UNIVERSIDAD POLITÉCNICA DE MADRID
Escuela Técnica Superior de Ingenieros Industriales

Machine Learning-Based Solution For Automatic Border Surveillance System

DOCTORAL THESIS

Submitted for the degree of Doctor by:

Khalifa Bellazi
Master of Telecommunications and Computer Networks Engineering from
London South Bank University

Madrid, 2024

UNIVERSIDAD POLITÉCNICA DE MADRID
Escuela Técnica Superior de Ingenieros Industriales

Doctoral Degree in Automatic Control and Robotics

Machine Learning-Based Solution For Automatic Border Surveillance System

DOCTORAL THESIS

Submitted for the degree of Doctor by:

Khalifa Bellazi
Master of Telecommunications and Computer Networks Engineering from London South Bank University

Under the supervision of:
Dr. Teresa Riesgo Alcaide
Dr. José Manuel Lanza Gutiérrez

Madrid, 2024

Title: Machine Learning-Based Solution For Automatic Border Surveillance System
Author: Khalifa Bellazi
Doctoral Programme: Automatic Control and Robotics

Thesis Supervision:

Dr. Teresa Riesgo Alcaide, Full Professor, Universidad Politécnica de Madrid, (Supervisor)

Dr. José Manuel Lanza Gutiérrez, Professor, Universidad de Alcála, (Supervisor)

External Reviewers:

Thesis Defense Committee:

Thesis Defense Date:

This thesis is dedicated to the memory of my mother. Although she was my inspiration to pursue my doctoral degree, she was unable to see my graduation. This is for her. Also, I would like to dedicate this thesis to my father, sisters and brothers who continually provide their moral, spiritual, emotional, and financial support. This thesis is dedicated to my wife who encouraged me to pursue my dreams and finish my thesis.

Acknowledgements

I would like to express my deepest gratitude and appreciation to all those who have been instrumental in the successful completion of this Ph.D. thesis:

First and foremost, I would like to extend my heartfelt thanks to my supervisors (Professor. Teresa Riesgo Alcaide and Professor. Jose M. Lanza-Gutierrez), for their guidance, mentorship, and invaluable support. Their expertise and dedication have been instrumental in shaping the direction of this research and ensuring its successful completion. I would also like to thank the Coordinator PhD program on Automation and Robotics, Professor. Manuel Ferre for his support of my study. Also, I would like to express my sincere gratitude to Rodrigo Marino for his invaluable contribution to completing the evaluation of the experiment in Section 5.3 of Chapter 5 of this thesis. Rodrigo's expertise and dedication were instrumental in achieving the successful completion of this task. I am truly grateful for his assistance and collaboration in this task.

Furthermore, I would like to acknowledge the support and assistance provided by the staff and faculty of Universidad Politécnica de Madrid. Their cooperation and willingness to assist have been pivotal in facilitating the smooth progress of this research.

I extend my heartfelt gratitude to my family for their unwavering support and encouragement during this journey. Their constant presence and words of encouragement have been a source of strength and motivation for me. I am deeply thankful for their love, guidance, and belief in my abilities, which have been instrumental in helping me navigate through challenges and achieve my goals.

In conclusion, this research would not have been possible without the assistance and contributions of the aforementioned individuals and the Universidad Politécnica de Madrid. I am sincerely grateful for their involvement and support in this endeavour.

Abstract

Developing and designing border surveillance systems to meet specific needs and requirements is a comprehensive process that involves careful assessment, customization, and consideration of environmental and operational factors. These systems are crucial for most nations worldwide, as they can provide real-time monitoring of vast areas, including remote and challenging terrains, which might pose a challenge for surveillance systems. Covering extensive areas with remote and difficult terrains presents significant challenges due to power limitations and high costs. To address the challenges faced by current border surveillance systems, such as power limitations, alternative energy sources and energy-efficient technologies are proposed. Additionally, cost management is achieved through the careful selection of equipment and modular designs. This enables surveillance systems for terrestrial environments to operate effectively in large, remote, and challenging terrains.

The thesis's contribution to the field of surveillance systems focuses specifically on monitoring the Libyan Desert border. It highlights the unique approach of the research and its relevance to addressing challenges in a specific geographical and environmental context. The system utilizes unmanned fixed platforms equipped with infrared cameras (FLIR) and employs edge computing in the Internet of Things (IoT) framework. Within this framework, two Automatic Target Recognition (ATR) systems based on machine learning algorithms, specifically Bag-of-Features for feature extraction and supervised classification, are implemented. These run on low-power microprocessors to address the energy and computing capacity limitations of IoT edge nodes. The first proposed approach segments the infrared image into regions of interest before processing, while the second ATR system works directly with the entire image. To evaluate the performance of these ATR systems, a dataset of images specifically relevant to the Sahara Desert environment is used, allowing comprehensive testing and assessment of the system's capabilities.

In this evaluation process, both approaches are considered in combination with four different classification algorithms: Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree (DT), and Naive Bayes (NB), along with three descriptors: SURF, SIFT, and ORB. As a conclusion of this experimental process, the use of generic classes is recommended due to the low resolution of infrared images. Additionally, the SURF-SVM prediction approach based on regions of interest achieved the highest detection capacity, reaching up to 97%, with frame rates of up to 5.71 on the IoT edge device and 59.17 on the workstation. This approach, focusing on classifying three generic categories (animal, vehicle, and person), resulted in reduced confusion between classes compared to identifying specific targets. By employing generic categories, the system achieved an increase in detection capacity. These results demonstrate the feasibility of using edge computing for border surveillance, even in challenging environments like the Sahara Desert.

Resumen

Desarrollar y diseñar sistemas de vigilancia fronteriza para cumplir con necesidades y requisitos específicos es un proceso integral que implica una evaluación cuidadosa, personalización y consideración de factores ambientales y operativos. Estos sistemas son importantes para la mayoría de las naciones en todo el mundo, ya que pueden proporcionar monitoreo en tiempo real de grandes áreas, incluyendo terrenos remotos y difíciles, que podrían ser un desafío para los sistemas de vigilancia. Cubrir grandes áreas con terrenos remotos y difíciles plantea desafíos significativos debido a limitaciones de energía y altos costos. Para abordar los desafíos enfrentados por los sistemas actuales de vigilancia fronteriza, como limitaciones de energía, se proponen fuentes de energía alternativas y tecnologías eficientes en energía, además de gestionar costos mediante la selección cuidadosa de equipos y diseños modulares. Esto permite que los sistemas de vigilancia para entornos terrestres operen de manera efectiva en terrenos grandes, remotos y difíciles.

La contribución de la tesis al campo de los sistemas de vigilancia se centra específicamente en el monitoreo de la frontera del Desierto Libio. Destaca el enfoque único de la investigación y su relevancia para abordar desafíos en un contexto geográfico y ambiental específico. El sistema utiliza plataformas fijas no tripuladas equipadas con cámaras de infrarrojos (FLIR) y emplea computación en el borde del Internet de las Cosas. Dentro de este marco, se implementan dos sistemas de Reconocimiento Automático de Objetivos (ATR) basados en algoritmos de aprendizaje automático, específicamente Bag-of-Features para la extracción de características y el uso de clasificación supervisada. Estos se ejecutan en microprocesadores de baja potencia para abordar las limitaciones de energía y capacidad de cómputo de los nodos en el borde del Internet de las Cosas. El primer enfoque propuesto segmenta la imagen de infrarrojos en regiones de interés antes del procesamiento, mientras que el segundo sistema ATR trabaja directamente con la imagen completa. Para evaluar el rendimiento de estos sistemas ATR se utiliza un conjunto de datos de imágenes específicamente relevante para el entorno del Desierto del Sáhara, permitiendo pruebas exhaustivas y evaluación de las capacidades de los sistemas.

Para llevar a cabo este proceso se evaluación, se consideraron ambas aproximaciones en su combinación de cuatro algoritmos de clasificación diferentes, Máquina de Soporte Vectorial (SVM), Vecinos Más Cercanos (KNN), Árbol de Decisiones (DT) y Naive Bayes (NB), en combinación con tres descriptores, SURF, SIFT y ORB. Como conclusión de este proceso experimental, se recomienda utilizar clases genéricas debido a la baja resolución de las imágenes de infrarrojos. Además, el enfoque de predicción basado en el uso de regiones de interés SURF-SVM logró la mayor capacidad de detección, alcanzando hasta un 97%, con frame rates de hasta 5.71 en el dispositivo de borde del Internet de las Cosas y 59.17 en la estación de trabajo. Este enfoque, que se centra en clasificar tres categorías genéricas (animal, vehículo y persona), resultó en una disminución de la confusión entre clases en comparación con la identificación de objetivos específicos. Al emplear categorías genéricas, el sistema logró un aumento en la capacidad de detección. Estos resultados muestran la viabilidad de utilizar la computación en el borde para la vigilancia fronteriza, incluso en entornos desafiantes como el Desierto del Sáhara.

Table of Contents

List of Figures

List of Tables

List of Codes

List of Acronyms

API	Application Programming Interface
ANN	Artificial Neural Network
AI	Artificial Intelligence
ANN	Artificial Neural Network
ATD	Automatic Target Detection
ATR	Automated Target Recognition
ATDR	Automatic Target Detection and Recognition
BRIEF	Binary Robust Independent Elementary Features
BoF	Bag-of-Features
BoVW	Bag of Visual Words
BSS	Border Surveillance System
CCTV	Closed-circuit television
CPU	Central Processing Unit
CNN	Convolutional Neural Network
CV	Computer Vision
DL	Deep Learning
Deep-IRTarge	An Automatic Target Detector in Infrared Imagery Using Dual-Domain Feature Extraction and Allocation.
DBS	Distributed Border Surveillance

DCNN	Deep Convolutional Neural Network
DT	Decision Tree
DVR	Digital Video Recording
EO	Electro-Optical
EIR-ViT	Edge Infrared Vision Transformer
IR	Infra-Red
ECM	extended confusion matrix
Faster R-CNN	Faster Region-based Convolutional Neural Network
FPN	eature Pyramid Network
FPS	frames per second
FLIR	Forward Looking Infra-Red
FP	False Positive
FN	False Negative
FPGA	Field-Programmable Gate Array
GPS	General Problem Solver
GPT3	Generative Pre-trained Transformer 3
GPU	Graphics Processing Unit
CogVSM	cognitive video surveillance management
GPS	Global Positioning System

ISS	Intelligent Surveillance Systems
IBS	Intelligent Border Surveillance System
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IoT	Internet of Things
IIoT	Industrial Internet of Things
IP	Internet Protocol
INES	Intelligent Edge Surveillance
HIFT	Hypercomplex Infrared Fourier Transform
IR	Infra-Red
K-NN	K-Nearest-Neighbor
KCF	Kernelized Correlation Filter
LSTM	long short-term memory
LIDAR	Long Range Radar and Laser
LIDAR	Laser Imaging Detection and Ranging
LWIR	Long wave infrared
ML	Machine Learning
mAP	mean Average Precision
MWIR	Medium-Wave Infrared
NLP	Natural Language Processing

NB	Naive Bayes
NCIE	nonlinear correlation information entropy
ORB	Oriented fast and Rotated brief
QCF	Quadratic Correlation Filters
ROI	Region of Interest
ROC	Receiver Operating Characteristic
RBF	Radial Basis Function
RES	RealEdgeStream
ResNet	Residual Network
ROIs	Regions of interest
JSR	sparse representation
SCVS	Smart-city Video Surveillance
SVS	smart video surveillance
SAR	Synthetic Aperture Radar
SIFT	Scale Invariant Feature Transform
SDD	Single Shot MultiBox Detector
SCS	Security Camera Systems
SURF	Speeded Up Robust Features
SVM	Support Vector Machine

SSD	Single Shot MultiBox Detector
SWIR	Short-Wave Infrared
SCS	Surveillance Camera System
TP	True Positive
TN	True Negative
PoE	Power over Ethernet
PD	Probability of Detection
Pfa	Probability of False Alarm
VSS	Video Surveillance System
VCE	vehicular edge computing
VPU	vision processor unit
VGG	Visual Geometry Group
VPU	vision processor unit
UAS	Unmanned Aircraft System
UV	unmanned vehicles
UAV	Unmanned Aerial Vehicle
USA	United States of America
WSN	Wireless Sensor Network
YOLO	You only look Once

Chapter 1

Introduction

This introductory chapter sets the stage for the subsequent discussions by establishing the context, motivation, objectives, and scope of the research. It lays the foundation for a comprehensive exploration of Border Surveillance System (BSS) and their intersection with Artificial Intelligence (AI) technologies. Highlighting the role of AI in shaping the evolution of BSS.

The development of an autonomous BSS using cutting-edge technologies such as Machine Learning (ML) and AI is the focus of the doctoral thesis, whose main aim is to address the challenges of surveillance in complex environments like the Sahara desert by embedding ML methods for vision-based applications in fixed Long wave infrared (LWIR) cameras to detect intruders. These LWIR cameras work within the edge computing to shift computation tasks from the cloud to the edge, resulting in a real-time BSS with high accuracy and low latency.

The term "AI" or "Artificial Intelligence" has indeed evolved over time and now has become a broad and encompassing field that covers various techniques and approaches. Originally, AI was associated with symbolic methods, such as formal logic and ontologies. However, as technology advanced, the scope of AI expanded to include statistical methods like ML, data mining, and probabilistic models.

In the context of this Thesis, AI is specifically associated with ML techniques. This is a common understanding in contemporary AI discussions. ML is a subset of AI that focuses on developing algorithms and models that enable computers to learn patterns and make decisions without being explicitly programmed based on data.(iDuan, 2023) ML can be categorized into various types, and one major categorization is supervised learning. In this type of ML, the algorithm is trained on a labelled dataset, meaning that the input data has corresponding output labels. The algorithm learns to map the input data to the correct output during training. This is widely used for tasks like classification and regression.

The subsequent section will provide a brief history of AI within the realm of surveillance technology, which is distinguished by noteworthy progress in security and surveillance systems.

1.1 The Past, Present and Future of Artificial Intelligence

The historical stages in AI research and development have paved the way for the most recent achievements and contributions to the field of BSS. In this section, a brief explanation is provided of the important historical stages in AI research and development.

Certainly, the field of AI has witnessed significant milestones and advancements over the past several decades, the roots of ML can be traced back to the 1940s and 1950s when early concepts of AI were being explored. Starting with the definition of the first mathematical model of a neuron in the Electronic Brain of (McCulloch and Pitts, 1943) Alan Turing's work laid the foundation for computational models of learning and intelligence. The 1950s saw the establishment of foundational concepts in AI and ML Researchers began exploring the idea of creating machines that could mimic human intelligence and perform tasks traditionally requiring human cognition. This era was characterized by the study of symbolic reasoning, logic, and problem-solving, laying the groundwork for the development of AI programs. The term "artificial intelligence" was coined at the Dartmouth Conference in 1956, organized by John McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon (McCarthy et al., 1956), this conference is often considered the birth of both AI and ML as fields of study, the conference brought together researchers from various fields to discuss the possibility of creating intelligent machines and laid the foundation for AI research. In the late 1950s and early 1960s, researchers developed some of the earliest AI programs, focusing on tasks like logical reasoning, language translation, and problem-solving. One notable example is the Global Positioning System (GPS), developed by Allen Newell and Herbert Simon in 1957 (Newell et al., 1958). GPS was an early attempt to create a computer program capable of solving a wide range of problems using heuristic search techniques. Arthur Samuel coins the term “machine learning”, and his contributions laid the groundwork for many modern machine learning algorithms and reinforcement learning techniques, pioneering work in ML set the stage for future developments in the field (Samuel, 1959). In the year 1957 Ray Solomonoff's pioneering work in algorithmic information theory has played a crucial role in advancing our understanding of information, prediction, and compression, and has influenced numerous areas of research, including ML and AI. The emergence of expert systems in the 1970s and 1980s represented a significant step forward in the automation of decision-making processes, including those related to surveillance. The advent of probabilistic and statistical methodologies within the field of ML signified a significant milestone in the field's development. These approaches revolutionized the way researchers approached complex problems, introducing formal frameworks for handling uncertainty and variability in data. These approaches continue to be foundational in modern ML research and applications, driving advancements in areas ranging from natural language understanding to computer vision (Karishma et al., 2023). In the context of surveillance, the systems could analyze sensor data and make decisions based on predefined rules. provided the necessary information to develop rules for detecting and responding to potential threats. And analyze data from various sources, such as cameras and sensors, and make decisions about the presence of suspicious activities or individuals. While the systems had limitations in scalability and adaptability, they paved the way for future advancements in artificial intelligence and machine learning, leading to more sophisticated approaches to automated surveillance and security.

In the 1990s and 2000s, the advent of video surveillance systems marked a significant advancement in the field of security and surveillance. This period saw the integration of AI algorithms to enhance the capabilities of surveillance systems. These systems provided advanced capabilities for detecting, recognizing, and classifying events in real-time, thereby enhancing the effectiveness of security operations and public safety efforts (Dufour, 2012).

In the 2010s and continuing into the present, ML has revolutionized surveillance technology, enabling advanced capabilities such as facial recognition, behaviour analysis, anomaly detection, and predictive analytics. These advancements have greatly enhanced the effectiveness and efficiency of surveillance systems, contributing to improved public safety and security across various domains (Elharrouss et al., 2021),(Yu et al., 2011).

Deep Learning (DL), a subset of ML that relies on neural networks, experienced a notable resurgence during the 2010s. This resurgence can be attributed to the breakthroughs achieved in image and speech recognition through the utilization of deep neural networks (Soundarya et al., 2023). A significant turning point for DL came in 2012 with the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). It was during this challenge that Deep Convolutional Neural Networks (DCNNs), such as AlexNet, showcased an unprecedented level of performance in image classification (Sharma, 2023).

The trend regarding the continuous growth of computational resources and the increasing number of parameters in trained models is a key characteristic of recent advancements in ML. This trend has significantly impacted the capabilities and performance of ML models across various domains. Recently, the intersection of ethical AI activism and the development of billion-parameter models has become a significant area of discussion and concern, exemplified by Generative Pre-trained Transformer 3 (GPT3) (Brown et al., 2020), marks a critical juncture in the field of Natural Language Processing (NLP) and AI as a whole.

The evolution of ML techniques has been closely intertwined with the parallel growth of hardware resources (Bavikadi et al., 2022). The advancements in ML algorithms, especially in DL, have been made possible by the availability of increasingly powerful and specialized hardware (Ali et al., 2022). The performance increase in Central Processing Unit (CPU) and Graphics Processing Units (GPUs) has played a pivotal role in the accelerated growth of ML methods (Nishanth et al., 2022).

In recent years have witnessed significant technological advancements in LWIR cameras, especially in their capability to capture thermal signatures and heat patterns. These advancements collectively contribute to the growing versatility and effectiveness of LWIR cameras, making them integral tools in various fields, including border monitoring. Furthermore, the enhancement in hardware and materials improvements have led to the successful development of a BSS. The evolution of uncooled LWIR imaging sensors has not only strengthened defence capabilities but has also led to a significant expansion into the consumer market. This shift has implications for various industries, creating opportunities for innovation (Jordan et al., 2023; Guan et al., 2021).

The recent advancements in AI, particularly in the field of computer vision, contributed significantly to more accurate and efficient object recognition in surveillance imagery. This has direct relevance to BSS, especially in terms of target recognition and detection.

The last decade has seen exponential growth in the research and application of sensor technologies, drones and Unmanned Aerial Vehicle (UAV), as well as the integration of AI and Machine Learning ML in BSS (Goyal et al., 2020; Ning et al., 2023; Laouira et al., 2021a; Yang, 2023; Blasch et al., 2021; Singh and Singh, 2022).

You only look Once (YOLO), Faster Region-based Convolutional Neural Network (Faster R-CNN) and similar architectures enhance real-time object detection. also, ML algorithms play a role in data fusion, combining information from various sensors for better decision support in surveillance systems. Optimized ML algorithms, especially those leveraging hardware acceleration, contribute to the real-time processing of video feeds in surveillance applications.

So the historical evolution of ML, from early symbolic AI to the recent dominance of DL, has significantly shaped the capabilities of systems relevant to border surveillance. Advancements in object recognition, target detection, and real-time processing have direct implications for enhancing the effectiveness of surveillance technologies in border security.

The present Doctoral Thesis aims to contribute to the various aspects of Intelligent Border Surveillance Systems (IBSs) by working on the integration of ML and Computer Vision (CV) for evolving new and innovative approaches to enhance border surveillance capabilities technology. The combination of sophisticated algorithms advanced sensing capabilities, and innovative connectivity solutions is significantly enhancing BSS. A cornerstone of Intelligent Border Surveillance Systems (IBSs) is object detection and tracking, enabled by ML and CV techniques. These technologies enable the real-time identification and monitoring of objects of interest, such as individuals, vehicles, and animals, even in challenging environments characterized by low visibility and complex terrain.

This thesis follows this hypothetical strategy for developing a surveillance system to be deployed in the Sahara Desert, whose border has a wide-area with a low density of crossing traffic (Cică and Filipoaia, 2017). Under this focus, the system should put much effort into detecting and classifying moving targets, ignoring stationary ones to save energy and computing needs, meaning low deployment and maintenance costs. This fact, along with a low occurrence of moving targets, means that the surveillance system could be designed considering a low processing speed and a low need for transmitting data over the network because only alerts will be sent, instead of all sensed data, avoiding also security problems. Thus, the solution proposed is built in an edge note of the Internet of Things (IoT) constrained in computing capacity. The sensor in the edge considers a LWIR camera, which is useful for detecting heat a few miles away and is especially powerful when the area has a low density of vegetation, as occurs in the Sahara Desert. The camera images are analyzed by an Automated Target Recognition (ATR) system embedded in the edge note, which is powered by CV and ML technologies, that represent the main concepts in the realm of technology and AI, below is an introduction to some of them:

- **Artificial Intelligence (AI):** refers to the simulation of human intelligence in machines, enabling them to perform tasks that typically require human intelligence, such as visual perception, speech recognition, decision-making, and language translation. AI systems learn from data, identify patterns, and make decisions with minimal human intervention.

- **Machine Learning (ML):** is a subset of AI that focuses on developing algorithms and statistical models that enable computers to learn from and make predictions or decisions based on data. ML algorithms improve their performance over time as they are exposed to more data, allowing them to identify patterns and make decisions without being explicitly programmed.
- **Deep Learning (DL):** is a subset of ML that uses artificial neural networks with multiple layers (deep neural networks) to learn representations of data through hierarchical layers of abstraction. DL algorithms have shown remarkable success in various domains, including image and speech recognition, natural language processing, and autonomous driving.
- **Internet of Things (IoT):** refers to a network of interconnected devices, sensors, and objects that collect and exchange data over the internet. IoT enables physical objects to be connected and controlled remotely, allowing for intelligent automation, monitoring, and optimization of processes in various domains, including smart homes, healthcare, agriculture, and industrial automation.
- **Edge Computing:** is a distributed computing paradigm that brings computation and data storage closer to the source of data generation (the edge), rather than relying on centralized data centres or cloud computing. edge computing enables real-time processing and analysis of data, reducing latency, bandwidth usage, and dependency on cloud services, making it ideal for applications requiring low latency and high availability, such as IoT, autonomous vehicles, and industrial automation.

1.2 Motivation

This thesis presents an empirical solution to the purpose of counter-intruders crossing the borders in the context of a BSS. In this context, two questions arise regarding the specific use case:

- For what purpose is the BSS to be designed?
- What are the characteristics of the border to be protected?

Due to the many challenges, the border security agencies have adopted innovative technologies to help in addressing those issues, such as AI. Then, two additional technological questions arise:

- How effective are new technologies and modern systems in detecting and identifying intruders across borders?
- What are the challenges faced in monitoring cross-border intruders?

This thesis will address all these previous questions to work on the following research question:

- Is it possible to design an alternative accurate and low-computing approach within the edge-computing paradigm for BSS?

