequence of sweeps.

3.5. YOLO for object detection

Another way to perform localization is by seeing the entire image simultaneously rather than using a sliding window, thus looking at it only once, in a global one-step framework. This technique is known as You Only Look Once (YOLO), developed by J. Redmon et al. in 2016 [61] and aims to yield real-time object detection. One of the co-author and contributor of this paper is R. Girshick, which is also the creator of all the aforementioned R-CNN techniques. Moreover, the authors showed that YOLO yields less than half of the total number of errors compared to Fast R-CNN [59, 61] and outperforms the latter in terms of computational speeds. However, while this technique is faster and less prone to

mistake the background for objects, in terms of localization accuracy is still behind with respect to the state-of-the-art.

This architecture works by taking the entire input image and splitting it into a mesh of grid cells. Each grid cell individually predicts whether an object is centered inside it and creates a number of bounding boxes and a confidence score for each box. The confidence score is a measurement of how sure the network is that the box contains an object, with 0 representing the case where there is no object inside the cell, and is given by the equation:

$$Confidence = Pr(Object) \times IOU, \tag{3.1}$$

where $Pr(Object)$ is the probability there is an object inside the cell, and IOU is the Intersection Over Union (IOU) score. The IOU is the overlap area between the predicted bounding box and the ground-truth inside the training data. If two boxes contain a very similar object, then the IOU score will be high and the bounding boxes will be merged together to remove duplicates. If the score is low, then the algorithm considers there are two distinct objects and leaves them separated. Furthermore, each cell also predicts the conditional probability of the class of the object, $Pr(Class_i|Object)$. Thus, the final class-specific confidence score is obtained for each bounding box and the final detections are obtained via the formula:

$$Pr(Class_i|Object) \times Pr(Object) \times IOU = Pr(Class_i) \times IOU \tag{3.2}$$

The entire process can be seen in Fig. 3.10, which takes an initial image and yields the final detected objects and their corresponding bounding boxes.

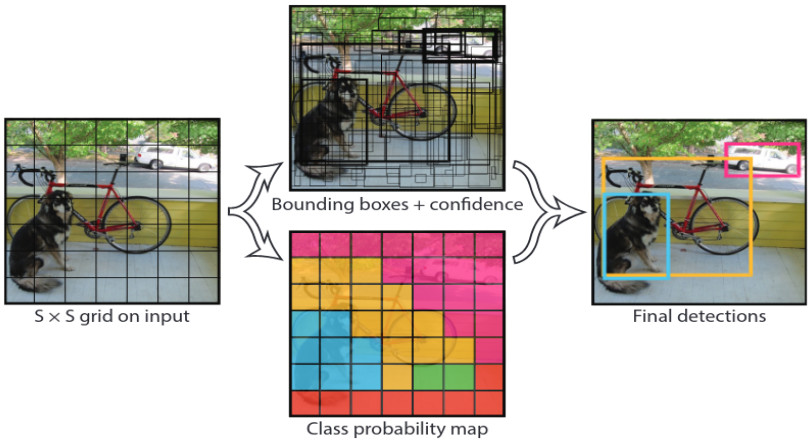


Figure 3.10: The complete process of YOLO, from unlabeled image input to final decision output, from [61]. The first stage of the architecture adds a grid of cells across the entire image, with each cell being responsible of detecting whether an object is located inside or not and creates an object class probability. Additionally, it predicts a number of bounding boxes with a confidence score, as seen in the middle top figure. Moreover, the class probability of each object is estimated based on the similarities of each cell, as seen in the middle bottom figure. In the end, based on the predicted bounding boxes, the confidence score and the class probability, the final object detections are obtained.

YOLO, like all the aforementioned networks, is based on CNNs. However, the first proposed architecture in 2016 and seen in Fig. 3.11 is a relatively deep CNN with almost 25 layers. Nonetheless, the main issue is the low accuracy and recall scores and not the computational speeds. To this end, YOLOv2 [62] was developed in late 2016 and focused uniquely on improving the localization performances and recall score, all without affecting the classification accuracy. Compared to YOLOv1, YOLOv2 leverages anchor boxes rather than spatial coordinates to predict bounding boxes, which greatly simplifies the computational load and facilitates the ability of the network to learn. Moreover, the spatial constrains of the bounding boxes in YOLOv1 assumes they one box may have only one class, which in real world applications may not be true due to objects overlapping.

While at the moment of release YOLOv2 was one of the fastest and most accurate computer vision pipeline, more improvements have been made in the past years. In 2018 J. Redmon et al. [63], the

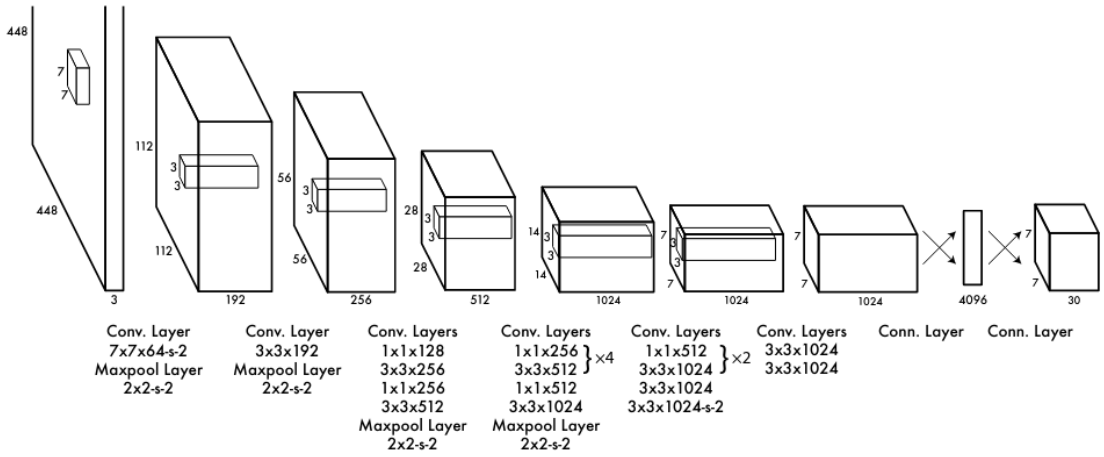


Figure 3.11: YOLO architeture from [61]. It is composed of 24 ConvNets, each followed by max pooling to reduce the feature maps. In the end, 2 fully connected layers are are present to yield the final classification outputs.

same authors of YOLOv2, created YOLOv3 which aimed to improve the accuracy of detecting small objects. Given YOLOv2 uses approximately 30 layers in total, it often struggled with properly detecting finer details. To achieve an increment in terms of accuracy, a more complex architecture of over 100 layers was used, but this leads to an increased complexity and thus slower computational speed.

After the development of YOLOv3, the original creator of YOLO, J. Redmon, quit pursuing the improvement of his technique due to ethical issues. He argued that the computer vision techniques he developed raised many issues in military applications and data protection. However, other scientists continued building on this highly popular method. A. Bochkovskiy et al. [64] developed in 2020 YOLOv4, which aimed at improving the computational speeds. Their results showed that the newly proposed model yielded a boost of 12% in frames per second, and also a 10% increase in accuracy.

In the end, soon after YOLOv4 was released, G. Jocher introduced YOLOv5 based on a PyTorch framework rather than Tensorflow [65]. Different blogs show that YOLOv5 may perform even faster than YOLOv4, but there is no scientific article documenting the architecture or the improvements, only an open-source code at GitHub and different contradicting information on different blogs. This is due to the main creator who ceased to further pursue improvements in computer vision, and enthusiasts taking over ever since.

It is worth emphasising the difference between traditional computer vision applications and remote sensing. CV is generally used with high-resolution cameras that capture a plethora of objects and details, such as the scenery of a street that contains cafes, chairs, buildings, pedestrians, cyclists, cars, bins, benches, trees, etc. To this end, the CV community focused on improving the accuracy to accurately detect and recognize smaller and smaller objects that are easily seen by cameras. In the field of radar engineering and especially in counter-drone situations, there are far fewer objects that induce distinguishable micro-Doppler modulations. Moreover, drones tend to have a very wide frequency spectrum due to their very fast rotating propellers, even though they are seen as point targets. Thus, a trade-off in the favor of small object detection while sacrificing real-time computational speeds is not the right direction to head towards in this field. Integrating computer vision algorithms into radar engineering requires novel approaches and fine-tuning of the existing techniques, with a focus on this obtaining the right trade-off.

3.6. Deconvolutional Neural Networks

While CNNs serve as means to scan input images and intrinsically learn the features in an unsupervised manner, it is meaningful to understand why do they reach such incredible performances. Thus, the aim

of Deconvolutional Neural Networks (DeconvNets) is to break down these black-boxes and see what information are the networks actually extracting, and how transferable are they. This ideology is also known as *explainable AI*.

M.D. Zeiler et al [66, 67] have created DeconvNet, a deconvolutional CNN that acts in a reversed manner to convolutional neural networks, i.e. using deconvolution and unpooling layers. By using such a reversed architecture as seen in Fig. 3.12, we are able to achieve feature visualization and see what each layer in a dense network learns. Moreover, by using these insights it is possible to pin down the downsides and limitations of current architectures and further improve them. H. Noh [68] used this approach in semantic segmentation and showed that by combining both convolutional and deconvolutional networks yield outstanding performances on various datasets.

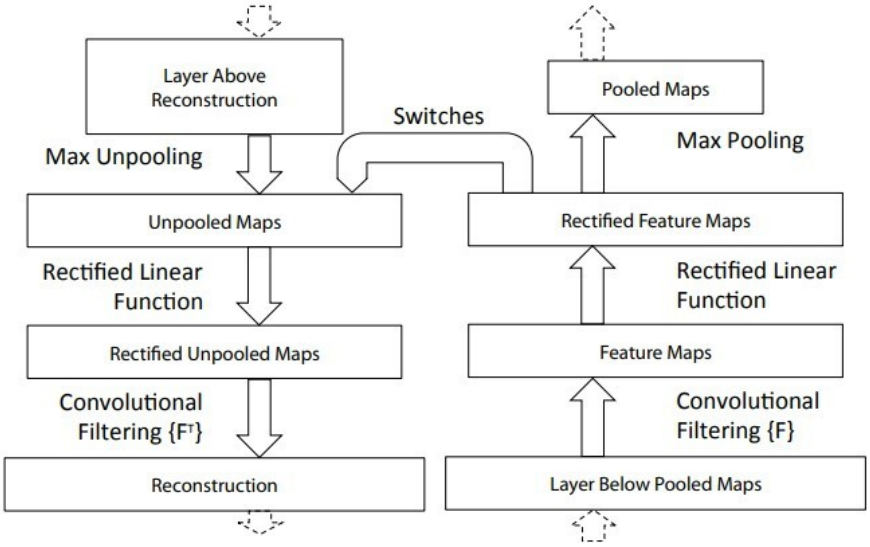


Figure 3.12: DeconvNet architecture on the left, juxtaposed by the traditional ConvNet on the right, from [66]. The deconvolutional neural network reconstructs the features learned from the CNN.

It is worth emphasizing the importance of this architecture. Traditionally, many techniques in DL were seen as *black boxes* and thoroughly understanding how they function remained a mystery. While even today there is still much to be unravelled, DeconvNets allowed us to visualize and better understand what happens inside CNNs. For instance, M.D. Zeiler et al. [66] proved that the more layers a deep neural network contains, the more abstract and complex features it is able to learn. In Fig. 3.13 a visual representation of the hierarchy feature learning inside a neural network is seen. This methodology confirmed the beliefs that the first layers extract the most prominent low-level features such as the geometry and contour of the object, while deeper layers are able to learn abstract high-level features, such as smaller shapes and finer details.

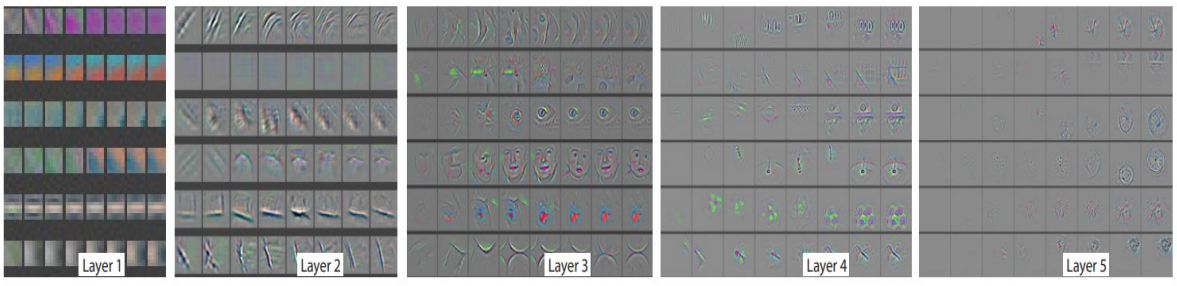


Figure 3.13: Feature learning evolution for multiple layers in a CNN, visualized by the DeconvNet architecture, from [66]. The deeper the number of layers, the more complex and abstract the learned features become.

This technique also proved that a very dense network may perform better with very intricate input

images, as it can properly learn the complex patterns. However, a deeper network is also more computationally complex and slow. Thus, if the input patterns are relatively simple, then a very deep neural network won't yield any improvements and will bottleneck the real-time computations.

3.7. Autoencoders

Autoencoders (AEs) are a type of self-supervised learning method that produce a reconstruction of the input image from solely the most relevant features coming from unlabeled data. This is a great advantage in the remote sensing community, as diversified labeled data is usually not available. Moreover, AEs enables us to extract the compressed representation of data and further use them for transfer learning or dimensionality reduction.

The general architecture of AEs consists of two parts, as seen in Fig. 3.14. The first level, the encoder, compresses the input data into the so-called latent embedding or bottleneck layer and may be built via CNNs. The second part, the decoder, reconstructs the original data from the compressed representations and is done via deconvolutional neural networks. Traditionally, the decoder only serves in the training stage. The predicted/reconstructed output, $\hat{x}$, is compared to the input, x, and the weights are computed by minimizing the loss function between the two [69]. Thus, the AEs can perform the heavy-lifting training stage a-priori and later be transferred as trained networks to bypass the computationally heavy training stage.

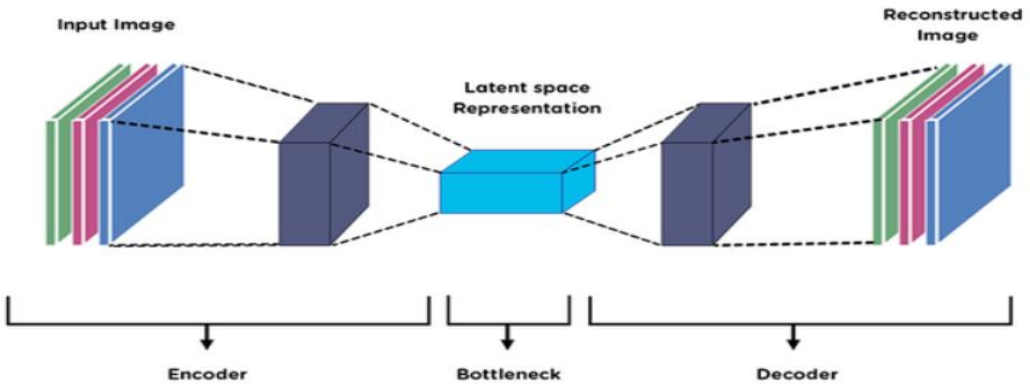


Figure 3.14: Convolutional autoencoder architecture from [70], where the 3 main levels are clearly seen.

When properly constructed, the encoder can efficiently extract features of the input and achieve dimensionality reduction in the latent space by retaining only the main features. For this reason, AEs are cut at the bottleneck layer, after the learning is completed, as seen in Fig. 3.15. This approach is widely popular in denoising radar images, where AEs are trained to recognize noise and then remove it [21]. Furthermore, given the difficulty of obtaining radar data that is kinematically relevant, AEs show great potential to generate and expand available datasets.

Moreover, A. Zeggada et al. [71] successfully used AEs to obtain multiple labels and classify UAVs in urban environments using time-frequency pictures in an unsupervised framework. The output of the encoder is the latent space, seen in Fig. 3.15, which contains the most predominant compressed features, which is then connected as input to a typical MLP classifier. The MLP consists in this case of a single hidden layer, trained via the classical backpropagation approach. The advantage of a single layer neural network is that it does not have many parameters that require fine tunning, hence the improved computational speeds.

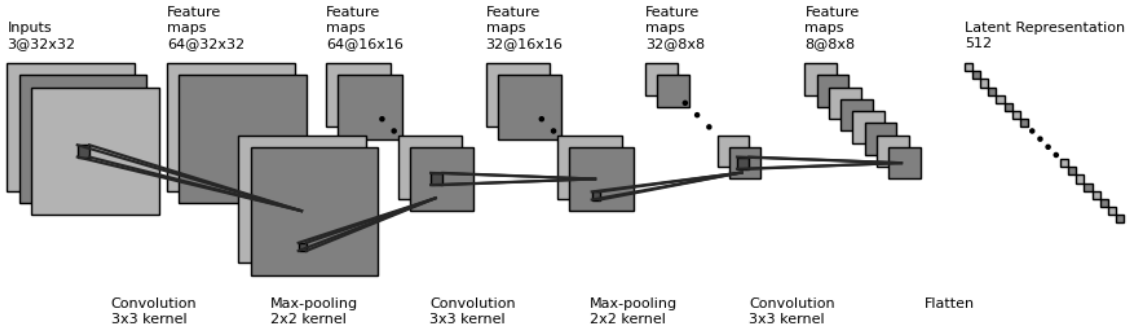


Figure 3.15: Architecture of a convolutional encoder and the latent embedding that already contain all the important features after training. These two levels may be further connected to another classifier, such as a multilayer perceptron classifier or fully connected layer to yield the final classification output.

3.8. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are another popular architecture encountered in DL and remote sensing. Compared to CNNs, a RNN is a based on directed cycle graphs that can leverages past information. The phenomena of short-term memory is achieved via the directivity of the information flow, allowing the network to perform a task and take in account all the prior information up to a specific point. This very nature help bypass the problem of vanishing gradients [21]. Traditionally, RNNs have been widely used to analyze non-stationary processes such as audio and speech processing, but drone sound detection & classification via RNNs was also investigated [19].

However, storing data for extended period of time takes significant amounts of time and memory to train the network via backpropagation. For this reason, Hochreiter and Schmidhuber [72] made a breakthrough for RNNs with their novel and improved gradient method named long short-term memory. By eliminating the gradient where it does not bring any improvements, the new optimization method drastically improved the neural network training time.

In brief, CNNs are widely used to leverage prior spatial information such as images and correlations amongst pixels. On the other hand, RNNs are used to leverage state variables to store past information and thus efficiently process sequential information such as audio signals, text strings or videos.

3.9. Summary

While there is a plethora of different DL & CV methods achieving different objectives, the main scope of this project is to classify and localize the drone on radar images. To this end, the CV pipeline objective can be framed as an *object detection* problem. Thus, the chosen classification pipeline will predict the bounding box around the target and a classification label. Moreover, the classification should be done in real-time as the processing chain is done continuously as the antenna is rotating. The chosen pipeline that is investigated in this project is YOLO, thanks to its optimal trade-off between inference speed and accuracy [64], alongside the open-source nature of the project. The performances of the proposed method used on radar images are presented in the following chapters.

4

Experimental Setup, Processing chain and Data analysis

In this chapter a preliminary analysis of the radar system, the experimental setup as well as the signal processing chain are covered, followed by an in-depth discussion of the generated data. Furthermore, the challenges of plot-based UAVs classification are examined and solutions to improve the plots resolution are proposed.

4.1. IRIS® 4D Counter-Drone Radar specifications

IRIS® C-UAS radar, seen in Fig. 4.1, was developed by Robin Radars B.V. for the detection, classification and tracking of rogue drones. IRIS® is an FMCW surveillance radar that transmits a narrow beam in azimuth of $6°$, and wide beam in elevation of $60°$ containing 6 stacked beams each covering $10°$. It is electronically steered in elevation via a phased array and steered in azimuth via a mechanically rotating antenna with a constant rotational speed of 30 RPM.

Figure 4.1: Robin IRIS® 4D radar.

The radar weights only 25 kilograms and is incredibly easy to install and deploy. It performs classification based on the range-Doppler plots and is able to differentiate between drones and other aerial point targets such as birds based on the width of the micro-Doppler signatures. The IRIS® radar spec-

ifications [73] are found in Table 4.1.

Technology	FMCW
Frequency bandwidth	X-band 50 MHz
Power output	2x 12W
Rotation/ Scan speed	30 rpm/ 1.0s
PRF	4 kHZ
CPI	100 sweeps
Nr. of RX channels	8
Azimuth beam width	$6°$
Elevation beam width	$60°$ for 6 stacked beams
Instrumented range	5 km
Azimuth coverage	360°
Elevation coverage	60°
Dimensions	554mm diameter x 623mm height

Table 4.1: IRIS® radar specifications from Robin Radar Systems.

4.1.1. Narrowband signal model

IRIS® uses a FMCW radar signal that uses a saw-tooth signal modulation. The signal modulation allows to estimate both the time delay and Doppler shift between the transmit and received signals, which in turn yield the estimated range and velocities of detected targets. An example of a FMCW saw-tooth signal model can be seen in Fig. 4.2.

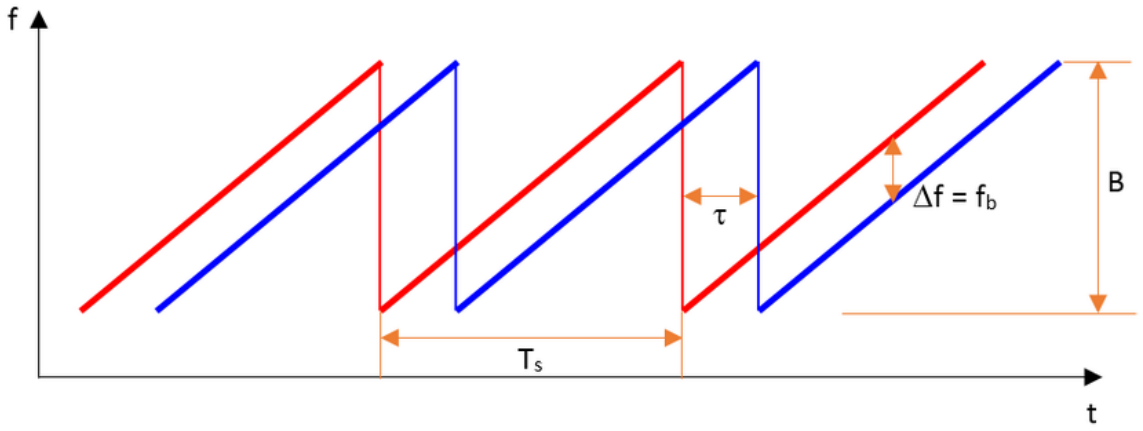


Figure 4.2: FMCW saw-tooth signal model example that highlights how the time delay, τ, and the Doppler frequency shift, Δf, can be estimated via the differences between the transmit and received signal, from [74].

Moreover, the array response is modeled as a narrowband signal. This assumption is of uttermost importance in the data processing chain and the modelling of the signals, beamforming and spatial filtering. Under the narrowband assumption, the real[1] signals received by the array with carrier frequency f_c are modelled [75] as:

$$z(t) = \textbf{Re}\{s(t)e^{j2\pi f_c t}\} = x(t)cos(2\pi f_c t) - y(t)sin(2\pi f_c t) \tag{4.1}$$

The signal $s(t)$ is refereed to as the complex envelope of real signal $z(t)$, or the baseband signal. The real and imaginary part of the baseband signal, $x(t)$ and $y(t)$ are called the in-phase and quadrature components of the real signal. In practice, the imaginary part of the signal is artificially created via a Hilbert transformation [76] for algebraic convenience.

[1] Antennas transmit and receive real signals without an imaginary component.

Furthermore, the real received signal, $z_\tau(t)$, is a delayed version of the transmitted signal, induced by the propagation time. Thus, the delayed real received signal is modeled as:

$$z_\tau(t) := \textbf{Re}\{s(t-\tau)e^{j2\pi f_c(t-\tau)}\} = \textbf{Re}\{s(t-\tau)e^{-j2\pi f_c\tau}e^{j2\pi f_c t}\} \tag{4.2}$$

The baseband signal of the delayed received signal is then modelled as:

$$s_\tau(t) = s(t-\tau)e^{-j2\pi f_c\tau} \tag{4.3}$$

Furthermore, given the narrowband assumption, the exponential component induced by the delay can be approximated as one, i.e. $e^{-j2\pi f\tau} \approx 1$. Thus, if $|2\pi f_c\tau| \ll 2\pi$ for all frequencies $|f| \leq B/2$, and $S(f)$ is the Fourier transform of the baseband signal, then we can approximate:

$$s(t-\tau) = \int_{-B/2}^{B/2} S(f)e^{2\pi ft}e^{2\pi f\tau}df \approx \int_{-B/2}^{B/2} S(f)e^{2\pi ft}df \approx s(t) \tag{4.4}$$

Hence, this assumption allows the complex envelope of the delayed received signal to be approximated as a simple phase shift, modelled as:

$$s_\tau(t) \approx s(t)e^{-j2\pi f_c\tau}, \qquad \text{for } B\cdot\tau \ll 2\pi \tag{4.5}$$

Thus, from an algebraic perspective, $B\cdot\tau \ll 2\pi$ is refereed to as the *narrowband condition*. In antenna theory, this condition is satisfied by creating the adequate design of the array [37]. More specifically, the bandwidth and the antenna aperture must be chosen such that:

$$B\cdot\tau \ll 2\pi \quad \Leftrightarrow \quad \frac{c}{B} \gg N\cdot d \tag{4.6}$$

where B is the bandwidth, τ is the time delay, c is the wave propagation speed (speed of light), N is the number of array elements and d is the inter-spacing between the said elements.

Alas, given IRIS® uses 8 receiver channels, a bandwidth of 50 MHz and the inter-spacing of the said channels is 20 millimeters, we can then conclude that IRIS® satisfies the narrowband condition, i.e.:

$$\frac{3\cdot 10^8}{50\cdot 10^6} \gg 8\cdot 20\cdot 10^{-3} \quad \Leftrightarrow \quad 6 \gg 0.16 \tag{4.7}$$

4.1.2. Spatial filtering and beamforming

Let us consider the receiver antenna of IRIS® containing 8 array elements placed along a line in elevation at 20 millimeters spacing, which is almost 4 millimeters higher[2] than $\frac{\lambda}{2}$, thus constructing an Uniform Linear Array (ULA). Let the delayed received signal, $x_\tau(t)$, defined in equation 4.5, arrive at the receiver channels at the angle θ with respect to the antenna boresight. In this case, θ is called the Direction of Arrival (DoA) of the plane wave, which may vary from $-90°$ up to $+90°$. Given the narrowband condition is satisfied, the signals collected by each individual array element are modelled as:

$$x_n(\theta) = e^{j2\pi(n-1)\frac{d}{\lambda}sin\theta}, \qquad n = 1,\dots,N \tag{4.8}$$

It is worth emphasizing that thanks to the narrowband condition, the phase progression across the N array elements is linear. The signals collected by all the array elements from any potential incoming DoA can be fit into the data vector $\mathbf{x}$, modelled as:

$$\mathbf{x}(\theta) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} 1 \\ e^{j2\pi\frac{d}{\lambda}sin\theta} \\ \vdots \\ e^{j2\pi(N-1)\frac{d}{\lambda}sin\theta} \end{bmatrix} \tag{4.9}$$

[2]The inter-spacing between the receiver elements is higher than $\lambda/2$ by 4 extra millimeters to optimize the trade-off between antenna aperture, beam width and gain, but also to operate at a wider frequency bandwidths with the same hardware.

Nonetheless, in the above equation there is no spatial directivity, making the DoA estimation impossible. Thus, spatial steering via beamforming on the receiver channels is used on one hand to achieve spatial filtering, i.e. separate the ground clutter from aerial targets, but also to estimate the elevation of a target. The steering vectors for IRIS® are constructed in elevation following the directivity of the ULA by using a similar framework as shown in equation 4.9, but rather than using a random DoA, the beam is steered towards a specific angle chosen a-priori, θ_0. Let the steering vector be modelled as:

$$\mathbf{s}(\theta_0) = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{bmatrix} = \begin{bmatrix} 1 \\ e^{j2\pi \frac{d}{\lambda} sin\theta_0} \\ \vdots \\ e^{j2\pi(N-1) \frac{d}{\lambda} sin\theta_0} \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \mathbf{w} \tag{4.10}$$

The total beamformer output, $\mathbf{y}$, sums the incoming signals from all the array elements from any possible angle, $\mathbf{x}(\theta)$, and achieves spatial filtering by steering the beam towards specific angle, θ_0. The final beamforming output for a single steering angle θ_0 and for N array elements is defined as:

$$\mathbf{y} = \mathbf{w}' \cdot \mathbf{x} = \sum_{n=1}^{N} x_n \cdot e^{j2\pi(n-1) \frac{d}{\lambda} sin\theta_0} \tag{4.11}$$

IRIS® uses six elevation receiver beamformers each covering $10°$ in elevation, centered around $[-30°, -20°, -10°, 10°, 20°, 30°]$ with respect to the antenna aperture. Moreover, Taylor tapers are added within the beamformers to decrease the side lobes levels to -25 dB. This in turn shifts the energy from the side lobes within the now wider main lobe. A visualization of spatial steering, as well as the effects of the tapers, juxtaposed by unitary weights without tapering, can be seen in Fig. 4.3.

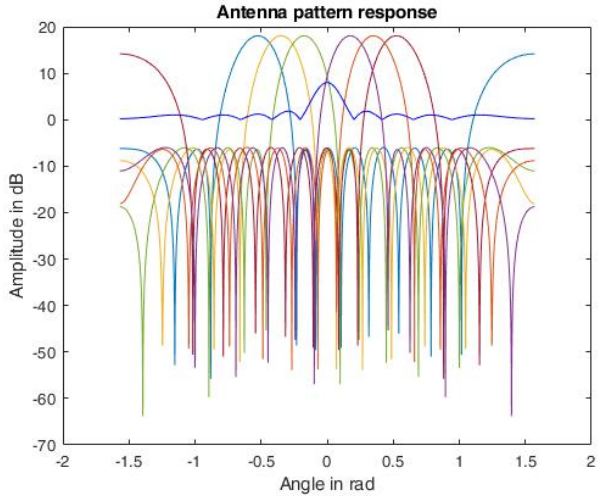


Figure 4.3: Visualization of the amplitude in dB for spatial steering via Taylor tapers weights steered between $-30°$ up to $+30°$ as well as unitary weights (in dark blue, centered around $0°$. The benefits obtained by including tapering within the spatial beamformers is clear, as it focuses all the power from the side lobes withing the main lobe towards a specific angle.

4.1.3. Signal processing chain

With the specification of the radar system covered and the signals modeled, we turn our attention towards the signal processing chain used to yield the input images for the computer vision classifier. A simplified processing pipeline can be visualized in Fig. 4.4, where the raw sweeps are recovered and fed into the chain after the Hilbert transform. The processing chain is highlighted in turquoise and begins after the received signals are passed by the Analogue to Digital converters, up to the computer vision classification pipeline.

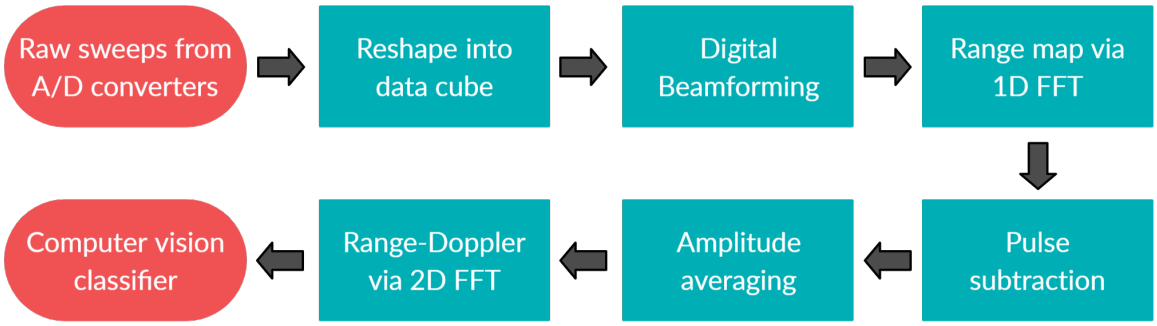
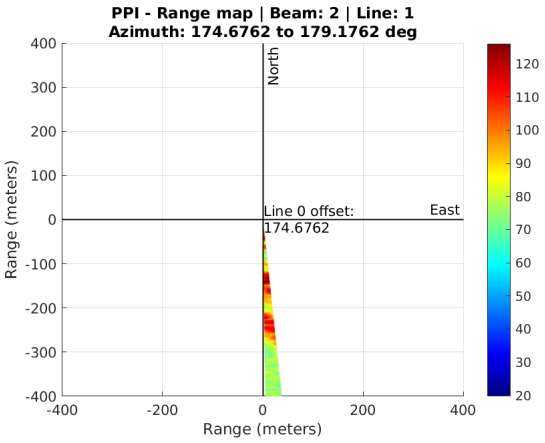
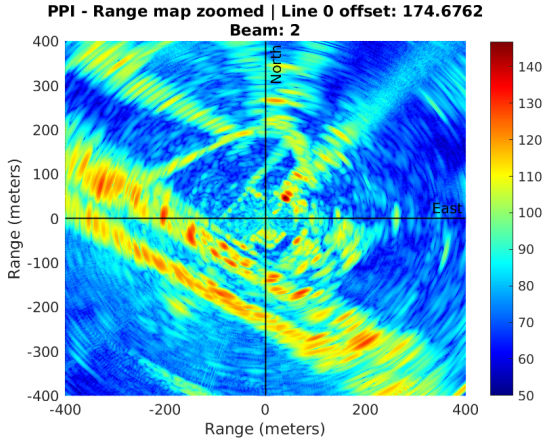


Figure 4.4: Digital signal processing chain from the received raw sweeps from the Analogue/Digital converters, up to the obtained range-Doppler plots used for ATC.