To research this topic, it would likely need to employ a mixed-methods approach that combines qualitative and quantitative analyses. The structure of the research is as follows:

- Literature Review: starting by conducting a comprehensive review of existing literature on Border Surveillance Systems (BSSs) and the use of modern technologies in these domains. Identify key technologies such as AI, BSS, target detection, etc., and their applications in Border Surveillance Systems (BSSs).
- Case Studies: Select several case study examples where modern technologies have been implemented to enhance border and surveillance systems. These case studies should cover different geographical regions, types of borders, and security challenges (illegal immigration, drug trafficking, terrorism, etc.).
- Data Collection: Gather data related to the effectiveness of modern technologies in addressing surveillance challenges. This could include quantitative data such as crossing border rates, apprehension statistics, seizure data, etc., as well as qualitative data such as stakeholder interviews, expert opinions, and policy documents.
- Analysis: Analyse the collected data to assess the impact and effectiveness of modern technologies in mitigating security threats. Compare security outcomes before and after the implementation of these technologies, identify trends and patterns, and evaluate the strengths and weaknesses of different technological solutions.
- Challenges and Limitations: Identify and analyse the challenges and limitations associated with the use of modern technologies in Border Surveillance Systems (BSSs). This could include technical challenges, privacy concerns, legal and regulatory issues, financial constraints, etc.
- Policy Implications: Consider the policy implications of the findings. Based on the effectiveness assessment, provide recommendations for policymakers on how to optimize the use of modern technologies in BSS strategies.
- Future Research Directions: Finally, outlining the areas for future research and development in this field. outlining emerging technologies, research gaps, and opportunities for innovation to further enhance border and maritime security in the future.

By following this structured approach, systematically investigate the extent to which modern technologies contribute to addressing BSS challenges and provide valuable insights for policymakers and security practitioners.

1.3 Objectives

The main objective of this thesis is to explore and validate an automated, cost-efficient wide-area surveillance system capable of detecting and classifying targets during both daytime and nighttime in a vast desert border region with minimal vehicular or human activity, which treats any moving object as a potential target. The solution is based on IoT technology, where IoT edge nodes consider fixed Forward Looking Infra-Red (FLIR) cameras for monitoring purposes. Each camera fully executes an intelligence engine powered by ML to detect and

classify targets. The system is divided into two stages the motion detection stage and the Classification stage, in first stage detects motion within the camera's field of view and captures silhouette images of the moving objects when motion is detected. then extracting silhouette images of the moving objects from the captured frames, by Isolating the shapes of the objects from the background, providing clear outlines. finally, Utilize shape-based features derived from the extracted silhouettes to classify the silhouettes based on their shape-based features using classification algorithms to categorize the silhouettes into predefined classes.
Reducing the number of categories to be classified by grouping them into broader classes, such as vehicles, animals, and people, to achieve accurate and real-time object classification in a BSS.

To reach the initial objective, it will be required to reach different sub-objectives related to the parts involved in a computer vision-embedded problem for ATR:

- System Architecture Design: The priority objective is developing a comprehensive system architecture for an intelligent and automated cost-effective surveillance system for long-range border areas with thousands of kilometres of desert with low traffic. This architecture should incorporate the necessary components to eliminate any requirement for human intervention in the system component levels. Then, it will focus on the integration of FLIR cameras as IoT edge nodes for surveillance purposes.

- Adaptive Motion Detection Techniques: The present work will investigate related motion detection techniques and integrate them into a border surveillance context to contribute valuable solutions to this field. Selecting an adequate motion identification algorithm is crucial for saving computing costs and ensuring computing capacity is used when needed. This is especially relevant is an environment whose usual state is having no movements.

- Innovation Machine Learning Models: The present work shall investigate low-cost ML methods for applications that rely on visual data. Techniques based on ML will be utilized and enhanced to make a valuable contribution to the field of target classification for computing vision, thereby enhancing the effectiveness of the methodologies. The final aim will be to implement advanced algorithms for autonomous decision-making and develop an integrated classification model for real-time object categorization with high accuracy.

- Experiment Validation It will be crucial to conduct experiments and validations to verify the findings of the thesis and additionally provide significant insights into the broader domain of border surveillance and the applications of ML.

- Community Contribution: The methods and techniques put forth in this study have been developed to conform to the limitations of reproducibility, along with the emphasis on open-source and open data. Adopting these practices, will not only contribute to the academic community but also the wider community with an interest in border surveillance and ML. Openness and collaboration serve as crucial catalysts for progress in research and technology.

1.4 Thesis outline

This thesis is divided into five chapters. The first chapter provides an introduction and motivation, as well as the objectives of the thesis, and highlights the role of AI in shaping the evolution of BSS. The following chapters are summarized as follows:

- Chapter 2 provides an overview of BSS and introduces ML technologies. Moreover, it introduces the concept of IoT and provides an overview of ATR.
- Chapter 3 provides a review of the current state-of-the-art methods and applications within the field of research. There is a particular focus on approaches for BSS, especially regarding mobile and fixed surveillance platforms as well as sensing sources. State-of-the-art techniques and technologies in ATR are also reviewed, presenting key research challenges.
- Chapter 4 introduces the solutions proposed in the thesis and the evaluation of the results obtained. It highlights the two ATR proposals (the bounding box prediction scheme, the frame-based prediction scheme), the database generated, and the experimental methodology.
- Chapter 5 presents the experimental results and the corresponding discussion for this thesis. It evaluates the two proposals when using different classification techniques. Also, the proposals are evaluated in a real edge platform.
- Chapter 6 presents the overall conclusions for this doctoral thesis and provides an overview of the future work that can be pursued. It includes a detail of the contributions to the field, as well as its diffusion.

Chapter 2

Background

This chapter starts with an overview of the ML technology, including the usual ML pipeline, as well as a detail of its main stages. Next, it is presented the IoT concept, focusing on edge computing and its target platforms. Then, a BSS background is provided highlighting the evolution, challenges, sensing sources, and innovative solutions within the market. It also presents the nature and characteristics of Libya's borders. Finally, an overview regarding ATR is presented.

2.1 Machine Learning Technology

ML is a sub-field of AI technology that empowers computers to learn from data and make predictions and decisions based on that data. The term "machine learning" encompasses the study and development of algorithms capable of making predictions or decisions based on data patterns. Unlike traditional programming, where instructions are fixed, ML algorithms adapt and improve their performance through exposure to sample inputs. ML has become a powerful tool for solving complex problems that cannot be addressed by explicitly programmed algorithms alone. It finds applications across various domains, including pattern recognition, email spam filtering, search result ranking, language translation, and scientific fields such as optics and photonics. (Bertolini et al., 2021) In practice, ML is often intertwined with data mining, a field focused on exploratory data analysis. Within the realm of data analytics, ML plays a crucial role in developing complex models and algorithms for predictive analysis. Commercially, this is known as predictive analytics, enabling researchers, data scientists, and analysts to make reliable decisions and uncover hidden insights by learning from historical data relationships and trends.

ML is a broad field with several subsets or branches, each focusing on specific techniques, tasks, or types of data. Some main subsets of ML include:

- Supervised Machine Learning: In supervised learning, the model learns from labelled data, where each input is associated with a corresponding output label. This allows the model to make predictions on new, unseen data by learning the relationship between input features and target labels (Burkart and Huber, 2021). The computer is presented with examples of input and the corresponding output, provided by a "teacher" or through labelled data. The objective is for the algorithm to learn a general rule or pattern that maps input data to the correct output. The input signal can be fully available, where both the input and output are provided for each example, or partially available, where only the input or output is provided. Additionally, the feedback provided to the algorithm can vary, ranging from simple binary feedback (correct or incorrect) to more nuanced feedback. Supervised learning is named as such because the learning process resembles that of a teacher supervising a student's learning.

- Unsupervised Machine Learning: Unsupervised learning involves finding patterns or structures in unlabelled data. Without explicit labels to guide the learning process, the algorithm identifies inherent structures in the data, such as clusters or latent variables. This can be useful for tasks like clustering similar data points together or reducing the dimensionality of the data while preserving its important features. In contrast to supervised learning, where the algorithm learns from labelled examples with predefined correct answers, unsupervised learning does not have access to such labels. Instead, the algorithm explores the data to identify underlying structures or relationships without explicit guidance (Verma et al., 2022). Enhanced learning, a form of unsupervised learning, occurs in dynamic environments where data is provided in the form of rewards and penalties based on the actions of the program. Examples include driving a car or playing a game against an opponent. In these scenarios, no instructor is providing correct answers, and the algorithm must independently discover meaningful patterns and make decisions based on the available information. Unsupervised learning algorithms are tasked with autonomously uncovering and presenting intriguing structures inherent within the data, even though the data lacks labels. Through this process, the algorithms gain knowledge of the inherent structure present within the input data, which can be valuable for various downstream tasks and analyses.

- Semi-Supervised Machine Learning: Semi-supervised learning falls between supervised and unsupervised learning. In this approach, the computer is provided with an incomplete training set, consisting of a dataset containing both labelled and unlabelled data. While some data points are labelled with their corresponding target outputs, a significant portion of the data remains unlabelled. The challenge in semi-supervised learning is to leverage the limited labelled data alongside the abundance of unlabelled data to improve the learning process. Techniques from both supervised and unsupervised learning can be employed to achieve this goal. The labelled data provides some guidance to the algorithm, helping it learn the underlying patterns in the data, while the unlabelled data allows the algorithm to explore and discover additional patterns. By combining information from both labelled and unlabelled data, semi-supervised learning algorithms can often achieve better performance than purely supervised or unsupervised approaches. This is particularly useful in scenarios where obtaining labelled data is expensive or time-consuming, as it allows for more efficient use of available resources (Yang et al., 2022).

In recent years, DL has indeed risen to prominence as the dominant computational approach within the field of machine learning ML. Considering DL is a branch of ML that employs artificial neural networks to enable computers to perform tasks traditionally carried out by humans. It has gained significant attention and is a key technology behind many modern applications, including self-driving cars, voice assistants, and image recognition systems. In DL, computers learn to perform classification tasks directly from raw data such as images, text, or audio. By leveraging large neural networks with many layers, known as deep neural networks, DL models can achieve high levels of accuracy, often surpassing human performance in certain tasks. Inspired by the structure and function of the human brain, neural networks consist of interconnected layers of artificial neurons. Each neuron receives input signals, applies a transformation, and produces an output signal that is passed to the next layer. The deeper the network, the more layers it has, allowing it to learn increasingly complex representations of the input data (Sen et al., 2020). Training a DL model typically involves feeding it with labelled data and adjusting the weights of the connections between neurons to minimize the difference between the predicted outputs and the true labels. However, there are also unsupervised learning techniques in DL, where the model learns to find patterns and structures in unlabelled data without explicit guidance from human supervisors. DL has revolutionized various fields such as computer vision, speech recognition, natural language processing, and robotics. Recent advancements in hardware, such as Graphics Processing Units (GPUs), and algorithms have significantly accelerated the progress of DL, leading to breakthroughs in tasks like image classification, object detection, and language translation. However, the computing cost, in both the training and prediction stages, of DL solutions is significantly greater than when using purely ML approaches. This is one of the reasons why ML is still considered instead of DL in some environments, such as when developing low-power edge-computing solutions. DL models, with their ability to learn complex patterns from large amounts of data, often require substantial datasets to achieve high performance. On the other hand, traditional ML models, which generally have simpler architectures and fewer parameters, can still perform well with smaller datasets. These models are often more interpretable and require fewer computational resources, making them suitable choices when data availability is limited or when transparency in the decision-making process is important.

2.1.1 Overview of the Machine Learning Pipeline

A ML pipeline streamlines the workflow of a ML task, automating the process from data input to model deployment. This automation is achieved by enabling the conversion and association of a series of data within a model, which can subsequently be analysed to ascertain the desired. Additionally, ML pipelines enable reproducibility and scalability, allowing us to easily replicate and scale their ML workflows across different projects and datasets. Next, it is presented the usual stages in this pipeline within a supervised learning focus (this is the type of system to be designed in this Thesis):

- Data Input: The Data input step is the first step in every ML pipeline, the raw data is collected, organized, and processed in preparation for subsequent stages of the pipeline. The data can come from various sources, such as databases, files, sensors, or Application Programming Interface (API)s.
- Validation of Data The collected data is validated to ensure its quality and suitability for training the model. Various statistical analyses and checks are performed to identify

anomalies or inconsistencies in the data. The statistics of the new data, such as the scope, number of classifications, distribution of subgroups, etc., are the main focus of data validation.

- Pre-processing of data One of the most important phases of each ML pipeline is data pre-processing. Before feeding the data into the model, it undergoes preprocessing, which includes tasks like data cleansing, attribute scaling, quality assessment, and reduction. This stage usually also includes feature extraction (extraction of knowledge in the form of features) and feature selection (selection of the most relevant features) to facilitate the training stage.
- Data Model Training In this central step, the preprocessed data is used to train the ML model. The model is trained to accurately predict the output based on the input data. This stage also includes a final step where the ML solution is evaluated based on indicative metrics for comparison purposes.
- Model Deployment Once the model is trained and evaluated, it is deployed for practical use. Deployment methods include hosting the model on a server, in a browser, or on edge devices. ML pipeline ensures smooth functioning of ML inference at edge-level devices where the data generation plays a crucial part and offers features like lower cost, real-time processing, and increased privacy.

2.1.2 Classification Techniques

Next, a description of the most usual classification algorithms in supervised learning is provided. Note that these techniques are considered in this Thesis.

- K-Nearest-Neighbor (K-NN) is a simple algorithm that stores all the available cases of training data and classifies the new cases according to the majority of its k nearest neighbours based on distance. The value of K is a hyperparameter that needs to be specified beforehand. The choice of distance metric can significantly impact the algorithm's performance and should be chosen based on the characteristics of the dataset. One limitation of K-NN is its computational complexity. Since K-NN does not learn a model during training, it requires storing the entire dataset in memory, making it memory-intensive and computationally expensive, particularly for large datasets. The K-NN method has the advantage of simple and stable performance. But, it requires many samples, as a disadvantage.
- Naive Bayes (NB) is a probabilistic classification algorithm based on Bayes' theorem with strong (naive) independence assumptions between features. It calculates the posterior probability of each class given the input features and selects the class with the highest probability. NB is simple, fast, and works well with high-dimensional data. It is commonly used for text classification tasks, such as spam filtering and document categorization. It assumes that the features are conditionally independent given the class label, which simplifies the computation of the posterior probabilities. NB is computationally efficient and requires a small amount of training data to estimate the parameters (Wickramasinghe and Kalutarage, 2021).

- Decision Tree (DT) are non-linear models that partition the feature space into regions based on simple decision rules. It is a versatile classification algorithm that builds a tree-like structure by recursively partitioning the feature space, as an internal node representing features, branches representing decisions, and leaf nodes representing class labels. DT are intuitive and easy to interpret, making them suitable for exploratory analysis. However, they can be prone to overfitting, especially with complex datasets. At each node of the tree, it selects the feature that best splits the data into homogeneous subsets based on impurity measures.

- Support Vector Machine (SVM) are powerful supervised learning models used for classification and regression tasks. SVM aims to find the hyperplane that maximally separates the classes in the feature space. It works well in high-dimensional spaces and is effective even in cases where the number of dimensions exceeds the number of samples. SVM allows for different kernel functions to handle non-linear decision boundaries. The objective of an SVM is to identify the optimal hyperplane that effectively separates the classes in the feature space while maximizing the margin between them. The support vectors, which are the data points closest to the hyperplane, determine the decision boundary. These support vectors are utilized to define the margin. SVM possesses the capability to efficiently handle non-linear classification tasks by employing kernel functions, such as linear, polynomial, radial basis function Radial Basis Function (RBF), or sigmoid, to map the input features into a higher-dimensional space. SVM incorporates a cost parameter (C), which regulates the trade-off between maximizing the margin and minimizing the classification error. A higher value of C permits a smaller margin, potentially resulting in improved classification performance, particularly in the presence of noisy datasets. Furthermore, SVM offers support for regularization techniques, such as L1 and L2 regularization, to mitigate overfitting.

2.1.3 Feature Extraction

Feature extraction in computer vision refers to the process of transforming raw input images into a format that is suitable for machine learning algorithms to analyze and make predictions. This involves identifying and extracting relevant information or features from the images, which can then be used as input for classification tasks. Several techniques and methods are commonly used for feature extraction in computer vision. Next, some usual techniques (also considered in this Thesis) are presented:

- Scale Invariant Feature Transform (SIFT) is a feature detection algorithm that extracts distinctive and invariant features from images. It identifies key points or interest points in an image that are invariant to scale, rotation, and illumination changes, and describes them based on the local gradient orientation histograms. Key points are detected by looking for local intensity extrema in different scales of the image. SIFT descriptors are then computed for each key point, which captures information about the local image gradients in the region surrounding the key point. These descriptors are robust to changes in viewpoint and lighting conditions, making them suitable for object recognition and image-matching tasks.

- Speeded Up Robust Features (SURF) is a feature detection and description algorithm that is similar to SIFT but is faster to compute. It detects key points based on the determinant of the Hessian matrix and describes them using a set of Haar wavelet responses. It uses a technique called integral images to compute image gradients and feature descriptors quickly. SURF descriptors are computed based on a combination of Haar wavelets and box filters, which makes them highly robust to noise and image transformations. While not as invariant as SIFT, SURF features are still effective for object detection, tracking, and image registration tasks.
- Oriented fast and Rotated brief (ORB) algorithm is a highly efficient and pixel-based method for detecting features. Its usage spans across a wide range of fields, including target recognition. In order to detect keypoints, ORB employs a variation of the FAST algorithm, which is known for its effectiveness in identifying corners and key points through a comparison of pixel intensities in a circular pattern surrounding each pixel. What sets ORB apart is its incorporation of rotation invariance, allowing it to robustly detect keypoints in a variety of orientations. Furthermore, for feature description, ORB utilizes a modified version of the Binary Robust Independent Elementary Features (BRIEF) descriptor. This combination of features makes ORB renowned for its computational efficiency, robustness, and ability to generate binary descriptors, rendering it suitable for real-time applications and environments with limited resources.
- Bag of Features: Bag-of-Features (BoF), also known as Bag of Visual Words (BoVW), is a popular technique for image representation in computer vision and ML. It is commonly used for tasks such as image classification, object recognition, and image retrieval. The BoF approach involves the following steps:

  1. Feature Extraction: The first step is to extract local features from the input images. These features capture distinctive information about the visual content of the images and are typically invariant to transformations such as translation, rotation, and scale. Common feature descriptors used in BoF include SIFT, SURF, and ORB.
  2. Feature Quantization: Next, the extracted features are quantized into a fixed number of visual words or codewords. This is typically done using clustering algorithms such as k-means. k-means clustering aims to partition the feature space into k clusters, where each cluster represents a visual word. The centroids of these clusters serve as the visual word representations. Each feature descriptor is assigned to the nearest cluster centroid, and the resulting cluster index represents the visual word associated with that feature.
  3. Histogram Generation: For each image, a histogram of visual word frequencies is constructed. This histogram represents the distribution of visual words in the image. Each bin of the histogram corresponds to a visual word, and the bin count indicates the frequency of occurrence of that visual word in the image.
  4. Normalization: Optionally, the histograms may be normalized to account for variations in image size or intensity.

5. Classifier Training and Prediction: Finally, the histograms are used as feature vectors to train a classifier. During the prediction phase, the classifier is used to classify new images based on their feature histograms.

2.1.4 System Assessment

Assessing the performance of an object recognition system involves evaluating its ability to accurately detect, localize, and classify objects within images or video frames, as well as evaluate its quality in hardware terms and compare such findings to other solutions. These three aspects are described below:

1. Performance Evaluation in Terms of Detection Capacity.

   The first stage of evaluating a system is to select an appropriate training-testing methodology to ensure the validity of future conclusions. Some of them are as follows:

   - Holdout Method: The holdout method involves dividing the available dataset into two distinct subsets: a training set and a validation set. The training set is utilized to train the ML model, whereas the validation set is employed to assess its performance. The model is trained on the training set and subsequently evaluated on the validation set by utilizing performance metrics such as accuracy, precision, and recall, among others. This procedure helps the evaluation of the model's ability to generalize to new data and provides an estimation of its performance.

   - Cross-validation: It refers to a resampling methodology wherein the dataset is partitioned into various subsets, known as folds. In each iteration, it divides the dataset into trained on a subset of the data (training set) and evaluated on the remaining subset (validation set). The performance metrics are then averaged across all iterations to provide a more robust estimate of the model's performance.

   ML performance indicators are crucial for evaluating the effectiveness of models in solving specific tasks. These performance indicators are helping to assess the quality and reliability of ML models across different tasks and domains. The most usual ones are as follows:

   - Confusion Matrix: Provides a detailed breakdown of true positives, true negatives, false positives, and false negatives for each class. It is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. It allows visualization of the performance of an algorithm. Helps identify specific areas where the system may be struggling and provides insights into potential improvements. A confusion matrix is a square matrix with dimensions $n{\times}n$ where n is the number as shown in Table 2.1. Each row of the matrix represents the instances in a predicted class, while each column represents the instances in an actual class. The four possible outcomes of a binary classification task are:

     - True Positive (TP): predicted to be positive and the actual value is also positive.

  - False Positive (FP): predicted to be positive but the actual value is negative.
  - True Negative (TN): predicted to be negative and the actual value is also negative.
  - False Negative (FN): predicted to be negative but the actual value is positive.

  From the confusion matrix, various performance metrics such as accuracy, precision, recall (sensitivity), specificity, F1 score, and others can be calculated to evaluate the performance of a classification model. In an ideal scenario, the confusion matrix would have 1.0 on the diagonal and 0.0 elsewhere, indicating perfect classification.

	Predicted Negative	Predicted Positive
Actual Negative	TN	FP
Actual Positive	FN	TP

Table 2.1: Confusion matrix structure.

- Accuracy: is a fundamental performance metric used in ML to evaluate the performance of classification models. It measures the ratio of correctly classified instances or objects to the total number of instances present in the dataset. From a mathematical standpoint, accuracy is defined as

  $$Accuracy = \frac{\text{Number of Correctly Classified Instances}}{\text{Total Number of Instances}}, \tag{2.1}$$

  which is calculated as

  $$accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{2.2}$$

  In the context of object recognition or classification tasks, accuracy serves as an indicator of how effectively a model can properly identify or classify objects based on the input data. A higher value for accuracy signifies that the model is committing fewer errors in its predictions. Nevertheless, it is important to note that accuracy alone may not always furnish a comprehensive overview of model performance, particularly in cases where there is an imbalance among classes or where the cost associated with misclassifications varies. Under such circumstances, supplementary performance metrics such as precision, recall, F1 score, and confusion matrices may be employed to yield a more comprehensive evaluation of the model's performance.

- Precision: This performance metric is used in classification tasks to evaluate the accuracy of positive predictions made by a model. It measures the ratio of correctly predicted positive instances to the total number of instances predicted as positive. Mathematically, precision is defined as

  $$Precision = \frac{\text{True Positives}}{\text{True Positives+False Positives}}, \tag{2.3}$$

which is calculated as

$$\text{precision} = \frac{TP}{TP + FP}. \tag{2.4}$$

A high precision value indicates that the model is making fewer false positive predictions, meaning that when it predicts an instance as positive, it is more likely to be correct. Precision is particularly important in applications where false positives are costly or undesirable.

- Recall: This metric indicates the ratio of correctly identified objects to all objects that truly belong to a specific class, known as sensitivity or true positive rate. Mathematically, recall is defined as

$$Recall = \frac{\text{True Positives}}{\text{True Positives+False Negatives}}, \tag{2.5}$$

which is calculated as

$$\text{recall} = \frac{TP}{TP + FN}. \tag{2.6}$$

Recall ranges from 0 to 1, where a higher value indicates better model performance.

- F1-score: It is the harmonic mean of precision and recall, providing a well-balanced assessment of the system's performance. It is calculated as

$$F1Score = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{2.7}$$

The F1 score ranges from 0 to 1, where a higher value indicates better model performance, the F1 score provides a more comprehensive evaluation of a classifier's performance compared to using either precision or recall alone.

2. Hardware Performance for Machine Learning.

Hardware performance indicators in ML refer to metrics that assess the computational efficiency and effectiveness of ML algorithms when executed on specific hardware platforms. These indicators are crucial for evaluating the scalability, speed, and resource utilization of ML models on different hardware architectures. The following are some prevalent hardware-related performance metrics:

(a) Processing Speed: It is crucial in ML tasks, especially for model training and inference, where large volumes of data need to be processed quickly. Faster processing speed allows ML algorithms to iterate through data more efficiently, leading to shorter training times and reduced latency during inference.

(b) Response Time: It indicates the time it takes for a system to react to a stimulus or input and provide a corresponding output or result. In the context of hardware performance in ML, response time typically refers to the latency or delay between feeding data into a ML model and receiving the predicted output, such as, in an image classification task, response time would be the time it takes for the system to process an input image and produce the predicted class label.

(c) Memory Capacity: It is the amount of memory that the system needs to work. It is a factor especially relevant for edge computing systems, where memories are constraints in capacity.

(d) Power Consumption: It refers to the amount of electrical energy consumed by a device or system over a specific period. The magnitude of electrical power consumed by the hardware constituents is typically quantified in watts (W) or kilowatts (kW). Reduced power consumption is preferable to enhance energy efficiency and cost savings. In the context of computing systems, power consumption is a critical factor that affects performance, cost, and environmental impact.

3. Performance Comparison.

   To conduct a comprehensive evaluation of various solutions for systems assessment, including both baseline and proposed solutions, it is essential to establish the specific performance metrics utilized, and the criteria employed for comparison. Comparing the performance of an object recognition system with baseline methods or state-of-the-art techniques involves evaluating various metrics to assess its effectiveness and efficiency. The following is a general approach to comparing performance indicators:

   (a) Selection of baseline methods and state-of-the-art techniques:

       - Baseline methods are typically elementary or traditional approaches that function as a reference point for comparison. They may encompass basic algorithms or standard techniques commonly used in the field.
       - State-of-the-art techniques represent the most advanced and leading-edge approaches currently available. These might include the most recent ML frameworks, advanced feature extraction methods, or novel algorithms.

   (b) Performance Metrics: Various performance metrics are typically used to evaluate the object recognition system. These include accuracy, precision, recall, F1 score, and mean Average Precision (mAP). These metrics provide quantitative measures of the system's ability to correctly identify and classify objects, as well as its computational efficiency.

   (c) Experimental Setup: Experiments should be conducted under controlled conditions to compare the object recognition system with both baseline approaches and state-of-the-art techniques. Datasets are selected and experiments are designed to evaluate the system's performance across different scenarios, such as varying object classes, backgrounds, lighting conditions, and occlusions.

   (d) Evaluation Process:

       - The process of evaluating the object recognition system involves training and testing it with the chosen datasets and evaluation protocols.
       - Quantitative assessment of the performance of each method or technique being compared is achieved through the computation of performance metrics.

   (e) Analysis: The results obtained from the evaluation are analysed to identify the strengths and weaknesses of the object recognition system compared to baseline methods and state-of-the-art techniques. Statistical tests may be conducted to determine if observed differences in performance are statistically significant. Insights gained from the analysis are used to conclude the relative performance of the object recognition system and its competitiveness compared to existing approaches.

(f) Discussion of findings: The findings of the comparison are discussed in the context of the objectives and requirements of the application or domain. Limitations of the object recognition system and areas for future improvement may be identified based on the analysis of the results.

2.2 Internet of Things

IoT has become one of the most familiar and popular expressions among various sectors of business and technology in the recent period. Currently, we have started to live in some of its aspects now that some of the things we use have the ability to connect to the Internet and enter under the concept of IoT, anything that can be attached to a processing unit and an Internet connection is something in the Internet world of things. The goal of the IoT is to minimize human intervention for data entry and employ different types of sensors to collect data from them and allow for automatic storage and processing of all of that data. The IoT has indeed garnered widespread recognition and acceptance across various industries due to its transformative potential. This phenomenon has attracted significant attention and enthusiasm, supported by empirical evidence of its benefits and applications(Lombardi et al., 2021). At its core, the Internet of Things envisions a framework where all objects in our daily lives can connect to the Internet or, to each other, facilitating data exchange and task execution through network-based communication. While this concept is gradually becoming integrated into our lives, some objects already possess internet connectivity, thus functioning within the IoT domain. Essentially, any entity with a processing unit and internet connectivity can be considered part of IoT, aiming to reduce human involvement in data input through sensor-based data collection, automatic storage, and processing. Modern technologies like Bluetooth, ZigBee, Wi-Fi, and 4G have revolutionized the establishment of Wireless Sensor Networks (WSNs), forming the backbone of IoT. These networks incorporate sensor packages that monitor various events such as sound, vibration, heat, magnetic fields, and noise, with the assistance of ML techniques (Sikimić et al., 2020). The relationship between IoT and AI is symbiotic, as IoT generates vast amounts of data that AI analyses to extract valuable insights. As IoT continues to grow, billions of connected devices are predicted to interconnect in the coming years, generating immense volumes of data. However, IoT devices exhibit substantial heterogeneity across multiple levels, posing challenges in data communication protocols, power requirements, and computing capacity. To manage IoT devices effectively, addressing these challenges is essential. Processing capabilities must be developed to extract meaningful information from vast datasets generated by IoT devices, thereby maximizing the potential of this transformative technology(Hassan et al., 2020).

2.2.1 Edge Computing

Edge Computing is a distributed computing paradigm, within IoT, that brings computation and data storage closer to the location where it is needed, which is typically at the edge of the network. This paradigm aims to address the limitations of traditional cloud computing, such as latency, bandwidth constraints, and privacy concerns, by moving computational tasks and data processing closer to the source of data generation. In edge computing, data processing and storage are performed on devices called edge devices, which are located near or at the edge of the network. These devices can range from sensors, smartphones, and IoT devices to edge servers and gateways. By processing data locally on edge devices, edge computing reduces the need to transmit raw data to centralized data centres for processing, thereby reducing latency and network bandwidth usage. One of the key advantages of edge computing is its ability to support real-time or low-latency applications, such as autonomous vehicles, industrial automation, and augmented reality. By processing data locally on edge devices, these applications can respond quickly to changing conditions without relying on a centralized cloud infrastructure (Nain et al., 2022).

Another benefit of edge computing is improved data privacy and security. Since sensitive data can be processed locally on edge devices, there is less risk of data exposure during transit to centralized data centres. Additionally, edge computing allows organizations to comply with data privacy regulations by keeping sensitive data within a specified geographic region. Edge computing offers a more efficient and scalable approach to data processing and analysis, especially for applications that require real-time processing, low latency, and enhanced data privacy and security. As the number of connected devices continues to grow, edge computing is expected to play an increasingly important role in enabling the next generation of IoT and edge AI applications (Chang et al., 2021).

Edge computing provides a variety of functional and financial advantages. It can help improve the performance and scalability of intelligent systems by unloading certain data and processing and analysis functions from applications running in the cloud (Al-Doghman et al., 2022). It accelerates the performance of local delay-sensitive applications such as real-time device automation by analysing data locally and avoiding wide-area network latency, it involves network costs by reducing traffic in the early stages over wan links Smart systems can be made more flexible and sustainable by distributing computational functions across the network and eliminating individual failures. It also enables simpler and less expensive IoT devices by converting the endpoint processor and memory capacity to edge gateways and servers. Some key benefits of edge computing include:

1. Reduced Latency: By processing data locally at the edge, edge computing minimizes the time it takes for data to travel from its source to the processing node and back, enabling real-time or near-real-time responses.

2. Bandwidth Optimization: edge computing reduces the amount of data that needs to be transmitted over the network to centralized data centres, conserving network bandwidth and reducing congestion.

3. Improved Data Privacy and Security: edge computing allows organizations to process sensitive data locally on edge devices, reducing the risk of data exposure during transit

to remote servers and enhancing data privacy and security.

4. Scalability and Flexibility: edge computing architectures can scale horizontally by adding additional edge devices as needed, providing greater flexibility and agility in deploying and managing distributed applications.

2.2.2 Edge Target Platforms

Edge computing targets various platforms that are positioned closer to the data source or end-users, enabling localized data processing, analysis, and decision-making. It is important to know the technical details of IoT devices as target platforms for edge computing. Some of their characteristics are that the IoT devices are equipped with various sensors, such as temperature sensors, humidity sensors, accelerometers, gyroscopes, and cameras, to collect relevant environmental data. These devices are powered by microcontrollers with embedded processing capabilities, enabling them to perform basic data preprocessing and analysis tasks. Furthermore, IoT devices support wireless communication protocols, such as Wi-Fi, Zigbee, Z-Wave, and LoRa, to allow connectivity.

Several edge devices are commonly used in edge computing, each tailored to specific applications and requirements. such as Raspberry Pi, NVIDIA Jetson Series, Intel NUC, Arduino, BeagleBone, UP Board and Xilinx Zynq-7000 SoC. The choice of edge device depends on factors such as performance requirements, connectivity options, form factors, and cost considerations, among others.

In this Thesis, it was considered the PYNQ-Z1 board which is a development board that combines the programmability and ease of use of Python with the flexibility and performance of an Field-Programmable Gate Array (FPGA). It is designed to enable rapid prototyping and development of embedded systems and digital circuits, particularly for applications requiring real-time processing and hardware acceleration. Below is the technical description of the PYNQ-Z1 board:

- Designed to be used with the PYNQ open-source framework that enables embedded programmers to program the onboard SoC with Python.
- The heart of the PYNQ-Z1 is the Xilinx Zynq-7000 SoC (System on Chip), which integrates both a dual-core ARM Cortex-A9 processor and an FPGA fabric on a single chip.
- The ARM Cortex-A9 processors offer the computational capability and adaptability akin to that of a traditional central processing unit CPU, thereby enabling the PYNQ-Z1 to execute high-level software applications.
- The field-programmable gate array fabric presents adaptable logical components that can be programmed to execute personalized digital circuits and expedite particular algorithms or tasks.

In the use case addressed in this Thesis, the edge device will be connected to a Infra-Red (IR) camera to capture thermal radiation emitted by objects in the environment. These devices are commonly used for temperature monitoring, surveillance, and object detection in various applications, including industrial automation, building management, and security systems. Further technical details of IR cameras are listed as follows:

- Sensor Technology: They utilize microbolometer-based thermal imaging sensors to detect and measure IR radiation emitted by objects. Microbolometers consist of tiny thermally sensitive resistors that change resistance in response to temperature variations, allowing them to convert thermal energy into electrical signals.
- Image Processing: They incorporate signal processing algorithms to convert the electrical signals from the thermal imaging sensor into digital images. These algorithms apply calibration, noise reduction, and contrast enhancement techniques to improve image quality and accuracy.
- Connectivity: They usually include wired interfaces, such as Ethernet or USB, to transmit data and supply power. Additionally, these cameras usually offer wireless connectivity options, including Wi-Fi, which facilitate remote monitoring and control based on standard Internet Protocol (IP)-based protocols, such as TCP/IP, UDP, FTP or HTTP.
- Power Supply: They can be powered via mains electricity, battery, or Power over Ethernet (PoE) depending on the deployment scenario and power requirements.
- Edge Analytics: Advanced cameras have the potential to integrate customized algorithms for edge analytics, enabling the direct execution of intricate data analysis tasks, such as object identification, categorization, and tracking on the device itself. Edge analytics serve to minimize latency, reduce bandwidth consumption, and lessen the reliance on centralized processing resources.
- Security: Usually, they implement encryption protocols to secure data transmission and protect sensitive information from unauthorized access or tampering.

2.3 Border Surveillance Systems

Border security and surveillance have become increasingly important for countries around the world in the present age. Most countries in the world invest significant resources and wealth in border security and surveillance. The global border security systems market is expected to reach USD 69.30 billion by 2028[1].

This fact is due to the illegal entrance on the territory is usually related to drug traffic, arms trade, and terrorism. As a result, governments are making continuous efforts to design accurate Border Surveillance Systems (BSSs) to mitigate such risks. However, the design of BSS is not trivial because each border has specific needs, which could be linked to the amount and type of crossing traffic, orography, area to cover, weather conditions, or available economic resources to deploy the system. Thus, there is a need for designing innovative surveillance systems, where cost, security, safety, and performance could be defined as the main goals (Martins and Jumbert, 2022).

[1] https://www.vantagemarketresearch.com/industry-report/border-security-systems-market-market-1344l

In a traditional border patrol system, border guards patrol the border according to predetermined intervals. However, in the case of long-distance border areas with thousands of kilometres, this approach requires extensive human resources. That means that intensive human involvement is the major challenge for protecting long border areas. For real-time border protection with high accuracy and minimizing the need for human support, multiple surveillance technologies, which complement each other, are required. This complementary technology could be based on different principles according to the needs, such as radars, seismic detectors, Closed-circuit television (CCTV), and Digital Video Recording (DVR) systems. This type of solution with complementary technology includes other challenges (Pawgasame and Wipusitwarakun, 2020; Sharma et al., 2021; Patino et al., 2022), such as the high cost of the sensors, the integration of multiple types of sensors in a single system, and adopting suitable communication protocols at different mediums.

The limitations associated with traditional border patrol systems have led to a shift towards camera surveillance systems, although other sensor types are also being considered (Olagoke et al., 2020; Bolakis et al., 2021). In these solutions, a significant number of cameras are installed along the border, and personnel are employed to monitor the captured images in real time. This approach addresses the challenges posed by long-distance border areas and the need for continuous surveillance (Al Fayez et al., 2019). However, it introduces its own set of challenges, particularly the requirement for individuals to analyze images around the clock. The necessity for continuous human monitoring was examined in a study, which delved into the repercussions of high human involvement in traditional border patrol systems and advocated for the integration of high-technology cameras mounted on surveillance towers.

The evolution in Surveillance Camera System (SCS) signifies a transition from conventional cameras to Intelligent Surveillance Systems (ISS), representing a shift towards integrating human intelligence into image analysis alongside the operational capabilities of camera systems. In such ISS, there is an aspiration to automate the tracking of intruders, including the potential identification of these intruders. This would be achieved with a minimized deployment of cameras to cover specific areas effectively (Elharrouss et al., 2021). On this basis, national and private defense companies worldwide are developing different systems to meet security needs. These solutions are based on providing effective connectivity and integration between the state apparatus competent of the State. These security services mainly concern border control to reveal penetration attempts by hostile groups or groups of people trying to enter the country illegally. The current trend in these systems is based on deploying sensors according to the needs, so that the border is divided into small sectors, enabling fast response due to the concentration focus.

Storing and analyzing the video stream from all cameras at the border can be a significant cost factor in centralized (BSS). so edge computing for video analysis in border surveillance systems offers a compelling solution to optimize resource utilization, improve response times, and enhance the overall effectiveness of the surveillance infrastructure. Employing a solution that triggers video recording and analysis only when movement or targets are detected can significantly optimize the efficiency and cost-effectiveness of a surveillance system. Integrating edge computing capabilities further enhances this strategy by shifting the processing and analysis tasks closer to the data source, i.e., the surveillance cameras or sensors deployed at the border. With edge computing, the detection and analysis of targets can occur in real-time at the edge of the network, reducing latency and alleviating the burden on central processing resources Figure 2.1 shows a system architecture of edge computing in a Video Surveillance System (VSS) Based on Edge Computing.

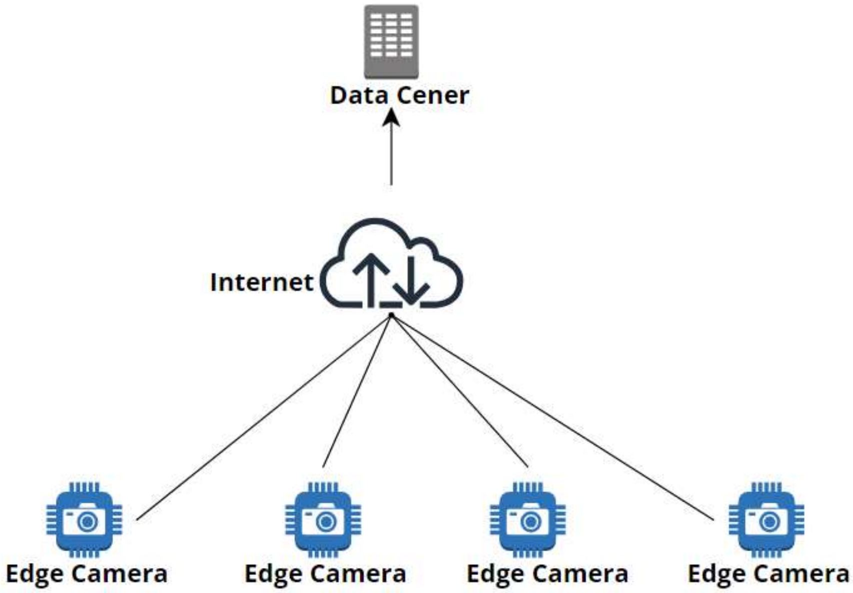


Figure 2.1: Video Surveillance System Based on Edge Computing

adopting this decentralized approach, the surveillance system can effectively detect and respond to security threats in real-time, without relying on centralized processing or coordination. This distributed architecture enhances the system's scalability, and reliability, making it well-suited for border security applications where rapid threat detection and response are critical(Singh and Singh, 2022).

The successful deployment of a decentralized BSS depends on the careful selection and integration of sensors that meet the system's specific requirements while addressing the challenges inherent in border surveillance operations, a complex challenge that demands advanced monitoring systems capable of covering expansive areas along international borders while maintaining flexibility and high performance. Modern monitoring systems deploy sensing technologies strategically, dividing the surveillance area into smaller sectors to enhance defence capabilities. This approach enables rapid response to potential threats, minimizing the risk of hostile actors exploiting delays in patrol response, as often seen in conventional border control systems, the key to this requirement to be achieved is the selection of sensor systems such as Electro-Optical (EO)/IR cameras, Laser Imaging Detection and Ranging (LIDAR) and radar, that offer both flexibility and high performance. These systems must be superior in early detection, providing real-time monitoring with low false alarm rates across diverse weather and lighting conditions (Singh and Singh, 2022). Such sensors play a crucial role in enhancing BSS by enabling fast and effective responses to emerging threats.

In line with the previous discussion, when designing a BSS for a particular situation, several open questions arise related to technology selection in the following two aspects. The first question is how a candidate sensor will perform in the system application. The second one is how the sensor will adapt to the geographical nature of the limits to be monitored.

The integration of ML techniques into border surveillance systems has revolutionized the way security threats are detected and managed. The application of ML techniques has transformed these systems by automating the detection and classification of targets of interest, thereby improving reliability and reducing human involvement (Verma et al., 2022; Yu et al., 2021). A comprehensive wide-area surveillance system for border security should include two key aspects. Firstly, it should utilize sensors to track targets within its field of view, enabling detection of potential threats. Secondly, it should incorporate an intelligence engine capable of performing tasks such as target detection, and recognition of the target into categories such as animal, human, or vehicle. Also, terrain conditions play a crucial role in the design of border security systems, as they influence the suitability and effectiveness of sensor deployment. Therefore, the choice of sensors for border surveillance should be tailored to the specific terrain and environmental conditions.

In addition to the territory type, sensor selection also depends on the length border determining the platform to be used. That is,

- Perimeter fences. It consists in deploying a variety of electronic surveillance technologies close the perimeter for intrusion detection and warning. These ground-based systems are primarily short-range, up to around 500 meters.
- Observation towers. It extends surveillance capabilities tens of kilometers further from a border installation and provides a platform for ground-based medium-range surveillance.
- Mobile observation platforms. It includes using land, maritime, and aerial vehicles (as drones), extending the reach of medium-range surveillance sensors.
- Stationary observation platforms. It includes aerostats, generally tethered balloons, which allow extended observation over wider areas, extending the reach of surveillance sensors beyond what can be seen from an observation tower.

2.3.1 Sensor Review

In border surveillance systems, there are four main types of terrain: desert, forest, mountains, and coastal areas. The deployment of sensors in these different terrains varies based on their specific capabilities and requirements. Each sensor type has different capabilities and effectiveness depending on the type of terrain being monitored. Table 2.2 shows an analysis of the applicability of seven usual sensors (EO cameras, radar, IR cameras, and seismic, acoustic, magnetic, and barrier sensors) in the four types of terrain for target detection (D field), target identification (I field), and target recognition (R field) tasks. In this table, "+" indicates suitability, "-" indicates little or no benefit, and "L" indicates limited applicability for each sensor type in different terrain types and tasks related to target detection, identification, and recognition.

The analysis of the table indicates that no single sensor performs optimally across all types of terrain. However, the IR technology emerges as the most suitable option, exhibiting

Table 2.2: Applicability of seven different sensors to four types of terrain for target detection (D field), identification (I field), and recognition (R field).

Means	Desert			Jungle			Mountain			Coastal		
	D	I	R	D	I	R	D	I	R	D	I	R
EO camera	L	+	+	L	+	+	L	+	+	L	+	+
Radar	+	−	−	−	−	−	+	−	−	+	−	−
IR camera	+	+	+	L	+	L	−	+	L	+	+	+
Seismic	+	−	−	+	−	−	+	−	−	+	−	−
Acoustic	+	L	−	+	L	−	+	L	−	+	L	−
Magnetic	+	L	−	+	L	−	+	L	−	+	L	−
Barrier	+	−	−	+	−	−	+	−	−	+	+	−

Table 2.4: Comparison of the existing intrusion detection sensors

Sensor	Low Power	Reliability	Cost
Infrared (Thermal)	Yes	Medium	Low
Ultrasound	No	High	High
Accelerometer (Seismic)	Yes	Low	High
Radar	No	High	High

adequate performance across all three tasks. Particularly for the detection and recognition tasks pertinent to the use case, the IR sensor remains the top choice. Sensor selection also depends on the action type to detect. For example, magnetic sensors have magnetic sensitivity then, it is only feasible in case of requiring detecting metals. Sensors designed to monitor sounds are a microphone that monitors sound generated by the movement of objects, such as car sounds and animal footsteps (Cenkeramaddi et al., 2020). Moving object detection can be achieved by seismic sensors due to the vibrations generated with the movement (Bai et al., 2020).

In this regard, Table 2.4 shows a comparison of different types of sensors when used to detect moving objects (Ji et al., 2022), which is one of the requirements in the surveillance system in this thesis. In this regard, Table 2.4 presents a comparison of different types of sensors, including IR, ultrasound, accelerometer, and radar, when used to detect moving objects (Ji et al., 2022). This comparison is particularly relevant to the surveillance system outlined in this thesis. The analysis reveals that the IR sensor emerges as the most efficient intrusion detection sensor, with the added advantage of requiring less power compared with radars.

A thermal imaging device enables seeing targets in darkness or smoke, creating a photographic image or video sequence of light emitted by an object at terrestrial temperatures. Its operating principle is based on the fact that IR light is emitted or absorbed by molecules during their rotational-vibrational movements. Note that IR is thermal imaging, also often known as thermography, is not the same as night vision. Night vision operates on the principle of light amplification, so in a dark environment, light amplification would yield no image whereas thermal imaging would be based on emitted energy then transmitted and reflected by the target.

Thus, thermal cameras are characterized by recognizing the temperatures of the targets, whether people, animals, or otherwise, and generating heat signatures. It means that it allows working in highly-sensitive areas, showing the infiltration of any object entering the restricted area in real-time, even in the most extreme darkness and harsh environments (fog, dust, smoke, snow, or rain), in addition to revealing the stealth and camouflage that may be practised by the intruder (Artan and Tombul, 2022). This technology can be used for many other purposes than surveillance, for instance, fire detection, manufacturing control in factories, and search and rescue tasks.

2.3.2 Market Research

The BSSs market is experiencing significant growth, driven by a confluence of factors including technological advancements, geopolitical considerations, and the changing landscape of security threats [1] The crucial role of the border surveillance industry is evident in addressing contemporary challenges by adopting advanced technology solutions and unmanned systems. The integration of advanced technologies reflects a commitment to enhancing the effectiveness and efficiency of BSSs. The combination of Surveillance Unmanned Aircraft System (UAS), IP technology, IR imaging, and compact surveillance radars contributes to a comprehensive and dynamic approach to border surveillance [2]

The European Union's commitment to securing its external borders is especially relevant in this context, especially maritime borders, due to mainly illegal immigration (Léonard and Kaunert, 2022). The allocation of up to 34.9 billion euros for the period 2021 to 2027, compared to 13 billion euros in the previous period, underscores the EU's dedication to enhancing its border control mechanisms (Conte and Savazzi, 2020).

In the Middle East Saudi Arabia, over $3.4 billion is spent on border fence security systems[3] Bordered by Yemen and Iraq. This system is the largest fully integrated border security solution worldwide, implemented by Airbus Defence and Space, to secure 9,000 km of borderline, including 5,000 km of coastline.

Finally, the national BSS in the United States of America (USA)[4] provides comprehensive situational awareness along the country frontier for border security and national security purposes, as well as to assist in detecting and identifying illegal entering. It highlights the 1.6 billion $ spent to experiment with different types of security fences in Arizona (Mann, 2020).

[1] Fortune Business Insights Border Security System Market (105208).

[2] Border Security Market Size Source & Share Analysis - Growth Trends & Forecasts (2023 - 2028).

[3] The graphic high-tech border fence [News Briefing].

[4] U.S. Border Patrol Technology (U.S. Customs and Border Protection.