The input data stream is saved as a long vector that for a single sweep represents the number of fast time samples multiplied by the number of receiver channels. Given the antenna is continuously rotating, the processing is done for each CPI of 100 consecutive sweeps which cover $4.5°$ in azimuth. Each CPI is called a *mini radar data cube*, or a *line*. For a complete $360°$ antenna rotation, there are a total of 80 lines. The differences between a line and a full rotation, as well as the dimensions of the radar data cube are highlighted in Table 4.2, and visualized in Fig. 4.5. Thus, for each line the entire processing chain must be done in real-time.



(a) Output of a polar range map covering $4.5°$ in azimuth obtained from a single line consisting of a CPI of 100 sweeps.



(b) Output of a polar range map of all the 80 lines concatenated together to obtain the full rotation.

Figure 4.5: Difference between a full rotation and a single line.

Data	Azimuth coverage	Number of sweeps	Number of lines	Range bins	Receiver channels	Beams	Memory
Single CPI	4.5°	100	1	3125	8	6	5 MB
Full rotation	360°	8000	80	3125	8	6	400 MB

Table 4.2: Summary of the IRIS® radar data format, for a single line and for an entire rotation.

Let us briefly explain each signal processing block, given the first two steps were covered in previous sections an the last three are discussed in-depth in the next:

Reshape into data cube: Changes the data structure of the received data from a long vector into the radar data cube of dimensions *Range bins by RX channels by Doppler bins* for each CPI. Thus, for a full rotation consisting of 80 lines, we will have a $4D$ data cube (or multiple mini radar data cubes/lines). The data format is summarized in Table 4.3;

Range bins	Receiver channels	CPI	Number of lines
3125	8	100	80

Table 4.3: Summary of radar data cube dimensions after reshape.

Digital beamforming: Changes the structure from 8 RX channels to 6 beams by summing and weighting the received signals, as described in Subsection 4.1.2 and highlighted in red in Table 4.4. The total elevation coverage is $60°$ by stacking all beams, thus each beam is covering $10°$ which allows us to find the elevation of the target by selecting the beam with the highest target gain, as seen in Fig. 4.9. The beams also contain Taylor tapers to decrease side lobes levels by 25 dB, as seen in Fig 4.3. While in theory the side lobes can be decreased to infinitely lower levels, in practice due to the antenna design relative to the aperture and the manufacturing constraints of hardware, the maximum attenuation possible is -25 dB. Thus, the gain in the main beam is 27 dBi, while the gain in the first side lobe is 3dBi, hence attenuating the energy received from other DoAs in each beam.

Range bins	Number of beams	CPI	Number of lines
3125	6	100	80

Table 4.4: Summary of radar data cube dimensions after digital beamforming.

Range map via 1D FFT: Obtain the unfiltered plan position indicator. Moreover, given the range map is symmetric, only half of the range bins are kept, as highlighted in red in Table 4.5. Considering the antennas is continuously rotating, each CPI will cover a specific azimuth angle. Thus, the full rotation can be converted into a complete map of the recorded area in a polar range map by concatenating all lines together. This visualization is used to correlate the objects present in plots with the real world environment via Google maps and have a better understanding of where the target of interest is located, as seen in Fig. 4.7 & 4.8.

Range bins	Number of beams	CPI	Number of lines
1562	6	100	80

Table 4.5: Summary of radar data cube dimensions after digital 1D FFT to obtain the range-map. This is also the final data format change.

Pulse subtraction & Amplitude averaging: Clutter filtering methods used to improve plots visibility and remove unnecessary information. The 2 pulse canceller and the amplitude averaging methods individually help improve the plots visibility, but create other artifacts, or are not powerful enough alone. Thus, when used in cascade, they compensate each others' weaknesses and yield the best results. The output images for each processing stage are clearly visualized and the effects of each block are described in detail in Fig. 4.15. Moreover, a Signal to Noise to Clutter Ratio (SNCR) is summarized in Table 4.10 highlight the improvements for each image brought by the proposed clutter filtering techniques.

Range-Doppler via 2D FFT: Obtain the filtered plots that are fed into the classifier. Contains a 2D Blackman Harris window for side lobes reduction following a similar framework as in the beamforming case. However, rather than decreasing the side lobes levels from nearby DoAs in elevation, this window aims to reduce the energy levels from neighboring lines covering nearby azimuth angles. Based on the bulk and micro-Doppler signatures, the distinction of drone vs. other targets is achieved, as clearly seen in Fig. 4.15. Hence, the pre-processing steps up to this point represent the most important part of the entire pipeline.

4.2. Real datasets analysis

4.2.1. Data collection and test scenarios

All the used datasets were collected at Maasdijk, in the province of South Holland in The Netherlands. Besides the main object of interest, i.e. the drone, there is a plethora of different targets that may be static or not, such as vegetation, birds, trains, ships, cars, cyclists, faraway wind turbines, or even helicopters, as seen in Fig. 4.7. Moreover, some datasets were recorded during summer when there was an abundance of birds flying in the proximity of the drone, while others were taken during winter when the ground was frozen which resulted in stronger static clutter. In other datasets, the drone was in the proximity of wind turbines which may mask the drone modulations entirely given their weak RCS compared to turbine blades, or may yield similar micro-Doppler modulations if they are far enough given the tip of the blade rotates very fast and the reflections are relatively weak.

Figure 4.6: An example of the recording location facing the river during winter when the ground was frozen. The presence of close and faraway wind turbines is shown, as well as railways and ships passing.

Thus, when analyzing the range-Doppler plots, there are different targets that may replicate similar modulations to those of an UAV. However, the selection of reliable ground-truth is possible by knowing a-priori at what range, azimuth and elevation the drone is located, as well as the direction of flight and its velocity. Moreover, processed plots obtained directly from Robin Radar Systems pipelines were used to further confirm the veracity of the data generated via the signal processing pipeline described above. Alas, the datasets were recorded using pre-defined scenarios, such as the simplified versions highlighted in Table 4.6 for the static case, and Table 4.7 for the on-the-move scenario.

First data set: Static radar					
Test scenario	Drone distance from radar (m)	Drone velocity (km/h)	Recording time (s)	Time on target (s)	Environmental conditions
No drone	N/A	N/A	8	0	Sunny
Hovering drone	100	0	34	0.425	Sunny with clouds
	500	0	36	0.45	Sunny with clouds
Drone flying in straight line	10 to 300	5	20	0.25	Sunny
	300 to 10	5	20	0.25	Sunny
	100 to 1000	40	50	0.625	Sunny
	1000 to 100	40	50	0.625	Sunny
Drone flying in circles	400	20	80	1	Sunny

Table 4.6: Datasets pre-defined test scenarios to obtain reliable ground-truth. The *Raw sweeps* were used to generate the data used in this project, while the *Processed plots* were obtained from Robin Radar Systems pipelines and used as means to verify the obtained results in similar scenarios.

Before looking at the data from the following chapters, the raised difficulty of the radar on-the-move dataset led to using a bigger mini-drone, DJI Inspire 2, that was used to compare the micro-Doppler sig-

natures captured in range-Doppler plots with the standard micro-drone used in the static radar datasets, Autel Evo II. The radar on-the-move scenario is analyzed more in-depth in Chapter 6.

Second data set: Radar On-The-Move						
Test scenario	Radar velocity (km/h)	Drone velocity (km/h)	Recording time (s)	Time on target (s)	Environmental conditions	Drone type
Slow radar and fast drone	30	60	24	0.3	Frozen ground	Autel evo II
	30	60	32	0.4	Frozen ground	DJI Inspire 2
	10	30	18	0.225	Sunny	Autel Evo II
Fast radar and slow drone	40	10	22	0.275	Frozen ground	Autel evo II
	40	10	24	0.3	Frozen ground	DJI Inspire 2
Equal speed with drone sideways	10	10	26	0.325	Sunny	Autel evo II
Almost equal speeds	50	40	22	0.275	Frozen ground	Autel evo II
	30	40	28	0.5	Frozen ground	Autel evo II
	30	40	22	0.275	Frozen ground	DJI Inspire 2
Fast radar and static drone	50	0	20	0.25	Frozen ground	DJI Inspire 2
Slow radar and static drone	10	0	26	0.325	Sunny	Autel evo II
Slowing radar and static drone	50-10	0	34	0.425	Frozen ground	Autel evo II

Table 4.7: Datasets of difficult pre-defined test scenarios to obtain reliable ground-truth. The range in this context is no longer constant, even if the drone is hovering. Moreover, the azimuth is always changing, depending on the direction of motion of the radar. The data type follows a similar pattern as in the earlier Static radar case but is not shown in this table.

It is important to emphasize that while there are multiple minutes of total recording time, a total rotation takes 2 seconds and the drone is visible only in a single line consisting of 100 processed sweeps. Thus, while the recording time is relatively high, for each rotation only a CPI of 25 milliseconds contains the target of interest, while the other 79 lines lasting 1.975 seconds contain only clutter. In terms of images used to train the classifier, only one generated range-Doppler plot contains the drone, making most of the other plots irrelevant.

Moreover, given IRIS® records for a single rotation 3125 range bins for each of the 8000 sweeps, saved for every single receiver channel, it takes 400 megabytes to save the data of a single rotation. Considering a lot of this data can not be used to train the classifier as it contains only clutter, huge amounts of data need to be saved to obtain relevant images.

In the end, a visualization using Google maps was done to draw the pre-defined paths of the drone and radar to help localize them in the polar range maps, as seen in Fig. 4.7 & 4.8.

4.2.2. Range-Time and Range-Doppler plots

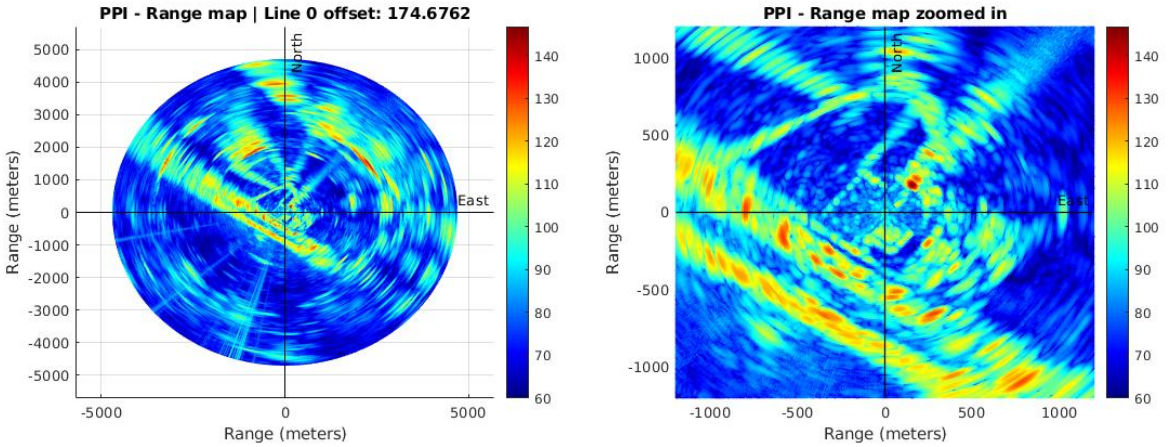
The range-time map serves only as means to visualize the plan position indicator obtained of the scanned area, as seen in Fig. 4.8 for a full rotation. Moreover, when a single CPI of 100 sweeps is processed, the Plan Position Indicator (PPI) plot shows exactly the region covered at a specific azimuth angle, which facilitates linking the data with the ground truth and understanding the plots. This further allows to compare the generated data with Robin Radar Systems plots and ensure the processing yields similar results, and also to localize the sectors of interest.

While the range maps are used as means to correlate the data with the real life environment, the

(a) Trajectories for the test scenarios of static radar datasets.

(b) Trajectories for the test scenarios for radar on-the-move datasets.

Figure 4.7: Google maps hand-crafted trajectories of the pre-defined setup for the field recordings.



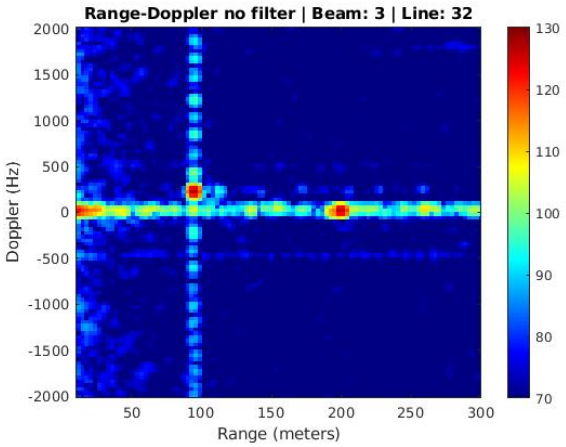
(a) Polar range map of a full rotation.

(b) Zoomed polar range map of a full rotation.

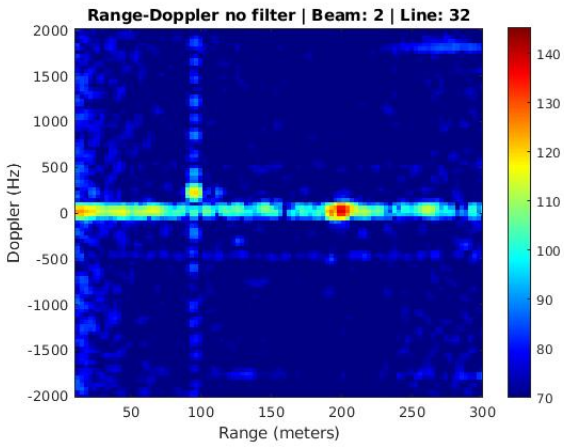
Figure 4.8: PPI plots used to compare the results generated from the matlab processing chain with Robin Radar Systems processing. Strong clutter induced by a line of nearby trees and road can be seen in the in the lower diagonal of the picture, with a parallel river. Moreover, the fences, railways and buildings can all be pinned down by correlating this plots with Google maps, hence their usefulness. In the end, the *Line 0 offset* represent the azimuth offset of where the recording begins with respect to the true North.

actual classification is done based on the range-Doppler plots, as seen in the surveillance counter-drone radars state-of-the-art [8], [9]. The distinction of drone vs other targets is achieved via the wide micro-Doppler generated by the propellers, compared to birds for example which yield only a narrow bulk Doppler. Nonetheless, the wide micro-Doppler induced by drones' rotary wings is only visible at relatively close distances, up to $1-1.5$ km, after which only the bulk Doppler from the main body remains visible, thus making reliable plot-based classification dependant on range. However, this method serves as a robust way to distinguish drones from other targets, and once the UAV is too far to observe the micro-Doppler and only the main body remains, classification based on previous tracks comes into role. However, track-based classification is beyond the scope of this thesis project and only the plot-based computer vision classifiers are discussed.

In Fig. 4.9 a drone at approximately 100 meters is clearly observed in a region without much clutter, except for the static one. The bulk Doppler induced by the main body is clearly seen at 200 Hz, around which the wide micro-Doppler is induced by the very fast spinning rotary wings. Moreover, as discussed in Subsection 4.1.2, spatial filtering and DoA estimation can be reliably achieved via beamforming on the receiver channels. By steering multiple beams at different elevation angles, on one side some of the ground clutter can be filtered, but also the intensity of the detected targets can be analyzed and thus their elevation estimated. This can be seen in Fig. 4.9a where the target is located in the main beam, compared to Fig. 4.9b where the target is located in a side lobe of a different beam.



(a) Range-Doppler plot for a single beam steered towards $0°$ w.r.t. the antenna aperture. The beam is centered directly around the drone.



(b) Range-Doppler plot for a single beam steered towards $-10°$ w.r.t. the antenna aperture. The target is not located inside the main beam making it less visible.

Figure 4.9: Two range-Doppler plots showing how plot-based UAV classification is possible via IRIS® FMCW surveillance radar. The bulk Doppler induced by the main body of the drone is colored is bright red, while the micro-Doppler induced by the propellers is extended across the entire Doppler spectrum. Moreover, it highlights for elevation estimation may be reliably achieved by identifying the beam with the highest target gain.

4.2.3. Image processing and contrast improvement

To visualize all the aforementioned radar plots, a transformation is required to map the received power into colors, as seen in Fig. 4.10.

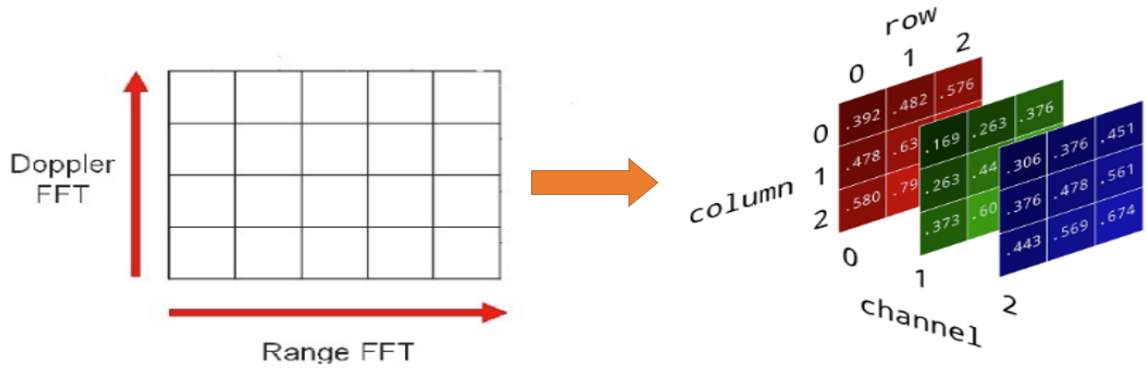


Figure 4.10: Range-Doppler 2D matrix containing the power values transformed into 3D RGB image.

More specifically, the power intensities from the range-Doppler snapshots are mapped on a pre-defined color scale, which allows to control the dynamic range and RGB values. Thus, the background noise is set to black, the micro-Doppler induced by the propellers is white, and the bulk Doppler induced by the main-body of the drone is red, as seen in Fig. 4.11

While the direct range-Doppler matrix for a single CPI could be directly fed as input to the neural network, the dynamic range of such a snapshot is much smaller than what can be obtained with a 3D RGB image. Moreover, there is a plethora of labeled big data sets of images based on color intensities encoded in the RGB channels. By mapping the received power from range-Doppler snapshots into traditional RGB images, transfer learning can be deployed from open-source pre-trained networks designed to operate on optical images from DL and CV, such as ResNets, YOLO, R-CNNs and many others. Thus, this transformation enables the radar community to benefit of largely available labeled data sets from the deep learning and computer vision communities.

In Fig. 4.12, we see the normalized intensity of the signal to vary between 0 and 40 dB for each CPI,

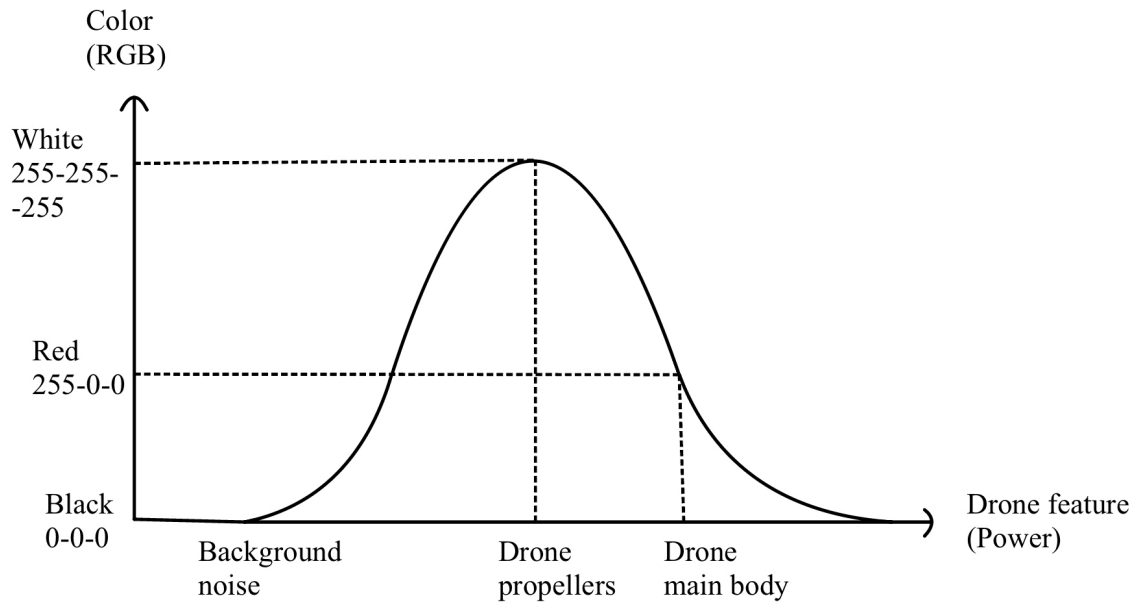


Figure 4.11: Mapping of the range-Doppler snapshot intensities into the custom RGB color map.

while the values of any pixel from the same plots vary between 0 and 255, as seen in Table 4.8. Thus, a higher dynamic range can be obtained for low RCS targets by transforming the snapshots into images.

Data type	Dynamic range interval
Radar snapshot	[0; 40]
RGB image	[0; 255]

Table 4.8: Dynamic range interval for the snapshot from the radar data cube based on the received power intensities, compared to an RGB image based on the pixel values.

While the plots from Fig. 4.9 are quite clear and without much noise, the classifier showed that it fails to catch weak micro-Doppler modulations when the noise floor is higher, or when the drone is far away and the propellers are not visible anymore, or if the drone is drown in clutter. To further improve the classifier performances without increasing the dataset, an extra step in the signal processing chain has been added: **image processing**. This very trivial step aims to change the colors used to visualize the data and improve the contrast and contour sharpness between the background and the micro-Doppler modulations.

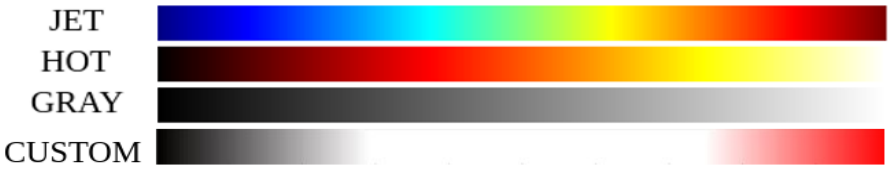


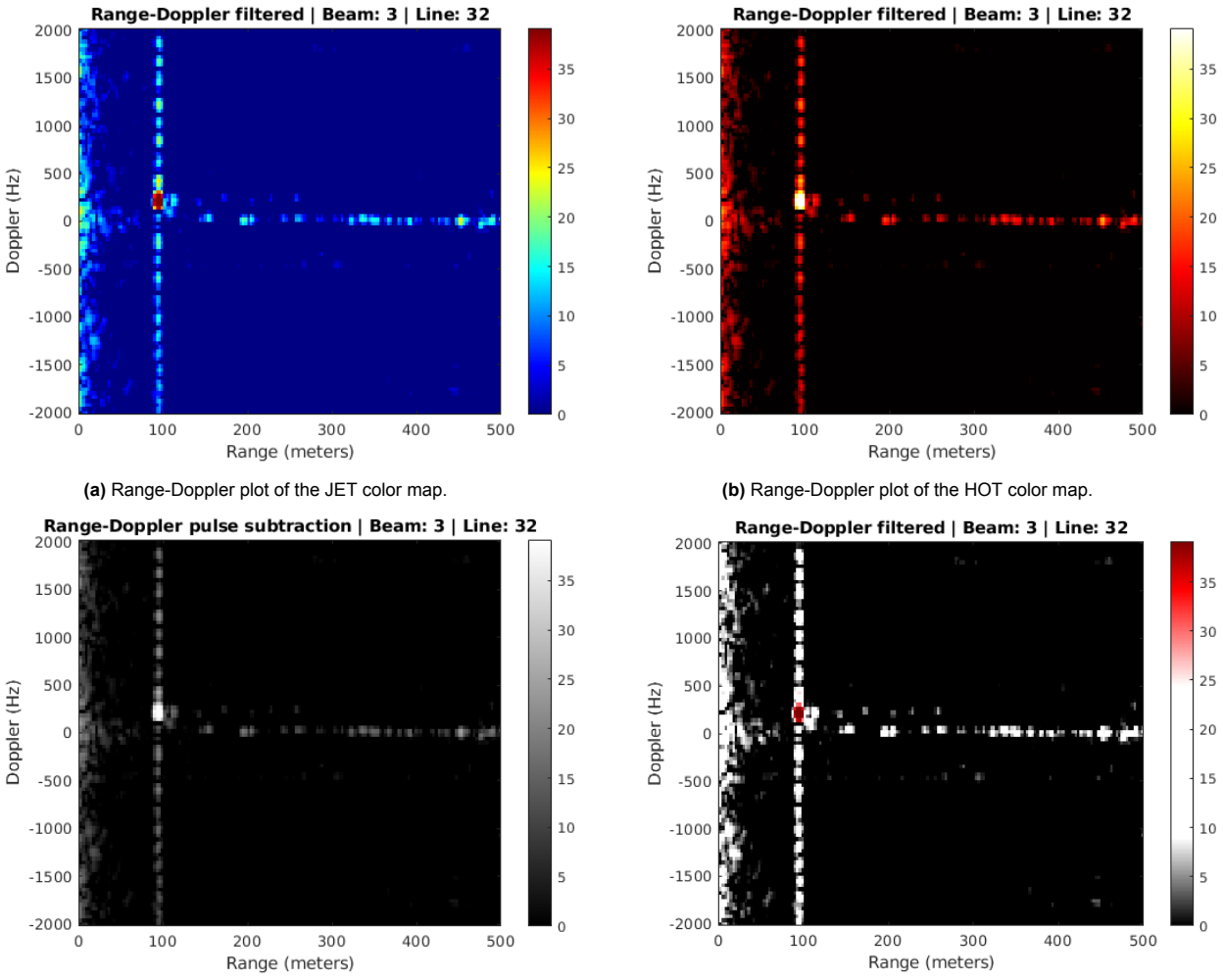
Figure 4.12: Intensity color map difference between Jet, Hot and Gray scale and color maps, from Matlab documentation.

The following standard maps were analyzed, seen in Fig. 4.12, plus a novel proposed map:

- Jet color map uses a blue background and a light blue micro-Doppler, with a red bulk Doppler;
- Hot color map uses a black background and a red micro-Doppler, with a white bulk Doppler;
- Gray scale color map uses only one channels, with a black background, gray micro-Doppler and white bulk Doppler;

- New custom color map that uses black background and a white micro-Doppler, with a red bulk Doppler. This choice maximizes the contrast between the background using the lowest RGB values $(0-0-0)$, and the micro-Doppler using the highest RGB values $(255-255-255)$, while being able to distinguish the main body of the drone.

First let us look at the the visual differences between the plots, as seen in Fig. 4.13. While initially there is no difference observed except for the visual colors, the pixel values greatly differ, as seen in Fig. 4.14, which will considerably influence the classifier performances.



(a) Range-Doppler plot of the JET color map.

(b) Range-Doppler plot of the HOT color map.

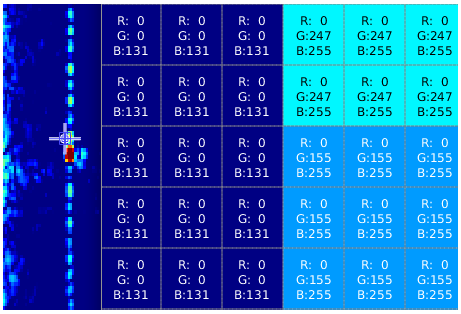
(c) Range-Doppler plot of the Gray scale color map.

(d) Range-Doppler plot of the custom color map.

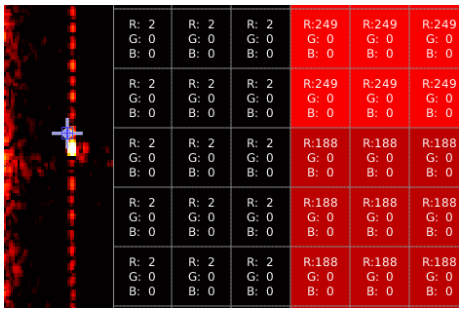
Figure 4.13: Range-Doppler plots highlighting the visual differences between the color maps.

Given the chosen DL & DL methods use CNNs as the architecture backbone, the cross-correlation between the pixels is computed. Therefore a high contrast and clear contour between objects yield better performances. Thus, let us analyze the differences in the images, namely the pixels and the contrast differences between the two color maps. In Fig. 4.14 the contrast between the edge of the drone micro-Doppler and the background is seen in the two middle columns.

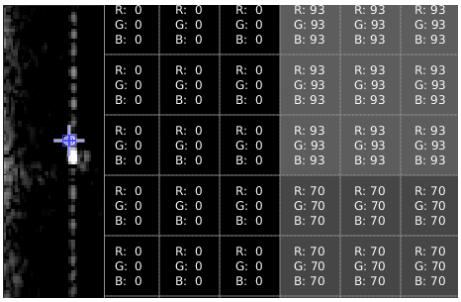
In Fig. 4.14a the values of the JET color map pixels of the contrast are shown, which show that given the background is dark blue and the micro-Doppler is light-blue, in the blue channel the contrast is lower given they share the same tint. In the HOT color map from Fig. 4.14b, the black background uses the lowest RGB values, while the red channels has the highest. Thus the dark background plays an important role to ensure no extra information is encoded within the pixels surrounding the target. In the gray scale figure from Fig. 4.14c, the RGB channels all use the same value to maintain the



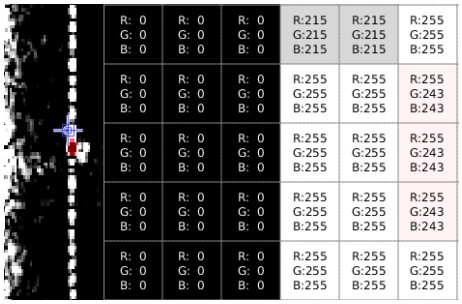
(a) Pixel map for the jet color map with relatively good contrast in a single channel but poor in the other.



(b) Pixel map for the hot color map with relatively good contrast in a single channel but non-existent in the others.



(c) Pixel map for the gray scale color map with average contrast across all duplicated channels.



(d) Pixel map for the custom color map with the highest contrast in all channels.

Figure 4.14: Two pixel maps that highlight the differences in the RGB pixel values for the two different color maps.

3D structure of the image, we can see that by maintaining a similar color across the micro-Doppler modulations, a good contrast across all channels can be achieved, albeit not being the best one since it uses a fade gray over a black background. Alas, we have seen that:

- A bright micro-Doppler with high RGB values has the strongest contrast (jet color map);
- A dark background boosts the contrast between the Doppler modulations and the neighbouring pixels (hot color map);
- Homogeneity across all the pixel values yields the best contrast and sharpest edges in all three color channels (gray scale image).

Thus, in the proposed custom color map from Fig. 4.14d, we combine all the advantages and set the background to the lowest possible values (black with $0-0-0$ RGB values) and the micro-Doppler to the highest (white with $255-255-255$ RGB values), while maintaining the red bulk Doppler to be able to localize the main body of the drone.

In Table 4.9 the pixel distances between the edge of the micro-Doppler of the drone and the background are highlighted for each color Colmap. The contrast line can be easily computing by subtracting the pixel values and taking the absolute value, i.e.