2.3.3 Geographical Features, and Characteristics of Libya's borders

Libya has a great variety of terrain including deserts, mountainous areas, and valleys, which constitute natural barriers to external threats. However, the importance of these barriers is constantly declining with the development of ways to penetrate the border, such as using rapid and effective four-wheel drive cars equipped with communication and navigation devices. The geographic characteristics of Libya are one of the biggest challenges facing the building of an integrated BSS, which amounts to 1,759,540 square kilometres. Libya's terrain is mostly the Sahara Desert and small population clusters with distances among them of several hundred kilometres, hardening long border protection by traditional means. Moreover, the Libyan desert climate involves high temperatures in the summer and reduced visibility levels due to surface wind and sandstorms (Phillips, 2020). In Table 2.6, the lengths of the Libyan border with other countries and the coastline are presented. Also, Figure 2.6 shows a Map of Libya displaying the illegal traffic routes, adapted from "Illicit Trafficking and Libya's Transition," by M. Shaw and F. Mangan, 2014, United States Institute of Peace. Copyright 2014 by the United States Institute of Peace.

Table 2.6: Libyan border length description.

Name of the country	Length of the border (in km)
Coastline	1,900
Algeria	982
Chad	1055
Egypt	1115
Niger	354
Sudan	383
Tunisia	459

Libya can develop a comprehensive and effective border security strategy by creating an effective border security strategy and establishing an effective border surveillance system, that enhances its ability to secure its borders, combat illicit activities, and promote stability and security within the country and the wider region. Libya needs to enhance its border control mechanisms, including the deployment of surveillance systems, to monitor the movement of people, goods, and illicit activities across its borders. It is especially relevant that there are large swathes of territory along Libya's border that remain largely ungoverned, presenting significant challenges for border security and control. Outside of population centres, Libya's armed forces have struggled to assert control over migration flows and the trafficking of goods and people passing through the country. This ungoverned space creates opportunities for various illicit activities, including human trafficking, arms smuggling, and drug trafficking, to flourish unchecked. The inability to effectively govern these border areas poses serious security concerns, both for Libya and its neighbouring countries. Without adequate control over its borders, Libya becomes a transit hub for illicit activities, exacerbating regional instability and insecurity. Additionally, the lack of governance in border regions undermines efforts to maintain law and order, facilitate legitimate trade and travel, and prevent the infiltration of extremist groups and criminal networks.

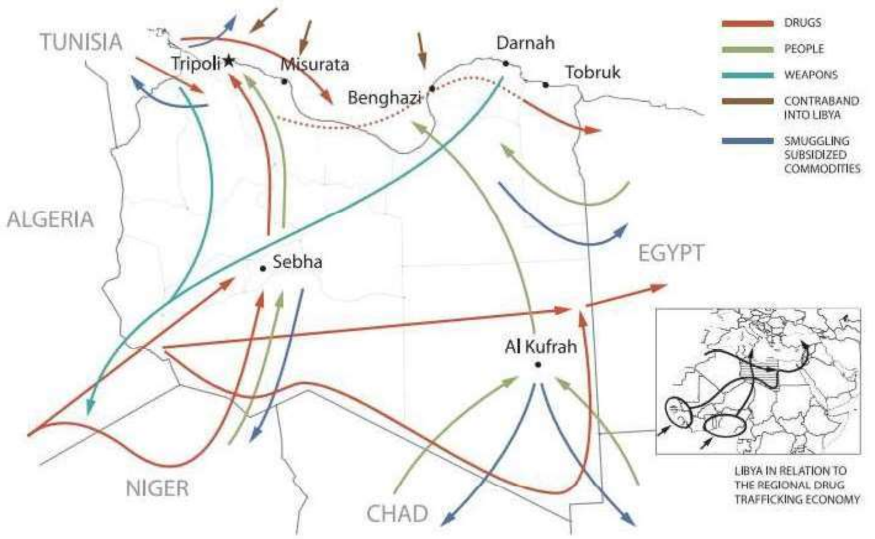


Figure 2.2: Illegal traffic routes in Libya.

2.4 Automatic Target Recognition

The detection and classification of targets based on sensor sources constitute one of the most significant challenges encountered in the realm of BSS To thwart cross-border infiltration attempts, it is of utmost importance to promptly identify targets. This is because the personnel responsible for monitoring multiple monitors cannot effectively carry out continuous surveillance for 24 hours. Furthermore, the targets in question are often too minute for human perception through visual examination of the screen. The primary goal of ATR systems is to conduct searches, detections, and identifications of targets in either real-time or near real-time mode.

The process of detecting and classifying targets based on sensing sources is one of the most important challenges in BSS. This identification task could be done manually by expert humans. However, the use of complex sensing sources as the ones introduced before, the huge amount of data generated, especially for long borders, and response time constraints imply that this identification task must be performed automatically.

ATR is used to identify objects such as vehicles and targets such as animals and humans, and to recognize objects on a battlefield and other military and security applications. several research works have been done in the employ of ATR to protect borders, security safety systems to identify objects or people, automated vehicles, and many others Goyal et al. (2020).

2.4.1 The advantages of Automatic Target Recognition

The integration of ATR technology into Border Surveillance Systems (BSSs) offers numerous advantages. ATR systems offer quick and precise identification and classification of targets, spanning individuals, vehicles, and other relevant objects. This capability empowers border security personnel to efficiently prioritize and address potential threats. Additionally, ATR systems automate the detection and tracking of targets, diminishing the requirement for manual monitoring and intervention. This automation guarantees uninterrupted surveillance coverage and amplifies situational awareness along the border. In terms of early threat detection and alert generation, ATR systems can detect suspicious activities or unauthorized intrusions early, triggering immediate alerts for security personnel. This early detection allows for timely response and intervention to prevent potential security breaches or illegal border crossings. As the ATR system supports and enhances decision-making by providing comprehensive and real-time situational information, the ATR empowers decision-makers with valuable insights for strategic planning and resource allocation. Enhanced situational awareness enables more informed and effective decision-making in border security operations.

The implementation of ATR technology results in the reduction of false alarms and human errors that are commonly associated with manual surveillance methods. ATR achieves this by effectively distinguishing between legitimate targets and those that are suspicious. Consequently, unnecessary responses are minimized and resource utilization is optimized. Furthermore, ATR can be seamlessly integrated with Unmanned Aerial Vehicles (UAVs), drones, and other autonomous platforms for enhanced border surveillance capabilities. These integrated systems provide continuous monitoring and rapid response capabilities, even in remote or challenging terrain. Finally, the integration of ATR technology with other AI applications in border surveillance can enhance their overall effectiveness in detecting, deterring, and responding to security threats. ATR-enabled systems play a crucial role in enhancing border protection and enforcement outcomes, ultimately safeguarding national security interests.

In summary, the integration of ATR technology into BSS offers a wide range of benefits, including improved target identification, automated detection and tracking, early threat detection, enhanced decision-making support, reduced false alarms, and enhanced border protection efficacy. These benefits underscore the value of ATR in strengthening border security and protecting national borders against evolving threats and challenges

2.4.2 Challenges of Implementing Automatic Target Recognition

Some challenges and critical considerations in the implementation of ATR systems in border surveillance contexts include:

1. **Robustness and Reliability:** ATR algorithms must be robust enough to handle variations in target appearance, environmental conditions, and potential distractions. Discriminating between different types of targets, such as people, vehicles, or animals, adds complexity and requires algorithms to be versatile and adaptable.
2. **Task Dependency and Data Set Concerns:** The effectiveness of ATR algorithms

is highly dependent on the specific surveillance task and scenario. Developing comprehensive and representative datasets for training and evaluation is essential but can be challenging due to the diversity of potential scenarios and the need for large labelled datasets.

3. **Hierarchy of Tasks:** ATR involves a hierarchical process, ranging from target detection to identification. While detection and classification may be relatively straightforward, identification, particularly in complex environments or with obscured or partially visible targets, can be extremely challenging.

4. **Sensor Data Processing:** ATR algorithms rely on sensor data for decision-making, and processing this data in real-time for rapid and reliable target acquisition is a significant technical challenge. Optimizing algorithms for efficient processing, especially in resource-constrained environments, requires careful consideration of computational complexity and data bandwidth.

5. **Response Time Reduction:** While ATR can potentially reduce response times for target detection and recognition, integrating ATR capabilities into command and control systems introduces additional challenges. Limited data communication and computational bandwidth may hinder the timely delivery of ATR information to decision-makers, impacting overall system effectiveness.

These challenges highlight the technical and operational complexities associated with the implementation of ATR in border surveillance contexts. Addressing these challenges is crucial to realizing the full potential of ATR technology in enhancing border surveillance system capabilities and decision-making processes.

Chapter 3

State of the Art

This chapter introduces an overview of the literature related to the methods developed in this work. The present dissertation has been formulated in the framework of BSS with broad coverage of state-of-the-art ML techniques and video surveillance based on edge computing. Then, it is presented first the state-of-the-art regarding the design of general BSS. Second, it is reviewed the state-of-the-art in the specific technology selected for the BSS, i.e., ATR based on IR sources.

3.1 Border Surveillance Arquitectures

This section aims to provide a comprehensive review of research findings from various sources regarding general BSS. The main objective is to explore the development and integration of IoT devices, specifically thermal cameras, within the context of BSS. The field of ISSs has witnessed significant advancements in hardware-related research and development. Various techniques, including target detection, tracking, and recognition, have played crucial roles in these advancements. Recent research efforts have been directed towards leveraging emerging technologies, particularly IoT devices and edge computing, to enhance the capabilities of BSS. Listed below are selected scientific papers that delve into the intersection of IoT, edge computing, and BSS:

- Ali et al. (2020) introduce RealEdgeStream (RES), a system designed for large-scale, high-performance data analytics at the edge. The proposed approach addresses video stream analytics through two main phases: filtration and identification. The filtration phase reduces data volume by applying configurable rules to filter out low-value stream objects. In contrast, the identification phase utilises DL inference to analyse the streams of interest. Results demonstrate that the proposed system outperforms a centralized cloud-only approach, achieving a 49% reduction in processing time and 99% savings in bandwidth for 10K element data streams with a frame rate of 15-100 per second. The paper concludes that edge-enhanced video stream processing offers greater efficiency compared to a central cloud-only approach.

- Gaikwad and Karmakar (2021) introduce a framework that integrates AI algorithms into energy-efficient embedded devices. Emphasis is placed on the detection, tracking, and re-identification of individuals through the utilization of edge computing. The proposed AI system exhibits the capability to successfully detect and track individuals within a surveillance setting. Consequently, the implementation of the smart camera node proves instrumental in the context of a distributed surveillance system.

- Zhang et al. (2022a) has formulated a methodology for creating an edge intelligence system. This system aims to establish reliable and real-time video surveillance capabilities. In this approach, edge devices that are dispersed geographically come together to form a decentralized network. The purpose of this network is to uphold restricted access and to facilitate the exchange of both data and computing resources. These resources are utilized for the execution of computationally demanding video analytics tasks.

- Myneni et al. (2022) introduced a novel and expansive distributed video surveillance service framework, denoted as Smart-city Video Surveillance (SCVS). The platform collects and processes video surveillance data on a large scale, intending to identify significant occurrences. This platform employs the utilization of edge cloud computing infrastructure to identify and conceal human faces within video surveillance data. The evaluation of its performance demonstrates that the privacy protection mechanism presented in this study is both efficient and effective when compared to conventional centralized computing models.

- Chen et al. (2022) deliberates on a framework for distributed real-time object detection, which relies on the collaboration between edge and cloud computing to enhance smart video surveillance (SVS) applications. The authors highlight the potential of edge computing in expediting the advancement of IoT within smart urban areas, particularly in the context of video surveillance. Nonetheless, the precise phrase "video surveillance based on edge computing" is not explicitly stated. The scholarly article presents a distributed real-time object detection framework that operates through the cooperation of edge and cloud computing, specifically tailored for the field of SVS. The framework is accompanied by an initial implementation and an evaluation of its viability and efficacy.

- Myagmar-Ochir and Kim (2023) provide a comprehensive overview of VSS within the context of smart cities. The functionalities of VSS, such as object detection, tracking, anomaly detection, classification, and secure data management, are thoroughly explored. Furthermore, the paper is an in-depth review of the integration of advanced technologies like DL, blockchain, edge computing, and cloud computing into VSS. These technologies play a crucial role in enhancing the capabilities of surveillance systems, enabling intelligent object detection and tracking, as well as efficient data management. In general, the study highlights the significant role of intelligent VSS in various aspects, including object recognition and tracking, identifying anomalies, and predicting potential incidents or emergencies.

- Fei and Han (2023) provide a comprehensive review of intelligent multi-camera video surveillance technologies for transportation, highlighting the effectiveness of different approaches in improving tracking success rates and accuracy in various scenarios, with emphasis on the importance of using datasets containing a wide range of data to train models, ensuring better robustness and adaptability to changing environmental conditions.

- Jiang et al. (2023) explore the utilization of vehicular edge computing (VCE) in the domain of on-road video surveillance services, specifically focusing on real-time video sharing. This technology enables secure video sharing in scenarios involving VCE, allowing for the transmission of video content to multiple recipients. The conducted experiments provide evidence that the suggested video-sharing approach is both efficient and secure.
- Ugli et al. (2023) examine a cognitive system for managing video surveillance, employing a hierarchical edge computing system. This system incorporates DL models for object detection and tracking. The cognitive video surveillance management (CogVSM) utilizes a long short-term memory (LSTM)-based model, which leads to remarkable predictive accuracy, as evidenced by a root-mean-square error metric of 0.795. The proposed framework demonstrates a substantial reduction in GPU memory usage, amounting to 32.1% less than the baseline and 8.9% less than previous studies.

These scientific papers offer valuable insights into the advancements, challenges, and potential applications of IoT, edge computing, and EO/IR cameras in surveillance systems.

Usually, BSS systems are commonly categorized into two types based on the deployment of surveillance platforms: mobile and fixed surveillance platforms. With the recent development of unmanned aerial vehicles such as drones, BSSs can be installed and operated in mobile devices which include specialized sensors and advanced navigation systems. In this chapter, an investigation and survey of research developments in the field of utilizing both fixed platforms and mobile platforms in border control systems are presented. The following is a list of some papers that address this topic:

Seunghan Lee (2019) This paper presents a comprehensive approach to designing and developing a BSS utilizing unmanned vehicles (UV)s and hybrid simulations. The system incorporates detection and classification algorithms to process real-time data from fixed and mobile sensors. Physics-based simulation is utilized to provide uncertainties for robust planning and control of UVs. The authors extend their dynamic-data-driven adaptive multi-level simulation framework to address challenges specific to border surveillance.

Laouira et al. (2021b) In this paper, the proposal introduces a multilayer hybrid structure, which integrates cameras, scalar sensors, and radars. These components function as both fixed platforms and mobile platforms, utilizing UAVs for the purpose of border surveillance. And discusses a detailed deployment strategy for each level of the architecture. It also presents an activation scheduling strategy based on load balancing and energy saving to efficiently manage the network and extend its lifetime.

Amutha et al. (2021). This research article presents a proposed Distributed Border Surveillance (DBS) system that functions as a fixed platform. It is specifically designed to be utilized in wireless sensor networks that are deployed in regions of interest. This system takes into account shadowing effects and the asymmetry in the sensing range, addressing energy conservation while evaluating the number of required barriers to monitor a given region, It employs a learning automaton approach for sensor nodes to adapt to coverage requirements effectively. Overall, the proposed DBS system offers significant advancements in border surveillance by leveraging wireless sensor networks and optimizing coverage, energy consumption, and adaptability to varying environmental conditions.

Richhariya et al. (2023) This paper proposed a surveillance deploying drone as a mobile platform, coupled with a reference infrastructure and diverse service and deployment models, can serve as a comprehensive framework for the design and deployment of surveillance-oriented drone cloud systems. The intelligent drone service model put forth in the paper can contribute to the optimal utilization of resources in surveillance activities, potentially resulting in cost savings and enhanced operational efficiency.

Some other researchers have focused on sensors used in surveillance system Two main technologies are usually considered as sensing sources in border surveillance: Synthetic Aperture Radar (SAR) and FLIR cameras. SAR is a type of radar used to create two-dimensional or three-dimensional reconstructions of objects and is typically mounted on moving platforms Renga et al. (2023); Deng et al. (2021).

3.2 Automated Target Recognition Based on Infrared Imagery

This section provides an overview of the state-of-the-art techniques and technologies in ATR and presents the key research challenges that have been addressed in this field. The survey covers various aspects of ATR, including sensor modalities, feature extraction methods, classification algorithms, and developed techniques. Additionally, it explores the impact of emerging technologies such as DL and AI on ATR systems. ATR based on infrared IR imagery is a crucial technology used in various applications, including surveillance, border surveillance, and object detection. IR imagery provides valuable information about the thermal characteristics of objects, making it particularly useful for detecting targets in low-light or nighttime conditions in a 24-hour operation. The IR sensors capture thermal radiation emitted by objects in the environment. Different types of IR sensors, such as Short-Wave Infrared (SWIR), Medium-Wave Infrared (MWIR), and LWIR, are used based on their sensitivity to different wavelengths of IR radiation Zhang et al. (2022b); Stančić et al. (2023); Altaf et al. (2022). Preprocessing techniques are applied to IR imagery to enhance the quality of the images and improve target detection Vadidar et al. (2022). These methodologies may encompass noise reduction, image stabilization, and contrast enhancement. Techniques for feature extraction are utilized to discern pivotal characteristics or patterns within the IR imagery that can be harnessed for target recognition. Conventional techniques for feature extraction embrace edge detection, texture analysis, and blob detection.

In the context of ATR systems, once features are extracted from the IR imagery, classification algorithms are employed to assign the detected entities into discrete classes or categories. ML algorithms, such as SVM, Random Forests, and Convolutional Neural Network (CNN)s, are commonly employed for classification within ATR systems Poster et al. (2023); Baili et al.

(2022). As it is well-known, the success of ATR systems is much influenced by the selection of the learning algorithm, as well as the way feature extraction is performed Dhaygude and Kinariwala (2022); Hameed et al. (2021). The process of target detection involves the identification of potential objects or regions of interest within the IR imagery, while target recognition encompasses the determination of the identity or class to which the detected targets belong. ATR systems make use of a combination of detection and recognition algorithms to accurately identify and categorize targets. The assessment of ATR system performance is based on various metrics, including detection rate, false alarm rate, classification accuracy, and processing speed. The evaluation of performance is essential for assessing the efficacy of the ATR system and identifying areas for improvement.

The design of ATR systems for the desert boundary has not been widely studied in the literature. As far as the candidate knows, there is only work with a focus similar to this thesis is as follows:

- Briones et al. (2012) proposed a radar system in combination with a decision engine based on an Artificial Neural Network (ANN) to classify several targets, which were pedestrians, trucks, and aeroplanes, getting a detection capacity of 80%.

This work includes some limitations for a low-power approach as the one considered in this paper, these are i) ANNs are usually executed in many-core oriented processors, such as GPUs, which are power-demanding devices, and ii) a radar system consumes twice the energy that an IR camera. Moreover, radar systems are more prone to failures due to they have moving parts. Therefore, this approach is not directly applicable to the edge layer for the desert use case.

Outside the desert use case, there are some interesting state-of-the-art works in the ATR field based on IR imagery, which are summarized in Table 3.1 and described as follows:

- Akula et al. (2020) presented a real-time framework for detecting and classifying targets in a forest landscape using thermal infrared cameras. ML techniques, such as adaptive thresholding, morphological operations, and template matching, have been used for animal detection in images obtained from UAVs and wildlife videos. The framework utilized a mixture of Gaussian background subtraction for target detection and a pre-trained deep CNN for feature extraction and classification. It achieves a preliminary testing accuracy of 95% and a frame rate of approximately 23 fps on an embedded computing platform. The real-time deployment of the framework is done on an embedded platform having an 8-core ARM v8.2 64-bit CPU and 512-core Volta GPU with Tensor Cores

- Zhang et al. (2021) introduced a novel framework called An Automatic Target Detector in Infrared Imagery Using Dual-Domain Feature Extraction and Allocation. (Deep-IRTarge) for target detection and recognition in infrared imagery. This framework combines discriminative features from both the frequency domain and spatial domain. It utilizes the Hypercomplex Infrared Fourier Transform (HIFT) module for infrared intensity saliency estimation and a CNN for spatial feature extraction. Deep-IRTarge comprises a frequency feature extractor, a spatial feature extractor, and a dual-domain feature resource allocation model as its backbone network. The proposed Deep-IRTarge framework achieves significant improvements in mAP scores compared to the current state-of-the-art methods on three challenging infrared imagery databases MWIR. Specifically, the paper reports a 10.14% improvement in mAP scores forMWIR, a 9.1%, demonstrating the effectiveness of Deep-IRTarge in target detection and recognition in infrared imagery.

Bolakis et al. (2021) This paper introduces a comprehensive system for border surveillance aimed at preventing unauthorized border crossings. This system integrates advanced aerial and space-based sensor platforms into a unified solution, addressing the need for a high-rising, fixed sensor platform with an unobstructed field of view. The multi-sensor platform incorporated in the system integrates various sensors, including LIDAR, visible and thermal cameras, and acoustic sensors. This approach enables comprehensive surveillance by combining the strengths of different sensing modalities. Ground sensors, quasi-static platforms, and satellite sensor technologies are utilized in the system design to support detection and tracking activities, particularly in foliated areas where traditional surveillance methods may be less effective. The proposed BSS represents a significant advancement in the field, leveraging a combination of advanced sensor technologies and data fusion techniques to enhance monitoring and detection capabilities along border regions.

- Chen and Kapadia (2022) presented a study that illustrates that fusing EO and IR images through pixel-based and decision-based sensor fusion can substantially enhance daytime ATR performance. The authors introduced a novel method for assessing ATR performance based on an extended confusion matrix (ECM), which effectively characterizes Probability of Detection (PD), Probability of False Alarm (Pfa), and the tradeoffs between them in ATR applications. By executing the ATR detector at various Confidence Score thresholds, the detection performance at different Pfa levels can be obtained and depicted using Receiver Operating Characteristic (ROC) curves. Through the fusion of EO and IR approaches utilizing the ECM, the study showcases improvements in Pd by 11% with pixel-based fusion and 13-17% with decision-based fusion.

- Khare et al. (2022) proposed a method for real-time automatic detection of moving ground targets in cluttered infrared imagery using a background modelling approach. This method leverages statistical variations of pixels over the temporal domain to compute the background model, thereby reducing computational costs compared to existing algorithms. Experimental results showcase a detection sensitivity of 0.88 and a false alarm rate of 0.001 in detecting intruding targets from infrared imaging video. The proposed algorithm's performance was evaluated using thermal IR imager video sequences captured in different scenarios from a stationary platform. The test dataset included various background scenes and different moving targets, such as humans and vehicles. Additionally, the algorithm was tested with targets exhibiting variable motion patterns to assess its efficiency. The performance analysis of the proposed algorithm was based on sensitivity and False Alarm Rates.

- Baili et al. (2022) proposed a multistage Automatic Target Detection and Recognition (ATDR) system using DL for infrared imagery. It aims to address the challenge of target detection and identification in computer vision applications, specifically in the context of infrared sensor data. It discusses the use of a state-of-the-art object detector YOLO for target localization and three different CNN architectures for target classification. The system utilizes the YOLO object detector for target localization and three different CNN architectures for target classification. The performance of YOLO in target detection is evaluated, showing high detection rates but also a high number of false alarms. The system is evaluated on a benchmark dataset of different vehicles and a frame rate of 10 frames per second.

- Miao and Xie (2022) used the Single Shot MultiBox Visual Geometry Group (VGG) backbone network to automatically classify and recognize images in the KAIST Pedestrian Dataset. The classification task includes four categories: person (clear pedestrians), people (fuzzy pedestrians), person? (unknown), cyclist (bicycle riders). The accuracy of the algorithm for the four target categories is 67%, 63%, 82%, 62%, the testing mAP is 67.83%, and the training mAP is 69.08%, the predicting mAP is 68.51%. The network recognition speed is 57.29 FPS. Faster RCNN is used as a comparison experiment.

- Zhang et al. (2022b) proposed method combines multiview infrared images for target recognition, using nonlinear correlation information entropy (NCIE) for internal correlation analysis and joint sparse representation (JSR) for classification. The experiments conducted on collected infrared images of multiple types of traffic vehicles, under different conditions, demonstrate the effectiveness and robustness of the proposed method. The recognition rates of buses, cars, trucks, and pickup trucks are 96.25%, 96.80%, 96.67%, and 96.63%, respectively. After calculation, the average recognition rate of the proposed method under current conditions is 96.60%.

- Adams et al. (2023) proposed a novel algorithm called Edge Infrared Vision Transformer (EIR-ViT) for Automatic Target Detection (ATD) using infrared images. This algorithm is lightweight and operates on the edge, making it easier to deploy in on-ground or aerial mission reconnaissance scenarios. The algorithm bridges the gap between traditional signal processing tools and machine learning, allowing for more efficient and accurate target detection and recognition. The EIR-ViT algorithm bridges the gap between traditional signal processing tools and ML, providing a more efficient and accurate solution for target detection and recognition.

- Hom et al. (2023) proposed a solution by combining the localization aspect of a Quadratic Correlation Filters (QCF) neural network layer with feature extraction layers of a purely CNN, resulting in a robust approach for target recognition in long-wave infrared video. The recognition accuracy of this approach is compared to the current state-of-the-art YOLO outputs for target localization and recognition in autonomous vehicles.

- Wu and Zhang (2023) proposed a vehicle target recognition method based on visible and infrared image fusion using Bayesian inference. The method combines the JSR and EdgeBox algorithms for target area detection in the visible light image. The proposed method achieved an accuracy of 77% in image target recognition when the vehicle target was not occluded. When the vehicle target was partially occluded, the accuracy of image target recognition reached 74%.

- Fan et al. (2023) proposed a real-time detection and tracking framework for infrared small targets under complex backgrounds, specifically focusing on vehicle targets. The framework consists of a CNN-based small target detection model, a lightweight CNN detection model combined with the Kernelized Correlation Filter (KCF) algorithm for an accurate evaluation of target location and size in cropped images and a target trajectory predictor based on an optimized Kalman filtering method. Overall, the proposed framework demonstrated promising results in enhancing the detection and tracking of infrared small targets, particularly in the context of vehicle targets, and showed adaptability to complex backgrounds and challenging scenarios.

- Zhao et al. (2023) proposed an improved Single Shot MultiBox Detector (SDD) for infrared small vehicle target detection based on DL. It addresses the limitations of existing approaches in detecting small targets and introduces several enhancements to improve detection performance. The proposed Infrared-SSD is evaluated using an infrared vehicle dataset created by the authors. The experiments were implemented using the Pytorch DL framework on a computer with an Intel I7 9900k CPU and NVIDIA Quadro P6000 GPU Experimental results show that Infrared-SSD achieves higher accuracy than the original SDD algorithm, with a mAP mean Average Precision (map) test score of 82.02% for an input of 300pixel×300pixel.

The main differences between the proposal in this thesis and the previous words described (those outside the desert focus for ATR systems based on IR imagery) are depicted in Tables 3.1 and are described as follows:

- Most of the works focused on identifying vehicles: Chen and Kapadia (2022); Adams et al. (2023); Baili et al. (2022); Wu and Zhang (2023); Fan et al. (2023). Others focused on detecting the presence of animals, cars, people, cyclic, military vehicles, and others Khare et al. (2022); Zhang et al. (2022b) by following a binary classification approach, and only one of the works considered vehicles and people in the same classification problem Akula et al. (2020); Hom et al. (2023). Our approach focuses on detecting targets within the three classes in the same system (animals, people, and vehicles) because these are the types of targets crossing the border in the use case.

- Most of the proposals were evaluated using a regular processor or a powerful GPU Akula et al. (2020); Zhang et al. (2021); Miao and Xie (2022); Fan et al. (2023); Zhao et al. (2023), providing frame rates which could limit the real-time applicability of the algorithm in a low-power edge device. Most proposals did not discuss the computing platform used nor any metric for analysing the computing cost, which limits the comparison. Moreover, some of these works Chen and Kapadia (2022); Hom et al. (2023) applied ANN without any optimization for low-power devices, which means that both works are related to powerful GPUs. In this thesis, the focus is on designing a multi-class ATR system ready to be executed on a constrained edge device with an adequate frame rate for the application.

Table 3.1: State-of-the-art works in the ATR field based on IR imagery

Ref.	Target (number of classes)	ATR algorithm	Computational platform	Frame rate	Detection	Edge capacity
Akula et al. (2020)	Elephant, Human, Vehicle and Others (4)	CNN	8-core ARM v8.2 64-bit CPU	23	95%	No
Zhang et al. (2021)	civilian vehicles, military vehicles , carriers and weapons(3)	Deep-IRTarget	NVIDIA 1080Ti Pascals		83.6%	No
Chen and Kapadia (2022)	vehicle ,military vehicle (2)	CoNNY-25 ATR	NVIDIA Quadro RTX 5000	74.4-92.6	83.5%	No
Khare et al. (2022)	Person and Truck (2)	GMM		-	89%	No
Baili et al. (2022)	Vehicle (10)	YOLO	GPU	10	80%-95%	No
Miao and Xie (2022)	Cyclist, Person, people and Person (4)	SSD	Intel i5-4590 CPU, 16 GB	57.29	69.08%	No
Zhang et al. (2022b)	Bus, pickup, Truck and Car (4)	JSR			96.6%	No
Adams et al. (2023)	Vehicles (15)	EIR-ViT			79%	No
Hom et al. (2023)	people, cars, trucks,..s(15)	YOLOv7			86.4%	No
Wu and Zhang (2023)	Vehicles (19)	SR EdgeBox			77%	No
Fan et al. (2023)	Vehicles (1)	CNN KCF	RTX3080Ti		94.7%	No
Zhao et al. (2023)	Vehicles (1)	SSD	Intel I7 9900k CPU , NVIDIA P6000	-	82.02%	No

Outside ATR systems based on IR imagery, the state-of-the-art is full of successful works applying feature extraction and classification methods for images in the visible field, as in Suganyadevi et al. (2023). Most of these works consider DL approaches, providing an excellent detection capacity but needing to be executed in powerful GPUs, with frame rates which limit the applicability in the edge. For instance, in the study by Miao and Xie (2022), a dataset referenced by the Single Shot MultiBox Detector (SSD) was analysed. The experiment involved training a neural network architecture on hardware consisting of an Intel i5-4590 CPU with 3.5 GB of memory and an NVIDIA GeForce GTX1080 GPU with 16 GB of memory. The GPU was chosen to accelerate the training process due to the network's complexity. The reported frame rate achieved during inference was 59 frames per second (FPS), indicating the speed at which the model could process input images. Additionally, the average accuracy achieved on the mixed dataset was reported to be 81.5%. This accuracy metric likely represents the model's ability to correctly detect and classify objects within the dataset, with higher values indicating better performance.

We also find approaches specially designed to work within the edge layer. For instance, Liu et al. (2023) proposed a lightweight object detection model called AT-YOLO for video surveillance in edge computing environments, the performance of AT-YOLO, achieved 29.9% mAP at 3.47 M parameters, achieved a reduction in bandwidth to 467.29 Kbps and a processing speed of 13.69 FPS on the Raspberry Pi 4B.

On this basis, this thesis is within the edge computing and approximate computing fields of knowledge, which aroused in the last years Townend et al. (2023); Rohith et al. (2021) and whose objective is to bring the computation closer to the sensing place. The novelty in this thesis is the possibility of embedding directly in the edge the capacity of detecting targets of interest in a specific border surveillance use case based on LWIR imagery, using to this end constrained hardware in computing capacity and to get both detection capacity and frame rate metrics within the state-of-the-art. As a way of emphasizing the applicability in the edge, the two proposals in this thesis are executed in an edge device to get a trade-off between computing cost and detection capacity to facilitate the decision-making under an approximate computing focus through the Pareto theory. As far as the authors know, this is the first study within the border surveillance field and IR sensing, which includes concepts from the edge computing and approximate computing fields.

Outside the case in this thesis, the state-of-the-art is full of successful works applying Video surveillance systems based on edge computing, and classification methods for images in the visible field, such as the works in Tables 3.2 and described below:

- Cob-Parro et al. (2021) presented a comprehensive approach to smart video surveillance, based on edge computing, DL algorithms, and efficient processing methods to achieve accurate and real-time detection and tracking of people in video streams. can detect, count, and track people's movements in an area using AI algorithms executed on low-power consumption embedded devices. The system is designed to be portable and can be easily installed anywhere. It can also be powered by portable batteries, eliminating the need for a connection to mains power. The edge node used in the system is based on an UpSquared2 device with CPU capable of accelerating AI CNN inference. The inference was executed both on vision processor unit (VPU) and CPU. An improvement in the processing speed of 64.59% was obtained when the processing was done with the VPU instead of the CPU. The system was compared with other state-of-the-art machine learning (ACF, PCL-MUNARO) and DL (DPOM, YOLOv3, YOLO-depth) methods for people detection. The achieved results are in line with the current state of the art, having a precision of 81.43% and a recall of 80.6% in the EPFL-corridor dataset, both precision and recall were above 87%.

- Zhao et al. (2020) proposed a deep learning-based Intelligent Edge Surveillance (INES) technique for a specific Industrial Internet of Things (IIoT) application, which combines edge computing with cloud computing to reduce network traffic and improve efficiency. The proposed INES technique achieved a detection speed of 16 frames per second on the edge device, with a detection precision of up to 89% after joint computing with the cloud. The operating cost at the edge device was only one-tenth of that of the centralized server. The experimental environment used the NVIDIA Jetson TX2 as the edge device and the nNVIDIA GTX 1080Ti graphics card for calculations on the server side. The paper also compared the computational cost of the depthwise separable convolutional with the standard convolutional. The specific ratio was not mentioned in the provided sources.

- Chen et al. (2022) proposed a distributed real-time object detection framework based on edge-cloud collaboration for smart video surveillance in smart cities. The framework leverages edge computing to consolidate media data from distributed edge devices and deploy AI models remotely for real-time surveillance. The performance evaluation of the framework includes assessing potential benefits, real-time responsiveness, and low-throughput media data transmission. It highlights the importance of edge-cloud collaboration in enabling real-time processing of media data for smart video surveillance, overcoming limitations of centralized cloud-based frameworks. The YOLOv3 model, which incorporates the Residual Network (ResNet) and eature Pyramid Network (FPN), is employed for object detection. The implementation is carried out on nNVIDIA® Jetson Xavier™ NX modules, serving as the distributed on-site edge infrastructure.

Table 3.2: State-of-the-art works based on EO imagery.

Ref.	**Target**	**ATR**	**Computational**	**Frame rate**	**Detection**	**Edge**
	(number of classes)	**algorithm**	**platform**	**rate**	**capacity**	
Cob-Parro et al. (2021)	people(1)	MobileNet-SSD	UpSquared	30	81.43%	Yes
Zhao et al. (2020)	pedestrians and helmets (2)	SSD , Tiny-YOLO	NVIDIA GTX 1080Ti	16	89%	Yes
Chen et al. (2022)	pedestrian, bicycle, car, motorcycle, truck, bus(6)	YOLOv3	NVIDIA® Jetson Xavier™ NX	15.7	94.4%	Yes

Chapter 4

Methodology

This chapter describes the two proposals developed in this thesis: the bounding-box prediction scheme and the frame-based prediction scheme. Both approaches work with IR video frames, in which a movement detection strategy is implemented to focus only on frames where a possible target could appear. Both approaches also consider a BoF strategy to express the information regarding the possible target which is going to be classified by a supervised algorithm. The main difference is that the first algorithm segments the input video frame into Region of Interest (ROI)s, whereas the second algorithm works with the whole frame, i.e., without applying segmentation at the level of a frame. This chapter also describes the generation of the datasets considered to evaluate both proposals, as well as the experimental methodology followed.

4.1 The Bounding box prediction scheme

This section describes the first intelligence system proposed, which is shown in Fig. 4.1. The system is characterized by segmenting the input IR video stream into ROIs, also named bounding boxes, where possible targets are. The proposed method involves a hierarchical process of ATR including a set of iterative stages which start from detecting video frames with movements (it should be noted that moving is exceptional in this use case, meaning a possible target). Then, the frame of interest is segmented into different ROIs. Finally, each ROI is classified based on features expressed in the form of the BoF technology. This process is described in the next subsections.

4.1.1 Moving object detection

As introduced before, movement detection is crucial in this system, because i) movement is considered exceptional meaning a possible target and ii) focusing on specific video frames saves computing cost, which is crucial in the edge computing paradigm. In this regard, there are some known methods in the literature, such as background subtraction, frame difference, and optical flow. Background subtraction is based on subtracting the background reference model from the current frame, resulting in a fast but imprecise method in dynamic

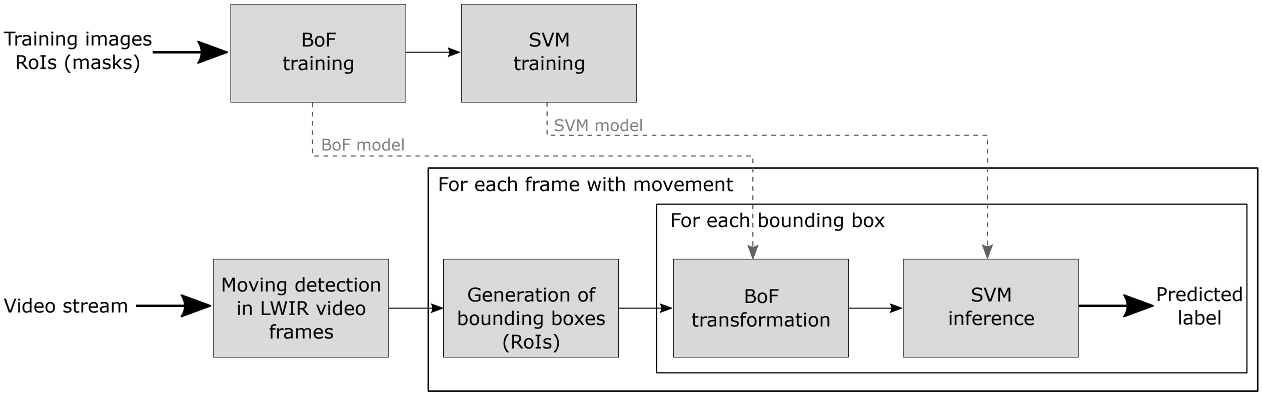


Figure 4.1: General procedure of the bounding box prediction method.

environments. Optical flow estimates moving areas by analysing the space-time gradient of the image sequence, resulting in complicated calculations and poor anti-noise performance. Frame difference sets a threshold by comparing two consecutive frames, resulting in good adaptability to dynamic environments but incorporating noise.

Additionally, it highlights the three-frame difference method, which is a simple and effective technique for detecting motion in video frames Luo et al. (2020); Zhang et al. (2020). This method involves comparing the binary frame difference of the current frame with the initial binary frame difference to determine the number of frames used to generate the background. The established background is then subtracted from the current frame to detect the moving object Boufares et al. (2021); Simsek and Ozyer (2021). Below, is included an overview of how it works and its key characteristics:

Principle:

1. **Capture Three Consecutive Frames:** The method involves capturing three consecutive frames from a video sequence.
2. **Pixel-wise Frame Difference:** Pixel-wise subtraction is performed between the first and second frames, as well as between the second and third frames.
3. **Thresholding:** A threshold is applied to the resulting frame differences to identify areas with significant changes.
4. **Combining Results:** The results from the two frame differences are often combined to enhance the accuracy of motion detection.

Characteristics:

1. **Adaptability:** The method is adaptable to dynamic environments, effectively detecting changes between frames.
2. **Noise Sensitivity:** While effective, the three-frame difference method can be sensitive to noise, leading to false positives.
3. **Real-time Processing:** It is computationally efficient, making it suitable for real-time

processing in video surveillance systems.

4. **Simple Implementation:** The simplicity of implementation makes it a popular choice for basic motion detection tasks.

Formally, the three-frame difference model works as follows (see Figure 4.2 for further clarification). First, the method starts by converting three consecutive LWIR frames (according to the use case) f_{k-1}, f_k, and f_{k+1} to greyscale images f'_{k-1}, f'_k, and $f'_k + 1$. Second, the frame different of image pairs (f'_k,f'_{k-1}) and (f'_{k+1},f'_k), denoted as $d_{k,k-1}$ and $d_{k+1,k}$, is calculated as

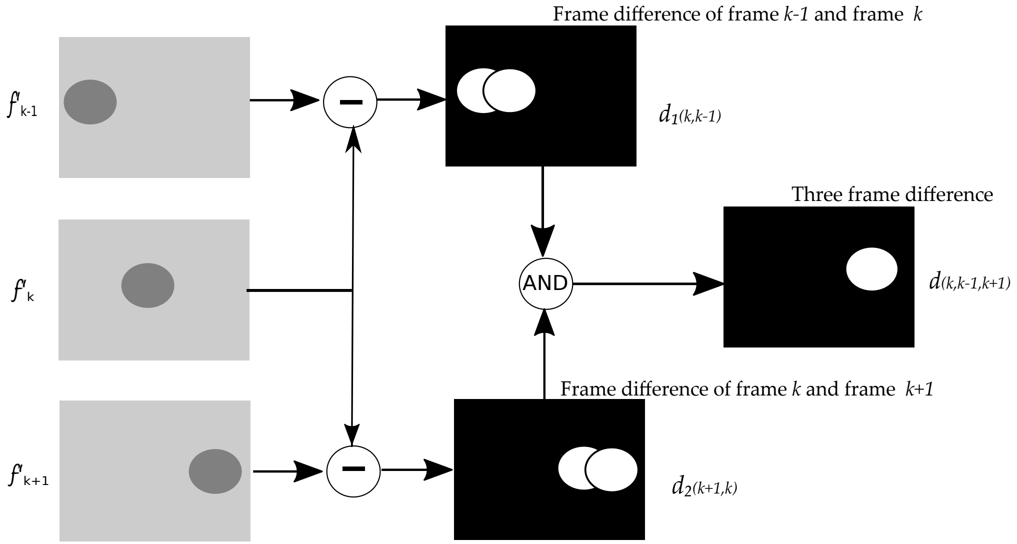

Frame difference of frame k-1 and frame k
f'k-1
−
d1(k,k-1)
Three frame difference
f'k
AND
d(k,k-1,k+1)
Frame difference of frame k and frame k+1
f'k+1
−
d2(k+1,k)

Figure 4.2: Moving object detection based on the improved three-frame difference model.

$$d(x,y)_{k,k-1} = \begin{cases} 1, & \text{if } |f(x,y)'_k - f(x,y)'_{k-1}| \geq th \\ 0, & \text{otherwise} \end{cases}, \tag{4.1}$$

$$d(x,y)_{k+1,k} = \begin{cases} 1, & \text{if } |f(x,y)'_{k+1} - f(x,y)'_{k}| \geq th \\ 0, & \text{otherwise} \end{cases}, \tag{4.2}$$

where $|\ |$ is the absolute value of a number, $th \in \mathbb{R}^+$ is a threshold based on experience, $d(x,y)_{k,k-1}$ is the value of the xy pixel in $d_{k,k-1}$, $d(x,y)_{k+1,k}$ is the value of the xy pixel in $d_{k+1,k}$, $f(x,y)'_k$ is the value of the xy pixel in f'_k, $f(x,y)'_{k+1}$ is the value of the xy pixel in f'_{k+1}, and $f(x,y)'_{k-1}$ is the value of the xy pixel in f'_{k-1}, with $x \in 1, 2 \ldots x_{max}$ and $y \in 1, 2 \ldots y_{max}$. Third, the three-frame difference image $d_{k,k-1,k+1}$ of tuple (f'_{k-1}, f'_k, f'_{k+1}) is calculated based on the AND logical operation of $d(x,y)_{k,k-1}$ and $d(x,y)_{k+1,k}$, that is

$$d(x,y)_{k,k-1,k+1} = \begin{cases} 1, & \text{if } d(x,y)_{k,k-1} \ \ \& \ \ d(x,y)_{k+1,k} = 1 \\ 0, & \text{otherwise} \end{cases}, \tag{4.3}$$

where $d(x,y)_{k,k-1,k+1}$ is the value of the xy pixel in $d_{k,k-1,k+1}$.

4.1.2 Target Segmentation in Regions of Interest

After applying the temporal difference technique on frames, several noise dots may still be part of the target shape. To obtain clear objects with certain shapes, morphological operations (such as erosion and dilatation) can be applied. Morphological operations process images based on shapes by applying a structuring element to an image, generating a new image that is sensitive to the shape of the structuring element.

After optimizing the frame obtained by the three-difference model, it is segmented into different Regions of interest (ROIs) in the form of a bounding box. Note that a bounding box is a rectangular box with sides parallel to the coordinate that contains the object (potential target). This approach means that each moving object will be its bounding box. After obtaining motion detection from the three frame differences stage the next step is to compute the grayscale, by the threshold of the frame difference to get a motion mask. To get bounding box detections, we will find the contours on the motion mask, and then draw a bounding box around them if they are large enough. The Python code effectively draws a bounding box around each contour detected in the frame as follows:

```python

ret,thresh = cv2.threshold(frame, 140, 255, cv2.THRESH_BINARY_INV);
_, contours, hierarchy = cv2.findContours(thresh,cv2.RETR_EXTERNAL,cv2.
    CHAIN_APPROX_SIMPLE)
for (i, c) in enumerate(contours):
    [x,y,w,h] = cv2.boundingRect(c)
    if h>300 and w>300:
        continue
    if h<40 or w<40:
        continue
    cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2
    roi = thresh[y:y+h,x:x+w]
```

Listings 4.1: Python code of Draw Bounding Box

This code snippet computes the bounding box for each contour detected in the frame and draws the bounding box on the frame.

- $cv2.boundingRect(c)$: This function computes the minimum area rectangle that encloses the contour. Which computes the bounding rectangle for the contour c. It returns four values: the x and y coordinates of the top-left corner of the rectangle (x, y), and the width and height of the rectangle (w, h).
- cv2.rectangle(frame, $(x, y), (x + w, y + h), (0, 255, 0), 2)$: This function draws a rectangle on the frame. It takes several arguments: frame: The image/frame on which the rectangle will be drawn. (x, y): The coordinates of the top-left corner of the rectangle. $(x+w, y+h)$: The coordinates of the bottom-right corner of the rectangle, are calculated by adding the width and height to the top-left coordinates.

4.1.3 Bag of Feature and Target Classification

The proposal is based on considering a feature representation for the ROIs in the form of a BoF, which will be used to feed a supervised classification algorithm. As usual, this strategy has two stages: training, where the model parameters are adjusted, and prediction, where the class of the input ROI is inferred.

The training stage is designed to be executed offline at a workstation. The procedure starts by collecting labelled binary images, which will be explained later in Section 4.3, corresponding to examples of ROIs. These images are considered to train the BoF feature extraction method, which allows representing an image through a histogram of visual words. BoF training is as follows:

- Each image in the collection is abstracted by several characteristic local patches, where there is the information of interest as provided by feature descriptors. In this thesis, it is evaluated using SIFT, SURF, or ORB, but not in its combination. The information in the patch is encoded in a vector form (a word in the context of BoF) whose size is determined by the descriptor used.
- All words are used to create a dictionary through an unsupervised ML algorithm, usually k-means. The dictionary size is given by the number of clusters (codewords) found by the unsupervised algorithm.
- Each word is associated with a codeword, resulting in each image represented by a histogram of codewords. Once the BoF model is trained, the histogram of codewords and the label for each image in the set are used to train the supervised classification algorithm. This thesis evaluates the use of three different classification algorithms.

The prediction stage is designed to be executed online on the edge device. Once moving detection and segmentation tasks are performed, a BoF histogram is generated for each ROI, according to the previously obtained codewords. Next, the classification algorithm infers the class of the possible target in the ROI, which is the system output.

4.2 The Frame-based prediction scheme

This section includes the second intelligence system proposed, which is shown in Fig. 4.3. The system is characterized by not segmenting the IR video stream into ROIs, using instead the full frames. As before, the system is divided into two stages: training and prediction:

- The training stage is designed to be executed offline on a workstation. The procedure starts by collecting labeled binary images corresponding to examples of full frames. If there is more than one type of target in a full frame, the label will refer to the most dominant content to the detriment of others. These images are considered to train the BoF model. Next, the histogram of codewords and the label for each image in the collection are considered to train the supervised classification algorithm.
- The prediction stage is designed to be executed online in the edge device. The procedure starts by detecting moving frames in the input IR video stream, according to the three-frame difference model. Once a moving frame is detected, a BoF histogram is generated for the full frame, i.e., without applying segmentation, based on the obtained codewords. Next, the classification algorithm infers the class of the possible target in the full frame, which is the system output.