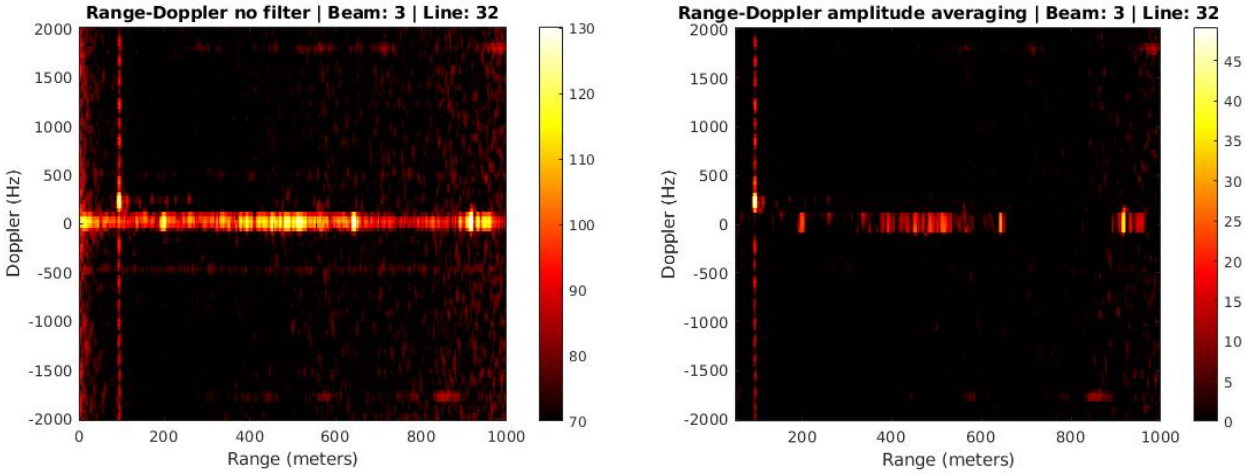
$$Contrast = |Pixel^{RGB}_{drone} - Pixel^{RGB}_{background}| = |Power^{dB}_{drone} - Power^{dB}_{background}| \tag{4.12}$$

Colormap	RGB contrast	Power contrast
Jet	0-247-124	15 dB
Hot	247-0-0	15 dB
Gray scale	93-93-93	15 dB
Custom	255-255-255	15 dB

Table 4.9: Contrast between the drone blades micro-Doppler and the background for each of the proposed color maps, juxtaposed by the differences in received power.

4.2.4. Clutter reduction (or image transformation)

Let us look further into the clutter reduction methods through each stage of the filtering. While these techniques were used on the radar data cube for the Doppler & range bins, this processing could also be seen as multiple image transformations, depending on the background of the reader.



(a) Range-Doppler plot without any filters to highlight how the raw image looks like.

(b) Range-Doppler plot which uses amplitude averaging to suppress the clutter and also lower the noise floor level.

(c) Range-Doppler plot which uses two pulse canceller to remove static clutter, but increases the background noise around high frequencies.

(d) Final range-Doppler obtained from the complete processing chain. The initial raw image is first passed through a pulse subtraction filter, then through the amplitude averaging step.

Figure 4.15: Range-Doppler plots using HOT color map which yield better contrasts. Moreover, the clutter filter chain is shown, from an unfiltered plot on the top left, up to the final plot fed into the classifier from bottom right.

The top left image, Fig. 4.15a, shows the image obtained without any clutter suppression techniques. The static ground clutter centered around 0 Hz is clearly seen.

The top right picture, Fig. 4.15b, shows how amplitude averaging is done in range and Doppler and averages all the range bins for each sweep. For every CPI of 100 sweeps, it takes all the N range bins and divides them by their mean absolute value in each sweep, i.e.

$$\text{Range bin} = \frac{\text{Range bin}}{\frac{1}{N}\sum_{i=1}^{N} |\ \text{Range bins}_{\text{per sweep}}\ |} \tag{4.13}$$

This filter helps remove background noise and also suppress clutter, but slightly decreases the visibility of the micro-Doppler modulations. However, the clutter notch remains relatively wide, albeit being suppressed.

In the bottom left plot, Fig. 4.15c, a two-pulse canceler is used that acts as a high-pass filter by subtracting two consecutive pulses and removing the similar information present, i.e. the static clutter. This may be implemented as a simple all-zeros FIR filter with binomial coefficients $[1 -1]$ that removes low-frequency components in the slow-time dimension via the matlab as:

```
radar_blocks_DopplerFilter = filter([1 -1],1,radar_blocks_noFilter,[],3);
```

The filter response can be seen in Fig. 4.16 which shows how the FIR filter acts as a high-pass filter that removes the static clutter centered around 0 Hz.

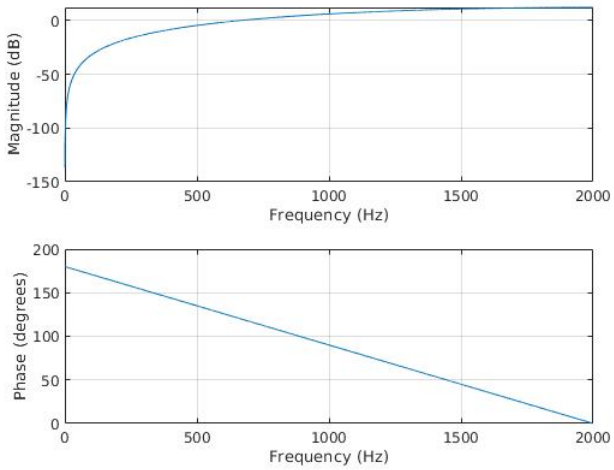


Figure 4.16: Pulse canceller frequency response highlighting how the filter suppresses the signals centered around 0 Hz.

Another less signal processing option and more pulse-radar oriented that yields the same results is to make a literal subtraction of all the range bins between two consecutive sweeps, i.e.

```
for sweep = 2 : nrOfSweepsPerCPI
        Cube_DopplerFilter(:,:,sweep) =
            Cube_noFilter(:,:,sweep) - Cube_noFilter(:,:,sweep-1);
end
```

This increases the visibility of the fast-spinning propellers, but it makes the distinction between the rotary wings and main body impossible and also increases the noise floor level around higher frequencies.

In the end, in the low right corner in Fig. 4.15d, the final plot used for classification uses both the two pulse canceller and the amplitude averaging clutter reduction methods in cascade. The pulse subtraction and the amplitude averaging steps compensate each other's weaknesses, and thus bring the best results. Most of the clutter is reduced and it is much narrower, the background noise is low, the main body can be properly distinguished from the propellers and the micro-Doppler is not cut around 0 Hz, thus yielding the clearest plots.

To quantify the improvements of the aforementioned plots, let us look at the SNCR differences analyzed for the images presented in Fig. 4.15. Given we work with images obtained from signal measurements, the traditional formulation for SNCR computation in image processing is given by:

$$SNCR = \frac{\mu}{\sigma} \tag{4.14}$$

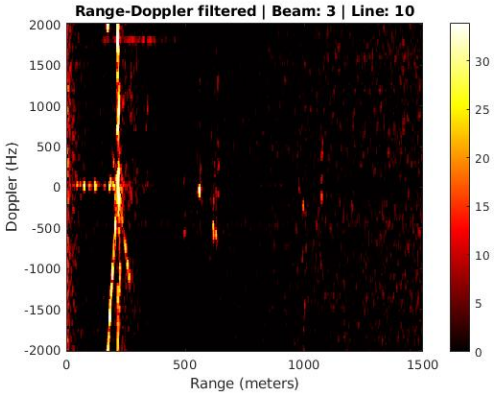
where μ is the mean pixels value, and σ is the standard deviation of the pixels [77, 78]. Thus, let us look at the SNCR improvements through each of the filtering steps, highlighted in Table 4.10.

SNCR	Unfiltered plot	Pulse subtraction	Amplitude averaging	Both techniques in cascade
Value	0.0916	0.112	0.118	0.127

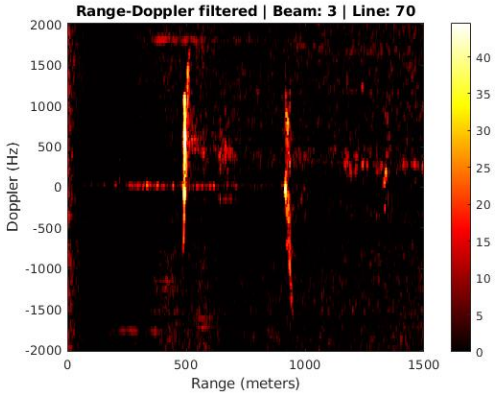
Table 4.10: SNCR values improvements through each filtering method for each image.

It is then clear that each of the clutter filtering techniques improve the SNCR. Moreover, the improvement brought by merging both techniques is further confirmed for the test image.

Let us look at another possible target and its signatures from the recording environment, besides the drones, namely wind turbines. In Fig. 4.17 multiple wind turbines are shown in range-Doppler plots. While some of them are clearly distinct from drones given their very strong reflectivity, such as those in Fig. 4.17a, depending on their orientation and range some may look similar to a drone, as the blades located at approximately 500 & 1 km range from Fig. 4.17b.



(a) Close wind turbine located close to the radar. The main body alongside the rotating blades ca be clearly seen.



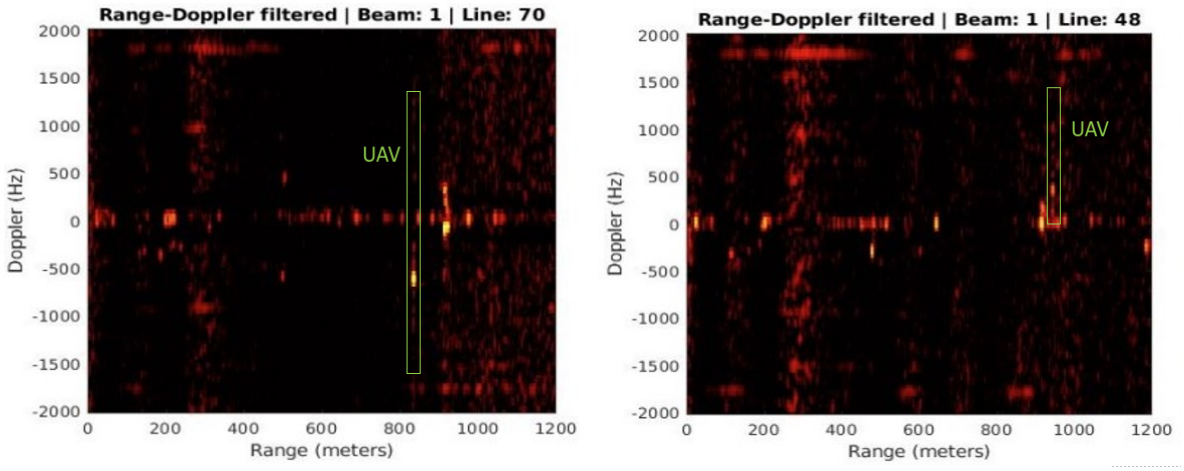
(b) Medium and relatively far range turbine blades with similar wide Doppler with those of drones that may yield false alarms.

Figure 4.17: Range-Doppler plots of wind turbines. Depending on their orientation with respect to the radar, the main body alongside the blades may be captured, or only the rotating blades. Moreover, depending on their range and orientation, the Doppler modulations may be similar to those of UAVs the classifier may falsely classify them.

4.2.5. Micro-Doppler range dependency

We have shown that the range-Doppler plots can be used as a robust way to differentiate targets. However, as shown by B-S. Oh et al. in [8] and further confirmed here, the visualization of the micro-Doppler modulations induced by the rotary wings is highly dependant on the range. While the mainbody of the drone can be seen very well from long range, the plastic propellers do not reflect much energy back. This phenomena can be seen in Fig. 4.18a where the drone located at approximately 800 meters is flying away from the radar with -500 Hz and the mainbody can be clearly seen, but the micro-Doppler begins to fade. In Fig. 4.18b, the drone reaches approximately 1 km and begins to return towards the radar, but the propellers can almost no longer be distinguished from the background noise.

Thus, reliable plot-based classification for micro-Drones such as Autel Evo II can be achieved up to approximately 1 - 1.3 kilometers, depending on the noise level and clutter. For bigger drones with stronger



(a) Drone flying away from the radar with 500 Hz, localized at approximately 800 meters. The micro-Doppler is still visible, but the resolution decreased.

(b) Drone turning around and flying towards the radar with 250 Hz at approximately 1 km. The drone propellers are almost not visible at all and without prior information, reliable classification may not be possible.

Figure 4.18: Range-Doppler plots highlighting the degradation of the micro-Doppler resolution with range due to the low RCS of rotary blades.

RCS such as DJI Inspire, classification may be achieved up to 2 kilometers according to Robin Radar, however only the first micro-drone was investigated in this chapter. The DJI Inspire 2 was however used in Chapter 6, but the focus revolves around classification performances via degraded plots obtained with radars on-the-move rather than maximum range. After this point, track-based classification can be used to complement the investigated method, but this scenario is beyond the scope of this thesis.

4.3. Summary

In this chapter we have highlighted the IRIS® counter-drone radar specifications, the signal processing chain used to process real-world data, as well as the data collection environment and scenarios. Moreover, the interaction with the environment of the radar and collected datasets were analyzed by displaying the signature of UAVs and other targets. The generated plots used for classification were examined in-depth, and a clear visualization of the signal processing chain choices were shown highlighting the advantages juxtaposed by the limitations and challenges. In the end, the degradation of micro-Doppler modulations with range was shown.

Moreover, improvements to the signal processing chain by blending two clutter mitigation techniques were proposed, i.e. pulse cancellers and amplitude averaging deployed both in cascade rather than individually. While they are not a novel algorithm, using them together brings a clear improvement and the benefits brought in terms of SNCR and plots visibility are clear.

Alas, while achieving reliable plot-based drone classification using surveillance radars with low dwell time is challenging, we have shown that it is indeed possible. The next chapters will cover how this can be achieved, in a novel framework based on computer vision.

5

Performances of Computer Vision for Object Detection via Radar Images

In this chapter the performances of the computer vision YOLOv5 classifier based on range-Doppler images are assessed and discussed to evaluate the feasibility of transferring computer vision and video processing algorithms within the radar domain, thus highlighting the main research objective of this thesis. The validation of the proposed framework is done on data generated from the same radar presented in the previous chapters. Moreover, this chapter presents and analyzes the results for differently balanced data sets, juxtaposed by methods to increase robustness, avoid overfitting, and boost overall performances.

5.1. Transferring computer vision into radar engineering

With the signal processing chain that generates the input to the object detector covered in Section 4.2.2, it is time to move on towards the computer vision pipeline. DL and CV techniques based on CNNs, as discussed in Chapter 3, compute the cross-correlation between the input image pixel values and the sliding kernel, which then yields the feature map. Thus, the higher the contrast between the objects and the sharper their contour is, the better the algorithm will be able to learn and predict accurate bounding boxes [79]. To this end, the contrast, sharpness and dynamic range improvements obtained via the proposed custom color map from Subsection 4.2.3 is used in the rest of the following sections. Additionally, in Appendix B, the Jet and Hot color maps performances are presented.

Moreover, to the best knowledge of the author, predicting bounding boxes to jointly localize & classify targets via YOLO was never used on radar data, i.e. range-Doppler images, but was conceived specifically for high-resolution cameras. The advantage of optical cameras is the much shorter wavelength compared to radars. This yields a much finer resolution of the objects' shapes and contours, a high contrast, and a superior sharpness. In radar engineering, such contrasts and clear contours of the targets rarely exist, and the shapes and edges of objects are intertwined with the background, hence making it impossible to clearly distinguish where an object ends and another begins. However, radar outputs are not images, but rather range-Doppler-angle data cubes. By applying Fourier transforms, the data can be visualized as an image, but the encoded information is inherently different from optical cameras. Each pixel in the radar data matrix encodes the kinematics of the objects, namely the range, velocity, and angle of arrival. Thus, while the radar output can be seen as an image based on the RGB channels such as optical images, the embedded information within the data matrix greatly differs between the two sensors.

Therefore, the main research objective of this thesis is highlighted in this chapter, namely assessing

the performances obtained by transferring computer vision algorithms within radar engineering.

5.2. Performance metrics and loss functions

A brief description of the traditional ML/DL performance metrics [80] is highlighted below. However, computer vision pipelines such as YOLO or R-CNN, are not only predicting a class per image, but rather a number of bounding boxes each with its own class label. Thus, more object detection focused performance metrics and loss functions are also discussed to further assess the performance of CV based YOLO detectors [61, 62, 63, 64]. Moreover, the data sets used are analyzed with only one possible object class per image, namely predicting the bounding box & label of a drone in different radar images. Thus, the accuracy and classification loss will not be representative in this case, as they are meant to assess the performances in multi-labels situations.

Intersection over Union (IoU): represents the overlap area between the predicted bounding box and the ground-truth inside the training data and is one of the most important metrics for object detection pipelines.

Confusion matrix: measures the number of predicted true positives, false positives, true negative and false negative for each class.

Accuracy: measures the number of correct predictions over all possible predictions, i.e. how balanced the number of predicted classes is. Given the used datasets aim to classify only drones, this score is irrelevant.

Recall: measures how many relevant predictions are made out of all predictions. It shows how many potential objects the model captures, i.e. if all possible detections are classified, then the recall score is 100% but with many false positives. In the computer vision framework, this score can be interpreted as the area of overlap between the ground truth bounding box and the detected one, divided by the detected bounding box.

Precision: measures how accurate the predicted classes are out of all positive predictions showing how precise the model is, i.e. the percentage of correct predictions. For example if the model classifies one and only one drone accurately, the precision score is 100% but misses many potential true positives. In computer vision, this metric measures the same area of overlap, but this time it is divided by the ground truth object.

Mean Average Precision (mAP): is based on the confusion matrix, precision, recall and IoU. It is a weighted mean of precision at each threshold. The said threshold represents the increment in the recall score compared to the prior threshold. Then, the results is averaged for each class. Thus, MAP highlights a trade-off between precision and recall that considers both false positives and false negatives, making it a very popular metric in all object detection applications. Moreover, $MAP@.5$ means the average precision is computed for IoU scores above 0.5. This metric is the most important in object detection applications as it represents the main criterion of choosing the best model.

Let us look at a more visual representation of the aforementioned precision and recall scores based on the predicted and grund truth bounding boxes, as seen in Fig. 5.1.

The loss functions specific to object detection that are minimized during training are described below.

Box loss: represents how well the algorithm localized the centre of the bounding box and how well it covers the entire object. By minimizing this function, the algorithm learns to better predict the location of the bounding box. Given drones are seen only as a relatively narrow line in range and wide in Doppler, the box is typically well centered in this application. The used loss function is based on the complete IoU score used to compare the consistency of the predicted width and heights of the bounding boxes, and uses the mean square error to compare the center of the predicted bounding box with the ground truth.

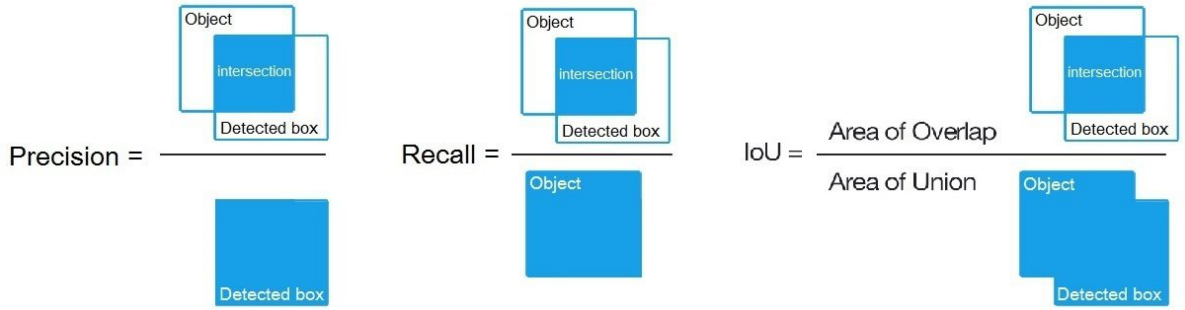


Figure 5.1: Visualization of the precision, recall and IoU computations in computer vision based on the predicted bounding boxes, the ground truth and their intersection, from [81].

Objectness loss: measures the probability that an object exists in a proposed region of interest. By minimizing this loss function, a better box is predicted by maximizing the IoU. High objectivity scores mean that the window is likely to contain an object. The used loss function is based on the binary cross-entropy.

Classification loss: measures how well the classifier predicts the adequate class of an object. Like the accuracy, this measurement in a single-class context is irrelevant. The used loss function is also based on the binary cross-entropy.

In Fig. 5.2 we can see a single image with multiple predicted bounding boxes. Given there are multiple shapes that can fit the detected object, the algorithm yields multiple predictions. Thus, based on these specific loss functions, the bounding box that fits the data the best is kept and the others are dropped.

Figure 5.2: Multiple detected boxes appear given the object fits multiple differently shaped bounding boxes, each having a different confidence score, from [82].

5.3. YOLOv5 model training, validation and selection

The chosen architecture is YOLO v5 [65] which uses the PyTorch implementation of YOLO v4 rather than Tensorflow, and has the following advantages that were documented for high-resolution cameras [64]:

- Real-time multi-label object detection;
- High accuracy for small objects distinction;
- Performs well with small datasets using transfer learning;

- High accuracy for bounding boxes prediction;
- Open-source code on Github[1];
- Many free labeling tools.

In Fig. 5.3, multiple YOLO architectures are seen. The main difference is the number of parameters used, which in turn affects the size of the model. According to the creators of YOLOv5 [65], and common intuition, the more parameters and the bigger the model, the better the results are. However, such models are slower to run and require more CUDA memory available to train. To this end, smaller models such as YOLOv5n/s are preferred for mobile deployments, while medium models such as YOLOv5m/l are recommended for cloud deployments, and the biggest model, YOLOv5x, traditionally being used on work stations.

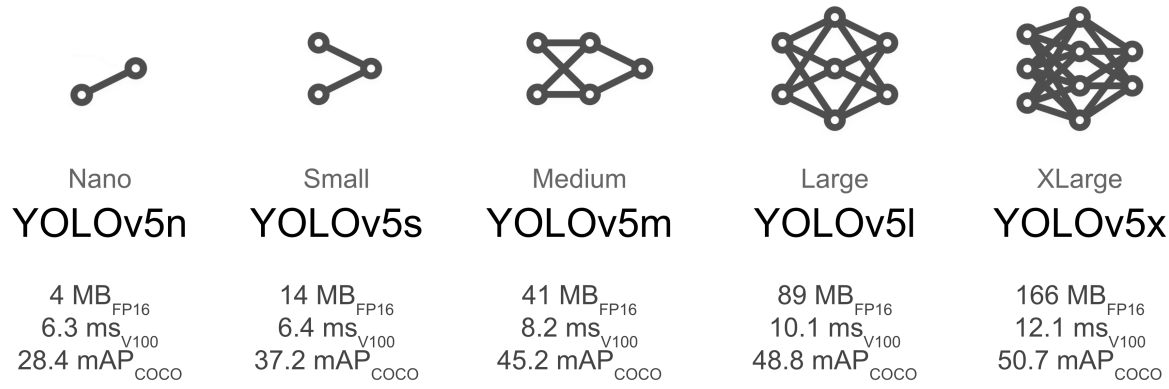


Figure 5.3: YOLOv5 architectures based on different number of parameters, from [61].

The model used in the future sections is based on the YOLOv5s, given radar images are not as complicated as the optical ones, but do have a reduced sharpness overall. Moreover, given the small size of the chosen architecture, the trained model could be transferred to a real radar system and easily be implemented on FPGAs. While bigger models could fit on the memory of GPUs, given the nature of small data sets available in radar engineering, a smaller model is less prone to overfitting and thus performs better on never before seen data.

5.3.1. Loss function

In order to train a model, traditionally a single loss function is minimized by comparing the algorithm prediction with the ground truth. However, in YOLO there are three cost functions that are minimized, as discussed in Section 5.2, namely:

- Class loss (BCE loss);
- Objectness loss (BCE loss);
- Location/Box loss (Complete IoU loss).

Given only one label is predicted for each bounding box, the class loss is always null and therefore does not contribute to the model improvement. Thus, we write the joint loss function as a weighted sum of all the cost functions as following:

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{cls} + \lambda_2 \mathcal{L}_{obj} + \lambda_3 \mathcal{L}_{loc} \tag{5.1}$$

where $\lambda_{1,2,3}$ are constants representing the gain of each loss over the total cost function. These gains are part of the model hyperparameters, and are defined as:

[1]YOLO v5 Github repository can be found at https://github.com/ultralytics/yolov5.

Hyperparameter	Class gain	Objectness gain	Box gain
Gain	0	1	0.05

Table 5.1: Individual gains for each of the cost function, highlighting how much each contributes to the total loss function.

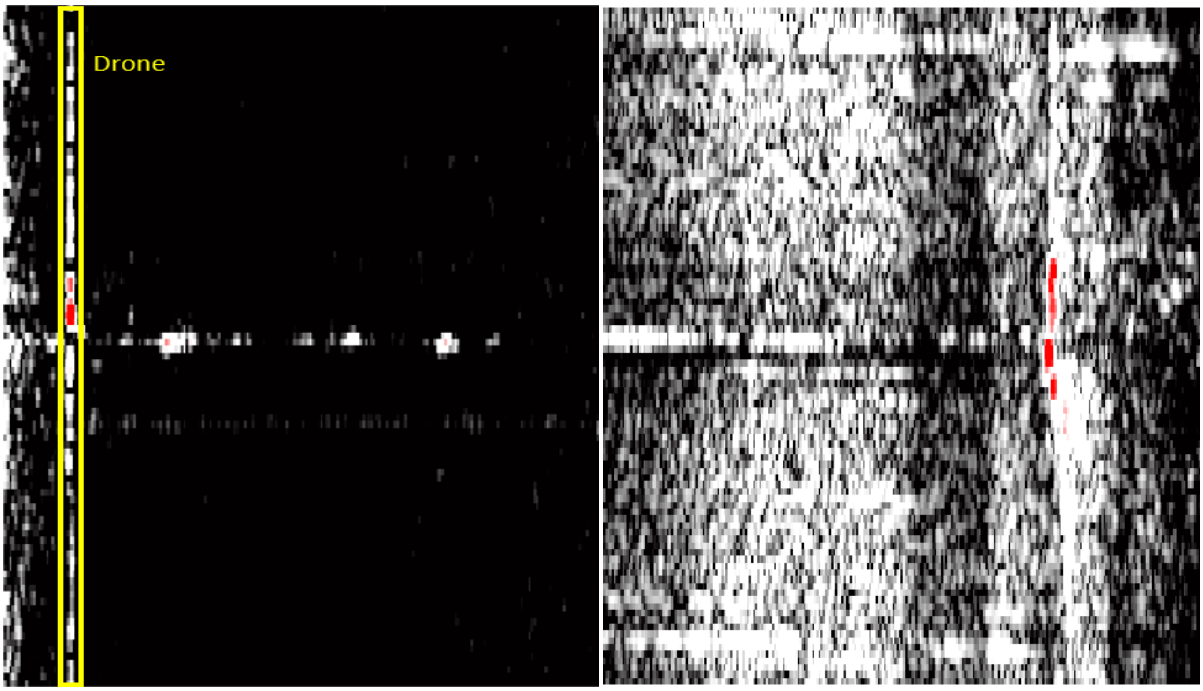
The objectness loss, $\mathcal{L}_{obj}$, has therefore the highest impact over total loss function, $\mathcal{L}_{total}$. Given the objectness loss measures the probability that an object exists withing a proposed region of interest and maximizes the IoU, this function is the main contributor to obtaining the detection results [65].

5.3.2. Data sets and validation

All the data sets use the proposed color map that has the highest contrast between the background and the micro-Doppler modulations, as discussed in Subsection 4.2.3. In Appendix B, the performances of the data sets created with default color maps are presented. The discussion in the following sections is therefore focused around the balance & complexity of the data sets, the model parameters & hyperparameters, their improvements, and the obtained performances.

The training is done on multiple data sets, as following:

- Simple drone-only data set containing 300 labeled range-Doppler images (bounding boxes centered around the target with the specific label, as seen in Fig. 5.4a);
- Balanced data set with equal ratios of labeled drone pictures and unlabeled clutter plots (range-Doppler images that contain only background noise or clutter, as seen in Fig. 5.4b);
- Unbalanced data set with 300 pictures of drones and 1000 pictures of clutter;



(a) Range-Doppler image containing the labeled drone.

(b) Range-Doppler image containing only clutter without any labels.

Figure 5.4: Data set used for training, that contains range-Doppler images with drones labeled using a bounding box, and clutter images that do not contain any labels.

The data is split for cross-validation with ratios of 60%, 20% and 20% between the training, validation and test sets. The objective of this split is to use each data point to understand how well does the model perform the task of learning from the training dataset, and then how well does it make predictions on

unseen data from the validation set that contains the ground truth a-priori. The test set is used to ensure the model does not overfit on either the training or the validation data.

Thus, these models aim to assess whether predicting bounding boxes on low-contrast radar images with poor objects shapes & contours is indeed possible. It is worth noting that the data is split only as means to check the model, not build the most robust one. The objective is not to build the best model possible, but verify the scallability of computer vision algorithms in novel applications from the radar domain. To build a more accurate and robust model, once it is verified that the technique performs indeed well in this novel scenario, more data can be gathered and then used for training.

5.3.3. Choosing the best model

After training, the model saves the last and the best weights generated during training. The best model is chosen using a weighted combination of the average precision at different thresholds [65]. More specifically, the mAP@0.5 with IoU threshold at 0.5 contributes for 10%, and the mAP@0.5:0.95 with a threshold varying from 0.5 to 0.95 with a step of 0.5 accounts for 90%. Thus, an average between the precision and recall scores computed for multiple IoU scores is taken to assess the best finesse of the model.

In the end, memory constraints and training times do not pose serious[2] problems given the training is done on work stations. Moreover, the pre-trained weights with the best scores could be transferred to real systems to operate in real-life scenarios.

To reiterate, the purpose of this chapter is to assess the feasibility of localizing drones based on range-Doppler plots by predicting bounding boxes via YOLO. Improvements in terms of data pre-processing to compensate for the lower sharpnesss and contrast of radar images are proposed through each of the following sections, which highlight the boost of the classifier performances.

5.4. Training, overfitting and performances

The model parameters of DL & YOLO are the weights of the neural networks that are learned during training. Overfitting occurs when the model learns too much from the training data and is not able to generalize over new data. Similarly, underfitting occurs when the data set is too complex and the network can not learn the underlying data pattern during training. In order to prevent overfitting and boost the capacities of the trained model to generalize, more data is usually a key component. However, due to the small nature of the data sets in radar engineering, data augmentation methods are investigated.

Additionally, YOLO has approximately 30 hyperparameters that require initialisation before training, such as the learning rate, weights decaying rate, momentum, and others. For different data sets, different hyperparameters may yield contrasting results. Thus, the better they are optimized, the better the performances of the model will be. In most cases however, the default hyperparameters are chosen for the COCO data set that was used to pre-train the network on.

Thus, this section, based on differently balanced data sets, aims to present the following analysis:

- Discuss how the neural network weights are initialized via transfer learning;
- Highlight the benefits of data augmentation to avoid overfitting and boost performances;
- Test different optimizers, namely SGD and Adam, and assess their performances;
- Show range-Doppler plots to visualize the performances of computer vision pipelines transferred into radar engineering.

[2] The limits are hard capped by the memory of the available GPUs, but given the complexity of the data set, the batch sizes are chosen such that they do not exceed the memory and are also relevant for the problem.

5.4.1. Transfer learning

It is important to emphasize that the used data set in this entire chapter was very small compared to traditional deep learning applications. The data set presented in the following section contains only 300 pictures of drones, while traditional deep learning data sets contain thousands of images.

Thus, the network weights were initialized from a pre-trained checkpoint of 300 epochs on the COCO dataset[3], which contains over 200.000 labeled images out of approximately 330.000, 1.5 million object instances and 80 object classes. Pre-training a network is a common practice for small and medium data sets, and it was shown to increase the classification performances for micro-Doppler applications such as human activity recognition or human voice & gesture recognition [80].

5.4.2. Data augmentation

In order to further increase the diversity of the images the network sees during learning, data augmentation is performed during training. Nonetheless, compared to optical images, radar plots contain information regarding the micro-Doppler modulations of targets. Therefore, when transforming the data, the kinematics of the target need to be coherent with the data seen in real-life. Thus, the following techniques were used to create new images:

Flip: flips the range-Doppler plots upside down. This method shifts the bulk Doppler shift into positive or negative values meaning the target may move away or towards the radar, but the modulations remain the same. Additionally, it may move the image left and right while maintaining the target velocity, but will change its range, as seen in Fig 5.5.

Figure 5.5: Flip left-right and up-down data augmentation technique, from [83].

Translation: moves the radar image across the X or Y axis, namely over the range or velocity dimension. Thus, the target may appear at difference ranges or velocities, but the modulations remain unchanged.

Mosaic: is a popular technique that creates a new image from a combination of multiple images. Thus, it takes multiple range-Doppler plots and maintains the coherence of the micro-Doppler modulations, but adds different noise and clutter from different azimuth angles.