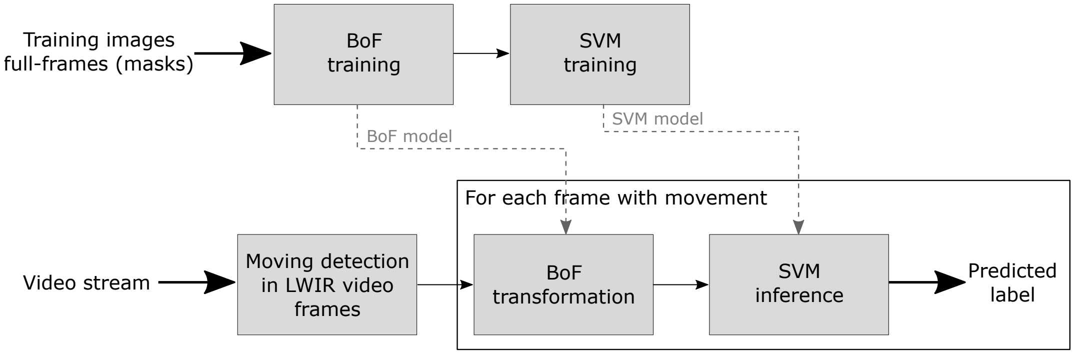


Figure 4.3: General procedure of the frame-based prediction method.

4.3 Database generation

The central focus of the thesis is compelling, addressing the application of object category and model classification through the BoF methodology. The study involves the compilation of datasets featuring images of cars, animals, and people, serving as an image database in the proposed machine learning for the target category classification system.

There are many datasets in the literature for object recognition, such as MPEG-7 Jeannin and Bober (1999), MSRA10K Wang et al. (2017), and MCL Lee et al. (2014). However, there is no dataset fitting with the requirements for this application (border surveillance in the Sahara): low-resolution silhouette images (this is the aspect after binarizing the IR image in the three-frame difference method) and up to thirteen classes. To address this challenge, four datasets were generated. These custom datasets were designed to closely align with the research objectives and enable a comprehensive analysis of the performance of the two proposed methods in the context of border surveillance in the Sahara Desert.

The main features of the four datasets are detailed below:

- dt_13_eq_ind includes 624 binary images from thirteen categories of individual targets (army-man, camel, car, cat, cow, deer, dog, elephant, giraffe, horse, person, walking-person, and truck). Images have the same size of 300x300 pixels and targets in the same class have a similar size, as Figure 4.4 shows.
- dt_3_eq_ind includes 750 binary images from three categories of individual targets (people, vehicles, and animals). Images have the same size of 480x270 pixels and targets in the same class have a similar size, as Figure 4.5 shows.
- dt_3_diff_ind includes 810 binary images from three categories (people, vehicles, and animals). Images have the same size of 480x270 pixels and targets in the same class can have a significantly different size, as Figure 4.6 shows.
- dt_6_eq_gr includes 1500 binary images from six categories combining individual and group targets (animal, vehicle, person, a group of animals, a group of vehicles, and a group of people). Images have the same size of 480x270 pixels and targets in the same class have a similar size, as Figure 4.7 shows.

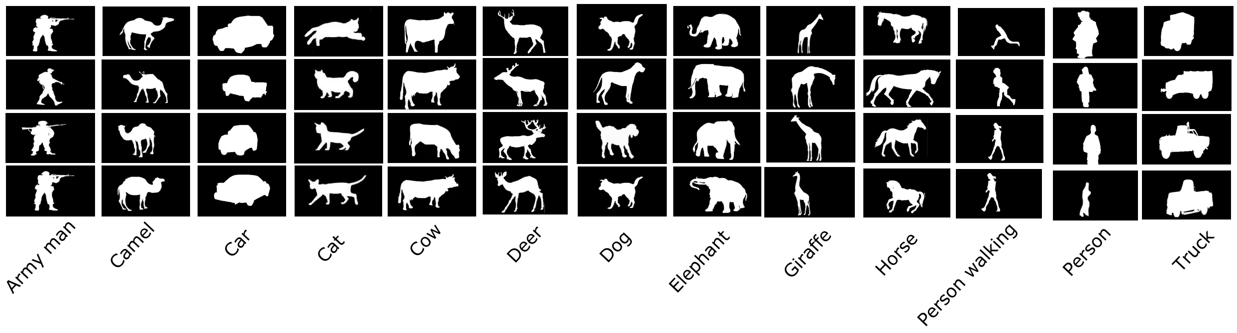
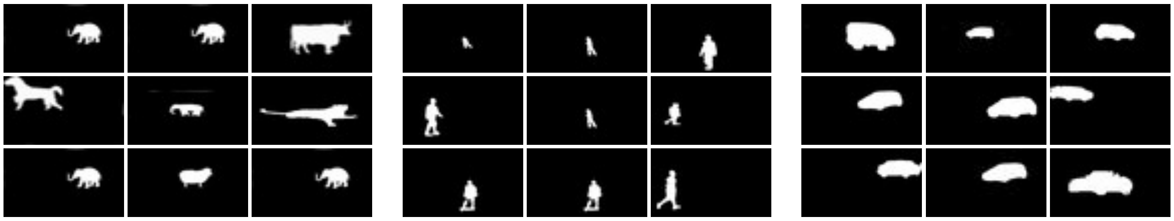


Figure 4.4: Example for dt_13_eq_ind.



(a) Class for animals.

(b) Class for people.

(c) Class for vehicles.

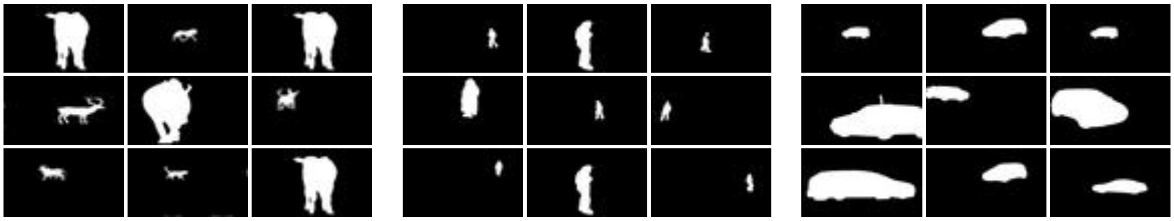
Figure 4.5: Example for dt_3_eq_ind.

For dt_13_eq_ind and dt_3_eq_ind The silhouette databases combine silhouettes from the two previously mentioned databases with ones that are generated from video frames. By identifying silhouette databases that contain the categories we are interested in. since they are poor in a diversity of silhouette images and since two databases may not cover all categories, it was essential. For categories where existing databases are insufficient, such as people and certain types of vehicles and animals, silhouette creation was done by extracting frames from video footage from an LWIR camera to isolate the silhouettes from the background.
For dt_3_diff_ind are obtained by modifying the previous dataset dt_3_eq_ind by applying transformations like rotation, and scaling of the size of objects in images. Finally, the dt_6_eq_gr image dataset primarily was generated from LWIR camera footage, with some manual modifications to scale sizes.

4.4 Experimental Methodology

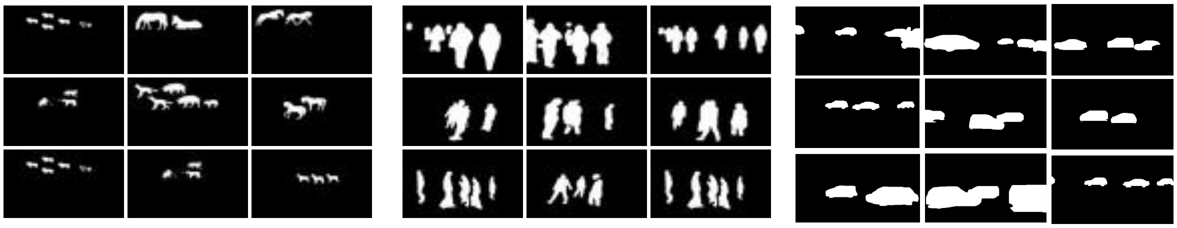
The two approaches are evaluated under the use case. To this end, first, the bounding box prediction scheme is evaluated under three different cases where ROIs are classified as follows:

- Classification of 13 categories with targets of similar size (use of the dt_13_eq_ind dataset).
- Classification of 3 categories with targets of similar size (use of the dt_3_eq_ind dataset).
- Classification of 3 categories with targets of different sizes (use of the dt_3_diff_ind



(a) Class for animals. (b) Class for people. (c) Class for vehicles.

Figure 4.6: Example for dt_3_diff_ind.



(a) class for groups of animals. (b) class for groups of people. (c) class for groups of vehicles.

Figure 4.7: Example for dt_6_gr.

dataset).

For each case, three supervised classification algorithms (SVM, KNN, Naive Bayes, and Decision trees) and three feature descriptors (SIFT, ORB, and SURF) are considered, resulting in 36 cases (4 classification algorithms × 3 feature descriptors × 3 use cases or datasets). In the evaluation of each of the 36 cases, a hold-out strategy was considered, randomly taking 80% of a dataset for training and using the remaining 20% of the same dataset for assessment purposes.

Second, the frame-based prediction scheme is evaluated under one case where whole frames are classified as follows:

- Classification of 6 categories with targets of similar size (use of the dt_6_eq_gr dataset).

All experiments were performed on a desktop with a 2.93GHz Intel Core 2 Duo E7 Processor 2.93 GHz and 2 GB RAM, running a 64-bit Ubuntu with Python and Open CV libraries. The source code generating and evaluating these models can be found in Annex 6.4. Additionally, an edge implementation was also done to validate the proposal in a real edge platform, as will be described in Section 5.4.

Chapter 5

Results and Discussion

This chapter presents and discusses the results obtained evaluating the two intelligent systems in the border surveillance use case based on LWIR video streams under different assumptions regarding the type of targets, the feature descriptors and the supervised classification algorithms. It also presents an analysis of the proposals in the edge to validate the two solutions in a real platform. Finally, the results are put in the context of the state-of-the-art.

5.1 Results of the bounding box prediction scheme

As introduced before, the bounding box prediction method is evaluated by applying three feature descriptors (SURF, SIFT, and ORB), four classification algorithms (SVM, KNN, Decision Trees, and Naive Bayes), and three types of scenarios (13 targets of similar size, 3 targets of similar sizes, and 3 targets of different sizes). Below, the results are organized in terms of the classification algorithm applied.

The bounding box prediction method with SVM

Table 5.1 shows precision, recall, f-score, and accuracy for the three feature descriptors, when applying SVM to the three datasets. Starting with dt_13_eq_ind, we can verify that the usage of bounding box prediction with SURF provided the best performance with up to 82% for all the metrics, while ORB and SIFT provided up to 78% and 73%, respectively. For dt_3_eq_ind, we can also verify that the usage of bounding box prediction with SURF provided the best performance with up to 97% for all the metrics, while ORB and SIFT provided up to 98% and 91%, respectively. For dt_3_diff_ind, we can verify again that the usage of bounding box prediction with SURF provided the best performance with up to 91% for all the metrics, while ORB and SIFT provided up to 83% and 91%, respectively. That means that using SVM in combination with the bounding box prediction method allows getting performance metrics higher than 80% when classifying among 13 targets and higher than 90% when classifying 3 targets.

The bounding box prediction method with KNN

Table 5.2 shows precision, recall, f-score, and accuracy for the three feature descriptors, when applying KNN to the three datasets. Starting with dt_13_eq_ind, it is evident that employing the bounding box prediction with SIFT gave the highest performance across all metrics, achieving up to 61%. In comparison, SURF and ORB achieved up to 58% and 52%, respectively. For dt_3_eq_ind, it is evident that employing the bounding box prediction with ORB gave the highest performance across all metrics, achieving up to 95%. In comparison, SURF and SIFT achieved up to 92% and 76%, respectively. For dt_3_diff_ind, it is evident that employing the bounding box prediction with ORB gave the highest performance across all metrics, achieving up to 91%. In comparison, SURF and SIFT achieved up to 83.3% and 74.7%, respectively. This means that, in this case, the performance is significantly higher when focusing on classifying among 3 targets, having performance metrics higher than 90%. However, when focusing on 13 targets the performance is significantly reduced, having performance metrics higher than 60%.

The bounding box prediction method with Naive Bayes

Table 5.3 shows precision, recall, f-score, and accuracy for the three feature descriptors, when applying Naive Bayes to the three datasets. Starting with dt_3_diff_ind, it is evident that employing the bounding box prediction with SURF gave the highest performance across all metrics, achieving up to 67%. In comparison, SIFT and ORB achieved up to 62% and 52%, respectively. For dt_3_eq_ind, it is evident that employing the bounding box prediction with ORB gave the highest performance across all metrics, achieving up to 87%. In comparison, SURF and SIFT achieved up to 78% and 68%, respectively. For dt_3_diff_ind, it is evident that employing the bounding box prediction with SURF gave the highest performance across all metrics, achieving up to 90%. In comparison, SIFT and ORB achieved up to 63% and 86%, respectively. This means that, in this case, the performance is significantly higher when focusing on classifying among 3 targets, having performance metrics higher than 90%. However, when focusing on 13 targets the performance is significantly reduced, having performance metrics higher than 60%.

The bounding box prediction method with Decision Trees

Table 5.4 shows precision, recall, f-score, and accuracy for the three feature descriptors, when applying Decision Trees to the three datasets. Starting with dt_3_diff_ind, it is evident that employing the bounding box prediction with SURF gave the highest performance across all metrics, achieving up to 68%. In comparison, SIFT and ORB achieved up to 62 % and 65%, respectively. For dt_3_eq_ind, it is evident that employing the bounding box prediction with SURF gave the highest performance across all metrics, achieving up to 89%. In comparison, SIFT and ORB achieved up to 83% and 55%, respectively. For dt_3_diff_ind, it is evident that employing the bounding box prediction with SURF gave the highest performance across all metrics, achieving up to 88%. In comparison, SIFT and ORB achieved up to 85% and 74%, respectively. This means that, in this case, the performance is significantly higher when focusing on classifying among 3 targets, having performance metrics higher than 80%. However, when focusing on 13 targets the performance is significantly reduced, having performance metrics higher than 60%.

5.2 Discussions on the bounding box prediction scheme

Differences in the number of classes

From the previous results, we check that the bounding box scheme works better predicting over a smaller number of classes. This fact could have two causes. First, it could be caused by an inherent problem from the multi-class effect in machine learning algorithms, meaning that it is hard to train a classifier to detect among many classes, working better with as few classes as possible. Second, it could be caused by the low resolution of LWIR images, meaning that it is especially difficult to distinguish between similar silhouettes. For instance, according to the confusion matrix applying the bounding box with SVM classifying 13 targets for SURF (see Figure 5.9 in Appendix I), the horse class is confused with deer and dog classes, being animals a more generic class. Some examples of such similarities can be found in Figure 5.1, where it can be seen that there are some similarities between the silhouettes of camels and cows.

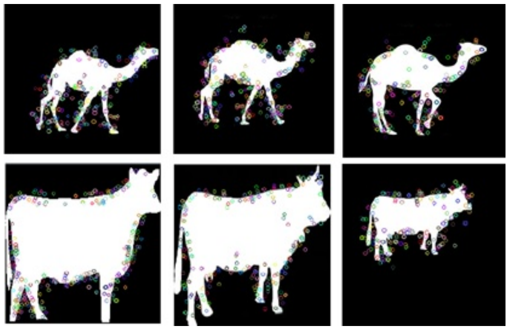


Figure 5.1: Similarity between the silhouette of the shapes objects

Then, according to this technological limitation, the proposal is to merge similar targets under the same class (for instance cars, truck, tractor, and vans can be merged in the vehicle class). To evaluate the advantages of following this scheme, it is possible to directly compare the results of solving dt_13_eq_indr to dt_3_eq_ind datasets, because it was the strategy when designing these two datasets. Then, SVM goes from 82% to 97%, KNN from 61% to 95%, Naive Bayes from 67% to 87%, and Decision Trees from 68% to 89%.

Consequently, by treating all moving objects as a single category, the classification task becomes simpler, reducing the complexity of the model and the number of distinct classes. Grouping similar objects into broader categories allows the model to generalize better to unseen data. Instead of learning specific features for each object type, the model learns more general patterns that apply to all types of general categories. In datasets with limited samples for each object class, grouping objects into broader categories can help also mitigate data sparsity issues. Grouping similar objects into one category can lead to improved classification performance, especially when dealing with noisy or ambiguous data. The model focuses on learning high-level features that are common across all types of vehicles, rather than fine-grained details that may vary between different classes. Training a model with fewer output classes requires less computational resources and training time compared to a model with a larger number of distinct classes. This can be particularly advantageous in resource-constrained environments or when working with large-scale datasets.

Differences in target sizes

From the previous results, it is also possible to identify that there is a significant difference in performance when working with targets of different sizes. It can be checked by comparing the results obtained from the dt_3_eq_ind to the dt_3_diff_ind, where three types of targets are classified. For instance, with SVM from 97% to 91%, with KNN from 95% to 91%, and with Decision Trees from 89% to 88%, except for Naive Bayes where there is an increase in performance from 87% to 90%. It should be noted that although the performance is usually reduced when considering targets with different sizes, the drop in performance is not as pronounced as when using many classes.

This situation is due to smaller objects may contain less discriminative information, leading to reduced classification accuracy or detection performance. As a solution, in future approaches, increasing data could help address differences in target sizes within the dataset by artificially generating additional training samples with variations in scale, rotation, and perspective.

Differences in descriptors and classifiers

From the previous results, it is observed that SVM with SURF outperforms any other approach for all datasets. Thus, for the 13-target dataset, it obtains 82% instead of 61% (KNN with SIFT), 67% (Naive Bayes with SURF), and 68% (Decision Trees with SURF), then, it performs very high performance in comparison to the other approaches. For the 3-target dataset with a similar size, it obtains 97% instead of 95% (KNN with ORB), 87% (Naive Bayes with ORB), and 89% (Decision Trees with SURF), then, it performs again a significantly better performance than the other approaches. For the 3-target dataset with a different size, it obtains 91% instead of 90% (Naive Bayes with SURF), and 88% (Decision Trees with SURF), getting the same performance as KNN with ORB. Then, it provides a better or similar performance than the other techniques. This suggests that SVM-based approaches offer more effective and discriminative classification models compared to other classifiers, especially in the context of the given datasets and descriptors. Overall, the results underscore the effectiveness of SVM classifiers in handling classification tasks for automatic target recognition, highlighting their potential for use in real-world applications in border surveillance and related domains.

5.3 Results and discussion on the frame-based prediction scheme

In this section, we analyse the performance of the frame-based prediction method while using SURF, SIFT, and ORB descriptors in combination with the four classification algorithms. To conduct this analysis, we consider the database that combines individual and group targets, known as the dt_6_eq_gr database. The reason for using this particular database is that the method does not have a procedure for identifying ROIs; instead, it aims to predict the prevalent class in the entire frame. By evaluating the performance of the frame-based prediction method across different descriptors using the database, we can assess its effectiveness in accurately classifying images based on their overall content. This analysis will provide insights into the suitability of the method for image category classification tasks and highlight any strengths or weaknesses associated with specific descriptors. Finally, this information can inform the selection of appropriate techniques for image classification in various applications, including border surveillance and object detection. Understanding the performance of the frame-based prediction method in this context is crucial for optimizing its use and maximizing its effectiveness in practical scenarios.

Table 5.5 shows precision, recall, f-score, and accuracy for the three feature descriptors when applying each classification algorithm to dt_6_eq_gr. Starting with SVM, ORB provided the best performance with up to 91%, followed by SIFT with up to 87% and ORB with up to 90%. For KNN, the best performance is provided by ORB with up to 83%, followed by 72% with SIFT and 75% with SURF. For Naive Bayes, SURF provided the best performance with up to 77%, followed by ORB with 72% and SIFT with 69%. For Decision Trees, ORB provided the best performance with up to 84%, followed by SIFT with 83% and SURF with 82%. Then, from a general point of view, SVM provides significantly better performance in combination with any descriptor, and especially in combination with ORB, providing the best performance (91%).

To compare the frame-based prediction scheme with the bounding box prediction scheme, it can be considered the results obtained while solving dt_3_eq_ind by SVM with SURF (bounding box) and dt_6_eq_gr with SVM with ORB (frame-based). In this comparison, it can be checked that the frame-based approach obtains very competitive results getting 0.91% instead of 0.97%. This finding is significant because the frame-based approach incurs lower computing costs than the bounding box method, as it does not require the study of all Region of Interests (ROIs) identified. Consequently, the frame-based approach could be effectively deployed in power or computing capacity-constrained sensing devices, potentially reducing the cost associated with such surveillance systems. This advantage in computing cost will be addressed in the next section, implementing both approaches in a real edge device.

5.4 Evaluating the Proposals in the Edge

In this section, the two proposals are evaluated based on a trade-off between computing cost and detection capacity. The evaluation considers two different hardware platforms: a workstation and an edge device. The workstation used for the evaluation features a 3.6 GHz Intel Core i7-7700 processor with 16 GB RAM, running a 64-bit Ubuntu operating system. On the other hand, the edge device is a PYNQ-Z1 platform by Xilinx, which includes a Field Programmable System-on-Chip (FPSoC) combining a Field-Programmable Gate Array (FPGA) and a 650MHz dual-core Cortex-A9 processor. However, for this evaluation, only a single-core implementation of the proposals is considered on the edge device, running under a Linux kernel. It is important to note that execution times could potentially be reduced by implementing parallel strategies, such as creating hardware accelerators in the FPGA for specific parts of the algorithm or utilizing both cores of the processor. Both proposals were fully encoded in Python 2.7, leveraging the OpenCV and Scikit-learn libraries for implementation. This standardized coding environment ensures consistency and allows for easy comparison of the performance of the proposals on different hardware platforms. By evaluating the execution times and resource utilization on both the workstation and the edge device, the assessment aims to measure the feasibility and efficiency of deploying the proposals in scenarios with varying computational constraints.

Starting with the bounding box prediction method, Table 5.6 shows the average computing times (in seconds) obtained while inferring about the dt_3_eq_ind dataset for each pair descriptor-classification algorithm in the workstation and the PYNQ-Z1 device. The total average time (*Total* field) for a run is split into the time needed to get the feature descriptors (*Descriptor* field), to apply the BoF transformation (*BoF* field), and to predict using the supervised algorithm (*ML* field). As each of the frames in the dt_3_eq_ind dataset only includes an ROI, the ROI rate is calculated as one second divided by the total time, meaning the number of ROIs that can be calculated in one second. The speed-up metric (*Speedup* field) is also shown, denoting the number of times that the implementation in the PYNQ-Z1 device is slower than the workstation. Analysing this table, we reach that i) ORB needs the lowest computing time among the descriptors, ii) DT needs the lowest computing time among the classification algorithms, and iii) speed-up is about 10.

Based on the computing times in Table 5.6 and the detection capacity analysis in Table 5.5 , Figure 5.2 shows a trade-off between ROI rate (related to computing effort) and detection capacity for each pair descriptor-classification algorithm in the edge device. Note that the detection capacity is obtained as the average for the three metrics considered: precision, f-score, and recall. The data in this figure is analysed by following a Pareto theory strategy for a multi-objective problem, where both ROI rate and detection capacity are the objective functions to maximize and have two constraints defined by the application: detection capacity $\geq 85\%$ and ROI rate $\geq 0.5 ROIs/s$. As a result, the non-dominated Pareto front is composed of three solutions with the same relevance. They are from lower to higher detection capacity and from higher to lower ROI rates as follows: ORB-NB, ORB-SVM, and SURF-SVM.

Following with the frame-based prediction method, Table 5.7 shows the average computing times obtained while inferring about the dt_6_eq_edr dataset for each pair descriptor-classification algorithm in the workstation and the edge device. In this case, as the frames in the dataset could include several targets and the frame-based prediction method infers without extracting ROIs, working directly with the whole frame, we opted for using the frame rate metric instead of the ROI rate, meaning the number of frames that can be calculated in one second. Analysing this table, we reach the same conclusions as before, that is i) ORB needs the lowest computing time among the descriptors, ii) DT needs the lowest computing time among the classification algorithms, and iii) speedup is about 10.

Based on the computing times in Table 5.7 and the detection capacity analysis in Table 5.5 , and Figure 5.3 shows a trade-off between frame rate and detection capacity for each pair descriptor-classification algorithm in the edge device. As before, the data is analysed following a Pareto theory strategy, where both frame rate and detection capacity are the objective functions to maximize, having the constraints: detection capacity $\geq 85\%$ and frame rate $\geq 0.5 Frames/s$. Thus, the non-dominated Pareto front is composed of three solutions with the same relevance. They are from lower to higher detection capacity and from higher to lower frame rate as follows: ORB-DT and ORB-SVM.