Hue, Saturation and Value (HSV): changes the colors and vibrancy of the plots, in the sense that the clutter, as well as the micro-Doppler and bulk Doppler may be more or less prominent. Similarly, a more desaturated image has more faded colours, corresponding to a drone that is further away, while a saturated plot resembles a drone that is close to the radar. An example is seen in Fig. 5.6

Thus, the architecture performs online augmentation via the aforementioned techniques to produce random combinations of the range-Doppler plots that are augmented in a stochastic manner. More specifically, the network doesn't see the data set itself during learning, but rather only the modified range-Doppler plots that are augmented over an uniformly sampled hyperparameters distribution that control the augmentation probabilities. This ensures that no augmented image is seen twice during training, no matter the training time [65].

[3] The COCO dataset can be found at https://cocodataset.org/#home.

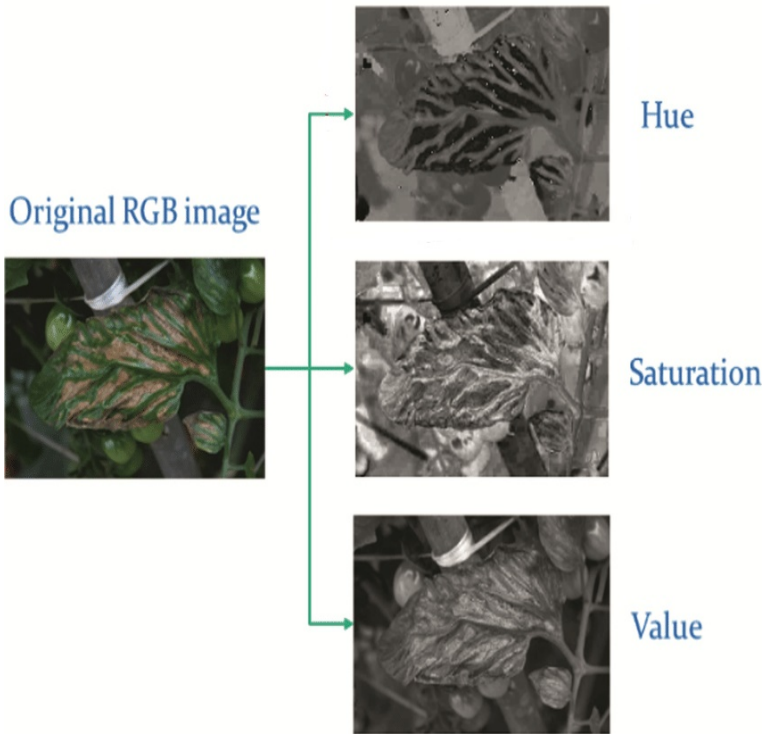


Figure 5.6: Image HSV data augmentation technique that changes the vibrancy and contrast of images, from [84].

Different data augmentation techniques exist, such as cropping, shearing, rotations, perspective transformations and others, but such techniques may completely alter the kinematics and micro-Doppler modulations of targets in ways that are not encountered in real-life situations. For example, the rotation method may flip the micro-Doppler horizontally instead of vertically, which would make the neural network learn completely wrong modulations that do not occur in real life.

Learning curves improvements via data augmentation

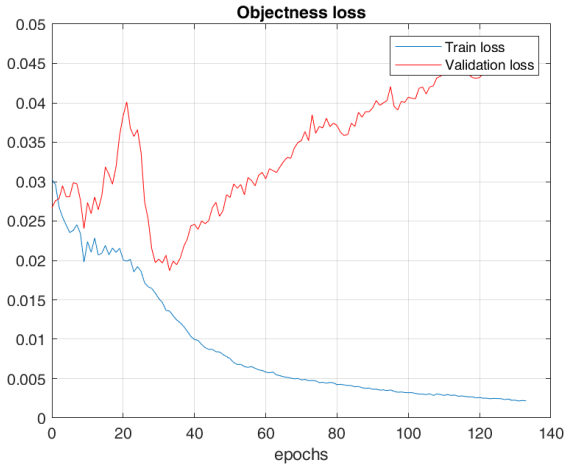
Let us look at the objectness training and validation losses for each of the proposed data sets with and without including data augmentation. These plots are important to assess whether the networks can indeed learn from the training data, and to verify if the models overfit or underfit. We can analyze the training performances of the model via the learning curves as following:

- **Underfitting**: Validation and training errors are both high;
- **Overfitting**: Validation error is high and the training error is low;
- **Good fit**: Validation error is low and slightly higher than the training error;
- **Unknown fit**: Validation error is low, but the training error is high.

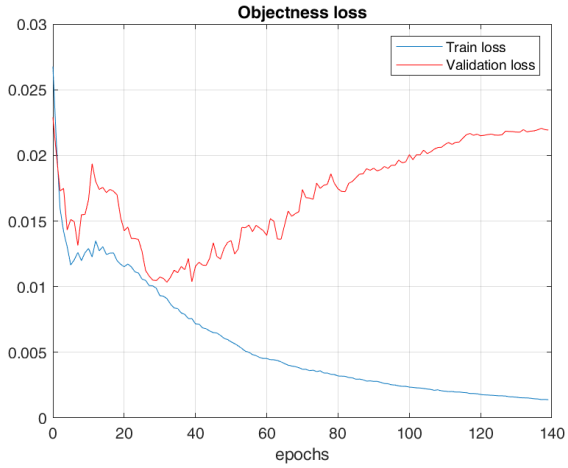
In Fig. 5.7 the effects of the complexity of the data set are highlighted, without including any data augmentation mechanisms. In all cases the models overfit over the training data as the distance between the training and validation loss is significant. However, by increasing the complexity of the data set by adding multiple clutter pictures, the overfitting can be reduced.

In Fig.5.7a, which contains the drone-only data set, the model severely overfits over the training data. By including multiple clutter images, as seen in Fig. 5.7b, the overfitting is reduced, but still present to an extensive degree. In the unbalanced data set that contains over a thousand pictures of clutter, the difference between the training and validation losses is twice smaller than in the other data sets, as

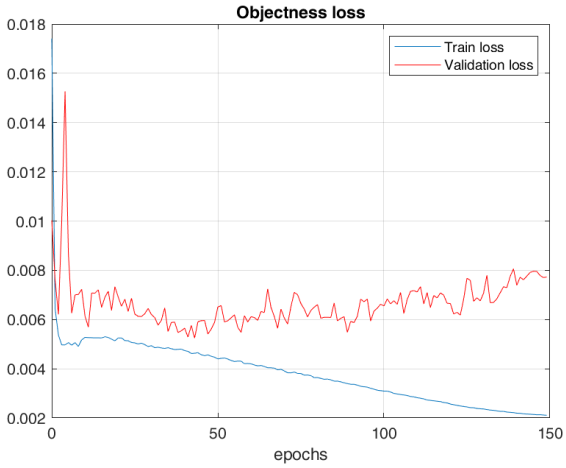
seen in Fig. 5.7c. This confirms that more complex data can indeed help the model generalize better, albeit not sufficient.



(a) Drone only data set. The difference between the train and validation losses highlight the model overfits on the training data and does not generalize well.



(b) Balanced data set with drone and clutter images. The difference between the train and validation losses highlight the model overfits on the training data and does not generalize well.



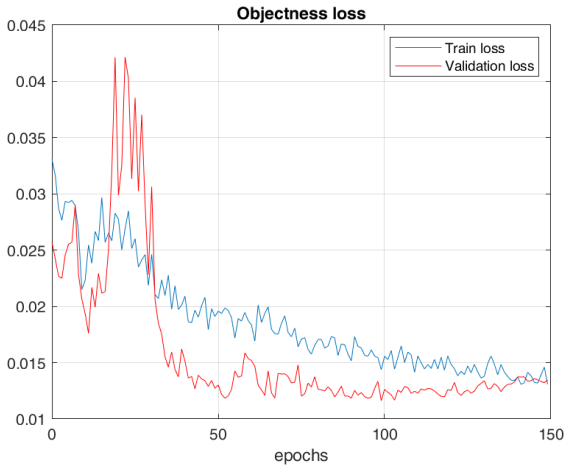
(c) Unbalanced data set with drone and clutter images. The validation loss is now closer to the train loss, however the model still overfits over the training data.

Figure 5.7: Differences between training and validation objectness losses highlighting the tendency of the model to overfit on simpler data.

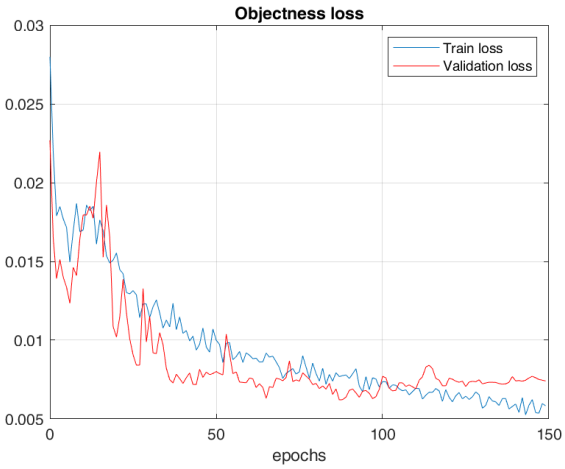
We now introduce data augmentation methods and analyze the training curves to see the improvements. In Fig. 5.8, the distance between the training and validation losses is significantly reduced, symbolizing that overfitting does not appear anymore.

Nonetheless, there are differences in all data sets. The drone-only data set from Fig. 5.8a shows that the validation loss is smaller than the train loss, which may represent that the validation test is either too simple, or not representative enough. This phenomena is similar in the unbalanced data set from Fig. 5.8c, where the validation curve is slightly smaller than the training loss. In the balanced data set however, seen in Fig. 5.8b, both curves approach the same minimum and present a good fit. However, the model tends to overfit towards the end, suggesting the number of epochs can be further decreased from 150 to 100.

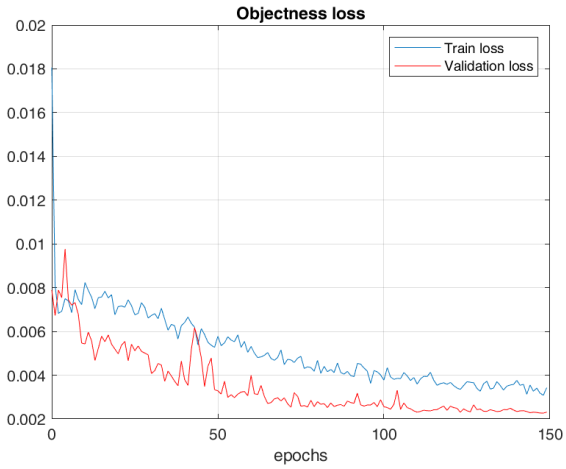
Thus, by comparing Fig. 5.7 & 5.8, it is then clear that data augmentation has a significant impact over the generalization capabilities of the model, and can successfully prevent them from overfitting over the training data and proving the are results meaningful. Moreover, the balanced data-set has the best



(a) Drone only data set. The difference between the train and validation losses are significantly reduced during training, but the validation curve is lower than the training loss.



(b) Balanced data set with drone and clutter images. Both loss functions converge towards the same minima, but the model tends to overfit if the training goes on.



(c) Unbalanced data set with drone and clutter images. With the increase of the data set complexity and the data augmentation in place, the model tends to have an unknown fit, given the unbalanced nature and extra complexity induced.

Figure 5.8: Differences between training and validation objectness losses highlighting that data augmentation can indeed prevent the model from overfitting.

overall learning performances.

Performance metrics improvements via data augmentation

Let us analyze the performance obtained in terms of precision, recall and mean average precision at different thresholds, with and without data augmentation.

In Fig. 5.9, the performances of the model without data augmentation are highlighted and summarized in Table 5.2. The model performs very poorly at identifying drones as the overall scores are very low. While the precision is relatively high for the drone-only and balanced data sets, the recall and mean average precision are dreadful, making the model redundant. Moreover, while the unbalanced data set helped reduce overfitting, its performances are unsatisfactory and substandard.

Dataset	Set	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Drone-only	Validation	75.74%	41.01%	49.78%	19.32%
Drone with clutter balanced	Validation	72.46%	35.3%	49.73%	22.98%
Drone with clutter unbalanced	Validation	25.42%	23.78%	19.65%	10.91%

Table 5.2: Table summarizing the differences of the training & validation performance metrics for each data set, without using data augmentation.

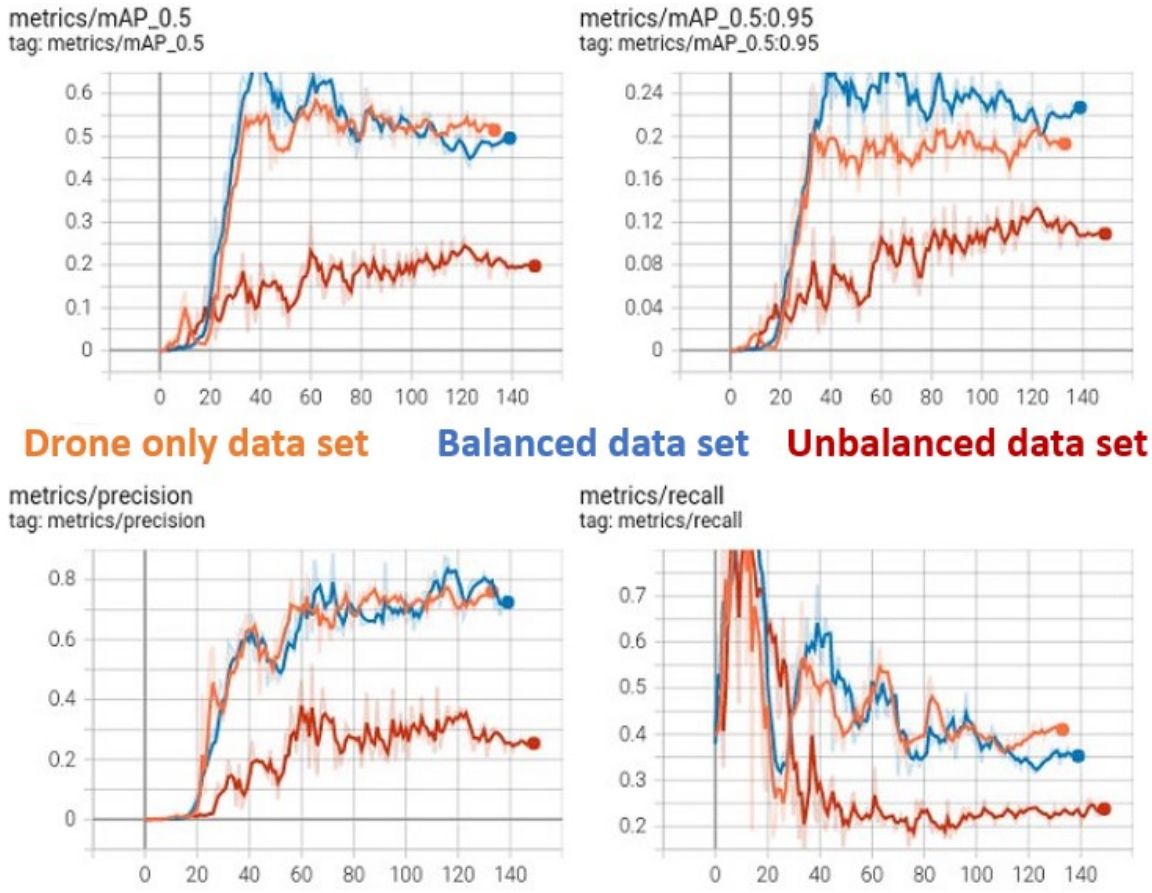


Figure 5.9: Performance metrics obtained for each of the proposed data sets, without any data augmentation techniques.

Let us now look at the improvements brought by data augmentation. In Table 5.3 the significant boost obtained in performances is summarized and the mAP at different thresholds improvement is highlighted in the last column, and further visualized in Fig. 5.10. All the models improve significantly and reach astonishing performances with over 95% mean average precision.

Dataset	Set	Precision	Recall	mAP@0.5	mAP@0.5:0.95	$\triangle$mAP@0.5:0.95
Drone-only	Validation	94%	96%	98.42%	62%	42.68%
Drone with clutter balanced	Validation	100%	89.3%	98.4%	59.6%	36.62%
Drone with clutter unbalanced	Validation	97.6%	87.9%	96%	58.2%	47.29%

Table 5.3: Table summarizing the differences of the training & validation performance metrics for each data set, including data augmentation techniques. In the last column on the right, the improvements for the mAP at multiple threshold is highlighted.

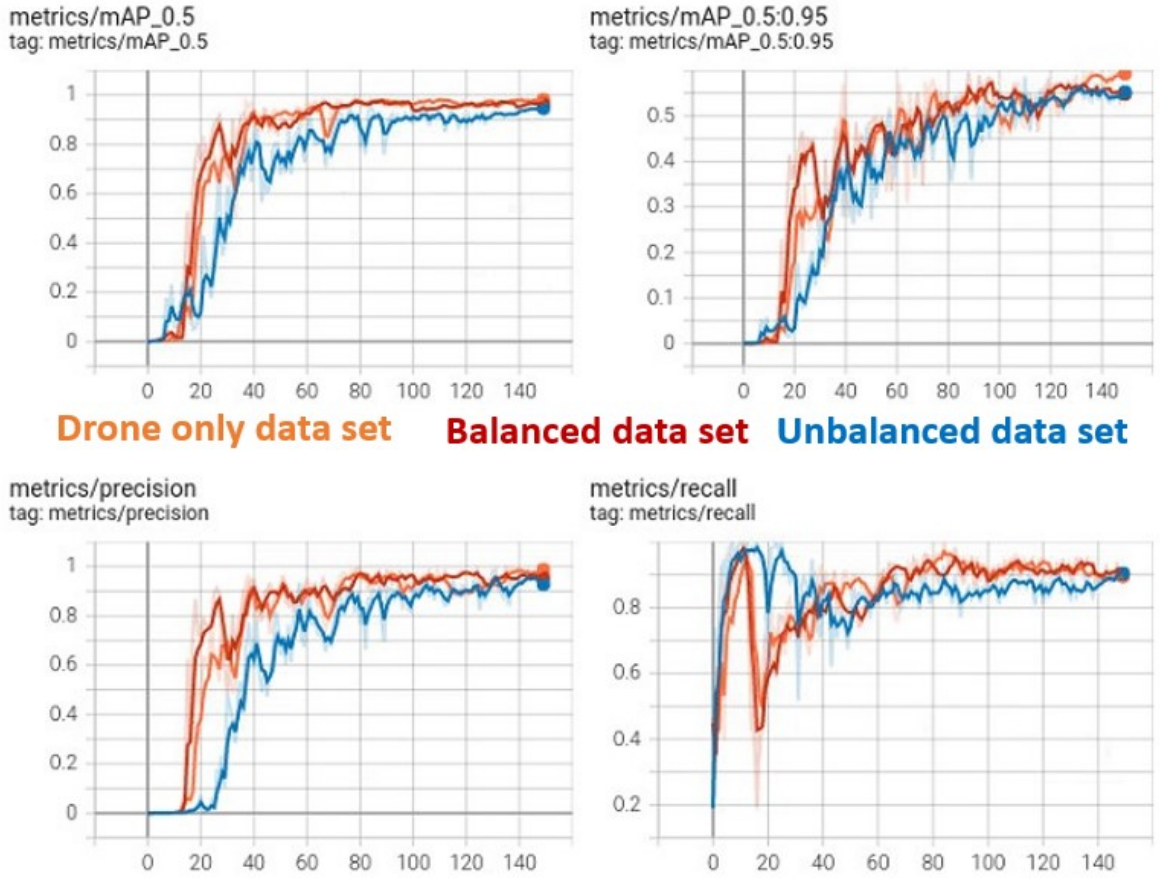


Figure 5.10: Performance metrics obtained for each of the proposed data sets, including the data augmentation techniques.

Comparing Fig. 5.9 & 5.10, it is clear that the data augmentation techniques significantly boost the performances of the pipeline. For the drone only and balanced data set, the mAP@0.5 grows from approximately 50% up to an impressive 95%, thus almost doubling the overall performances. Moreover, for the unbalanced data set that is also the most difficult, these benefits are even more pronounced as the mAP@0.5 grows almost 5 times in total.

We can then conclude that data augmentation plays a crucial role in the performances of the computer vision pipeline. Firstly, it prevents the model from overfitting and helps generalize better on unseen data. Secondly, it considerably boosts the performances of the model, with the balanced data set approaching 99% accuracy.

Moreover, we see that the drone-only, the balanced and highly unbalanced data sets can achieve similar performances thanks to data augmentation.

5.4.3. Optimizers

As discussed in Section 3.3, in order to train the neural networks a gradient is computed in order to minimize the loss function. To this end, multiple optimization algorithms exist that iteratively decrease the loss by following the gradient. These algorithms are influenced by the chosen learning rate and momentum, which are part of the YOLO hyperparameters. The learning rate decides the step size at each iteration, while the momentum is an extension of the gradient descent techniques that allow to build inertia in the direction of search space and thus bypass oscillations of noisy gradients. While there is no consensus on what are the optimal values for these hyperparameters, a high learning rate results in a very abrupt and divergent behaviors of the optimizer, while a low rate will require too many

iterations. Hence, a middle-ground is typically preferred, as seen in Fig. 5.11.

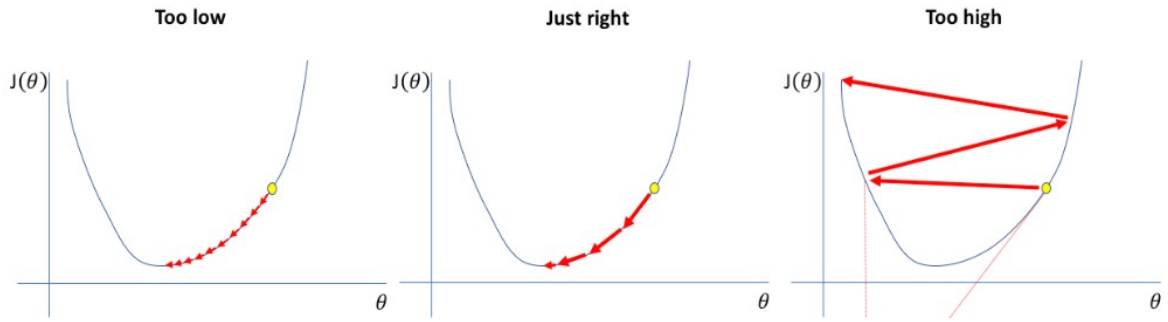


Figure 5.11: Learning rate influence over gradient descent algorithms.

Amongst the most common is the Stochastic Gradient Descent (SGD) that computes the gradient over a randomly selected subset of the data, but has a fixed learning rate. Another very popular choice that adaptively computes the learning rates at each iteration is Adam. However, the choice of the optimizer is influencing the performances of the model, as seen in Fig. 5.12.

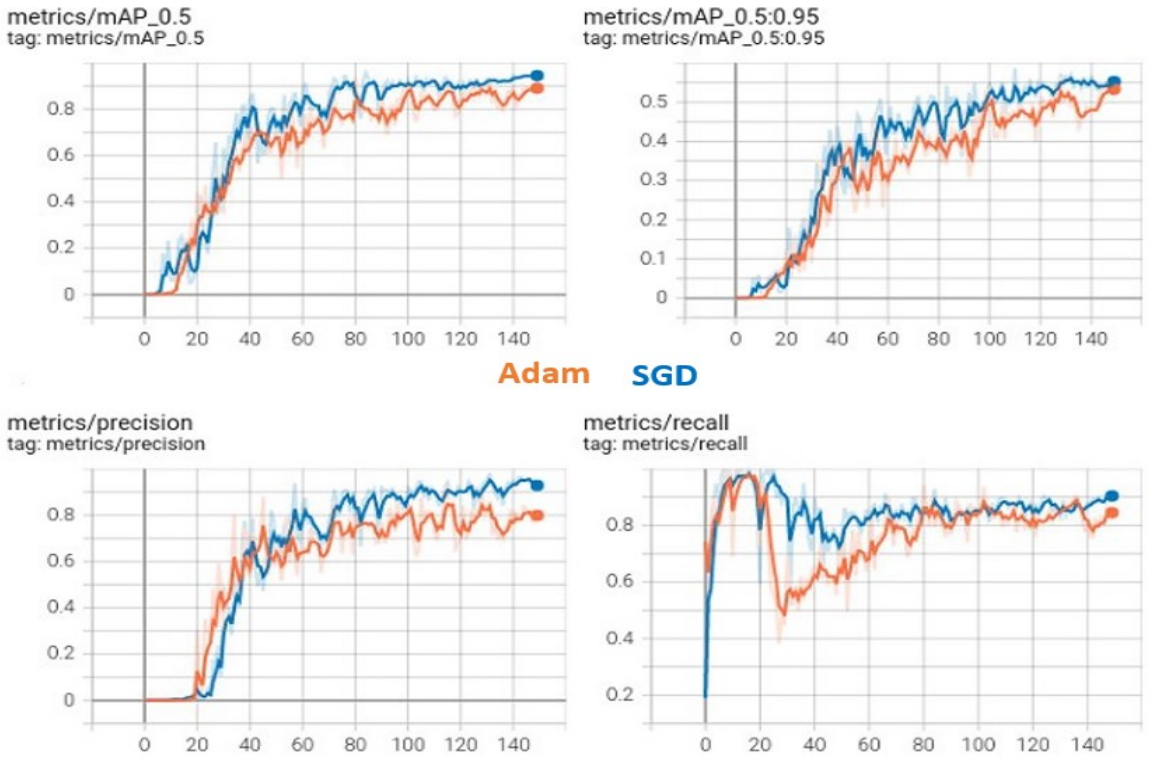


Figure 5.12: Differences in terms of performances between Adam and SGD optimizers.

In 5.12 we can see that Adam tends to perform worse than SGD for the complex & unbalanced data set. Albeit Adam being faster, SGD is more adequate as it attains higher performances overall. While the performances are relatively similar, given training times do not pose any issues due to the small size of the data set, the better performing optimizer is chosen, namely SGD. However, if training speeds would be preferred, then Adam would also be an adequate choice as the performances are relatively similar.

5.4.4. Object detection results for radar images on test sets

With the model learning and performances covered, we turn our attention towards visualizing the results. The following plots were obtained from the balanced data set which presented the best training curves and considerable overall performances.

The performances obtained over the test set are summarized below in Table 5.4. The objective of the test set is to assess the performances of the model on a data set that the model has not seen before, but which contains the ground truth. It has the same distribution as the validation set, with 116 images in total, 58 of them being drones, and the other 58 clutter.

Dataset	Set	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Drone with clutter balanced	Validation	100%	89.3%	98.4%	59.6%
Drone with clutter balanced	Test	94.5%	93.1%	95.2%	60.6%

Table 5.4: Performance results shown for the validation and test sets of the balanced data set.

In Fig. 5.13, the labels containing the ground truth are shown. It is worth nothing that the drones have different widths of the micro-Doppler, depending on their range. The narrower the micro-Doppler, the smaller the probability of the CV pipeline to catch the target.

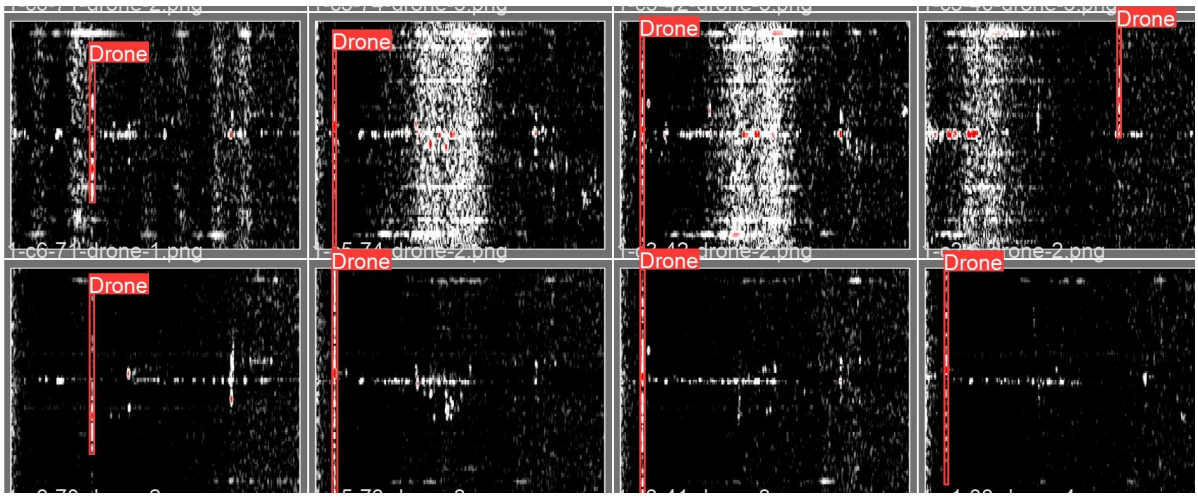


Figure 5.13: Ground truth labels of the radar images.

In Fig. 5.14 the predicted bounding boxes and labels can be seen. The object detection pipeline has shown a very good precision, but a smaller recall score. Thus, as seen in the top right image, it fails to capture the target when the micro-Doppler is too faded. Nonetheless, it does a great job at detecting and localizing the targets in most cases.

As discussed in Section 4.2.5, the pitfall of plot-based classification is not necessarily the computer vision pipeline, but the hard constraints induced by the physics of radars and low reflectivity of drones' propellers. If the target approaches 1 km, the micro-Doppler is significantly reduced and the main classification feature is lost. Moreover, the data set contained very few pictures with narrower micro-Doppler modulations, hence making their detection even more challenging for the CV pipeline. Nonetheless, albeit these downsides, the object detector makes a formidable job in accurately localizing the drones and predicting the right labels.

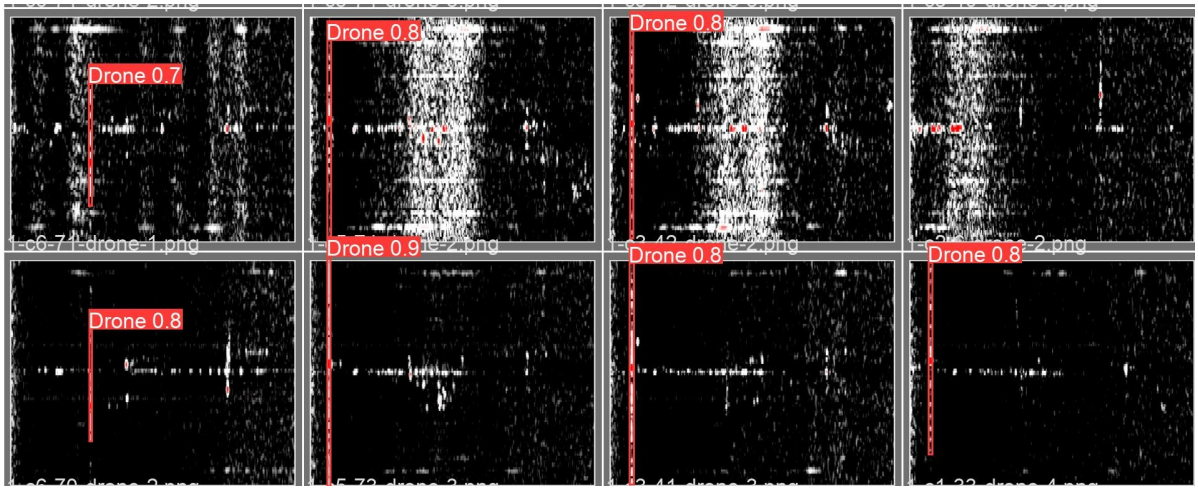


Figure 5.14: Predictions made by YOLO.

5.5. Improvements proposed to YOLO

In this section, modifications to YOLO are brought in order to tailor the architecture to be more suitable for implementation on real radar systems. It aims to:

- Introduce weights decaying to further prevent model overfitting;
- Extend object detection with state estimation for range & Doppler information extraction;

5.5.1. Weights decaying

In order to prevent the model from becoming too complex and overfit the data, a popular technique is introducing weights decaying, i.e. minimizing the weights of the neural networks during training.

One option would be to include the weights into the loss function gradient and minimize them during each epoch. However, this would not work since some weights are positive, others are negative. While one solution would be to take their square, the loss then would grow significantly and all the parameters would be set to zero.

Another way to penalize the growing complexity of the model is to subtract a percentage of the original weights at each iteration, hence the term *decay*. This constant represents another hyperparameter that can be chosen before training, however choosing the optimal value is not obvious. Choosing a decay that's too strong will block the model from learning, and thus it will not fit the data well enough. If the value is too small, then it won't have any effects.

In Table 5.5 the effects of introducing weight decaying are highlighted. We can notice that the higher the percentage of the decay, the lower the precision becomes, but the recall increases. This means that by introducing weights decay, the model is able to catch multiple detections, but the precision worsens. Nonetheless, the mAP score at multiple thresholds remains relatively the same, since it represents an average between precision and recall.

Thus, the choice of introducing weight decays enables an extra degree of freedom in fine-tunning the model, depends on what the objectives are. If the aim is to catch as many detections as possible at the price of a decreased precision, then a higher decay rate is beneficial.

Moreover, we have seen that an adequate trade-off is to introduce a decay rate of 0.75%, which only slightly decreases the precision, but compensates in the recall score, thus yielding the highest mean average precision at different threhsolds.

Dataset	Set	Weight decay	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Drone with clutter balanced	Validation	0%	100%	89.3%	98.4%	59.6%
Drone with clutter balanced	Validation	0.5%	98.1%	88.8%	97.4%	59.5%
Drone with clutter balanced	Validation	0.75%	98.2%	91.6%	98.6%	59.8%
Drone with clutter balanced	Validation	1%	91.1%	93.1%	96.6%	59.7%

Table 5.5: Training & validation performance metrics changes by introducing weight decay during training.

Let us analyze the performance metrics on the test set via using the new model, to assess how weight decaying influences the results over data that wasn't seen before. In Table 5.6, the test sets performance metrics with and without weight decaying are shown, which shows that the new model scales and generalized better on new data with the introduction of weight decaying, hence proving the improvements of the model.

Dataset	Set	Weight decay	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Drone with clutter balanced	Test	0%	94.5%	93.1%	95.2%	60.6%
Drone with clutter balanced	Test	0.75%	93%	92.7%	95.2%	63.7%

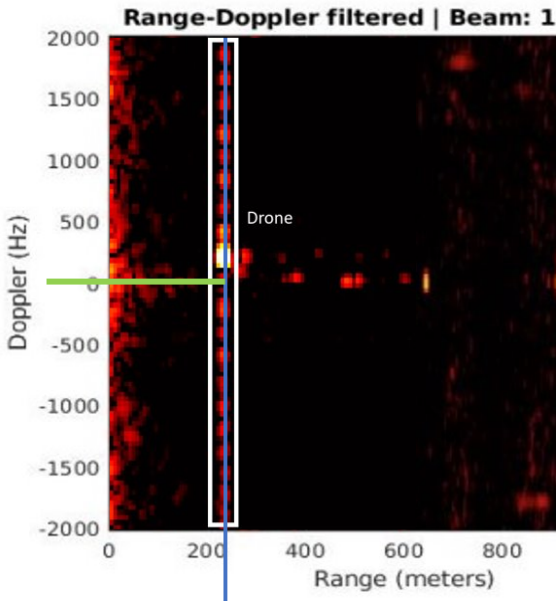
Table 5.6: Performance metrics over the test set, further confirming that weight decaying boosts the overall performances of the model over never-before-seen data.

5.6. State estimation

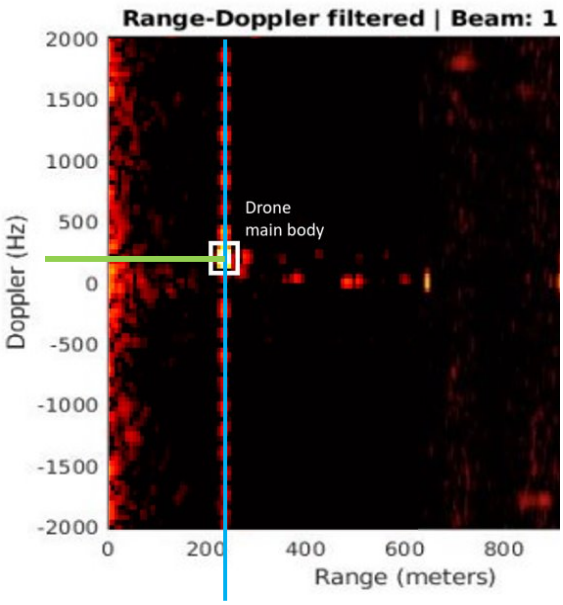
The focus of the trained pipeline was to classify drones as robustly as possible with very few images using the width of the drone's micro-Doppler modulations. Thus, using the same bounding box to estimate the range and Doppler may not accurate because it is not the objective of the pipeline and the training data was focused on the micro-Doppler width rather than the bulk Doppler.

However, range may be estimated from the center of the used bounding box, albeit not being the main objective, as seen in Fig. 5.15a. The location of the predicted bounding box is therefore a main point of interest to achieve reliable state estimation. However, the center of the box is located around the main body of the drone and the range resolution of IRIS® is 3 meters, which is significantly larger than a drone. Therefore, the bounding box is always well centered around the corresponding target range bin, which makes the range estimation accurate.

Moreover, joint range and Doppler estimation can be done in a similar framework using the same algorithms, but rather then focusing on the width of the micro-Doppler modulations, the focus should lie within the main body of the drone. Thus, in order to achieve accurate range & Doppler estimations, a square box that entails uniquely the bulk Doppler induced by the main body is far more adequate, as seen in Fig. 5.15b. Given it is a square box, the center of the said box represents the exact range of the drone, as well as the exact Doppler shift. Therefore while the estimation of the state of the drone would be indeed possible, the classification would not be possible given no micro-Doppler modulations are taken into account, as seen in Fig. 5.15. However, the objective of this thesis project is UAVs classification using low dwell-time surveillance radars and the estimation of range and velocity may represent future work.



(a) Classification focused framework used to classify drones. The center of this box is centered around the micro-Doppler modulations and may be used for range estimation, but not for Doppler.



(b) State estimation focused framework used to estimate the range and velocity of drones. The center of this box is centered around the main body solely and may be used for range and for Doppler.

Figure 5.15: Two possible predicted bounding boxes for different objectives. On the left, the predicted box is meant to classify the wide micro-Doppler of drones, while the right one aims to find the main body of the target and estimate its range & Doppler.

5.7. Summary

To conclude, the overall results are very promising and confirmed the main research objective of this thesis project. The proposed computer vision algorithms have been validated on real radar data and shown to detect & label drones with high performances using conventional static sensors.

The chosen architecture in this project is based on YOLOv5, and the network was pre-trained over 300 epochs from scratch on the COCO data set containing thousands of images to bypass small data sets constraints. Additionally, data augmentation was implemented during training to randomly generate new versions of the training plots, which showed to significantly boost the performances of the model. Moreover, improvements to the network were proposed by introducing weight decaying as a mechanism to further prevent the model from overfitting, but also to improve the performances of the model over both the validation and test sets. This induced an extra degree of freedom, allowing to improve the recall score at the cost of a reduced precision.

While radar images have lower contrasts and poorly defined contours with low sharpness, the object shapes are significantly simpler, which allows the model to obtain accuracy for drone detection based on range-Doppler plots. Therefore, with adequate pre-processing, the computer vision pipeline attained performances close to 99% mean average precision validated on real world data using a static radar.

Moreover, thanks to the localization of the objects, the range may be accurately estimated. The velocity of the main body may be as well estimated, under a different problem setting as described above. Nonetheless, these are consequences of the pipeline to enable multi-class objet detection, and do not represent the main objective of this project.

Hence, the first research objective of this thesis consisting of assessing the feasibility of computer vision pipelines for object detection & localization based on radar images has been proven a success. In the following sections, the last research objective of this thesis will be investigated, namely the assessment of this technique's abilities to achieve even more reliable UAVs classification based on clutter rich, low resolution and degraded range-Doppler plots via a completely novel scenario using radars on-the-move.

6

Radars On-The-Move and Computer Vision Performances Changes

This chapter highlights the analysis performed for a novel C-UAS framework, namely using counter-drone surveillance radars on-the-move. The changes of the problem settings between static and moving radars is presented, focusing on how the clutter and drones micro-Doppler modulations change in this scenario. Moreover, the performances of UAVs detection using the same radar and YOLO pipeline are presented, alongside the proposed improvements to boost the robustness of the CV architecture. It is shown that the proposed methods achieved precision, recall and mean average precision scores close to 100%.

6.1. Differences between static and moving radars

6.1.1. Real-time calibration requirements

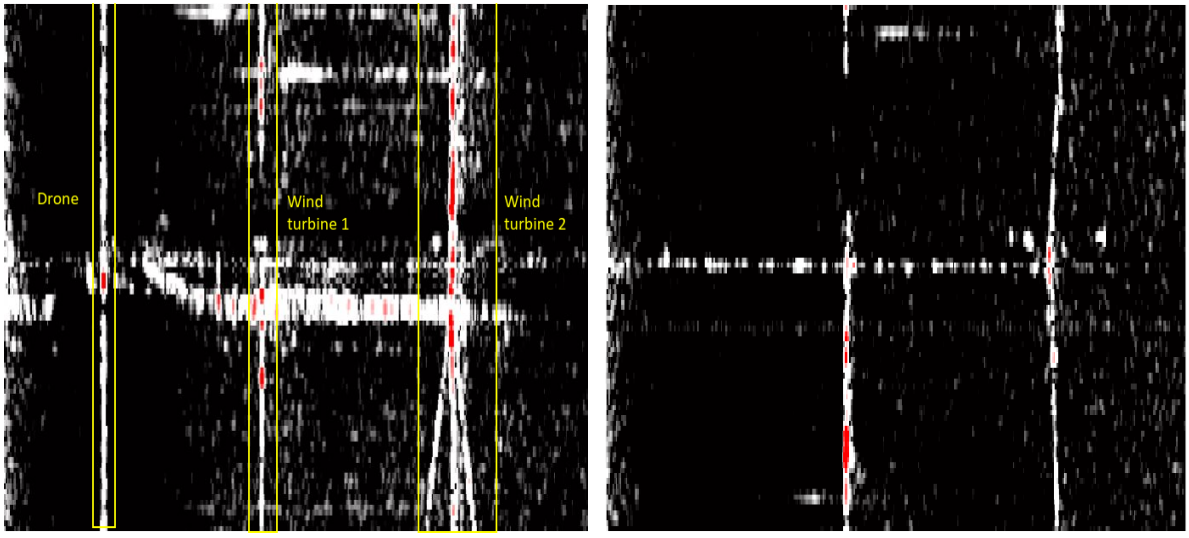
With the fundamentals of static surveillance counter-drone radars covered, and the performances of video processing algorithms evaluated, it is time to look at the final research objective of this thesis project, namely how does the drone signature, the clutter, and the object detection performances change when the radar is anchored on a moving platform. This development of a counter-drone radar on-the-move is different from many state-of-the-art work in C-UAS scenarios, where radars are traditionally fixed in a place and either rotate to cover full spatial coverage, or cover large angular areas, such as the staring holographic radar developed by Aveillant [13].

From a practical perspective, if the radar is switching between static and moving dynamic modes, the characteristics of the clutter and target signatures change over time, even more with the platform velocity changes with time. Moreover, if the radar is on-the-move and it reaches a crosswalk, it has to decelerate and perhaps even stop for a brief moment, after which it starts moving again. Hence, switching between static and OTM modes is a stochastic and common scenario. To this end, all the processing & adaptation must be done in real-time to ensure no blind zones appear due to sensor dynamics changes. Thus, the proposed data processing architecture must be robust to movement changes and ensure no calibration down-time, whether the radar is static, on-the-move, or switching between different modes.

6.1.2. Micro-Doppler changes

The emergence of novel artifacts within the range-Doppler snapshots represents one of the main differences between static and moving radars. Given the sensor is now dynamically changing its position, new objects continuously enter the field of view and previous faraway targets that yielded weak reflections are now appearing within the radar images. More specifically, due to the location where the data set was recorded, wind-turbines are now a common target. Moreover, depending on their orientation, the fast spinning blades may show Doppler modulations that are extended across all the Doppler frequencies, similar to drones.

In Fig. 6.1, two typical range-Doppler plots taken in the same location that were used for detection are seen, highlighting the differences between moving and static radars. The drone is clearly seen on the left, with the bulk Doppler shift induced by the main body represented by the red dot as in the static radar case, and the micro-Doppler modulations induced by the propellers appearing as a continuous vertical line. However, in this scenario the drones may be surrounded by wind-turbines. Depending on the orientation of the wind-turbine blades, all three blades may be visible, as seen on the right, or only one blade that resembles the modulations of a drone, as seen in the middle.



(a) Range-Doppler plot obtained with a radar moving at approximately 20 km/h, highlighting the possible targets present in a single picture. The drone is clearly seen on the left, located near two wind-turbines with the blades rotating in different orientations.

(b) Range-Doppler plot obtained with a static radar. The two wind-turbines are visible, with different orientation angle. The dynamic clutter projected with a specific Doppler shift correlated to the moving platform velocity is seen as static.

Figure 6.1: Range-Doppler plots used for object detection and recorded in a similar position, highlighting the difference between static and moving images obtained with the same sensor.

The drone modulations are therefore similar for both static and moving sensors. Nonetheless, the probability that wind-turbines appear within the recorded data set is significantly increased. The problem setting remains a binary detection decision, i.e. is there a drone in the image or not, but the probability of detecting false alarms is increased due to a more challenging environment and passing near wind-turbines that may be mistaken for drones. In Subsection 6.2.3, this issue is further analyzed.

6.1.3. Clutter changes

With the bulk and micro-Doppler modulations covered, let us turn our attention towards the clutter. Fig. 6.2 highlights an unfiltered range-Doppler plot obtained with Robin's IRIS® On-The-Move Radar.

The most important difference is the emergence of the dynamic clutter, shifted from 0 Hz to approximately 800 Hz in this case. This shift is correlated to the velocity of the moving platform and azimuth angle, and are continuously changing. However, both these values are known a-priori and can be incorporated within the clutter filtering methods, as discussed in Subsection 6.1.4.

Moreover, a weak static clutter centered around 0 Hz is still present. These reminiscences come from the car on which the sensor is anchored and is seen as static, and the ground clutter reflected from backlobes that are facing the opposite direction of movement and therefore the velocities cancel each other and yield a 0 Doppler shift.

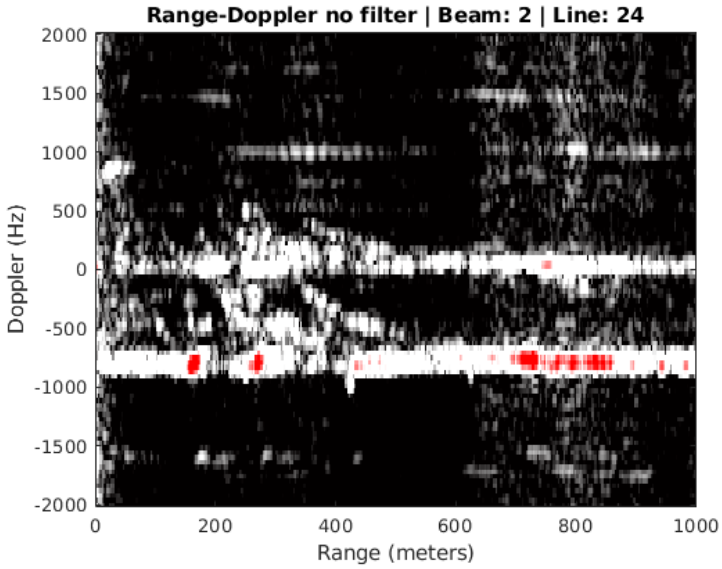


Figure 6.2: Unfiltered range-Doppler plot showing the static clutter at 0 Hz, and the new dynamic clutter centered around 900 Hz which is induced by the moving radar and correlated with the azimuth angle and velocity of the platform which runs at approximately 50 km/h.

6.1.4. Adaptive processing for clutter reduction

Given the extra dynamic clutter would not be entirely suppressed by the previous filters, and that it is changing over time depending on the velocity of the platform, a new adaptive filter is introduced.

Following the same non-statistical framework presented in Chapter 4, the proposed solution encapsulates a cut-off filter that is dynamically changing the notch frequency. Given the velocity of the platform is known a-priori, we incorporate this information within an adaptive filter that aims to suppress the clutter at the specific Doppler frequencies. The notch filter follows an Autoregresive (AR) Moving Average (MA) (ARMA) model [85], that can be written as following:

$$H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^{q} b(k) z^{-k}}{1 + \sum_{k=0}^{p} a(k) z^{-k}} \tag{6.1}$$

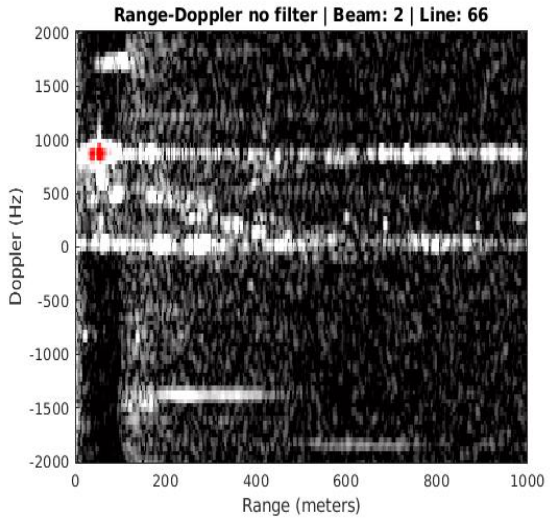
To estimate the ARMA filter coefficients, the platform velocity is translated into Doppler frequency in order to control the cut-off notch frequency. Moreover, given the antenna is continuously rotating, the radial velocity is taken into account, and the azimuth angle is included, as following:

$$f_d = 2\frac{v}{\lambda} cos(\varphi) \tag{6.2}$$

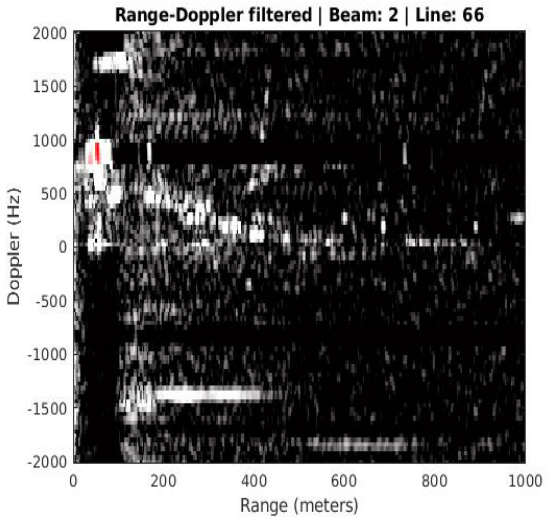
where v is the platform velocity, λ is the radar wavelength, and φ is the azimuth angle.

With the radar velocity transformed into Doppler frequency, the AR(p) poles and the MA(q) zeroes are computed, which yields the final ARMA model. In the static radar case, the pulse canceller used can be modeled as a simple MA filter since it contains only zeros and the AR poles are null, which yields a high-pass filter.

In the moving sensor scenario however, the chosen ARMA filter model uses both poles and zeroes that allow to calibrate the notch cut-off frequency on the fly. A frequency response of the adaptive filter is seen in Fig. 6.3a, that aims to suppress the clutter centered around a specific velocity. Following the proposed framework, the notch will be shifted at different frequencies according to the platform velocity and azimuth angle. The original unfiltered image is seen above in Fig. 6.2, and the final filtered output is seen in Fig. 6.3b, which shows how most of the clutter and background noise are successfully suppressed.



(a) Unfiltered range-Doppler plot showing the static clutter at 0 Hz, and the new dynamic clutter at 900 Hz induced by the moving platform and correlated with the velocity of the radar and azimuth angle.



(b) Filtered range-Doppler plot showing the filtered output, where the static and dynamic clutter, alongside the background noise were successfully suppressed.

Figure 6.3: Adaptive notch filter and filtered Range-Doppler plot examples.

It is worth emphasizing that the processing was done offline based on test scenarios defined a-priori. This means that the platform is assumed to have a specific velocity for the entire test, but in reality it is influenced by acceleration and deceleration. To bypass this error, the notch of the filter was widened, i.e. if the notch frequency -3 dB cut-off frequency is widened by an additional ± 40 Hz around the notch frequency. This allowed to account for platform velocity measurement error of approximately ± 5 km/h. However, in a real system where the real platform velocity can be extracted from a sensor in real-time, the width of the notch can be further reduced.

It is worth noting that this filter does not influence the previous clutter reduction methods. If the radar is static, then the pulse canceller and notch filter will overlap. When the radar begins moving, the velocity is transformed into the cut-off notch frequency in real-time and the notch is shifted accordingly.

Thus, the proposed processing chain works in a variety of scenarios and does not require any down-time to scan & adapt to the environment when switching between static and OTM modes. This allows the radar to continuously process the sweeps, filter the output range-Doppler images and detect drones in real-time and without any changes in the signal processing chain, whether the radar is static or moving.

6.2. Radar on-the-move object detection performances

6.2.1. On-the-move data sets

The new data set was taken in the same location as in the static scenario. Moreover, it contains equal ratios of labeled drone range-Doppler plots and unlabeled clutter plots as described in Table 6.1, split as following:

Thus, a similar amount of data as in the static radar case from Chapter 5 was used. However, as seen

Data	Training	Validation	Test
Number of drone images	188	62	62
Number of total images	376	124	124
Ratio	60%	20%	20%

Table 6.1: New data set highlighting the amount of drone & clutter plots, where plots or images refer to range-Doppler snapshots treated as images.

in Fig. 6.1 & 6.2, in the moving radar the complexity of the data set is significantly higher. This is due to the added artifacts induced by the moving platform, such as the radar passing in front of multiple wind-turbines that are rotating at different angles, as well as the extra dynamic clutter.

Additionally, the training and validation data sets were taken during winter, in December, and for two completely different test scenarios (i.e. different radar and drone speeds, and different direction of motion). The test set on the other hand was taken during summer, in June. This results in different clutter characteristics, as in December the ground was frozen and therefore more reflective but the vegetation was more scarce, while in June the tree line was fully blossomed and there were more birds flying around. These differences between the data sets ensure the following benefits:

- No correlation between the training, validation and test sets. This aspect is important to ensure the object detector pipeline did not see any information during training that is almost identical (or strongly correlated) with the validation or test sets.
- Natural data diversity & artifacts due to different meteorological conditions which can not be entirely synthetically modelled. For example, during winter the tree line is less dense and there are no leaves blowing, the canals are more visible, the ground is frozen and therefore more reflective. During summer, there are many other targets such as cyclists, horses, or birds, but also a denser vegetation.

6.2.2. Performances metrics specific for radars on-the-move

Let us analyze the performance metrics obtained after training a model specific for the new data set, with and without weights decaying, and using both SGD and Adam as optimizers. In Table 6.2 the obtained performance results are highlighted, both for training and test data sets.

Model	Set	Weights decay	Optimizer	Precision	Recall	mAP@0.5	mAP@0.5:0.95
1	Validation	0%	SGD	98.4%	99.5%	99.5%	75.1%
1	Test	0%	SGD	99.8%	100%	99.5%	74.3%
2	Validation	0.75%	SGD	98.3%	98.4%	99.4%	75.2%
2	Test	0.75%	SGD	99.9%	98.4%	99.4%	76.6%
3	Validation	0%	Adam	100%	98%	99.5%	75.8%
3	Test	0%	Adam	99.6%	98.4%	99.5%	76.2%
4	Validation	0.75%	Adam	99.9%	98.4%	99.5%	75.2%
4	Test	0.75%	Adam	99.8%	100%	99.5%	74.9%

Table 6.2: Performance metrics for the validation and test OTM radar data sets, and for SGD and Adam.

The overall results are very promising. The high precision shows that the predictions are accurate, while the high recall score indicates that the number of total predictions with respect to the ground truth is high. The mAP over multiple IoU thresholds avoids the ambiguity of picking an optimal IoU for evaluating the accuracy of the model, while the one set at 50% is a common choice in CV [86].

Moreover, the results between the validation and test sets are similar, which show the object detector does not overfit over the training data set and scales well over never before seen data.

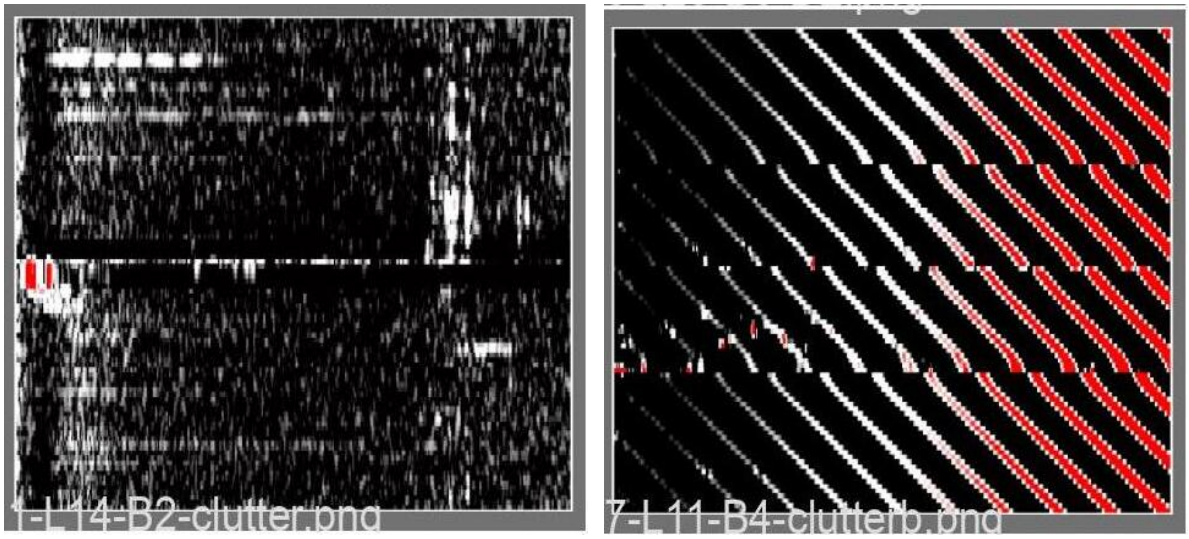
The performances for the moving radar data set are slightly higher than the static radar case, especially in the mAP at multiple thresholds. This is thanks to a more diverse data set for the moving radar case, i.e. data sets taken in different times of the year. However, comparing two different models built on different data sets and for different objectives remains challenging.

6.2.3. Object detection results for radars on-the-move

Let us visualize the results of the performance metrics and predicted detections to better understand the outcomes and differences between both optimizers. The following examples are shown from the test and validation sets using the models trained over 150 epochs with 0.75% weights decaying.

Ground truths

In Fig. 6.4 the ground truths of a clutter-only range-Doppler plots from the test and validation sets are. Fig. 6.4a contains a noise only picture from the test set, while Fig. 6.4b shows interferences induced by nearby radars that are either using pulse compression, or slower modulations that match the phase. This phenomena is more common with moving radars since they may randomly enter the range of other sensors.

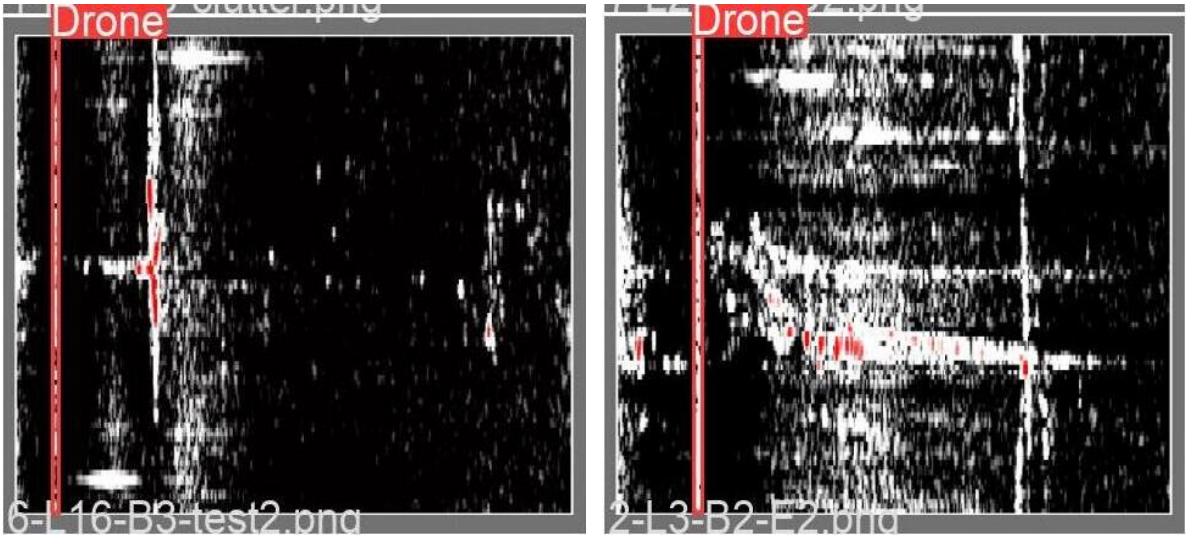


(a) Ground truth sample containing only clutter, taken from the test set.

(b) Ground truth sample containing only clutter, taken from the validation set.

Figure 6.4: Clutter only image samples taken from the validation and test sets.

Fig. 6.5 shows the two examples of drone labels, juxtaposed by two wind-turbines that may represent the main source of false alarms due to similar wide micro-Doppler modulations. In Fig. 6.5a shows a sample taken from the test set with a labeled drone, located near a reflective wind-turbine. Fig. 6.5b on the other hand is a sample from the validation set which also contains the labeled drone, but the wind-turbine is located far away this time, reducing the reflected energy and yielding a similar target profile with those of drones.



(a) Ground truth sample containing a drone near a wind-turbine which may induce false alarms, taken from the test set.

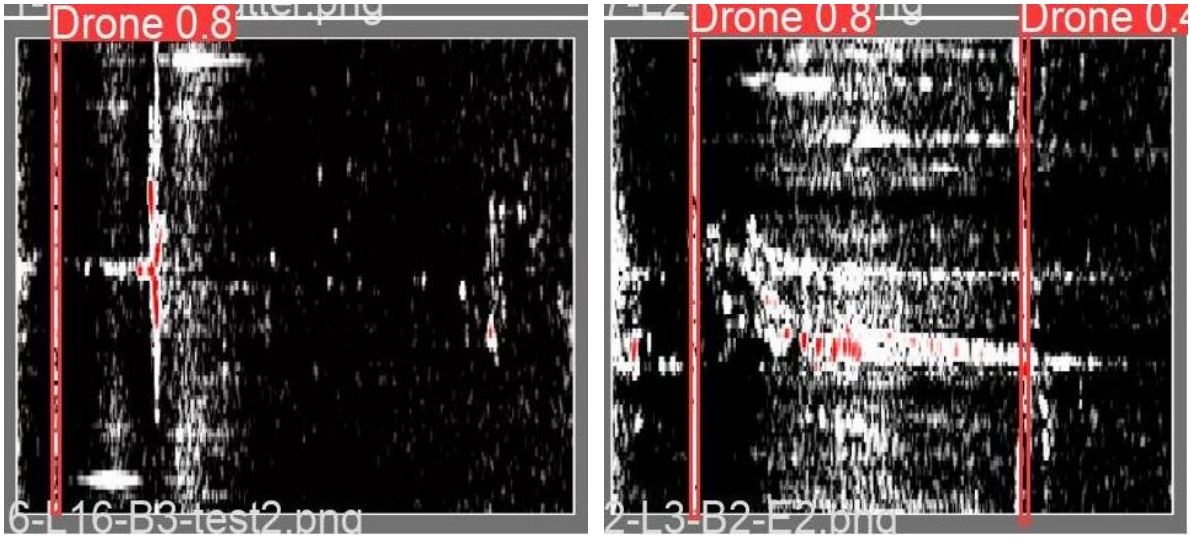
(b) Ground truth sample containing a drone near a wind-turbine which may induce false alarms, taken from the validation set.

Figure 6.5: Drone image that contain possible false alarms, samples taken from the validation and test sets.

SGD optimizer

In Fig. 6.6, the predicted bounding boxes made by YOLO using SGD as optimizer are highlighted. The model successfully predicts the accurate location of all drones and labels them accordingly. Moreover, while the wind-turbine from the test set seen in Fig. 6.7a that has similar Doppler modulations does not yield any false alarms.

While initially the results seem promising, Fig. 6.7b highlights the predictions made during the learning process. Albeit the overall good performances of SGD, the far-away wind-turbine that is seen as a weaker target yields false alarms.



(a) Predicted bounding box and label done over the test set. No false alarms were generated.

(b) Predicted bounding box and label done over the validation set. The far-away wind-turbine yields false alarms.

Figure 6.6: Predictions made by YOLO with SGD over the test and validation data sets, highlighting the emergence of false alarms induced by wind-turbines.