Up to this point, we analysed both approaches through a trade-off between computing effort and detection capacity. However, both studies are not comparable because they consider different metrics defined in the context of each algorithm, i.e., frame rate and ROI rate. As a way of comparing both proposals, we opted for calculating a synthetic frame rate metric in Table 5.6 for the bounding box model, which is calculated as

$$Frame\_Rate* = ROI_n/ROI_t. \tag{5.1}$$

where ROI_n and ROI_t are the average number of ROIs in a video frame and the time needed for inferring about one ROI (the Total field in Table 5.6). Based on the average number of ROIs in dt_6_eq_edr for a frame, the synthetic frame rate in Table 5.6 is calculated assuming ROI_n equalling 4. Comparing both frame rates in Tables 5.6 and 5.7,we reach that the frame rate metric is significantly higher when applying the frame-based model, as expected because there is no need to iterate over ROIs. From this analysis, we also note that the difference observed in computing effort depends on the number of ROIs in a frame for the bounding box method, meaning that the frame-based approach is more robust in expected computing time.

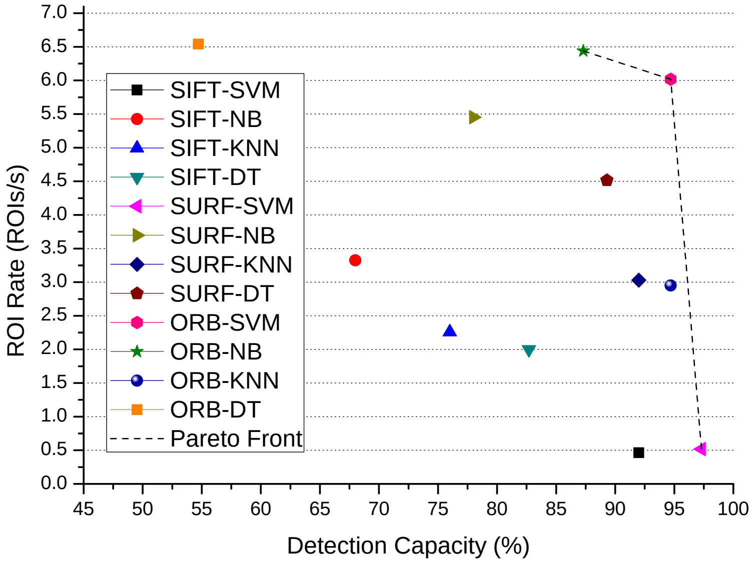


Figure 5.2: ROI rate vs. detection capacity for each pair descriptor-classification applying the bounding box prediction method to dt_3_eq_ind in the PYNQ-Z1 device.

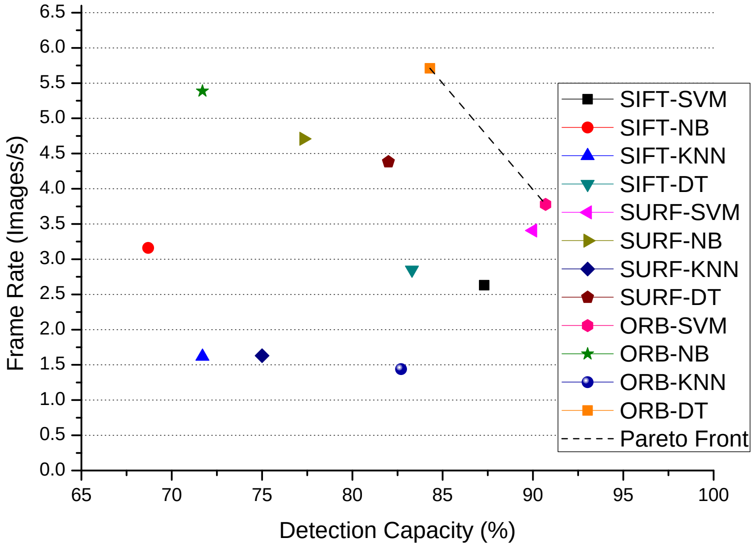


Figure 5.3: Frame rate vs. detection capacity for each pair descriptor-classification applying the frame-based prediction method to dt_6_eq_edr in the PYNQ-Z1 device.

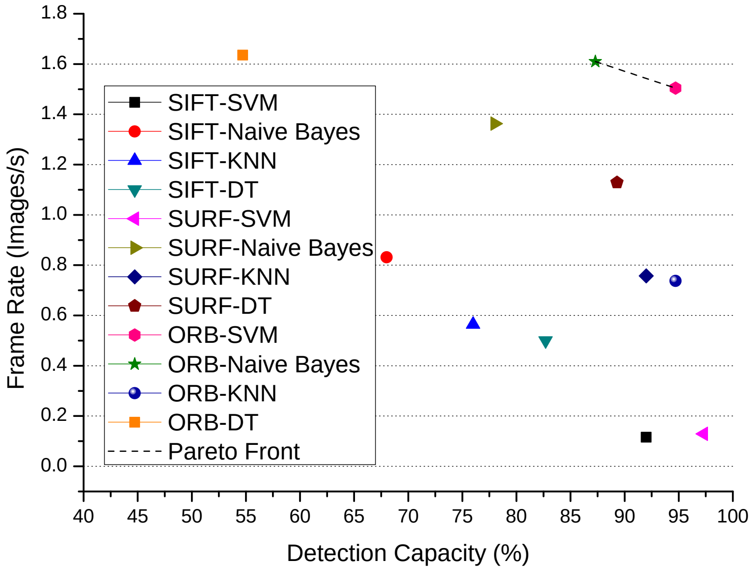


Figure 5.4: Frame rate vs Detection Capacity of each pair descriptor-classification algorithm implemented in the PYNQ platform for dt_3_eq_ind.

Table 5.1: Performance metrics applying the bounding box prediction method with SVM.

dt_13_eq_ind

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Army man	73%	80%	76%	75%	90%	82%	88%	70%	78%	10
Camel	25%	20%	22%	43%	30%	35%	67%	40%	50%	10
Car	100%	70%	82%	100%	70%	82%	69%	90%	78%	10
Cat	90%	90%	90%	67%	80%	73%	89%	80%	84%	10
Cow	83%	100%	91%	100%	90%	95%	77%	100%	87%	10
Deer	77%	100%	87%	91%	100%	95%	83%	100%	91%	10
Dog	71%	50%	59%	58%	70%	64%	75%	60%	67%	10
Elephant	78%	70%	74%	100%	80%	89%	82%	90%	86%	10
Giraffe	91%	100%	95%	100%	100%	100%	83%	100%	91%	10
Horse	56%	50%	53%	62%	50%	56%	100%	80%	89%	10
Person	50%	80%	62%	80%	80%	80%	82%	90%	86%	10
Person walking	91%	100%	95%	67%	100%	80%	100%	100%	100%	10
Truck	67%	40%	50%	78%	70%	74%	78%	70%	74%	10
Avg./Total		**73%**			**78%**			**82%**		**130**

dt_3_eq_ind

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
animal	100%	92%	96%	100%	100%	100%	100%	100%	100%	50
person	88%	92%	90%	92%	92%	92%	94%	98%	96%	50
Vehicle	88%	92%	90%	92%	92%	92%	98%	94%	96%	50
Avg./Total		**92%**			**95%**			**97%**		**150**

dt_3_diff_ind

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	95%	98%	96%	98%	100%	99%	98%	98%	98%	54
Person	81%	63%	71%	89%	57%	70%	92%	81%	86%	54
Vehicle	69%	81%	75%	68%	91%	78%	85%	94%	89%	54
Avg./Total		**91%**			**83%**			**91%**		**162**

Table 5.2: Performance metrics applying the bounding box prediction method with KNN.

dt_13_eq_ind

	BoF with SIFT			**BoF with ORB**			**BoF with SURF**			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Army_man	73%	80%	76%	100%	30%	46%	100%	30%	46%	10
Camel	100%	20%	33%	0%	0%	0%	33%	20%	25%	10
Car	46%	60%	52%	67%	60%	63%	41%	70%	52%	10
Cat	67%	80%	73%	75%	60%	67%	80%	80%	80%	10
Cow	90%	90%	90%	44%	80%	57%	43%	100%	61%	10
Deer	70%	70%	70%	37%	70%	48%	86%	60%	71%	10
Dog	33%	10%	15%	25%	10%	14%	0%	0%	0%	10
Elephant	88%	70%	78%	100%	40%	57%	100%	60%	75%	10
Giraffe	67%	100%	80%	43%	100%	61%	71%	100%	83%	10
Horse	100%	30%	46%	43%	30%	35%	100%	20%	33%	10
Person	41%	90%	56%	88%	70%	78%	47%	70%	56%	10
Person walking	56%	90%	69%	67%	100%	80%	67%	100%	80%	10
Truck	0%	0%	0%	25%	30%	27%	56%	50%	53%	10
Avg./Total	**61%**			**52%**			**58%**			**130**

dt_3_eq_ind

	BoF with SIFT			**BoF with ORB**			**BoF with SURF**			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	93%	84%	88%	100%	94%	97%	100%	82%	90%	50
Person	85%	46%	60%	91%	96%	93%	83%	96%	89%	50
Vehicle	63%	98%	77%	94%	94%	94%	96%	98%	97%	50
Avg./Total	**76%**			**95%**			**92%**			**150**

dt_3_diff_ind

	BoF with SIFT			**BoF with ORB**			**BoF with SURF**			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	100%	93%	96%	100%	91%	95%	100%	83%	91%	54
Person	72%	39%	51%	82%	98%	89%	69%	91%	78%	54
Vehicle	60%	93%	73%	94%	83%	88%	89%	76%	82%	54
Avg./Total	**75%**			**91%**			**83%**			**162**

Table 5.3: Performance metrics applying the bounding box prediction method with Naive Bayes.

NB

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Army_man	69%	90%	78%	67%	60%	63%	42%	50%	45%	10
Camel	62%	50%	56%	67%	20%	31%	43%	30%	35%	10
Car	75%	90%	82%	47%	80%	59%	91%	100%	95%	10
Cat	67%	60%	63%	50%	50%	50%	80%	40%	53%	10
Cow	77%	100%	87%	100%	60%	75%	100%	90%	95%	10
Deer	50%	80%	62%	58%	70%	64%	58%	70%	64%	10
Dog	33%	20%	25%	14%	10%	12%	50%	40%	44%	10
Elephant	86%	60%	71%	50%	70%	58%	73%	80%	76%	10
Giraffe	100%	80%	89%	80%	80%	80%	83%	100%	91%	10
Horse	33%	40%	36%	29%	40%	33%	70%	70%	70%	10
Person	50%	10%	17%	12%	10%	11%	70%	70%	70%	10
Person walking	53%	100%	69%	91%	100%	95%	50%	100%	67%	10
Truck	60%	30%	40%	33%	30%	32%	100%	30%	46%	10
Avg./Total	**62%**			**52%**			**67%**			**130**

dt_3_eq_ind

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	64%	92%	75%	100%	98%	99%	91%	100%	95%	50
Person	65%	22%	33%	88%	72%	79%	90%	38%	54%	50
Vehicle	74%	90%	81%	77%	92%	84%	65%	96%	77%	50
Avg./Total	**68%**			**87%**			**78%**			**150**

dt_3_eq_diff

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	91%	91%	91%	89%	94%	92%	95%	98%	96%	54
Person	47%	39%	42%	84%	85%	84%	90%	80%	84%	54
Vehicle	51%	59%	55%	84%	78%	81%	86%	93%	89%	54
Avg./Total	**63%**			**86%**			**90%**			**162**

Table 5.4: Performance metrics applying the bounding box prediction method with Decision Trees.

dt_13_eq_ind

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Army_man	38%	30%	33%	57%	40%	47%	67%	40%	50%	10
Camel	22%	20%	21%	100%	30%	46%	60%	30%	40%	10
Car	100%	80%	89%	80%	80%	80%	50%	60%	55%	10
Cat	50%	80%	62%	80%	80%	80%	62%	80%	70%	10
Cow	100%	80%	89%	89%	80%	84%	56%	90%	69%	10
Deer	69%	90%	78%	73%	80%	76%	67%	100%	80%	10
Dog	80%	40%	53%	44%	40%	42%	67%	60%	63%	10
Elephant	60%	30%	40%	33%	40%	36%	86%	60%	71%	10
Giraffe	67%	80%	73%	83%	100%	91%	91%	100%	95%	10
Horse	56%	50%	53%	57%	40%	47%	75%	60%	67%	10
Person	70%	70%	70%	62%	100%	77%	60%	60%	60%	10
Person_walking	53%	100%	69%	83%	100%	91%	91%	100%	95%	10
Truck	50%	40%	44%	33%	40%	36%	57%	40%	47%	10
Avg./Total	**62%**			**65%**			**68%**			**130**

dt_3_eq_ind

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	89%	94%	91%	70%	84%	76%	94%	96%	95%	50
Person	72%	82%	77%	50%	54%	52%	86%	84%	85%	50
Vehicle	90%	72%	80%	36%	26%	30%	88%	88%	88%	50
Avg./Total	**83%**			**55%**			**89%**			**150**

dt_3_diff_ind

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	93%	96%	95%	91%	94%	93%	98%	93%	95%	54
Person	77%	76%	77%	77%	43%	55%	87%	83%	85%	54
Vehicle	83%	81%	82%	61%	85%	71%	80%	87%	83%	54
Avg./Total	**85%**			**74%**			**88%**			**162**

Table 5.5: Performance metrics applying the frame-based prediction method to dt_6_eq_gr.

SVM

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	86%	100%	93%	94%	100%	97%	86%	100%	93%	50
Group of animals	100%	100%	100%	98%	100%	99%	100%	100%	100%	50
Group of people	91%	86%	89%	88%	92%	90%	96%	86%	91%	50
Group of vehicle	79%	90%	84%	89%	94%	90%	85%	94%	90%	50
Person	82%	84%	83%	83%	86%	84%	85%	88%	86%	50
Vehicle	86%	64%	74%	92%	72%	81%	90%	72%	80%	50
Avg./Total	**87%**			**91%**			**90%**			**300**

KNN

	BoF with SIFT			BoF with ORB			BoF with SURF				
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**	
Animal	77%	77%	86%	81%	98%	80%	88%	81%	86%	83%	50
group of animals	100%	100%	100%	91%	100%	95%	100%	100%	100%	50	
group of people	72%	36%	48%	94%	68%	79%	100%	60%	75%	50	
group of vehicle	60%	78%	68%	88%	74%	80%	82%	64%	72%	50	
person	57%	64%	60%	68%	98%	80%	51%	84%	63%	50	
Vehicle	69%	66%	67%	70%	76%	73%	62%	56%	59%	50	
Avg./Total	**72%**			**83%**			**75%**			**300**	

Naive Bayes

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	71%	72%	71%	81%	78%	80%	84%	84%	84%	50
group of animals	86%	100%	93%	84%	96%	90%	89%	96%	92%	50
group of people	65%	52%	58%	60%	58%	59%	76%	70%	73%	50
group of vehicle	74%	50%	60%	77%	60%	67%	60%	72%	65%	50
person	58%	62%	60%	66%	74%	70%	89%	68%	77%	50
Vehicle	59%	76%	67%	62%	64%	63%	71%	74%	73%	50
Avg./Total	**69%**			**72%**			**77%**			**300**

Decision Trees

	BoF with SIFT			BoF with ORB			BoF with SURF			
Target	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**precision**	**recall**	**f-score**	**support**
Animal	89%	100%	94%	88%	100%	93%	79%	100%	88%	50
group of animals	94%	100%	97%	98%	100%	99%	98%	100%	99%	50
group of people	82%	66%	73%	81%	60%	69%	93%	74%	82%	50
group of vehicle	86%	86%	86%	85%	100%	92%	83%	86%	84%	50
person	71%	74%	73%	71%	72%	71%	70%	80%	75%	50
Vehicle	76%	74%	75%	82%	74%	78%	70%	52%	60%	50
Avg./Total	**83%**			**84%**			**82%**			**300**

Table 5.6: Computing time analysis applying the bounding box prediction method to dt_3_eq_ind.

Workstation

	SVM			KNN			NB			DT		
Descriptor	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**
Descriptor(s)	0.0287	0.0104	0.0141	0.0320	0.0102	0.0184	0.0312	0.0108	0.0180	0.0326	0.0111	0.0199
BoF(s)	0.0052	0.0033	0.0042	0.0010	0.0031	0.0011	0.0010	0.0034	0.0011	0.0154	0.0033	0.0028
ML(s)	0.1305	0.0021	0.1247	0.0100	0.0146	0.0107	0.0005	0.0009	0.0005	0.0003	0.0007	0.0003
Total	**0.1643**	**0.0158**	**0.1430**	**0.0431**	**0.0280**	**0.0301**	**0.0326**	**0.0152**	**0.0196**	**0.0484**	**0.0151**	**0.0231**
ROI_Rate	6.0868	63.4339	6.9919	23.2304	35.7145	33.1603	30.6413	65.9489	51.1381	20.6790	66.2866	43.2950
Frame_Rate*	1.5217	15.8585	1.7480	5.8076	8.9286	8.2901	7.6603	16.4872	12.7845	5.1697	16.5717	10.8237

PYNQ-Z1 (single-core)

	SVM			KNN			NB			DT		
Descriptor	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**
Descriptor(s)	0.2586	0.0934	0.1275	0.2889	0.0923	0.1662	0.2815	0.0976	0.1621	0.2945	0.1003	0.1800
BoF(s)	0.0676	0.0437	0.0553	0.0134	0.0410	0.0140	0.0129	0.0448	0.0143	0.2011	0.0430	0.0367
ML(s)	1.8336	0.0290	1.7520	0.1405	0.2056	0.1500	0.0064	0.0129	0.0070	0.0047	0.0096	0.0048
Total(s)	**2.1597**	**0.1662**	**1.9349**	**0.4429**	**0.3390**	**0.3302**	**0.3008**	**0.1553**	**0.1834**	**0.5003**	**0.1529**	**0.2215**
ROI_Rate	0.4630	6.0176	0.5168	2.2578	2.9502	3.0289	3.3246	6.4386	5.4526	1.9986	6.5408	4.5143
Frame_Rate*	0.1158	1.5044	0.1292	0.5644	0.7375	0.7572	0.8311	1.6097	1.3631	0.4997	1.6352	1.1286

Speedup	13.1460	10.5415	13.5283	10.2890	12.1058	10.9479	9.2167	9.3787	10.2427	10.3466	10.1343	9.5907

Table 5.7: Computing time analysis applying the frame-based prediction method to dt_6_eq_edr.

Workstation

	SVM			KNN			NB			DT		
Descriptor)	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**
Descriptor(s)	0.0321	0.0109	0.0216	0.0317	0.0111	0.0227	0.0328	0.0116	0.0198	0.0369	0.0115	0.0226
BoF(s)	0.0014	0.0043	0.0019	0.0013	0.0048	0.0017	0.0013	0.0048	0.0018	0.0015	0.0047	0.0017
ML(s)	0.0053	0.0074	0.0052	0.0216	0.0360	0.0264	0.0007	0.0013	0.0009	0.0004	0.0007	0.0004
Total(s)	**0.0388**	**0.0227**	**0.0287**	**0.0547**	**0.0519**	**0.0508**	**0.0347**	**0.0176**	**0.0225**	**0.0387**	**0.0169**	**0.0247**
Frame_Rate	25.7916	44.0615	34.8286	18.2894	19.2616	19.6838	28.7826	56.7161	44.4180	25.7732	59.1716	40.4858

PYNQ-Z1 (single-core)

	SVM			KNN			NB			DT		
Descriptor	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**	**SIFT**	**ORB**	**SURF**
Descriptor(s)	0.2836	0.0967	0.1903	0.2798	0.0982	0.2002	0.2896	0.1020	0.1751	0.3254	0.1015	0.1999
BoF(s)	0.0186	0.0582	0.0263	0.0182	0.0651	0.0232	0.0171	0.0651	0.0243	0.0203	0.0631	0.0225
ML(s)	0.0778	0.1099	0.0769	0.3198	0.5316	0.3903	0.0097	0.0186	0.0129	0.0054	0.0105	0.0058
Total(s)	**0.3800**	**0.2647**	**0.2935**	**0.6177**	**0.6950**	**0.6137**	**0.3165**	**0.1856**	**0.2123**	**0.3511**	**0.1751**	**0.2282**
Frame_Rate	2.6316	3.7773	3.4069	1.6188	1.4389	1.6295	3.1592	5.3868	4.7098	2.8484	5.7099	4.3828

Speedup	9.8007	11.6647	10.2228	11.2979	13.3860	12.0797	9.1107	10.5288	9.4309	9.0644	10.3720	9.2359

5.5 Comparisons to the state-of-the-art

Up to this point, it is reached that the bounding box prediction method provided a detection capacity from 87% to 97% with a frame rate from 0.13 to 1.61 running on the edge device and a frame rate from 1.75 to 16.49 running on the workstation. For the frame-based prediction method, the detection capacity obtained was from 84% to 91% with a frame rate from 3.77 to 5.71 running on the edge device and a frame rate from 44.06 to 59.17 running on the workstation. Comparing such results with the works previously discussed in Table 3.1, we reach that

- The two proposals achieved detection capacity metrics in line with the state-of-the-art ATR systems based on IR imagery, being the highest detection capacity metric a value of 97%. In fact, this value is above the average detection capacity for the works in Table 3.1, which is 90.33%. Note that most of the works analysed provided the accuracy metric as a detection capacity metric instead of using more confident metrics, such as the ones considered in this thesis. Therefore, the detection capacity analysis in these works could be biased. Focusing on the works providing higher detection capacity metrics than ours (97%) Akula et al. (2020), the we reach that two of the works Akula et al. (2020); Fan et al. (2023) were based on CNNs whose approach might not be applied in edge platforms due to its resource utilization, such as convolutional layers with 256 filters.

- Regarding the computational cost, only the works in Akula et al. (2020); Baili et al. (2022); Miao and Xie (2022) provided frame rates running in a workstation. For the case of Akula et al. (2020), the frame rates shown in these works are comparable with the frame rates from our proposals running on the workstation, which is relevant because they considered powerful GPUs. For the case of Miao and Xie (2022), the algorithm obtained a frame rate of 57.29, but in a GPU and with a lower detection capacity than ours. For the case of Wu and Zhang (2023), the algorithm obtained a lower detection capacity than our approaches. For the case of Miao and Xie (2022), the frame rate is comparable, but our approach provides a higher detection capacity when considering a high frame rate. For the case of Akula et al. (2020), the frame rate could be comparable with our worst case in computational cost, but ours provides other alternatives at a different detection capacity. From the analysis of these works, we reach that the computational cost of the proposals is within the state of the art.

- There has been no mention of the computational cost at the edge in the existing literature.

Note that the comparison with previous works is limited because i) the problem addressed was not solved before in the literature, ii) the hardware used in the workstation is different in each work, and iii) the hardware used in the edge is different in each work.

Chapter 6

Conclusions

This chapter ends this thesis dissertation. To this end, it discusses the conclusions obtained from performing this research. Then, the main contributions to the field are outlined, detailing the dissemination. Finally, future lines of research are outlined, highlighting potential areas for further exploration and development.

6.1 Conclusions

This work investigated research topics related to ATR for border surveillance applications to enhance their performance, expand their capabilities, and offer new insights. The development of accurate border surveillance systems is an imperative objective for numerous nations worldwide because unauthorized entry into a country's territory is associated with significant concerns such as terrorism and drug trafficking. This thesis is focused on a specific border surveillance use case, the Sahara Desert, whose border has a wide area with a low density of crossing traffic. Under these circumstances, the surveillance system should focus on only identifying and tracking moving objects, using to this end sensors which could work at all times and weather conditions.

Under this context, the proposed surveillance system comprises multiple unattended fixed platforms with long-range sensing capabilities, implemented following an IoT approach. Edge nodes equipped with LWIR cameras and minimal hardware compute the ATR algorithm for object tracking and classification within the field of view. The thesis primarily focuses on designing the ATR algorithm, considering the limitations of energy and computing capacity at the edge layer. With an emphasis on silhouette and shape-based feature techniques, two machine learning-based approaches, suitable for execution on low-power microprocessors, are proposed. Feature extraction from IR images, known for their low resolution and quality, is emphasized, utilizingBoF based on feature descriptors. After feature extraction, content is classified using supervised algorithms. The two proposed approaches differ in segmenting the image into ROIs for independent classification or working with the entire image, offering potential trade-offs between computing cost and performance.