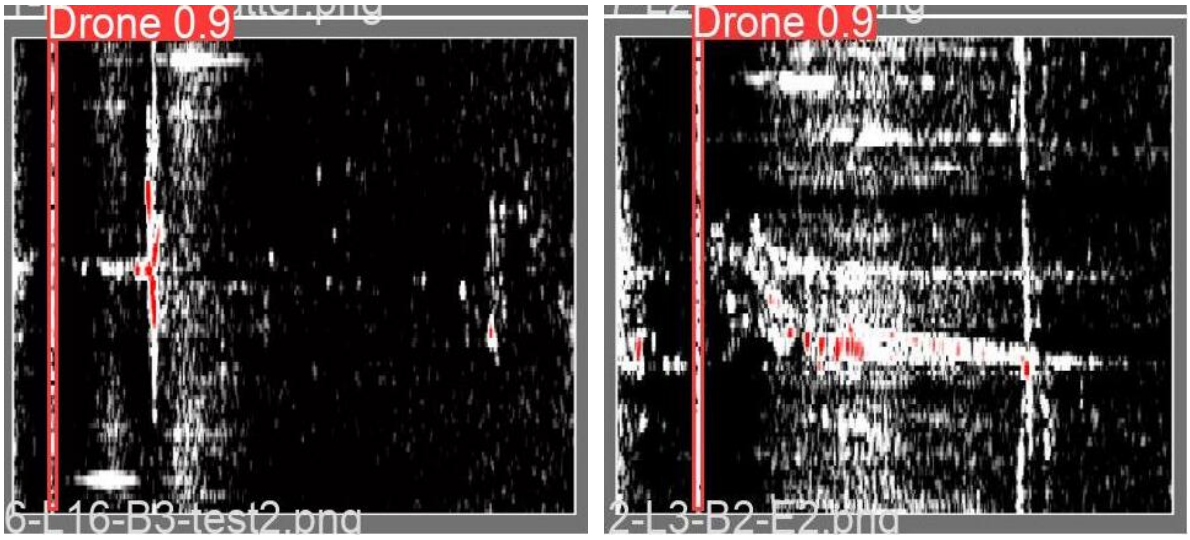
Thus, during training the detectors wrongly predicts extra bounding boxes over modulations induced by wind-turbines. While this problem could be solved with an *if* loop to check whether the target is present

in all the beams (given wind-turbines are extended targets in elevation), it may also be solved during training with a more adequate optimizer.

Adam optimizer

Adam showed a slightly higher precision and recall, and generally tends to be amongst the most popular optimizers. Moreover, it is faster to train and the adaptive nature allows to control extra hyperparameters, such as changing the learning rate during training, the warm-up epochs or the momentum. The downside of adaptive optimizers however is that they are more difficult to initialize due to the increased number of hyperparameters.

In Fig. 6.7, we see that the false alarms completely disappear by changing the optimizer from SGD to Adam, alongside an improved confidence score for each assigned label. This phenomena is further observed across multiple training batches, and also reflected in the improved performance scores that reach higher precision than SGD, as seen in Table 6.2.



(a) Predicted bounding box and label done over the test set. No false alarms were generated.

(b) Predicted bounding box and label done over the validation set. No more false alarms are generated with Adam.

Figure 6.7: Predictions made by YOLO with Adam over the test and validation data sets, highlighting the emergence of false alarms induced by wind-turbines.

It is worth noting that initially SGD was slightly better in the static radar case, while Adam is performs better in the moving radar scenario. This difference further highlights that there is no single optimizer that performs the best in every single case, but rather multiple choices exist that should be tested, and the best one chosen for the specific application or data set.

6.3. YOLO improvements

6.3.1. Genetic algorithms for hyperparameters tuning

In the aforementioned sections, we have seen that the choice of the optimizer and hyperparameters is of uttermost importance for the success of the model, but finding optimal values can be challenging. While the choice of these values come from experience and empirical testing, there are methods to generalize this choice via different kind of algorithms, such as random or grid search, different libraries such as Optuna, evolutionary algorithms, Bayesian optimizations and more.

However, for YOLO there are approximately 30 hyperparameters that need to be initialized. Thus, traditional methods like grid search can lead to high dimensional search spaces and computational

complexity due to assessing the fitness of the model at each point. The creators of YOLO suggested genetic algorithms as a viable solution. It is worth nothing however that genetic algorithms are also very computationally expensive and require hundreds of generations to yield adequate results [65]. However, it was shown that tuning hyperparameters via genetic algorithms can improve the model performances by approximately $10 - 15\%$ [87].

Thus, the chosen technique is based on genetic algorithms. These allow to test a plethora of different hyperparmeter combinations, and the model is trained for hundreds of times for each of the offsprings. The best combination of hyperparameters is chosen based on fitness function that is dependant on performance metrics and which defines the best model choice, as described in Subsection 5.3.3.

A mutation for the proposed data set takes approximately 5 minutes, and a total of 600 mutations were used. Thus, to obtain the new hyperparameters by using genetic algorithms, it took approximately 2 days to obtain the results. Hyperparameters such as the learning rate, the loss functions gain, data augmentation parameters, and weight decaying percentage were changed to find the most optimal values. It is worth emphasizing that with more mutations, better combinations could be discovered. However, given the computational complexity, a cap of 600 generations was chosen. As seen in the following subsections, this amount of mutations has shown to improve performances and remove false alarms. Nonetheless, more generations could be further introduced to find even better tailored hyperparameters values, but it is beyond the scope of this thesis project due to very high GPU load and time required.

6.3.2. Final performance metrics

In Table 6.3 the performance metrics are shown for both SGD and Adam for the validation and test sets, and include the results of the models initialized with hyperparameters chosen manually and via evolutionary algorithms.

Set	Hyperparameters tuning	Optimizer	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Validation	Hand picked	SGD	98.3%	98.4%	99.4%	75.2%
Test	Hand picked	SGD	99.9%	98.4%	99.4%	76.6%
Validation	Genetic algorithm	SGD	98.2%	100%	99.5%	75.7%
Test	Genetic algorithm	SGD	100%	99.7%	99.5%	75.1%
Validation	Hand picked	Adam	100%	98%	99.5%	75.8%
Test	Hand picked	Adam	99.6%	98.4%	99.5%	76.2%
Validation	Genetic algorithm	Adam	100%	98.2%	99.5%	76.5%
Test	Genetic algorithm	Adam	99.9%	100%	99.5%	77.3%

Table 6.3: Performances differences between hand picked hyperparameters, and those obtained via genetic algorithms.

For the SGD, the overall performance metrics improved for both the validation and test set. However, the mAP at multiple thresholds slightly decreased for the test set. This may be due to lower weights decaying percentage, that the genetic algorithms found to be better suited during training, but slightly decreased the ability of the model to generalize better. However, this slight decrease is a better trade-off compared to false alarm generation.

For Adam, all the performance metrics improved for both validation and test sets. Given Adam is an adaptive optimizer, it requires four times more hyperparameters to be initialized compared to SGD, but can also be more tailored for the desired application. Thus, the genetic algorithm tried multiple combinations and chose the best hyperparameters for this very specific data set, which allowed to obtain the best results via adequate initialization.

Moreover, these results could be further fine-tuned by running the genetic algorithms for even longer

periods of time. To obtain Table 6.3, approximately one week worth of computations was required. This served to assess both optimizers, which showed Adam to be a better fit with the adequate initialization. Further hyperparameters fine-tuning could be done for thousands of generations rather than hundreds, which would allow to test even more possible combinations of the hyperparameters and choose an even better one. However, this would require hundreds or even thousands of GPU hours.

6.3.3. Optimizers and false alarms suppression

In Subsection 5.4.3, we showed that SGD tends to perform slightly better for the static radar data set, while Adam tends to perform better on the more difficult moving radar data set, as seen in the Subsection 6.2.3.

This performance difference may be due to a wrong initiation of hyperparameters, since Adam requires setting the initial and final learning rates, the momentum, warm-up epochs and warm-up momentum. While these values were initialized based on the recommendations of the YOLO creators [65], they were optimized for the COCO data set containing images obtained with optical cameras. Thus, the genetic algorithms aim to choose specifically tailored hyperparameters for data sets containing radar images rather than optical cameras.

Moreover, the main error was the emergence of false detections that appeared by using SGD, but were successfully removed via Adam. Thus, we investigate the results obtained by fine-tuning the hyperparameters for both optimizers.

SGD optimizer

Compared to Fig. 6.6 that used the model with hand-picked hyperparameters, in Fig. 6.8 we can see that the false alarms were successfully suppressed by using genetic algorithms to fine-tune the hyperparameter initialization. This further highlights what a major role the initialization of the hyperparameters plays in the success of the model. While hand picked values can achieve over 99% performances, by fine-tuning these values for the desired application, false detections can be successfully suppressed since as early as the learning stages.

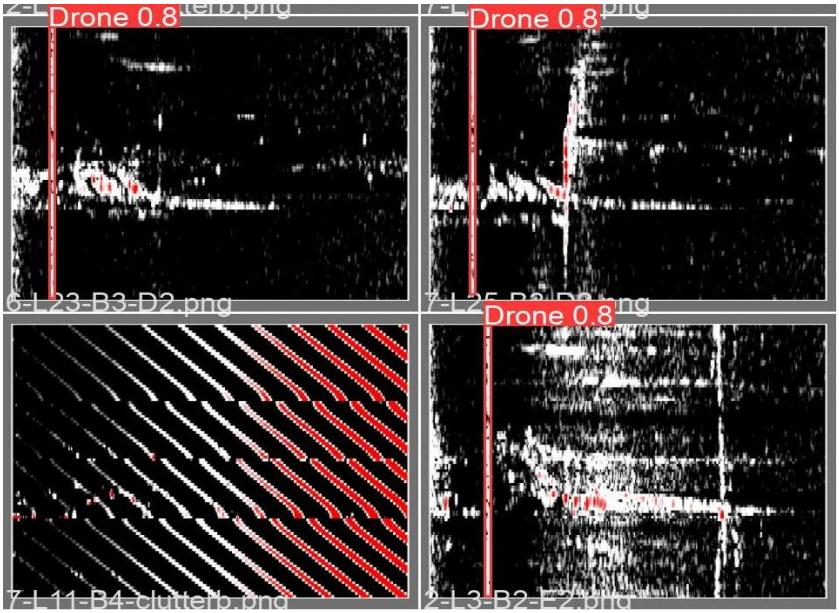


Figure 6.8: Predictions made by YOLO over the test set using genetic algorithms and SGD as optimizer.

Adam optimizer

In Fig. 6.9, the model trained with Adam makes the same accurate predictions using genetic algorithms as it did with hand-picked hyperparameters. However, compared to SGD, the confidence score for each label grows from 80% to 90%.

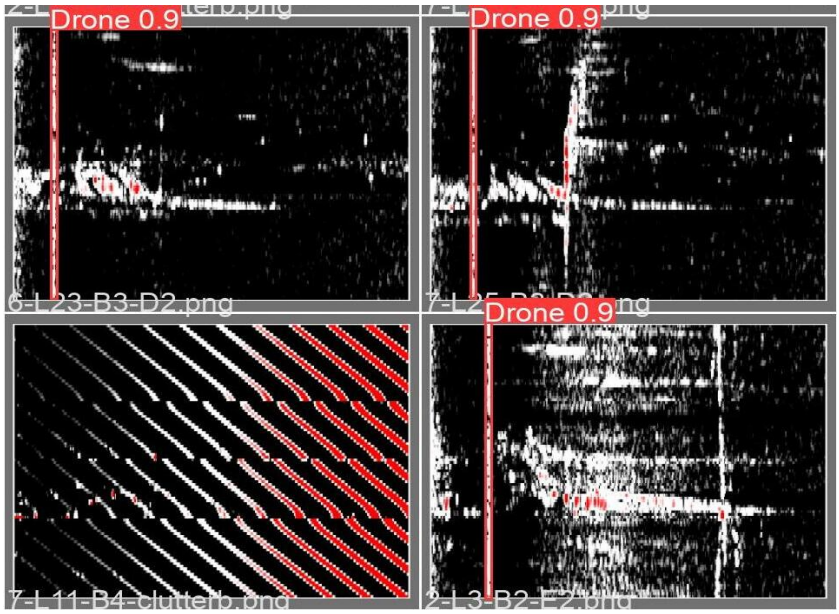


Figure 6.9: Predictions made by YOLO over the test set using genetic algorithms and Adam as optimizer.

We can then conclude that using genetic algorithms can successfully improve the robustness and accuracy on the model, and facilitate the minimization of false detections. However, it took approximately a week to obtain these results due to the very computational heavy nature of evolutionary algorithms which require hundreds of generations to yield adequate estimations. The above results may be further improved if the number of generations is increased from 600 to thousands, as the algorithm will try multiple possible combinations of the hyperparameters.

6.4. Performance comparison between radars and optical cameras

YOLO was initially created for optical cameras that use a significantly shorter wavelength compared to radars. This wavelength difference yields a much finer resolution and sharper contour of objects for high-resolution optical cameras, which yields high-fidelity models that greatly facilitate the classification task. However, if the environmental conditions are unfavorable, such as rain, fog, snow, or any other phenomena that reduces the visibility, reliable classification is not possible anymore. This is one of the main vulnerabilities of cameras, whereas radars are robust against poor environmental or atmospherical conditions, glaring effects or low-light. Radars moreover function at longer distances, and can extract the kinematic information of the targets such as their range, velocity, and angle of arrival.

The purpose of the comparison between the two sensors is that to the best of the authors' knowledge, there is no comparable algorithm or paper that investigates the performances of object detection using surveillance drone radars. Thus, an optical data set was taken because the chosen computer vision technique was specifically developed for this type of sensor. While the comparison is not entirely relevant due to the inherently different data structure of the outputs between the two sensors, it serves as a starting point for comparing performances of YOLO for its initial design purposes, and assess how well it performs on other types of sensors.

Moreover, radar range-Doppler plots contain simpler shapes of the objects and have a lower resolution, which allowed for the chosen YOLO model to be amongst the smallest, taking only 14 MB of memory.

The camera used to obtain the drone images is Sony FCB-EV7500 Full HD colour camera, seen in Fig. 6.10. It has a high-quality 1080 by 1080 pixels resolution, records at 60 frames per second, and has optical zoom of up to 30 times.

Figure 6.10: Sony optical camera used for drone and bird classification.

In Fig. 6.11 two drones images can be seen. In the case where the drone is faraway, Fig. 6.11a, the drone is only seen as a dot within the image. However, when the UAV is close, the main-body of the drone alongside the propellers can be easily seen, as shown in Fig. 6.11b. These images were taken at the very same location as the previous data sets obtained with radars.

(a) Faraway drone that is seen as a single point, with a flock of birds flying in front of the camera.

(b) Close drone flying near a wind-turbine.

Figure 6.11: Optical images captured with the optical camera.

6.4.1. Performance metrics

Let us train a model based on the optical camera data set using the same training parameters that yielded the best results, i.e. 150 epochs and Adam as optimizer. The performance metrics comparison is summarized in Table 6.4, alongside the difference in performances for the test sets between the two types of sensors.

It is important to emphasize the hyperparameters fine-tuning choice. The handpicked values were initially chosen for optical images, and showed that they perform well on radar plots as well. However, the genetic algorithms were run for hundreds of generations over the radar data set, and the hyperparameters initialization is therefore fine-tuned for this very specific data set. Further fine-tuning via evolutionary algorithms could be done specifically for drone detection via optical cameras, which would

Sensor	Set	Hyperparameters	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Radar	Validation	Handpicked	100%	98%	99.5%	75.8%
Radar	Test	Handpicked	99.6%	98.4%	99.4%	76.2%
Camera	Validation	Handpicked	89.6%	88.4%	93.2%	54.1%
Camera	Test	Handpicked	92.6%	89.1%	92.2%	46%
Radar	Validation	Genetic algorithms	100%	98.2%	99.5%	76.5%
Radar	Test	Genetic algorithms	99.9%	100%	99.5%	77.3%
Camera	Validation	Genetic algorithms	95.7%	92.3%	96.2%	49.2%
Camera	Test	Genetic algorithms	86%	79.8%	82%	35.3%
Radar vs camera difference	Test	Handpicked	7%	9.3%	7.2%	30.2%
Radar vs camera difference	Test	Genetic algorithms	13.9%	20.2%	17.5%	42%

Table 6.4: Performance metrics difference between the radar and camera, obtained for the validation and test sets.

in turn improve the performances of the model. However, the high-resolution nature of optical cameras yields very large images, which in turn requires a significantly larger amount of time for training. Thus, running evolutionary algorithms would take weeks, and hyperparameters fine-tuning for optical images is beyond the scope of this thesis project.

Beyond the learning and fine-tuning process, the choice of the model is again relevant for this application. The model chosen for radar data set is amongst the smallest, given radar images have a low-resolution and the shapes present are simpler, albeit the extra information they contain with respect to the targets' kinematics. A bigger model performs better with more complex images [65], but will also cause lag and may bottleneck real-time application.

6.4.2. Object detection results

Table 6.4 showed that the created model performs relatively well with both optical and radar images, albeit being fine-tuned for range-Doppler plots. Let us visualize the detection results obtained from the test set. In Fig. 6.12 the ground truth for the test set is shown, while in Fig. 6.13 the predictions made by YOLO are highlighted for the handpicked hyperparameters that were optimized for cameras.

Figure 6.12: Ground truth labels for the optical camera test set.

Albeit the low mAP score obtained at multiple threshold, the precision, recall and mAP at 50% IoU

Figure 6.13: Predictions made by YOLO for the optical camera test set.

are relatively good. This is further seen in Fig. 6.13 where the confidence score of the detections is high. The low mAP score at multiple threshold may result from the bounding box size. Given the drone occupies only a few pixels, the center of the predicted bounding box may be off, hence an intersection between the ground truth and predicted boxes may be low, albeit being correct. Moreover, labeling a point target with such a small surface in a very large image is difficult. This leaves room for human error, which in turn contributes directly to the model evaluation.

6.5. Performance comparison between proposed method and IRIS® radar benchmarks

Let us compare the performances of the proposed pipeline with the performances of separate detection and classification algorithms that are used on Robin IRIS® Radar. It is however worth emphasizing that the radar performances were defined by specialists with years of experience, and investigated for multiple types of drones and different experimental conditions. Moreover, the method proposed in this thesis performs detection and labeling in a single algorithm, while the conventional methods use firstly use a Constant False Alarm Rate (CFAR) detector to identify targets of interest, and then feed their tracks into a DL model to classify the detected target. Thus, the outputs of the two methods are inherently different.

The Probability of Detection (P_D) used by Robin Radar Systems B.V. for the Swerling case IV (i.e. the target RCS varies from pulse to pulse) with a pre-defined Probability of False Alarm (P_{FA}) of 10^{-3} and a SNR threshold of $16dB$, is defined as following:

$$cst = \frac{1}{\frac{1+SNR}{2}} \tag{6.3}$$

$$P_D = 1 - cst \cdot (1 - cst) \cdot log(P_{FA}) \cdot P_{FA}^{cst} = 78.08\% \tag{6.4}$$

The above parameters were chosen via theoretical and experimental observations for multiple types of drones. The RCS of the drone influences the SNR threshold, meaning that for a smaller drone with lower RCS, a smaller SNR threshold is chosen. However, for the drone used in this project, an experimental RCS of approximately -18 dB was recorded, and the above parameters were chosen specifically for targets of this size and reflectivity. For smaller drones, the detection performance with the same false alarm rate is approximately 75.6%.

After a detection is done, classification is achieved via deep learning on the plot extracted from detected objects of interest from the range-Doppler snapshot. It is important to note that the output of the deep

learning model used in IRIS® is then a class for each detected track from the targets of interest. In YOLO, the output is a bounding box that has a specific confidence score. Thus, the following metrics serve as performances benchmarks to be used as a reference.

The current DL model has a precision score of 80.65% and a recall score of 97.60%. A large part of the false alarms were generated by wind-turbines, similar to the problem encountered with YOLO. Moreover, it is worth emphasizing that these metrics were obtained for multiple drones, some even smaller than the ones used in this project.

Thus, the proposed method improved the precision and recall scores, but works based on plots rather than tracks. Moreover, the output is fundamentally different, and the data sets were significantly simpler than the benchmarks set by Robin Radar Systems. In order further assess the performances of the proposed method, multiple types of drones should be used, and the system should run the new model in real-time to obtain the confusion matrix with more real world experimental data.

6.6. Summary

In this chapter we have presented the novel C-UAS scenario based on moving radars. The real-time processing requirements were established, followed by the unique micro-Doppler and clutter changes induced by the moving platform. In order to combat the newly emerged artifacts, an adaptive filter was discussed that uses the azimuth angle and platform velocity to remove the dynamic clutter.

In the end, the model was evaluated based on the performance metrics obtained for the validation and test sets, using different optimizers and hyperparameter spaces. SGD and Adam were compared, and Adam was shown to perform better in terms of accuracy and false alarms suppression than SGD, at the cost of being more difficult to initialize. However, by introducing genetic algorithms for hyperparameter fine-tuning, SGD was shown to improve and bypass false detections via proper initialization. The final results showed close to 100% performances across all metrics.

Moreover, the results were compared to optical cameras to further assess the capabilities of transferring video processing algorithms within the radar domain, which further confirmed the usefulness and robustness of the novel proposed framework in a cross-domain set-up. The model yielded better performances for the radar, with approximately 8% overall improved scores compared to the camera data set.

To conclude, transferring object detection algorithms from computer vision towards remote sensing has been proven a success. YOLO is capable of detecting drones based on radar plots with incredible precision, and the overall performance metrics are adequate.

7 Conclusion and Future Work

This chapter provides a concise summary and discussion of the motivation, methodology and results obtained in this work, which complements the introduction presented in Chapter 1. To summarize the work, Chapter 2 highlighted the current research done around drone classification and challenges using staring and surveillance radars. In Chapter 3 the backbone of deep learning and computer vision architectures were presented, while in Chapter 4 the radar system and data processing chain were analyzed. The performances of the proposed method were analyzed in Chapter 5 for a controllable scenario based on static radar, and in Chapter 6 the novelties of moving radars alongside the improved signal processing and object detection chains were presented and evaluated. This chapter then highlights the achieved milestones for each of the research questions, and provides recommendations for future research.

7.1. Conclusion

Given the exponential growth of the drone market, drone detection and tracking systems are now a clear necessity in most public, private or political events to ensure adequate safety. Thus, the objective of this thesis project was to build a novel framework that is able to jointly detect and label drones using static and moving surveillance FMCW radars.

The research questions therefore focused on achieving the following milestones:

- Developed an adaptive C-UAS processing chain for clutter removal that works with static and on-the-move radars without calibration down-time;
- Jointly solved the detection & classification problem with a single algorithm (YOLO) from computer vision, rather than using a CFAR detector and DL classifier individually;
- Validated the proposed method on experimental data, and successfully proposed methods to suppress false alarms;

The high-level contributions of this thesis can therefore be outlined as:

- **Counter-drone radars on-the-move**
  One of the main novelties in this work is the development of a novel counter-drone system, namely surveillance moving radars. To the best of my knowledge, this is the first time UAVs detection & classification with a radar anchored on a moving platform has been achieved with close to 100% precision, recall and mean average precision at a set IoU threshold. Coupled with the novel computer vision pipeline and sustained by the signal processing chain, the end-to-end framework is

capable of promising results in a wide array of scenarios, and it sets the first benchmarks and foundation for future development. However, the proposed method was validated uniquely on a small data set. While this proved the potential of these techniques, a study of the scalability and generalisation of this technique over significantly larger data sets is required.

- **Feature extraction with surveillance radars**
  There is very few information around automatic drone recognition using surveillance radars, and classification performances using such systems are not available to the best of the author's knowledge. Thus, this thesis project presented and analyzed the interaction with the environment of FMCW surveillance radars, and the performances of drone detection using such systems showing over 99% precision, recall and mAP at a set threshold, complementing the work of B-S. Oh et al. [8] and H. Sun et al. [9].

- **Plot-based multi-object detection & recognition**
  Traditionally, target classification is done after detection due to computational requirements. The presented method enabled multi-label and multi-instance joint object detection & classification in a single snapshot, without requiring mechanisms such as a sliding window, or a CFAR detector. Additionally, the inference time per image is 1.7 milliseconds with 0.2 milliseconds pre-processing, which satisfies the real-time requirements.

- **Integration of computer vision with radar engineering**
  To the best of the author's knowledge, object detection algorithms from computer vision did not achieve substantial success when transferred from optical cameras onto radar systems.

  This thesis showed how video processing algorithms can be successfully transferred into radar engineering for drone detection and labeling. This was possible thanks to the very specific and distinct modulations induced by UAVs that are not encountered by any other targets of similar sizes. Given the low RCS, only drones that have very fast rotating propellers can cover all the Doppler bins and yield a wide micro-Doppler modulation. Thus, object detection via computer vision and radar plots is possible, under the specific constraint that the modulations induced by the targets of interest are clearly distinct from other objects with similar geometry.

7.2. Future Work

While there is a myriad of further investigations to be done, and this work explored many new avenues for object detection and surveillance radars, there is still much left to be researched and documented. The following topics are amongst the most important:

- **Switch from plot-based to snapshot-based detection**
  In this work, the generated range-Doppler image was used to visualize the data, detect the drone and then assign a label. This allowed to clearly visualize and assess the performances and inherent mechanisms of the underlying structure of the algorithm. Moreover, this transformation allowed to compare optical images with range-Doppler radar plots. However, the underlying structure of the radar data is not an RGB image, but rather a data cube with dimensions represented by the range bins, Doppler bins and number of beams. The next and perhaps one of the most interesting next steps is then to assess the performances of computer vision methods used directly on range-Doppler snapshots of $3D$ data cubes, rather than transformed images.

- **Create a single deep learning model for both static and moving radars**
  There are two YOLO models that aim to detect drones for each static ad moving radar cases. It is therefore an open question to investigate whether building a single object detection model that works for both scenarios is possible. In order to assess this possibility, more data is necessary to capture more instances of drones in both cases. Due to time constraints required to go to the field and record and then label so much data, this analysis is beyond the scope of this thesis project.

- **Include track-based classification**
  We have seen in Subsection 4.2.5 that the pitfall of this technique is the dependency over the range. If the drone flies too far from the sensor, the micro-Doppler induced by the fast-spinning propellers will not be seen anymore due to their low RCS. However, the main body of the drone can still be seen. Thus, after a target has been assigned a label, track-based classification can take over when the micro-Doppler signature is no longer visible, which would increase the total classification range. Additionally, in Subsection 5.6 we have seen how range and Doppler information can be extracted with the proposed technique. This feature comes as a consequence of enabling multi-class & multi-instance object detection, but represents a crucial point of interest nonetheless. While Doppler estimation can not be extracted by using the current model due to the center of the bounding box not being centered around the main body of the drone, range estimation is indeed possible. The identification of the main-body of the drone and its range could therefore be used to correlate multiple consecutive detections, and hence create track classification.

- **Extend possible target classes**
  In this thesis project only one object class was considered, namely drones. However, the chosen architecture is capable of detecting multiple different objects in the same snapshot. In turn, this requires extra labeling and data to capture enough instances of each object type. Nonetheless, detecting swarms of drones, flock of birds, helicopters, wind-turbines and any other possible object that has a distinct micro-Doppler signature can be reliably achieved via this technique, and is worth exploring.

- **Investigate radar system performances at different platform velocities and environment**
  The final recommendation for this project is to investigate the limits of the entire radar system. We have shown that it is possible to take a counter-drone radar designed to be static and anchor it on a moving platform, and still be able to detect & classify drones via adequate software adaptation. Thus, the next practical question is to discover what are the limits of smart software by pushing the same hardware into never-before-seen scenarios. The moving platform speed limit was set to 50 km/h, which is the legal road speed in some roads in The Netherlands. It is however important to assess whether the same system can be used on highways at velocities of over 100 km/h. Additionally, it is important to further assess the performances of the radar system and data processing chain for varying velocities of the platform, but also with stop-and-go motions.

References

[1] Grand View Research. *Commercial Drone Market Size, Share & Trends Analysis Report By Product (Fixed-wing, Rotary Blade, Hybrid), By Application, By End-use, By Region, And Segment Forecasts, 2021 - 2028*. 2021. URL: https://www.grandviewresearch.com/industry-analysis/global-commercial-drones-market (visited on 11/30/2021).

[2] D. LCdr Ehredt. "UAS Yearbook - UAS: The Global Perspective". In: *NATO - Joint Air Power Competence Centre*. 2011, pp. 62/214.

[3] U.S. Federal Aviation Administration. *UAS by the Numbers*. 2021. URL: https://www.faa.gov/uas/resources/by_the_numbers/ (visited on 11/30/2021).

[4] The British Broadcasting Corporation. *Gatwick Airport drone attack: Police have 'no lines of inquiry*. 2019. URL: https://www.bbc.com/news/uk-england-sussex-49846450 (visited on 11/30/2021).

[5] The Guardian. *Thousands forced to spend night at airport as chaos continues – as it happened*. 2018. URL: https://www.theguardian.com/uk-news/live/2018/dec/20/gatwick-airport-drone-travel-chaos-disruption-live-updates (visited on 02/22/2022).

[6] Robin Radar Systems. *Evolution of the Drone Threat: Top Ten Drone Incidents*. 2019. URL: https://www.robinradar.com/press/blog/evolution-of-the-drone-threat-top-ten-drone-incidents (visited on 11/30/2021).

[7] Peter Wellig et al. "Radar Systems and Challenges for C-UAV". In: *2018 19th International Radar Symposium (IRS)* (2018), pp. 1–8.

[8] Beom-Seok Oh, Xin Guo, and Zhiping Lin. "A UAV classification system based on FMCW radar micro-Doppler signature analysis". In: *Expert Systems with Applications* 132 (2019), pp. 239–255. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2019.05.007. URL: https://www.sciencedirect.com/science/article/pii/S0957417419303203.

[9] Hongbo Sun et al. "Improving the Doppler Resolution of Ground-Based Surveillance Radar for Drone Detection". In: *IEEE Transactions on Aerospace and Electronic Systems* 55.6 (2019), pp. 3667–3673. DOI: 10.1109/TAES.2019.2895585.

[10] Stamatios Samaras et al. "UAV Classification with Deep Learning Using Surveillance Radar Data". In: *Computer Vision Systems*. Ed. by Dimitrios Tzovaras et al. Cham: Springer International Publishing, 2019, pp. 744–753. ISBN: 978-3-030-34995-0.

[11] Stephen Harman. "A comparison of staring radars with scanning radars for UAV detection: introducing the Alarm™ staring radar". In: *2015 European Radar Conference (EuRAD)*. IEEE. 2015, pp. 185–188.

[12] Mohammed Jahangir and Chris Baker. "Robust Detection of Micro-UAS Drones with L-Band 3-D Holographic Radar". In: *2016 Sensor Signal Processing for Defence (SSPD)*. 2016, pp. 1–5. DOI: 10.1109/SSPD.2016.7590610.

[13] Mohammed Jahangir, Chris J Baker, and Gordon A Oswald. "Doppler characteristics of micro-drones with L-Band multibeam staring radar". In: 2017, pp. 1052–1057. DOI: 10.1109/RADAR.2017.7944360.

[14] Matthew Ritchie et al. "Monostatic and bistatic radar measurements of birds and micro-drone". In: *2016 IEEE Radar Conference (RadarConf)*. 2016, pp. 1–5. DOI: 10.1109/RADAR.2016.7485181.

[15] Svante Björklund. "Target Detection and Classification of Small Drones by Boosting on Radar Micro-Doppler". In: *2018 15th European Radar Conference (EuRAD)*. 2018, pp. 182–185. DOI: 10.23919/EuRAD.2018.8546569.

[16] R. I. A. Harmanny, J. J. M. de Wit, and G. Prémel Cabic. “Radar micro-Doppler feature extraction using the spectrogram and the cepstrogram”. In: *2014 11th European Radar Conference*. 2014, pp. 165–168. DOI: 10.1109/EuRAD.2014.6991233.

[17] Beom-Seok Oh et al. “An EMD-based micro-Doppler signature analysis for mini-UAV blade flash reconstruction”. In: Aug. 2017, pp. 1–5. DOI: 10.1109/ICDSP.2017.8096105.

[18] Nima Mohajerin et al. “Feature extraction and radar track classification for detecting UAVs in civillian airspace”. In: *2014 IEEE Radar Conference*. 2014, pp. 0674–0679. DOI: 10.1109/RADAR.2014.6875676.

[19] Sungho Jeon et al. “Empirical study of drone sound detection in real-life environment with deep neural networks”. In: *2017 25th European Signal Processing Conference (EUSIPCO)*. 2017, pp. 1858–1862. DOI: 10.23919/EUSIPCO.2017.8081531.

[20] Bilal Taha and Abdulhadi Shoufan. “Machine Learning-Based Drone Detection and Classification: State-of-the-Art in Research”. In: *IEEE Access* 7 (2019), pp. 138669–138682. DOI: 10.1109/ACCESS.2019.2942944.

[21] John E. Ball, Derek T. Anderson, and Chee Seng Chan Sr. “Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community”. In: *Journal of Applied Remote Sensing* 11.4 (2017), pp. 1–54. DOI: 10.1117/1.JRS.11.042609. URL: https://doi.org/10.1117/1.JRS.11.042609.

[22] Thomas Perrott Gill. *The Doppler effect: an introduction to the theory of the effect*. Academic Press, 1965.

[23] V.C. Chen et al. “Micro-Doppler effect in radar: phenomenon, model, and simulation study”. In: *IEEE Transactions on Aerospace and Electronic Systems* 42.1 (2006), pp. 2–21. DOI: 10.1109/TAES.2006.1603402.

[24] Mark S Zediker, Robert R Rice, and Jack H Hollister. *Method for extending range and sensitivity of a fiber optic micro-Doppler ladar system and apparatus therefor*. US Patent 5,847,817. Dec. 1998.

[25] Kevin J Parker, Robert M Lerner, and Sung-Rung Huang. *Method and apparatus for using Doppler modulation parameters for estimation of vibration amplitude*. US Patent 5,086,775. Feb. 1992.

[26] Shie Qian and Dapang Chen. “Joint time-frequency analysis : methods and applications”. In: 1996.

[27] Shie Qian and Dapang Chen. “Joint time-frequency analysis”. In: *IEEE Signal Processing Magazine* 16.2 (1999), pp. 52–67. DOI: 10.1109/79.752051.

[28] V.C. Chen. “Analysis of radar micro-Doppler with time-frequency transform”. In: *Proceedings of the Tenth IEEE Workshop on Statistical Signal and Array Processing (Cat. No.00TH8496)*. 2000, pp. 463–466. DOI: 10.1109/SSAP.2000.870167.

[29] Victor C Chen and Hao Ling. *Time-frequency transforms for radar imaging and signal analysis*. Artech house, 2002.

[30] Samiur Rahman and Duncan Robertson. “Radar micro-Doppler signatures of drones and birds at K-band and W-band”. In: *Scientific Reports* 8 (Nov. 2018). DOI: 10.1038/s41598-018-35880-9.

[31] Chenchen J. Li and Hao Ling. “An Investigation on the Radar Signatures of Small Consumer Drones”. In: *IEEE Antennas and Wireless Propagation Letters* 16 (2017), pp. 649–652. DOI: 10.1109/LAWP.2016.2594766.

[32] J. J. M de Wit, R. I. A. Harmanny, and G. Prémel-Cabic. “Micro-Doppler analysis of small UAVs”. In: *2012 9th European Radar Conference*. 2012, pp. 210–213.

[33] Albert Huizing et al. “Deep Learning for Classification of Mini-UAVs Using Micro-Doppler Spectrograms in Cognitive Radar”. In: *IEEE Aerospace and Electronic Systems Magazine* 34.11 (2019), pp. 46–56. DOI: 10.1109/MAES.2019.2933972.

[34] Kyung-Tae Kim, Dong-Kyu Seo, and Hyo-Tae Kim. “Efficient radar target recognition using the MUSIC algorithm and invariant features”. In: *IEEE Transactions on Antennas and Propagation* 50.3 (2002), pp. 325–337.

[35] Siddharth Gupta et al. “Target Classification by mmWave FMCW Radars Using Machine Learning on Range-Angle Images”. In: *IEEE Sensors Journal* 21.18 (2021), pp. 19993–20001. DOI: 10.1109/JSEN.2021.3092583.

[36] J. Ward. “Space-time adaptive processing for airborne radar”. In: *1995 International Conference on Acoustics, Speech, and Signal Processing*. Vol. 5. 1995, 2809–2812 vol.5. DOI: 10.1109/ICASSP.1995.479429.

[37] J.R. Guerci. *Space-Time Adaptive Processing for Radar, Second Edition*. Artech House radar library. Artech House Publishers, 2014. ISBN: 9781608078219. URL: https://books.google.nl/books?id=5efmBgAAQBAJ.

[38] Børge Torvik, Karl Erik Olsen, and Hugh Griffiths. “Classification of Birds and UAVs Based on Radar Polarimetry”. In: *IEEE Geoscience and Remote Sensing Letters* 13.9 (2016), pp. 1305–1309. DOI: 10.1109/LGRS.2016.2582538.

[39] Pavlo Molchanov et al. “Classification of small UAVs and birds by micro-Doppler signatures”. In: *International Journal of Microwave and Wireless Technologies* 6.3-4 (2014), pp. 435–444.

[40] Byung Kwan Kim, Hyun-Seong Kang, and Seong-Ook Park. “Drone classification using convolutional neural networks with merged Doppler images”. In: *IEEE Geoscience and Remote Sensing Letters* 14.1 (2016), pp. 38–42.

[41] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.

[42] Daniel A. Brooks et al. “Temporal Deep Learning for Drone Micro-Doppler Classification”. In: *2018 19th International Radar Symposium (IRS)*. 2018, pp. 1–10. DOI: 10.23919/IRS.2018.8447963.

[43] Dinesh Kumar Behera and Arockia Bazil Raj. “Drone Detection and Classification using Deep Learning”. In: *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*. 2020, pp. 1012–1016. DOI: 10.1109/ICICCS48265.2020.9121150.

[44] Syed Ali Hassan, Tariq Rahim, and Soo Young Shin. “Real-time UAV Detection based on Deep Learning Network”. In: *2019 International Conference on Information and Communication Technology Convergence (ICTC)*. 2019, pp. 630–632. DOI: 10.1109/ICTC46691.2019.8939564.

[45] Lars Sommer and Arne Schumann. “Deep learning based UAV type classification”. In: *Pattern Recognition and Tracking XXXII*. Vol. 11735. International Society for Optics and Photonics. 2021, p. 1173508.

[46] Jacco J.M. De Wit, Daniel Gusland, and Roeland P. Trommel. “Radar Measurements for the Assessment of Features for Drone Characterization”. In: *2020 17th European Radar Conference (EuRAD)*. 2021, pp. 38–41. DOI: 10.1109/EuRAD48048.2021.00021.

[47] Lucas Prado Osco et al. “A review on deep learning in UAV remote sensing”. In: *International Journal of Applied Earth Observation and Geoinformation* 102 (2021), p. 102456. ISSN: 0303-2434. DOI: https://doi.org/10.1016/j.jag.2021.102456. URL: https://www.sciencedirect.com/science/article/pii/S030324342100163X.

[48] Prince Grover. *Evolution of Object Detection and Localization Algorithms | by Prince Grover | Towards Data Science*. https://towardsdatascience.com/evolution-of-object-detection-and-localization-algorithms-e241021d8bad. (Accessed on 06/30/2022). Feb. 2018.

[49] Y. Bengio and Yann Lecun. “Convolutional Networks for Images, Speech, and Time-Series”. In: (Nov. 1997).

[50] Vincent Dumoulin and Francesco Visin. “A guide to convolution arithmetic for deep learning”. In: *ArXiv* abs/1603.07285 (2016).

[51] Sumit Saha. *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way | by Sumit Saha | Towards Data Science*. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. (Accessed on 06/30/2022). Dec. 2018.

[52] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012), pp. 1097–1105.

[53] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[54] Hyun Min Oh, Hyunki Lee, and Min Young 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)*. 2019, pp. 87–90. DOI: 10.23919/ICCAS47443.2019.8971699.

[55] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[56] Christian Szegedy et al. "Rethinking the Inception Architecture for Computer Vision". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[57] Ross Girshick et al. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2014. arXiv: 1311.2524 [cs.CV].

[58] Zhong-Qiu Zhao et al. "Object Detection With Deep Learning: A Review". In: *IEEE Transactions on Neural Networks and Learning Systems* PP (Jan. 2019), pp. 1–21. DOI: 10.1109/TNNLS.2018.2876865.

[59] Ross Girshick. *Fast R-CNN*. 2015. arXiv: 1504.08083 [cs.CV].

[60] Shaoqing Ren et al. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. 2016. arXiv: 1506.01497 [cs.CV].

[61] Joseph Redmon et al. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV].

[62] Joseph Redmon and Ali Farhadi. *YOLO9000: Better, Faster, Stronger*. 2016. arXiv: 1612.08242 [cs.CV].

[63] Joseph Redmon and Ali Farhadi. *YOLOv3: An Incremental Improvement*. 2018. arXiv: 1804.02767 [cs.CV].

[64] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. *YOLOv4: Optimal Speed and Accuracy of Object Detection*. 2020. arXiv: 2004.10934 [cs.CV].

[65] G. Jocher. *YOLOv5 documentation*. 2020. URL: https://docs.ultralytics.com/ (visited on 05/09/2022).

[66] Matthew D. Zeiler and Rob Fergus. "Visualizing and Understanding Convolutional Networks". In: *Computer Vision – ECCV 2014*. Ed. by David Fleet et al. Cham: Springer International Publishing, 2014, pp. 818–833. ISBN: 978-3-319-10590-1.

[67] Matthew D Zeiler, Graham W Taylor, and Rob Fergus. "Adaptive deconvolutional networks for mid and high level feature learning". In: *2011 International Conference on Computer Vision*. IEEE. 2011, pp. 2018–2025.

[68] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. "Learning deconvolution network for semantic segmentation". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1520–1528.

[69] *TensorFlow Autoencoder: Dataset with Deep Learning Example*. https://www.guru99.com/autoencoder-deep-learning.html. (Accessed on 29/12/2021).

[70] *Auto-Encoders for Computer Vision: An Endless world of Possibilities*. https://www.analyticsvidhya.com/blog/2021/01/auto-encoders-for-computer-vision-an-endless-world-of-possibilities/. (Accessed on 29/12/2021).

[71] Abdallah Zeggada and Farid Melgani. "Multilabeling UAV images with Autoencoder networks". In: Mar. 2017, pp. 1–4. DOI: 10.1109/JURSE.2017.7924544.

[72] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[73] Robin Radar Systems. *IRIS Counter Drone Radar*. 2021. URL: https://www.robinradar.com/iris-counter-drone-radar (visited on 02/23/2022).

[74] *RF Tutorial Lesson 17: Simulating a Frequency-Modulated Continuous-Wave (FMCW) Radar System - Emagtech Wiki*. http://www.emagtech.com/wiki/index.php/RF_Tutorial_Lesson_17:_Simulating_a_Frequency-Modulated_Continuous-Wave_%28FMCW%29_Radar_System. (Accessed on 06/30/2022).

[75] van der Veen, A-J. *Array signal processing - an algebraic approach, TU Delft, CAS Department, EE4715 Spring 2022*. 2022. URL: https://cas.tudelft.nl/Education/courses/et4147/reader.php (visited on 04/17/2022).

[76] Michael Inggs et al. "Experimental Coherent Networked Radar Using GPS-Disciplined Reference Oscillators". In: Jan. 2010.

[77] *Image Mean, Standard Deviation, and Correlation Coefficient - MATLAB & Simulink - MathWorks Benelux*. https://nl.mathworks.com/help/images/image-mean-standard-deviation-and-correlation-coefficent.html. (Accessed on 08/02/2022).

[78] A. Rose. *Vision: Human and Electronic*. Optical physics and engineering. Plenum, 1973. URL: https://books.google.nl/books?id=XV6FswEACAAJ.

[79] Arash Akbarinia and Raquel Gil-Rodríguez. "Deciphering image contrast in object classification deep networks". In: *Vision Research* 173 (2020), pp. 61–76. ISSN: 0042-6989. DOI: https://doi.org/10.1016/j.visres.2020.04.015. URL: https://www.sciencedirect.com/science/article/pii/S0042698920300766.

[80] S.Z. Gurbuz. *Deep Neural Network Design for Radar Applications*. Radar, Sonar and Navigation. Institution of Engineering and Technology, 2020. ISBN: 9781785618529. URL: https://books.google.nl/books?id=TgENEAAAQBAJ.

[81] Rohit Singh. *SimpleEDA + Visualizations | Kaggle*. https://www.kaggle.com/code/rohitsingh9990/simpleeda-visualizations/notebook. (Accessed on 05/24/2022). May 2020.

[82] Uri Almog. *YOLO V3 Explained. In this post we'll discuss the YOLO… | by Uri Almog | Towards Data Science*. https://towardsdatascience.com/yolo-v3-explained-ff5b850390f. (Accessed on 05/24/2022). Oct. 2020.

[83] Arun Gandhi. *Data Augmentation | How to use Deep Learning when you have Limited Data*. https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/. (Accessed on 05/24/2022). 2021.

[84] Alvaro Fuentes et al. "Spectral Analysis of CNN for Tomato Disease Identification". In: May 2017, pp. 40–51. ISBN: 978-3-319-59062-2. DOI: 10.1007/978-3-319-59063-9_4.

[85] M.H. Hayes. *Statistical Digital Signal Processing and Modeling*. Wiley, 1996. ISBN: 9780471594314. URL: https://books.google.nl/books?id=N%5C_VSAAAAMAAJ.

[86] Mark Everingham et al. "The Pascal Visual Object Classes (VOC) Challenge". In: *International Journal of Computer Vision* 88 (2009), pp. 303–338.

[87] Laurits Tani et al. "Evolutionary algorithms for hyperparameter optimization in machine learning for application in high energy physics". In: *The European Physical Journal C* 81.2 (Feb. 2021). DOI: 10.1140/epjc/s10052-021-08950-y. URL: https://doi.org/10.1140%2Fepjc%2Fs10052-021-08950-y.

A

Publication paper

Below the research paper written based on this thesis project can be found.

It aims to be sent at IEEE Radar Conference from May $1-$ May 5 2023 in San Antonio, Texas, US.

Drone Detection & Classification with Surveillance 'Radar On-The-Move' and YOLO

Hani Haifawi*†, Francesco Fioranelli*, Alexander Yarovoy*, Rob van der Meer†
*Microwave Sensing, Signals and Systems (MS3) Group, Delft University of Technology, Delft, The Netherlands
†Robin Radar Systems B.V., The Hague, The Netherlands
{hani.haifawi, rob.vandermeer}@robinradar.com; {f.fioranelli, a.yarovoy}@tudelft.nl

***Abstract*—A new method to jointly detect and classify drones using a moving surveillance radar system (*'radar on-the-move'*) and computer vision is presented. While most conventional counter-drone radar-based techniques focus on time-frequency distributions to obtain classification features, such approaches are limited in volumetric spatial coverage. To compensate for this, surveillance radars that offer full spatial coverage are used, but the determination of the best detection and classification approach to be applied on the resulting data is still an open challenge. In this paper a framework is proposed that combines deep learning approaches from computer vision, specifically the You Only Look Once (YOLO) network, to radar data from the moving surveillance radar system produced by Robin Radar Systems B.V. This framework allows to jointly detect and label targets based on range-Doppler snapshots generated in real-time. The method is validated on experimental data, with initial results on a small dataset showing precision, recall, and mean average precision (mAP@0.5) of over 99%.**

***Index Terms*—drone detection, drone classification, surveillance radar, YOLO**

I. Introduction

Drone tracking and classification systems have received significant attention due to the exponential growth of the Unmanned Aerial Vehicles (UAVs) market, and the related concerns for accidental or intentional misuses of such platforms. To address this need of reliable monitoring capabilities, radar systems can provide robust detection and classification in any weather or light condition, as well as the direct determination of range and velocity of targets [1]. For this reason, counter-drone radar systems have become a safety requirement in a wide array of public and private events and at critical locations such as airports or power plants. Moreover, drone detection and tracking systems that are anchored on moving vehicles have gained remarkable interest due to an ever increasing need for protection against rogue UAVs that might appear anywhere near and around an asset to be protected.

The ability to distinguish drones from other types of targets is typically achieved in literature thanks to the fast-rotating propellers that induce unique micro-Doppler modulations. Conventionally, micro-Doppler signatures are obtained using radars with fixed beams with high dwell time on target [2]–[4]. While this allows to capture a continuous sequence of samples necessary to obtain good-quality time-frequency distributions and signatures, such systems suffer of limited spatial coverage due to their static nature, so that the targets of interest may not be always present in the radar beam. In order to achieve full spatial coverage in a cost-effective manner, surveillance radars that use rotating antennas are adopted [5], [6]. However, this may result in losing the most salient features for drone classification due to the limited time on target. This limitation can be solved by staring radar systems that ensure wide spatial coverage continuously at all time [7], [8], but at the cost of higher system complexity and a very large amount of data to be processed. In terms of the actual classification algorithms, machine and deep learning methods have gained a lot of momentum given their high performances for drone-related applications [9]–[13]. However, these approaches aim to solve only a classification problem, i.e. assigning labels to detected targets. Hence, a detector is firstly needed to find the targets of interest, after which the classifier assigns a label to each target.

In this paper, an approach that jointly solves the detection and classification problem of drones with a single algorithm is presented. The 'You Only Look Once' (YOLO) [14] framework from computer vision is chosen given the real-time inference speed of its small models, while retaining high accuracy for small objects identification and bounding boxes prediction. Moreover, this technique can scale well with small datasets and is an open-source project, hence easier to use and interpret than proprietary codes. To the best of our knowledge, the usage of the YOLO framework and networks on radar range-Doppler plots for drone detection and classification is novel, as it has been primarily used only on optical images or videos. The system considered in this paper for the validation of the proposed approach is a surveillance radar with rotating antennas, but with the additional complexity of its movement on the ground to patrol a certain area or asset of interest. The data used for classification via YOLO are range-Doppler plots generated by this radar, achieving multi-class and multi-instance object detection. The initial validation with experimental results reported in this paper showed promising results with precision, recall and mean average precision scores of over 99% in real-time.

The rest of the paper is organized as follows. Section II presents the radar system used. The signal processing chain and computer vision detector are presented in Section III. The dataset is discussed in Section IV. Section V presents the experimental results, and the conclusion is drawn in Section VI.

II. Moving Radar System Characteristics

The radar system used is the *IRIS* FMCW Drone Radar developed by Robin Radar Systems B.V., operating at 9.25 GHz with a bandwidth of 50 MHz [15]. The sampling rate is 15.625 MHz with a Sweep Repetition Frequency (SRF) of 4 kHz. A full rotation in the azimuth plane takes 2 seconds per antenna, and given there are two antennas facing opposite directions, a full scan takes 1 second in total. The Coherent Processing Interval (CPI) consists of 100 consecutive sweeps that take 25 milliseconds and cover 4.5 degrees in azimuth. In the elevation plane, the radar has two phased array antennas, each with 8 elements. The radar was anchored on a moving platform on the ground and can detect, track and classify drones while cruising at approximately 50 km/h ('radar on-the-move').

A. Data model

The signal model of each individual sweep uses a saw-tooth modulation that satisfies the narrowband condition [16]:

$$\frac{c}{B} \gg N \cdot d \tag{1}$$

where c is the speed of light, B is the signal bandwidth, N is the number of array elements, and d is the inter-spacing between the said elements. Thanks to the narrow-band condition, the phase progression across the array elements is linear. Thus, the signals collected at each of the N array receiver channels incoming from any possible angle θ are modelled as the data vector $\mathbf{x}$:

$$\mathbf{x}(\theta) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} 1 \\ e^{j2\pi \frac{d}{\lambda} sin\theta} \\ \vdots \\ e^{j2\pi(N-1)\frac{d}{\lambda} sin\theta} \end{bmatrix} \tag{2}$$

In this case, conventional beamforming is performed on the phased array in elevation. A total of 6 beams are used which cover 60° in elevation, ranging from $-30°$ up to $+30°$ with respect to the antenna aperture.

The final data structure before the pre-processing takes place is a radar data cube with dimensions of 1562 by 6 by 100, representing the number of range bins by the number of beams by the number of Doppler bins, per CPI.

III. Data Processing Pipeline

A. Signal processing chain

The signal processing chain on the radar data after beamforming is described in this section. Given that the radar is both moving on the ground and rotating, care needs to be taken when applying clutter filtering techniques. Specifically, three different pre-processing steps are applied one after the other on the data:

- **Pulse subtraction:** this acts as a moving target indicator (MTI) or low-pass filter. It removes the static clutter centered around 0 Hz that emerges from the moving platform that is seen as static, and the reflections obtained from the back-lobe that are moving in the opposite direction of the sensor and yield a null Doppler shift.
- **Notch filter:** this is an adaptive autoregressive moving average filter that cuts the frequency band correlated to the radar own velocity and azimuth angle with respect to the antenna. It removes the dynamic clutter created from the ground reflections which is no longer static due to the movement of the sensor.
- **Amplitude averaging:** this processing step divides the absolute value of each range bin by the mean absolute value of all range bins in a single sweep. This helps attenuate the effect of background noise and remaining clutter in view of the subsequent steps based on image processing.

The output of the signal processing chain is the filtered range-Doppler plot obtained via 2D Fast Fourier Transform (FFT), which is then used as the input to the computer vision pipeline. Conventionally, the absolute value of these range-Doppler matrices can be directly fed as input to neural networks for classification. However, given that YOLO was initially created for optical images, a further image processing step is applied to the range-Doppler matrices to generate visualisations such as the example shown in Figure 1.

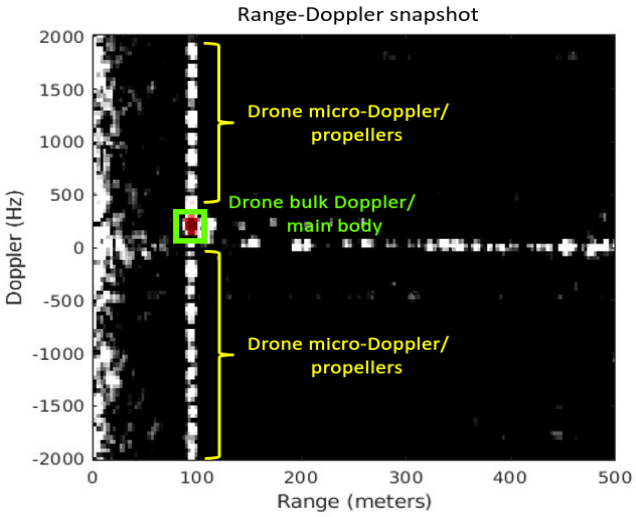


Fig. 1. An example of filtered range-Doppler image containing a drone that is used as input for YOLO for detection and classification.

Essentially, this step is a colour transformation that maps the received signal power in the range-Doppler matrix to a color map designed to maximize the RGB contrast between the drone micro-Doppler (colored in white) and the background (colored in black), while retaining the drone bulk Doppler in red. The proposed color mapping is seen in Fig. 3.

This visualization facilitates the interpretation of the data by the YOLO network, making the contrast between different elements in the image more evident (e.g., the modulation due to the blades of the drones). The signal processing chain discussed in this sub-section is summarised in Figure 2.

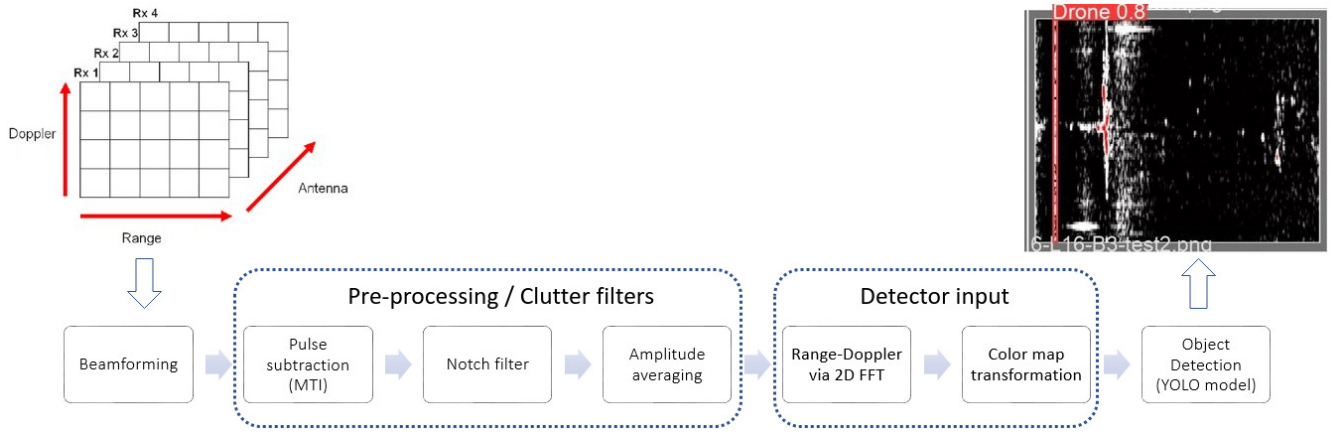


Fig. 2. Signal processing chain block diagram, from input data cube from the radar to an example of detection output generated by the YOLO network.

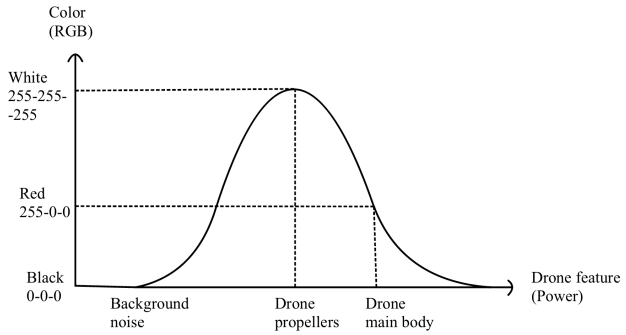


Fig. 3. Mapping of the range-Doppler matrix intensity values into the custom-defined RGB color map.

B. YOLO detector and its training

The chosen object detection pipeline is YOLOv5s, the PyTorch implementation of YOLOv4 which was initially built in Tensorflow [14]. There are multiple YOLO models with higher number of parameters, tailored for complex high-resolution images, such as YOLOv5l or YOLOv5x. However, due to the small size of the dataset used alongside the relatively simple micro-Doppler modulations shapes, the small YOLO model was chosen as the best fit for this application. This choice is further supported by the open-source nature of the project, alongside the real-time inference speeds of small models, and an adequate trade-off between optimal speed and accuracy.

Nevertheless, one of the main issues of deep learning approaches applied to radar engineering remains the small size of datasets. To address this issue, the network weights were initialized using transfer learning from a pre-trained model for 300 epochs over the COCO dataset [17]. The model was then trained over range-Doppler plots of drones for an additional 150 epochs, using the Stochastic Gradient Descent (SGD) and Adam as optimizers.

Moreover, data augmentation was included during the training stage to help mitigate overfitting and improve the performance metrics without going to the field to record more experimental data. It should be noted that radar images embed the kinematics of targets that should not be altered, otherwise, the augmented micro-Doppler modulations risk to be not realistic and not encountered in actual true data. Thus, the data augmentation methods used in this approach are flip, translation, mosaic, hue, saturation, and value, denoted as albumentations [18]. While the library contains over 60 data augmentation methods, techniques that preserve the kinematics coherence were chosen. For example, the flip is only done from left to right or up and down, but the image is never rotated because the micro-Doppler profile would then be horizontal instead of vertical, which is never encountered in real world. The transformation of images is done stochastically, such that the model never sees twice the same image.

A second step in helping the model generalize better over unseen data is including weights decaying. At each epoch, a specific percentage of the weights is removed in order to prevent the model from learning too much from the specific training data.

As discussed also in Section V, it was noted that the model initially yielded false alarms due to wind-turbines in the field of view that had similar micro-Doppler modulations due to the rotation of their blades. By fine-tuning the training hyperparameters for this specific application using a genetic algorithm, the model was shown to be more robust against these false detections [**Cite own thesis**].

IV. Dataset Generation and Composition

The data was recorded using pre-defined test scenarios with a drone and the radar moving back and forth at different velocities. The IRIS Drone Radar was anchored on a moving vehicle, as seen in Figure 4, and the tests were taken with radar velocities ranging from 0 km/h up to 50 km/h. The Autel Evo II drone was used in these tests. It should be noted that the data was collected in a clutter-rich area with a plethora of other targets, such as wind turbines in the background, vegetation moving with the wind, birds, trains, ships, cars, cyclists, and other objects that can occur in real-life environments and may yield false alarms.

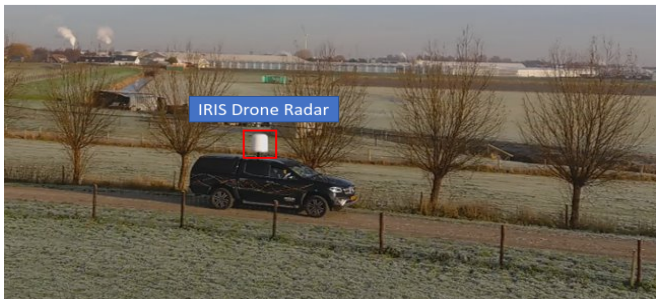


Fig. 4. IRIS radar by Robin Radar Systems B.V. anchored on the moving vehicle.

The final model was trained using only 188 drone range-Doppler images, and an equal amount of plots containing non-drone targets as described in Table I. As shown, the data was split as 60-20-20% between training, validation, and test sets to verify the performances of the model during training, but also the scalability over unseen data. It should be noted that, to de-correlate the training and test sets and to obtain natural data diversity, the training and validation data sets were taken in June, while the test set was taken in December.

TABLE I
DATASET COMPOSITION AND SPLITTING

Data	Training	Validation	Test
Number of drone images	188	62	62
Number of total images	376	124	124
Ratio compared to whole dataset	60%	20%	20%

V. RESULTS

In this section the results generated by the proposed method are presented and discussed. In computer vision, recall can be interpreted as the area of overlap between the ground truth bounding box and the predicted one, divided by the area of the detected bounding box. Similarly, precision measures the same area of overlap, but this time it is divided by the ground truth object. Lastly, the mean Average Precision (mAP) is a weighted mean of precision scores at multiple thresholds. The said threshold represents the increment in the recall score compared to the prior threshold. The result is then averaged for each class. Thus, mAP highlights a trade-off between precision and recall that considers both false positives and false negatives, making it a very popular metric in all object detection applications. Moreover, mAP@0.5 means the average precision is computed for Intersection over Union (IoU) scores of 0.5, while mAP@0.5 : 0.95 represents the mAP taken for IoUs ranging from 0.5 up to 0.95 with a step of 0.05.

A. Handpicked hyperparameter space

The performances of two optimizers were analyzed: SGD and Adam. Additionally, the model was tested using weight decaying of 0.75%, or no weight decaying at all. The results are summarized in Table II.

Firstly, the effects of weight decaying are different for the two optimizers. While this technique improves the performances of SGD over the test for the mAP at multiple thresholds, it negatively affects Adam as the mAP@0.5 : 0.95 decreased by approximately 1% using the decay. On one hand, this may be due to Adam being more difficult to initialize given the extra adaptive hyperparameters (such as start and end learning rates, momentum, warm-up epochs, among others). Nonetheless, all models yield overall high performances with mAP@0.5 scores of approximately 99%.

However, one main difference between SGD and Adam is the emergence of false alarms. During training, SGD creates false detections over wind turbines that yield wide Doppler modulations due to their fast-rotating blades, as seen in Fig. 5. Adam on the other hand avoids such false alarms, as seen in Fig. 6. This is further reflected from the performance metrics, where Adam tends to have higher precision than SGD. Thus, while one simple solution to distinguish drones from wind turbines would be to verify the reflectivity and size of the target, the false alarms can be successfully removed by choosing a more adequate optimizer.

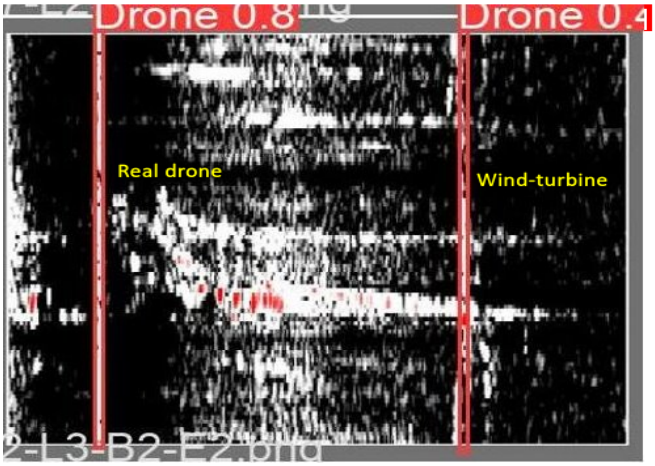


Fig. 5. Example of output of the proposed model, with detections obtained using SGD optimizer. A false alarm induced by the a wind turbine is reported, near a true detection.

B. Hyperparameters fine-tuning via genetic algorithms

The proposed solution to further adapt the model for the desired application is by fine-tuning the hyperparameters to better fit the data. While there are multiple methods to explore the hyperparameter space, in this paper genetic algorithms were investigated and the new model results are summarized in Table III. The best combination of the hyperparameter space is chosen based on the best performance metrics obtained. A mutation for the proposed dataset took approximately 5 minutes, and a total of 600 mutations were used for each optimizer. Thus, approximately 5 days were required to obtain the new hyperparameters' values.

TABLE II
PERFORMANCE METRICS FOR THE VALIDATION AND TEST SETS USED FOR SGD AND ADAM OPTIMIZERS.

Model	Set	Weights decay	Optimizer	Precision	Recall	mAP@0.5	mAP@0.5:0.95
1	Validation	0%	SGD	98.4%	99.5%	99.5%	75.1%
1	Test	0%	SGD	99.8%	100%	99.5%	74.3%
2	Validation	0.75%	SGD	98.3%	98.4%	99.4%	75.2%
2	Test	0.75%	SGD	99.9%	98.4%	99.4%	76.6%
3	Validation	0%	Adam	100%	98%	99.5%	75.8%
3	Test	0%	Adam	99.6%	98.4%	99.5%	76.2%
4	Validation	0.75%	Adam	99.9%	98.4%	99.5%	75.2%
4	Test	0.75%	Adam	99.8%	100%	99.5%	74.9%

TABLE III
PERFORMANCES DIFFERENCES BETWEEN HAND-PICKED HYPERPARAMETERS AND THOSE OBTAINED VIA GENETIC ALGORITHM.

Model	Set	Hyperparameters tuning	Optimizer	Precision	Recall	mAP@0.5	mAP@0.5:0.95
1	Validation	Hand picked	SGD	98.3%	98.4%	99.4%	75.2%
1	Test	Hand picked	SGD	99.9%	98.4%	99.4%	76.6%
2	Validation	Genetic algorithm	SGD	98.2%	100%	99.5%	75.7%
2	Test	Genetic algorithm	SGD	100%	99.7%	99.5%	75.1%
3	Validation	Hand picked	Adam	100%	98%	99.5%	75.8%
3	Test	Hand picked	Adam	99.6%	98.4%	99.5%	76.2%
4	Validation	Genetic algorithm	Adam	100%	98.2%	99.5%	76.5%
4	Test	Genetic algorithm	Adam	99.9%	100%	99.5%	77.3%

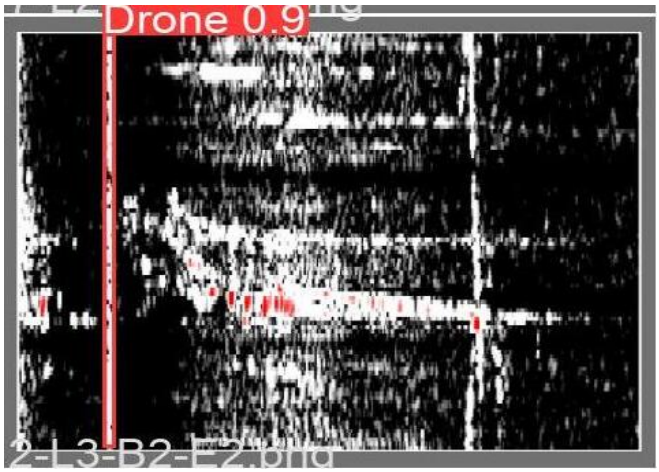


Fig. 6. Example of output of the proposed model for the same range-Doppler plot of Fig. 5, but using Adam as optimizer. The choice of this optimizer did not yield false detections.

Fig. 7 shows an example of results generated with the hyperparameter space from the genetic algorithm, without the false detection from the wind turbine. While Adam is less computationally expensive and did not yield false alarms as compared to SGD, these examples highlight the importance of the optimizer choice and proper hyperparameters initialization. Table III shows the improvements of the model obtained via genetic algorithms, especially for the results obtained over the test set. Nonetheless, while the false alarms were successfully suppressed for SGD, the mAP at multiple threshold slightly decreased when the hyperparameters were initialized using genetic algorithms. For Adam meanwhile, all the performance scores improved. This may be due to Adam being an adaptive optimizer which is more difficult to initialize, but also because more mutations were required to find an optimal hyperparameter space. Hence, the genetic algorithms were able to find a better hyperparameter space to yield the best overall results.

Fig. 7. Example of output of the proposed model for the same range-Doppler plot of Fig. 5 and 6, but using SGD optimizer and genetic algorithm. The genetic algorithm combined with SGD did not yield false detections.

C. Discussion and comparison with traditional algorithms

In these preliminary results, the proposed method was shown to accurately detect and label drones with high precision and recall scores based on range-Doppler plots. Moreover, the YOLO inference time is 1.7ms per image with approximately 0.2ms pre-processing time using a Nvidia GeForce RTX 3080 graphics card with 10 gigabyte total memory.

To the best knowledge of the authors, no similar computer vision method was investigated for the specific application of jointly detecting and classifying drones using surveillance radars' data. In the literature, a detector is firstly used to obtain the targets of interest, after which a classifier is used to assign the detected targets' labels. Thus, two different algorithms with different performance metrics and outputs are used to detect and label targets, while the proposed approach with YOLO solves this problem jointly. This makes the benchmarking of the method difficult.

For a drone with an experimental radar cross-section of approximately -18 dBsm, the same value as the Autel Evo II drone used in this project, the probability of detection using a constant false alarm rate detector for a Swerling case IV is given by [19]:

$$cst = \frac{1}{1 + \frac{SNR}{2}} \tag{3}$$

$$P_D = \left(1 - cst \cdot (1 - cst) \cdot log(P_{FA})\right) \cdot P_{FA}^{cst} = 78.08\% \tag{4}$$

The classification however is done based on plots & tracks compared to YOLO which labels targets based on range-Doppler plots. The current deep learning model used by Robin Radar has a precision score of 80.65% and a recall score of 97.60%. However, these benchmarks were exhaustively analysed for many test scenarios with multiple types of drones, different experimental conditions and a static radar, while the computer vision model was tested for a radar on-the-move using the same drone.

VI. Conclusion

In this paper, a novel approach for joint detection and classification of drones is proposed. This approach uses the computer vision YOLO framework on range-Doppler plots captured with a surveillance 'radar on-the-move', i.e. a radar with a rotating antenna that is simultaneously mounted on a moving platform on the ground. Different from the conventional approaches using micro-Doppler signatures from a staring radar, the proposed approach is shown to be effective when operating on single range-Doppler images corresponding to relatively short observation times (or CPI).

The approach has been validated with experimental data collected by the *IRIS* FMCW radar developed by Robin Radar Systems B.V., on a small dataset with a single drone flying in the scene of interest where other moving targets were present, including birds, targets on the ground, and wind turbines. Results in terms of precision, recall, and mean average precision (mAP@0.5) over 99% were obtained, showing the potential of the method and the opportunity to perform a broader verification on a larger dataset.

VII. Acknowledgments

The authors are grateful to colleagues at Robin Radar Systems and TU Delft who supported the master thesis project of Hani Haifawi and related research.

References

[1] P. Wellig, P. Speirs, C. Schüpbach, R. Oechslin, M. RenkerM U. Boeniger and H. Pratisto (2018). Radar Systems and Challenges for C-UAV. 1-8. 10.23919/IRS.2018.8448071.

[2] Molchanov, P., Harmanny, R., De Wit, J., Egiazarian, K., & Astola, J. (2014). Classification of small UAVs and birds by micro-Doppler signatures. International Journal of Microwave and Wireless Technologies, 6(3-4), 435-444. doi:10.1017/S1759078714000282

[3] J. J. M. de Wit, R. I. A. Harmanny and G. Prémel-Cabic, "Micro-Doppler analysis of small UAVs," 2012 9th European Radar Conference, 2012, pp. 210-213.

[4] R. I. A. Harmanny, J. J. M. de Wit and G. P. Cabic, "Radar micro-Doppler feature extraction using the spectrogram and the cepstrogram," 2014 11th European Radar Conference, 2014, pp. 165-168, doi: 10.1109/EuRAD.2014.6991233.

[5] H. Sun, B. -S. Oh, X. Guo and Z. Lin, "Improving the Doppler Resolution of Ground-Based Surveillance Radar for Drone Detection," in IEEE Transactions on Aerospace and Electronic Systems, vol. 55, no. 6, pp. 3667-3673, Dec. 2019, doi: 10.1109/TAES.2019.2895585.

[6] B. -S. Oh, X. Guo and Z.Lin, A UAV classification system based on FMCW radar micro-Doppler signature analysis, Expert Systems with Applications, Volume 132, 2019, Pages 239-255, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2019.05.007.

[7] M. Jahangir, C. J. Baker and G. A. Oswald, "Doppler characteristics of micro-drones with L-Band multibeam staring radar," 2017 IEEE Radar Conference (RadarConf), 2017, pp. 1052-1057, doi: 10.1109/RADAR.2017.7944360.

[8] M. Jahangir and C. Baker, "Robust Detection of Micro-UAS Drones with L-Band 3-D Holographic Radar," 2016 Sensor Signal Processing for Defence (SSPD), 2016, pp. 1-5, doi: 10.1109/SSPD.2016.7590610.

[9] Ritchie, M., Fioranelli, F., Borrion, H. and Griffiths, H. (2017), Multistatic micro-Doppler radar feature extraction for classification of unloaded/loaded micro-drones. IET Radar Sonar Navig., 11: 116-124. https://doi-org.tudelft.idm.oclc.org/10.1049/iet-rsn.2016.0063

[10] Rahman, S. and Robertson, D.A. (2020), Classification of drones and birds using convolutional neural networks applied to radar micro-Doppler spectrogram images. IET Radar Sonar Navig., 14: 653-661. https://doi-org.tudelft.idm.oclc.org/10.1049/iet-rsn.2019.0493

[11] S. Björklund, "Target Detection and Classification of Small Drones by Boosting on Radar Micro-Doppler," 2018 15th European Radar Conference (EuRAD), 2018, pp. 182-185, doi: 10.23919/EuRAD.2018.8546569.

[12] D. A. Brooks, O. Schwander, F. Barbaresco, J. Schneider and M. Cord, "Temporal Deep Learning for Drone Micro-Doppler Classification," 2018 19th International Radar Symposium (IRS), 2018, pp. 1-10, doi: 10.23919/IRS.2018.8447963.

[13] A. Huizing, M. Heiligers, B. Dekker, J. de Wit, L. Cifola and R. Harmanny, "Deep Learning for Classification of Mini-UAVs Using Micro-Doppler Spectrograms in Cognitive Radar," in IEEE Aerospace and Electronic Systems Magazine, vol. 34, no. 11, pp. 46-56, 1 Nov. 2019, doi: 10.1109/MAES.2019.2933972.

[14] A. Bochkovskiy, C. -Y. Wang and H. -Y. M. Liao, YOLOv4: Optimal Speed and Accuracy of Object Detection. 2020. arXiv: 2004.10934 [cs.CV], 2020, doi: 10.48550/ARXIV.2004.10934

[15] Robin Radar Systems B.V., IRIS Counter Drone Radar. 2021. URL: https://www.robinradar.com/ iris-counter-drone-radar (visited on 18/07/2022).

[16] J.R. Guerci. Space-Time Adaptive Processing for Radar, Second Edition. Artech House radar library. Artech House Publishers, 2014. ISBN: 9781608078219. URL: https://books.google.nl/books?id=5efmBgAAQBAJ

[17] Lin, Tsung-Yi & Maire, Michael & Belongie, Serge & Hays, James & Perona, Pietro & Ramanan, Deva & Dollár, Piotr & Zitnick, C.. (2014). Microsoft COCO: Common Objects in Context.

[18] Buslaev, A.; Iglovikov, V.I.; Khvedchenya, E.; Parinov, A.; Druzhinin, M.; Kalinin, A.A. Albumentations: Fast and Flexible Image Augmentations. Information 2020, 11, 125. https://doi.org/10.3390/info11020125

[19] Richards, M.A. Fundamentals of Radar Signal Processing, Second Edition. McGraw-Hill Education, 2014, ISBN: 9780071798327, URL: https://books.google.nl/books?id=qGZCAQAACAAJ

B

Color map differences

In this appendix, the preliminary performances obtained for different colormaps are presented, namely for the JET and HOT colormaps. Additionally, the model is tested on an unlabeled video to asses real-time performances.

B.1. Static radar simple dataset

The dataset contained 180 pictures of drones and 100 pictures of clutter, split between training and validation with a $70 - 30\%$ ratio. The model was trained over 40 epochs with SGD optimizer. The weights were initialized from a checkpoint after 300 epochs on the COCO dataset, as described in Subsection 5.4.1. Moreover, the data augmentation methods described in Subsection 5.4.2 were used during training.

Let us look at the preliminary results of the chosen technique applied on an easy dataset recorded with a static radar, as highlighted in Table 4.6, and that contains only two objects: drone vs distinct clutter, i.e. objects that are not resembling drones at all. This simple test serves as an early assessment to verify whether computer vision can indeed predict accurate bounding boxes around targets based solely on low-resolution & low contrast range-Doppler images captured using surveillance radars.

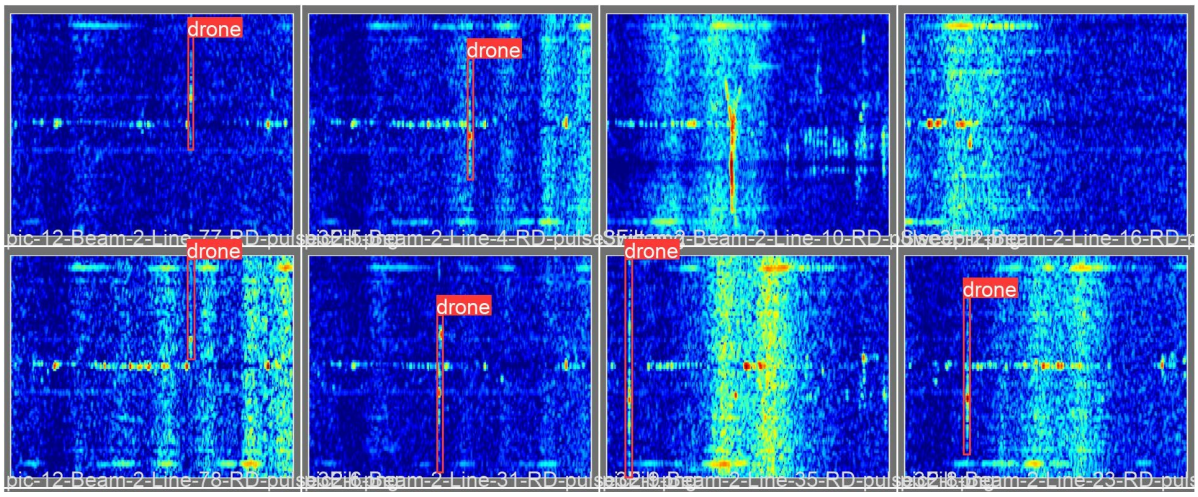


Figure B.1: JET colormap ground-truth for a single batch.

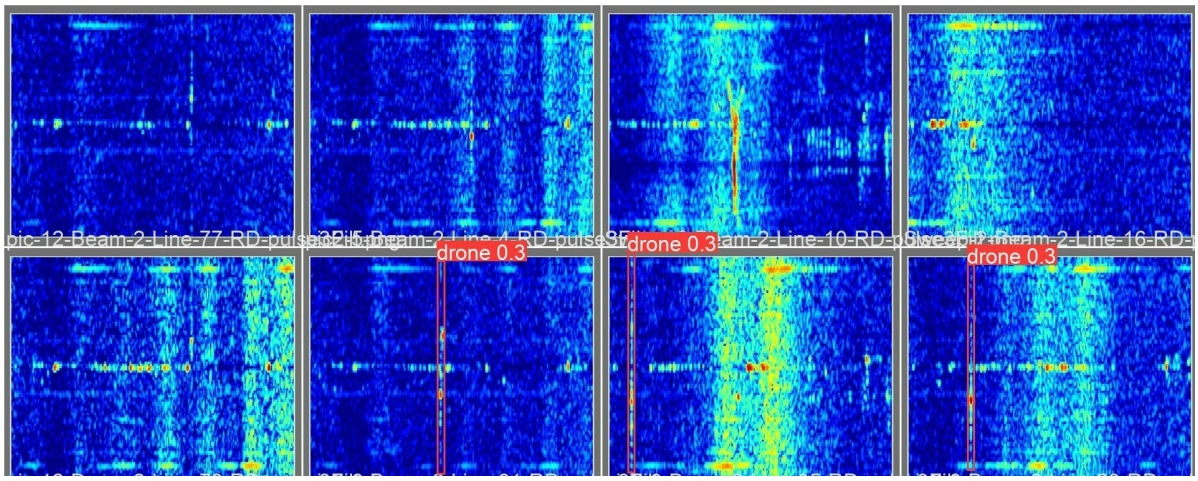


Figure B.2: JET colormap YOLO predictions for a single batch.

Without any extra image processing, Fig. B.1 shows the ground-truth labels of drones, juxtaposed by clutter-only pictures. In Fig. B.2 the predictions generated by YOLO after training show that while the pipeline can properly predict the bounding boxes in some cases, it fails to properly detect the UAV drown in clutter (bottom left picture), or when the micro-Doppler is not clearly visible (top left picture). Given the pixel values between the drone propellers are very close to the background noise, the contrast is low and the classifier can not distinguish the target of interest from clutter.

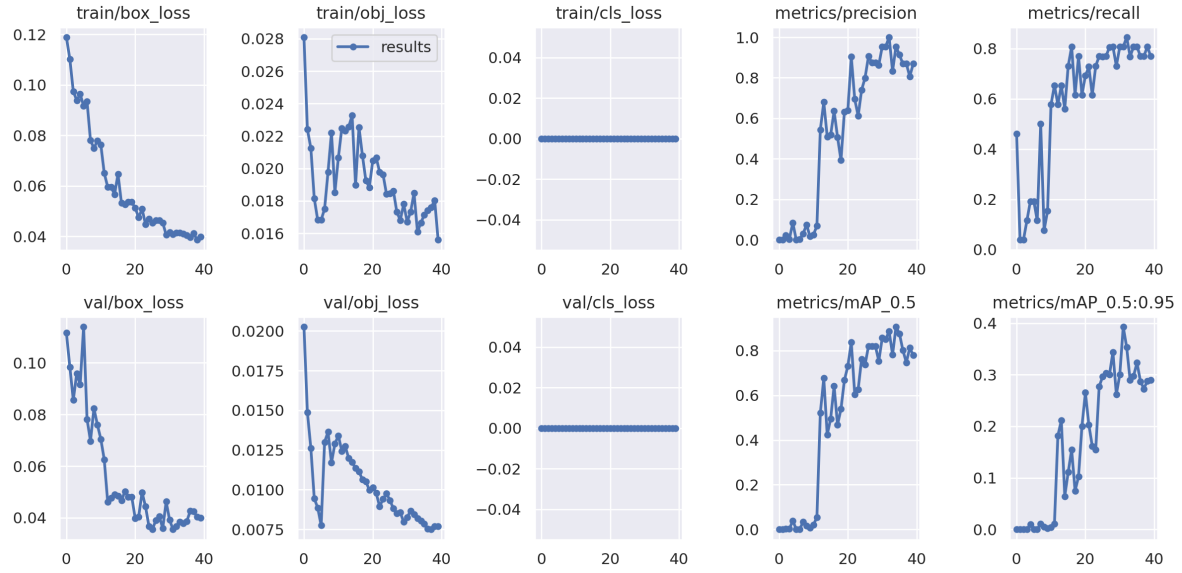


Figure B.3: JET colormap YOLO performance metrics as a function of each epoch (in this case, up to 40 epochs in total).

In Fig. B.3 the performance metrics are highlighted and summarized in Table B.1. While the precision, recall and MAP scores are relatively high and the losses are relatively small, the model fails to capture true positives. The main initial conclusions is that the performances may be increased with more training over more epochs, more diversified data, but also better data pre-processing.

B.2. Static radar complex dataset

Compared to the previous model, the new dataset contains the following novelties:

- An improved contrast with an increased noise-floor level;
- More diversified data with 400 clutter images that may yield possible false alarms due to similar micro-Doppler modulations. The object detection problem remains single-class, i.e. localizing and classifying drones;
- The number of training epochs is increased from 40 to 60.

The novel images with their ground truth labels are seen in Fig. B.4, while the predictions are seen in Fig. B.5. The new model is able to accurately predict the bounding boxes of almost all drones, with the exception of suspected drones which may have have very narrow micro-Doppler modulations (top left plot from Fig. B.5). Nonetheless, the classifier can catch much shallower target signatures that in the previous case were completely missed.

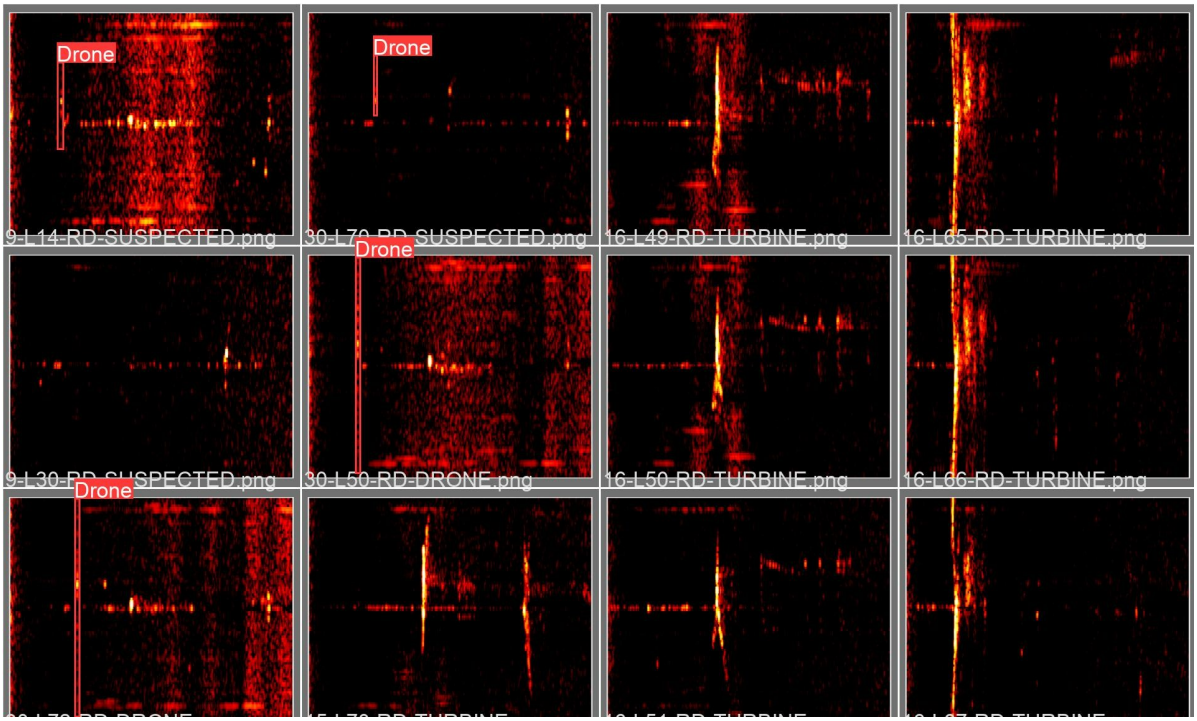


Figure B.4: HOT colormap ground-truth for a single batch.

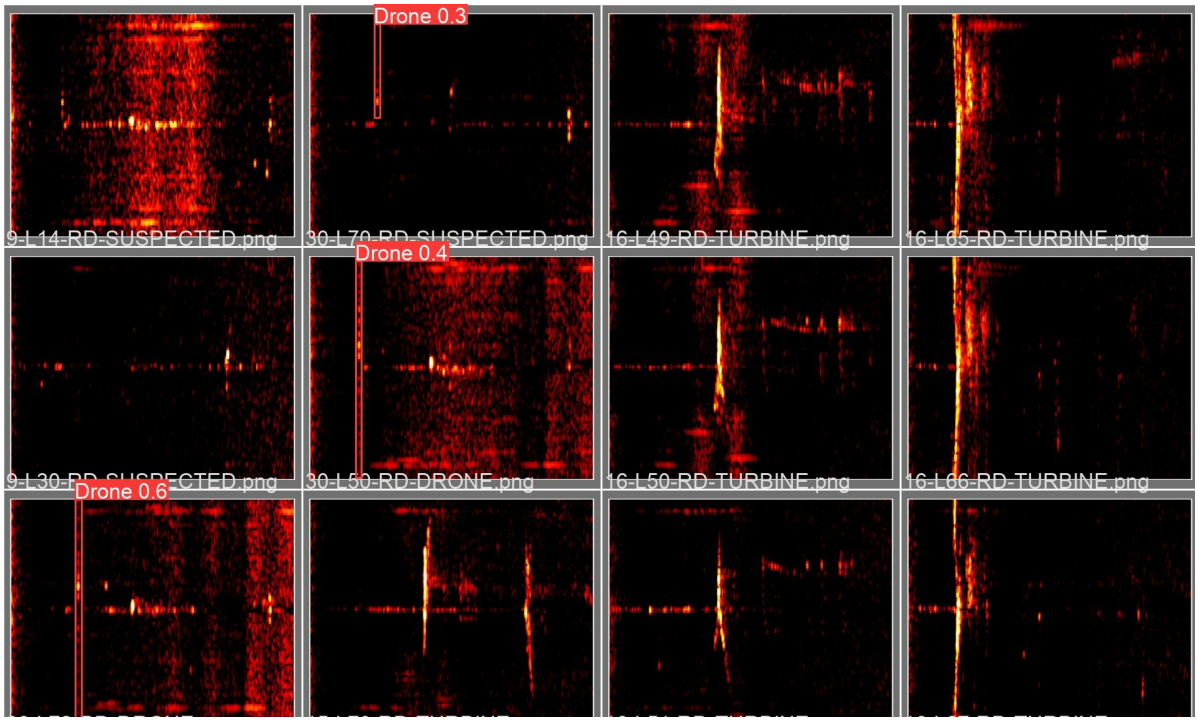


Figure B.5: HOT colormap YOLO predictions for a single batch.

It is then clear that thanks to the extra image processing step, the increased number of training epochs and showing the model multiple clutter images, the new model performs better and is able to capture drones with a much narrower micro-Doppler. Before looking at the performance metrics, it is worth noting that predicted labels scores are twice of the previous model. Moreover, the model does not yield false positives for strong, reflective wind turbines that show wide Doppler modulations.

The performance metrics can be seen in Fig. B.6 and summarized in Table B.1. While the overall precision and recall scores are similar, the objectness loss is significantly smaller. This suggests that there is a much higher probability that an object exists within the regions of interest. This can further be seen in the predicted bounding boxes, where the label score increases twice in the clear plots, while it retains the same prediction score for narrower, lower resolution micro-Doppler plots.

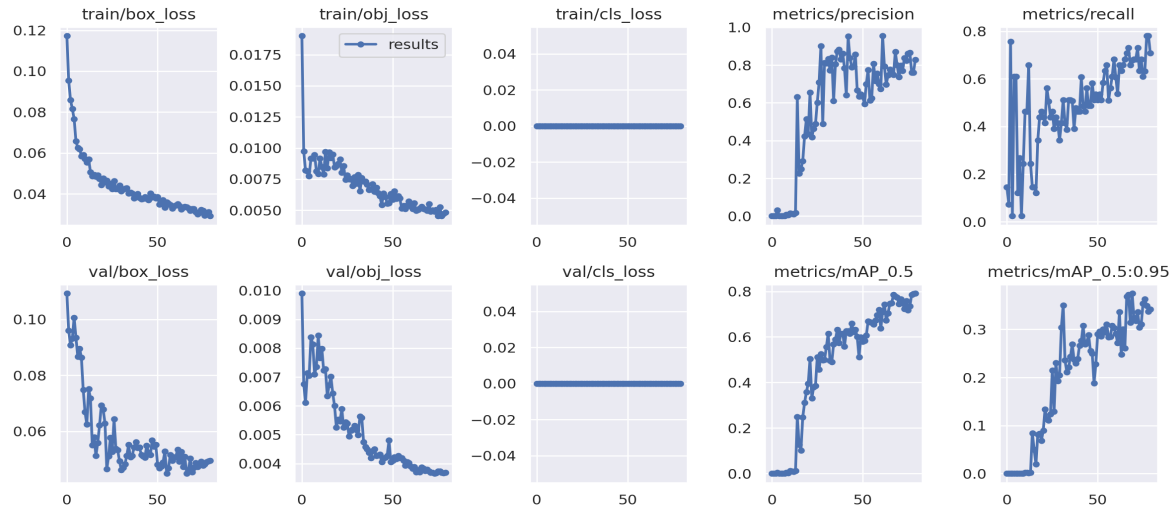


Figure B.6: HOT colormap YOLO performance metrics as a function of each epoch (in this case, up to 60 epochs in total).

B.2.1. Test on real-time range-Doppler video

The final test was done on 20 consecutive full rotations of the radar. The trained classifier from the aforementioned section was fed a video of approximately 40 seconds of consecutive range-Doppler plots with a drone flying back and forth, unseen before and which did not contain any ground truth for validation. Each video frame consists of a CPI of 100 sweeps, or one line, covering $4.5°$ in azimuth, as discussed in Subsections 4.1.3 and 4.2.2. Thus, a full rotation of 80 lines yields 80 separate range-Doppler plots, each containing different possible targets. An example of this test can be seen in Fig. B.7, where the classifier accurately detected the main body of the drone as well as the propellers.

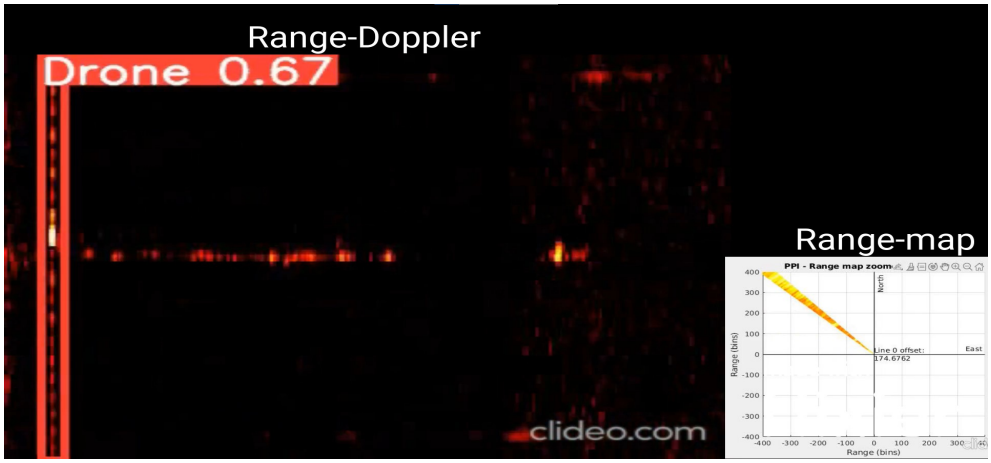


Figure B.7: Detection of a drone based on the range-Doppler plot from a single in a frame from multiple consecutive lines and rotations. The polar range map for a line pointing towards a specific azimuth can be seen in the lower right corner.

This test was not meant to verify the precision of this technique, but rather assess its real-time capabilities and scalability on unlabeled data as the last part of the feasibility study. Moreover, the model used was not the best possible one, but this test highlighted the following aspects with regards to the classifier:

- It performs very well on real-time systems that yield almost instantaneous detection decisions;
- It is able to catch drones that are relatively far away, up to 1.3 kilometers, and have weak micro-Doppler modulations, but may also yield more false alarms;
- It yields false positives on faraway wind turbines that yield weak wide micro-Doppler modulations similar to those of drones.

The problem with wind turbines can be easily fixed by verifying whether the detection is present in all beams with a simple *if* loop since it is the only possible extended target that yields such modulations. However, such problems are to be expected in real-life scenarios.

To emphasize, better models can be created that will remove the false alarms by further improving the average precision. To this end, more images are required to diversify the learning process, but obtaining such data is a time consuming process that requires a lot of memory given 99% of the data saved being clutter, as well as trained engineers and technicians to operate the radars and go onto the field. Another option would be to synthetically create more data, or increase the number of epochs, as described in the following chapter.

B.3. Performance comparison between color maps

The differences of the performance metrics between the two color maps are highlighted in Table B.1, where albeit most of them being similar, the objectivness loss is much lower for the improved dataset.

This improvement means that there is a much higher probability that an object exists within the detected boxes. Moreover, the lower recall score confirms the multiple missed detections for the JET color map.

Simple dataset with JET colormap performance metrics				
Precision	Recall	Mean Average Precision	Box loss	Objectness loss
0.88	0.74	0.8	0.04	0.016
Complex dataset with HOT colormap performance metric				
Precision	Recall	Mean Average Precision	Box loss	Objectness loss
0.88	0.79	0.8	0.04	0.0075

Table B.1: Summary of the YOLO model trained on the simple data set using JET colormap which yielded poor contrast, versus the complex dataset with HOT colormapthat used an improved image pre-processing step to improve the contours.

Thus, the improved model is more robust with respect to the background noise and clutter, as it is accurately localizing multiple drones with weaker Doppler signatures. Moreover, it is able to *see* more narrow micro-Doppler modulations, which in turns improves the classification range from approximately 800 meters up to approximately 1.3 kilometers.