Both ATR systems demonstrate successful results regarding detection capacity and frame rates, showcasing the feasibility of employing edge computing and ML algorithms for border surveillance in challenging environments like the Sahara Desert. The evaluation of the bounding box prediction method in detection capacity yielded several important conclusions. In scenarios with many classes, such as dt_13_eq_ind, the combination of SURF with SVM performed the best, achieving up to 82% detection capacity. Additionally, it was recommended to use generic classes due to the low resolution of IR images, which can improve performance. For classification problems with fewer classes, like dt_3_eq_ind, the SURF with SVM combination also proved to be the most effective, reaching up to 97%. Moreover, it was observed that the confusion between classes decreased with an increase in the number of classes, indicating an improvement in detection capacity. In scenarios with fewer classes but no constraint on target distance, such as dt_3_diff_ind, the detection capacity was negatively affected due to the low resolution of IR images, getting SURF with SVM the best performance with up to 91%. For the frame-based prediction scheme, the best performance was obtained by SVM with ORB, getting up to 91%, meaning a robust performance with fewer computing requirements than the bounding box prediction scheme.

When evaluating the method on the edge, a Pareto strategy was considered with a trade-off between ROI rate (related to computing effort) and detection capacity for each pair descriptor-classification algorithm. The Pareto front was composed of three solutions with the same relevance, from lower to higher detection capacity and from higher to lower ROI rates as follows: ORB-NB, ORB-SVM, and SURF-SVM. These solutions achieved detection capacities ranging from 87% to 97% and frame rates ranging from 0.13 to 1.6. Following the frame-based prediction method, a Pareto front was obtained with ORB-DT and ORB-SVM solutions and frame rates ranging from 3.7 to 5.7 and detection capacity from 84% to 91%. It means that the frame-based prediction method provides robust performance with good detection capacity at a lower computing cost than the bounding box scheme. Also, some other conclusions were obtained, ORB needed the lowest computing time among the descriptors, DT needed the lowest computing time among the classification algorithms, and iii) speed-up was about 10, meaning the edge versions needed 10 times longer than its workstation version.

In conclusion, the thesis contributes to advancing the field of border surveillance by proposing efficient ATR algorithms tailored for edge computing environments. The findings demonstrate the feasibility and effectiveness of employing ML techniques in border surveillance systems, particularly in challenging environments like the Sahara Desert

6.2 Contributions

This thesis contributes to the development of ATR systems for border surveillance applications in difficult environments, such as the Sahara Desert. To this end, it adopted an intelligence solution with ML supervised classification algorithms fed by information expressed as BoFs in combination with specific descriptors. Integrating this scheme into an BSS edge computing solution, it was possible to achieve a balance between detection capacity, efficiency, and computational complexity, ultimately enhancing the effectiveness of border surveillance while optimizing resource utilization and reducing operational costs. In this regard, the following contributions to the field are made in this thesis:

- The proposal of two architectures based on BoF in combination with feature descriptors and supervised classification algorithms to detect targets of interest through LWIR video streams. The first architecture works within a segmentation focus. The second works with the whole frame.
- The proposal of a collection of images fitting the use case.
- The use of several classification algorithms and feature descriptors in both architectures was explored, reaching that specific combinations of classification algorithms and feature descriptors outperformed others.
- The evaluation of the methods regarding the number and sizes of targets, resulting in specific recommendations for further designs.
- Both proposals were evaluated in an edge device, checking the suitability of the proposals to run on the edge. The response times of the method working without segmentation were especially relevant, meaning it is more suitable to be used than the ROI-based method.

6.3 Publications

The present work has been published in a scientific journal:

Bellazi, Khalifa M , Marino, Rodrigo and Lanza-Gutierrez, Jose M and Riesgo, Teresa,. *Towards a machine learning-based edge computing oriented monitoring system for the desert border surveillance use case,.* journal IEEE Access, volume 8, pages 218304–218322, year 2020, publisher IEEE.

6.4 Future work

Future lines of research include Parallel Strategies on Low-Power Multicore Microprocessors and Field-Programmable Gate Arrays (FPGAs). Implementing parallel processing strategies on low-power multicore microprocessors and FPGAs could significantly improve the computational capabilities of border surveillance systems. By distributing tasks across multiple cores or hardware resources, parallel processing techniques can enhance processing speed and efficiency, leading to faster and more effective execution of ATR algorithms. The existing model is based on only detecting moving targets. An extension could enhance the system to detect and classify both moving and stationary targets simultaneously, generating to this end a more complex model which could be integrated into a parallel architecture as described before, paying special attention to the hardware-software co-design.

Reference

F. iDuan, "Artificial intelligence." *AIAA Meets IEEE. Springer, Singapore.*, 02 2023.

W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, 1943.

J. McCarthy, N. Rochester, and C. Shannon, "Dartmouth workshop," ., 1956.

A. Newell, J. C. Shaw, and H. A. Simon, "Elements of a theory of human problem solving." *Psychological review*, vol. 65, no. 3, p. 151, 1958.

A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal of research and development*, vol. 3, no. 3, pp. 210–229, 1959.

V. Karishma, T. Vigneswaran *et al.*, "Artificial intelligence in video surveillance," in *Handbook of Research on Deep Learning Techniques for Cloud-Based Industrial IoT*. IGI Global, 2023, pp. 1–17.

J. Dufour, *Intelligent Video Surveillance Systems*, ser. ISTE. Wiley, 2012. [Online]. Available: https://books.google.com.ly/books?id=tqEOLOKT_fAC

O. Elharrouss, N. Almaadeed, and S. Al-Maadeed, "A review of video surveillance systems," *Journal of Visual Communication and Image Representation*, vol. 77, p. 103116, 2021.

C. Yu, X. Zheng, Y. Zhao, G. Liu, and N. Li, "Review of intelligent video surveillance technology research," in *Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology*, vol. 1. IEEE, 2011, pp. 230–233.

M. Soundarya, P. Karthikeyan, and G. Thangarasu, "Automatic speech recognition trained with convolutional neural network and predicted with recurrent neural network," in *2023 9th International Conference on Electrical Energy Systems (ICEES)*. IEEE, 2023, pp. 41–45.

P. Sharma, *Deep Learning and Neural Networks: Methods*, 2023.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

S. Bavikadi, A. Dhavlle, A. Ganguly, A. Haridass, H. Hendy, C. Merkel, V. J. Reddi, P. R. Sutradhar, A. Joseph, and S. M. P. Dinakarrao, "A survey on machine learning accelerators and evolutionary hardware platforms," *IEEE Design & Test*, vol. 39, no. 3, pp. 91–116, 2022.

D. Ali, A. U. Rehman, and F. H. Khan, “Hardware accelerators and accelerators for machine learning,” in *2022 International Conference on IT and Industrial Technologies (ICIT)*. IEEE, 2022, pp. 01–07.

S. Nishanth, M. S. Rao, B. Sagar, T. Padmashree, and N. Cauvery, “Performance of cpus and gpus on deep learning models for heterogeneous datasets,” in *2022 6th International Conference on Electronics, Communication and Aerospace Technology*. IEEE, 2022, pp. 978–985.

S. Jordan, R. Driggers, O. Furxhi, P. Leslie, C. Cavanaugh, K. Renshaw, and E. Jacobs, “Comparison of scene contrast temperature in mid-wave infrared and long-wave infrared,” *Optical Engineering*, vol. 62, no. 11, pp. 113 107–113 107, 2023.

X. Guan, X. Yu, D. Periyanagounder, M. R. Benzigar, J.-K. Huang, C.-H. Lin, J. Kim, S. Singh, L. Hu, G. Liu *et al.*, “Recent progress in short-to long-wave infrared photodetection using 2d materials and heterostructures,” *Advanced Optical Materials*, vol. 9, no. 4, p. 2001708, 2021.

A. Goyal, S. B. Anandamurthy, P. Dash, S. Acharya, D. Bathla, D. Hicks, A. Bhan, and P. Ranjan, “Automatic border surveillance using machine learning in remote video surveillance systems,” in *Emerging Trends in Electrical, Communications, and Information Technologies: Proceedings of ICECIT-2018*. Springer, 2020, pp. 751–760.

Z. Ning, H. Hu, X. Wang, L. Guo, S. Guo, G. Wang, and X. Gao, “Mobile edge computing and machine learning in the internet of unmanned aerial vehicles: A survey,” *ACM Computing Surveys*, 2023.

M. L. Laouira, A. Abdelli, J. B. Othman, and H. Kim, “An efficient wsn based solution for border surveillance,” *IEEE Transactions on Sustainable Computing*, vol. 6, no. 1, pp. 54–65, 2021.

Z. Yang, “The development and application of uav intelligent machine learning system based on artificial intelligence,” *Advances in Education, Humanities and Social Science Research*, vol. 6, no. 1, pp. 140–140, 2023.

E. Blasch, T. Pham, C.-Y. Chong, W. Koch, H. Leung, D. Braines, and T. Abdelzaher, “Machine learning/artificial intelligence for sensor data fusion–opportunities and challenges,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 36, no. 7, pp. 80–93, 2021.

R. Singh and S. Singh, “Smart border surveillance system using wireless sensor networks,” *International Journal of System Assurance Engineering and Management*, vol. 13, no. Suppl 2, pp. 880–894, 2022.

C. Cică and L. Filipoaia, “Sustainability enabling research methods for development of integrated border security systems,” in *2017 9th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*. IEEE, 2017, pp. 1–4.

M. Bertolini, D. Mezzogori, M. Neroni, and F. Zammori, “Machine learning for industrial applications: A comprehensive literature review,” *Expert Systems with Applications*, vol.

175, p. 114820, 2021.

N. Burkart and M. F. Huber, “A survey on the explainability of supervised machine learning,” *Journal of Artificial Intelligence Research*, vol. 70, pp. 245–317, 2021.

K. K. Verma, B. M. Singh, and A. Dixit, “A review of supervised and unsupervised machine learning techniques for suspicious behavior recognition in intelligent surveillance system,” *International Journal of Information Technology*, vol. 14, no. 1, pp. 397–410, 2022.

X. Yang, Z. Song, I. King, and Z. Xu, “A survey on deep semi-supervised learning,” *IEEE Transactions on Knowledge and Data Engineering*, 2022.

P. C. Sen, M. Hajra, and M. Ghosh, “Supervised classification algorithms in machine learning: A survey and review,” in *Emerging Technology in Modelling and Graphics: Proceedings of IEM Graph 2018*. Springer, 2020, pp. 99–111.

I. Wickramasinghe and H. Kalutarage, “Naive bayes: applications, variations and vulnerabilities: a review of literature with code snippets for implementation,” *Soft Computing*, vol. 25, no. 3, pp. 2277–2293, 2021.

M. Lombardi, F. Pascale, and D. Santaniello, “Internet of things: A general overview between architectures, protocols and applications,” *Information*, vol. 12, no. 2, p. 87, 2021.

M. Sikimić, M. Amović, V. Vujović, B. Suknović, and D. Manjak, “An overview of wireless technologies for iot network,” in *2020 19th International Symposium INFOTEH-JAHORINA (INFOTEH)*. IEEE, 2020, pp. 1–6.

R. Hassan, F. Qamar, M. K. Hasan, A. H. M. Aman, and A. S. Ahmed, “Internet of things and its applications: A comprehensive survey,” *Symmetry*, vol. 12, no. 10, p. 1674, 2020.

G. Nain, K. Pattanaik, and G. Sharma, “Towards edge computing in intelligent manufacturing: Past, present and future,” *Journal of Manufacturing Systems*, vol. 62, pp. 588–611, 2022.

Z. Chang, S. Liu, X. Xiong, Z. Cai, and G. Tu, “A survey of recent advances in edge-computing-powered artificial intelligence of things,” *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 13 849–13 875, 2021.

F. Al-Doghman, N. Moustafa, I. Khalil, Z. Tari, and A. Zomaya, “Ai-enabled secure microservices in edge computing: Opportunities and challenges,” *IEEE Transactions on Services Computing*, 2022.

B. O. Martins and M. G. Jumbert, “Eu border technologies and the co-production of security ‘problems’ and ‘solutions’,” *Journal of ethnic and migration studies*, vol. 48, no. 6, pp. 1430–1447, 2022.

W. Pawgasame and K. Wipusitwarakun, “Mobile sensors’ patrol path planning in unobservable border region,” *International Journal of Intelligent Computing and Cybernetics*, vol. 13, no. 3, pp. 311–329, 2020.

M. K. Sharma, G. Singal, S. K. Gupta, B. Chandraneil, S. Agarwal, D. Garg, and D. Mukhopadhyay, “Intervenor: intelligent border surveillance using sensors and drones,” in *2021 6th International Conference for Convergence in Technology (I2CT)*. IEEE, 2021, pp. 1–7.

L. Patino, M. Hubner, R. King, M. Litzenberger, L. Roupioz, K. Michon, Ł. Szklarski, J. Pegoraro, N. Stoianov, and J. Ferryman, "Fusion of heterogenous sensor data in border surveillance," *Sensors*, vol. 22, no. 19, p. 7351, 2022.

A. S. Olagoke, H. Ibrahim, and S. S. Teoh, "Literature survey on multi-camera system and its application," *IEEE Access*, vol. 8, pp. 172 892–172 922, 2020.

C. Bolakis, V. Mantzana, P. Michalis, A. Vassileiou, R. Pflugfelder, M. Litzenberger, M. Hubner, G. Pastore, D. Oricchio, M. Desplas *et al.*, "Foldout: A through foliage surveillance system for border security," in *Technology Development for Security Practitioners.* Springer, 2021, pp. 259–279.

F. Al Fayez, M. Hammoudeh, B. Adebisi, and K. N. Abdul Sattar, "Assessing the effectiveness of flying ad hoc networks for international border surveillance," *International Journal of Distributed Sensor Networks*, vol. 15, no. 7, p. 1550147719860406, 2019.

J.-Y. Yu, Y. Kim, and Y.-G. Kim, "Intelligent video data security: A survey and open challenges," *IEEE Access*, vol. 9, pp. 26 948–26 967, 2021.

L. R. Cenkeramaddi, J. Bhatia, A. Jha, S. K. Vishkarma, and J. Soumya, "A survey on sensors for autonomous systems," in *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA).* IEEE, 2020, pp. 1182–1187.

Y. Bai, L. Lu, J. Cheng, J. Liu, Y. Chen, and J. Yu, "Acoustic-based sensing and applications: A survey," *Computer Networks*, vol. 181, p. 107447, 2020.

X. Ji, K. Chen, M. Chen, Y. Li, and X. Qian, "Secure olympics games with technology: Intelligent border surveillance for the 2022 beijing winter olympics," *Journal of Systems Architecture*, vol. 129, p. 102634, 2022.

G. G. Artan and G. S. Tombul, "The future trends of eo/ir systems for isr platforms," in *Image Sensing Technologies: Materials, Devices, Systems, and Applications IX*, vol. 12091. SPIE, 2022, pp. 76–88.

S. Léonard and C. Kaunert, "The securitisation of migration in the european union: Frontex and its evolving security practices," *Journal of Ethnic and Migration Studies*, vol. 48, no. 6, pp. 1417–1429, 2022.

C. Conte and V. Savazzi, "The financial dimension of eu external policies on migration," in *The External Dimension of EU Migration and Asylum Policies.* Nomos Verlagsgesellschaft mbH & Co. KG, 2020, pp. 325–350.

C. T. Mann, *Military Funding for Southwest Border Barriers.* Congressional Research Service, 2020.

M. Phillips, "Managing a multiplicity of interests: The case of irregular migration from libya," *Migration and Society*, vol. 3, no. 1, pp. 89–97, 2020.

M. Ali, A. Anjum, O. Rana, A. R. Zamani, D. Balouek-Thomert, and M. Parashar, "Res: Real-time video stream analytics using edge enhanced clouds," *IEEE Transactions on Cloud Computing*, vol. 10, no. 2, pp. 792–804, 2020.

B. Gaikwad and A. Karmakar, “Smart surveillance system for real-time multi-person multi-camera tracking at the edge,” *Journal of Real-Time Image Processing*, vol. 18, no. 6, pp. 1993–2007, 2021.

M. Zhang, J. Cao, Y. Sahni, Q. Chen, S. Jiang, and L. Yang, “Blockchain-based collaborative edge intelligence for trustworthy and real-time video surveillance,” *IEEE Transactions on Industrial Informatics*, vol. 19, no. 2, pp. 1623–1633, 2022.

S. Myneni, G. Agrawal, Y. Deng, A. Chowdhary, N. Vadnere, and D. Huang, “Scvs: On ai and edge clouds enabled privacy-preserved smart-city video surveillance services,” *ACM Transactions on Internet of Things*, vol. 3, no. 4, pp. 1–26, 2022.

Y.-Y. Chen, Y.-H. Lin, Y.-C. Hu, C.-H. Hsia, Y.-A. Lian, and S.-Y. Jhong, “Distributed real-time object detection based on edge-cloud collaboration for smart video surveillance applications,” *IEEE Access*, vol. 10, pp. 93 745–93 759, 2022.

Y. Myagmar-Ochir and W. Kim, “A survey of video surveillance systems in smart city,” *Electronics*, vol. 12, no. 17, p. 3567, 2023.

L. Fei and B. Han, “Multi-object multi-camera tracking based on deep learning for intelligent transportation: a review,” *Sensors*, vol. 23, no. 8, p. 3852, 2023.

B. Jiang, Q. He, P. Liu, S. Maharjan, and Y. Zhang, “Blockchain empowered secure video sharing with access control for vehicular edge computing,” *IEEE Transactions on Intelligent Transportation Systems*, 2023.

D. B. R. Ugli, J. Kim, A. F. Mohammed, and J. Lee, “Cognitive video surveillance management in hierarchical edge computing system with long short-term memory model,” *Sensors*, vol. 23, no. 5, p. 2869, 2023.

Y. Y. Seunghan Lee, Saurabh Jain, “Design and development of a dddams-based border surveillance system via uvs and hybrid simulations,” *Expert Systems with Applications,*, vol. 128, pp. 109–123,, 2019.

M. L. Laouira, A. Abdelli, J. B. Othman, and H. Kim, “An efficient wsn based solution for border surveillance,” *IEEE Transactions on Sustainable Computing*, vol. 6, no. 1, pp. 54–65, 2021.

J. Amutha, J. Nagar, and S. Sharma, “A distributed border surveillance (dbs) system for rectangular and circular region of interest with wireless sensor networks in shadowed environments,” *Wireless Personal Communications*, vol. 117, pp. 2135–2155, 2021.

S. Richhariya, K. Wanaskar, S. Shrivastava, and J. Gao, “Surveillance drone cloud and intelligence service,” in *2023 11th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud).* IEEE, 2023, pp. 1–10.

A. Renga, A. Gigantino, and M. D. Graziano, “Multi-platform image synthesis for distributed synthetic aperture radar in long baseline bistatic configurations,” *IEEE Transactions on Aerospace and Electronic Systems*, 2023.

J. Deng, H. Bi, Y. Yin, X. Lu, and W. Liang, “Sparse sar image based automatic target recognition by yolo network," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*. IEEE, 2021, pp. 3597–3600.

J. Zhang, Y. Rao *et al.*, "A target recognition method based on multiview infrared images," *Scientific Programming*, vol. 2022, 2022.

I. Stančić, A. Kuzmanić Skelin, J. Music, and M. Cecić, "The development of a cost-effective imaging device based on thermographic technology," *Sensors*, vol. 23, no. 10, p. 4582, 2023.

M. A. Altaf, J. Ahn, D. Khan, and M. Y. Kim, "Usage of ir sensors in the hvac systems, vehicle and manufacturing industries: a review," *IEEE Sensors Journal*, vol. 22, no. 10, pp. 9164–9176, 2022.

M. Vadidar, A. Kariminezhad, C. Mayr, L. Kloeker, and L. Eckstein, "Robust environment perception for automated driving: A unified learning pipeline for visual-infrared object detection," in *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022, pp. 367–374.

D. Poster, S. Hu, and N. M. Nasrabadi, "Long-range thermal target detection in data-limited settings using restricted receptive fields," *Sensors*, vol. 23, no. 18, p. 7806, 2023.

N. Baili, M. Moalla, H. Frigui, and A. D. Karem, "Multistage approach for automatic target detection and recognition in infrared imagery using deep learning," *Journal of Applied Remote Sensing*, vol. 16, no. 4, pp. 048 505–048 505, 2022.

M. A. Dhaygude and S. Kinariwala, "A literature survey on content-based information retrieval," *J. Comput. Technol.(JCT)*, vol. 11, pp. 1–6, 2022.

I. M. Hameed, S. H. Abdulhussain, and B. M. Mahmmod, "Content-based image retrieval: A review of recent trends," *Cogent Engineering*, vol. 8, no. 1, p. 1927469, 2021.

J. C. Briones, B. Flores, and R. Cruz-Cano, "Multi-mode radar target detection and recognition using neural networks," *International Journal of Advanced Robotic Systems*, vol. 9, no. 5, p. 177, 2012.

A. Akula, V. Kaur, N. Guleria, R. Ghosh, and S. Kumar, "Real-time thermal infrared moving target detection and recognition using deep learned features," in *Automatic Target Recognition XXX*, vol. 11394. SPIE, 2020, pp. 239–247.

R. Zhang, L. Xu, Z. Yu, Y. Shi, C. Mu, and M. Xu, "Deep-irtarget: An automatic target detector in infrared imagery using dual-domain feature extraction and allocation," *IEEE Transactions on Multimedia*, vol. 24, pp. 1735–1749, 2021.

H.-W. Chen and R. Kapadia, "Improving automatic target recognition (atr) performance with electro optics (eo) and infrared (ir) sensor fusion," in *2022 IEEE Aerospace Conference (AERO)*. IEEE, 2022, pp. 1–16.

Y. Khare, T. V. Chandran, A. Ramesh, K. Akhil, and L. S. Nair, "Automatic target detection in cluttered infrared imagery using background modeling approach," in *2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)*. IEEE, 2022, pp. 1615–1619.

A. Miao and H. Xie, “Infrared target classification based on single shot multibox detector,” in *2022 International Conference on Image Processing and Media Computing (ICIPMC)*. IEEE, 2022, pp. 85–91.

E. R. Adams, A. C. Depoian II, A. G. Kurz, C. P. Bailey, and P. Guturu, “Automatic target detection utilizing an edge ir vision transformer (eir-vit),” in *Automatic Target Recognition XXXIII*, vol. 12521. SPIE, 2023, pp. 192–197.

K. L. Hom, B. A. Millikan, and R. Z. Shilling, “Enhanced target recognition using quadratic correlation filter networks,” in *Automatic Target Recognition XXXIII*, vol. 12521. SPIE, 2023, pp. 170–179.

J. Wu and X. Zhang, “Vehicle target recognition method based on visible and infrared image fusion using bayesian inference,” *Applied Sciences*, vol. 13, no. 14, p. 8334, 2023.

J. Fan, J. Wei, H. Huang, D. Zhang, and C. Chen, “Irsdt: A framework for infrared small target tracking with enhanced detection,” *Sensors*, vol. 23, no. 9, p. 4240, 2023.

X. Zhao, Y. Xia, M. Xu, W. Zhang, J. Niu, and Z. Zhang, “An infrared small vehicle target detection method based on deep learning,” in *Third International Seminar on Artificial Intelligence, Networking, and Information Technology (AINIT 2022)*, vol. 12587. SPIE, 2023, pp. 480–487.

S. Suganyadevi, V. Pavatharani, K. Koushick, R. Manimaran, A. M. Sameer, and K. Murugan, “Multisalient object detection and recognition using deep learning techniques,” in *2023 International Conference on Applied Intelligence and Sustainable Computing (ICAISC)*. IEEE, 2023, pp. 1–7.

Y. Liu, Z. Yu, D. Zong, and L. Zhu, “Attention to task-aligned object detection for end-edge-cloud video surveillance,” *IEEE Internet of Things Journal*, 2023.

P. Townend, A. P. Martí, I. De La Iglesia, N. Matskanis, T. O. Timoudas, T. Hallmann, A. Lalaguna, K. Swat, F. Renzi, D. Bocheński *et al.*, “Cognit: Challenges and vision for a serverless and multi-provider cognitive cloud-edge continuum,” in *2023 IEEE International Conference on Edge Computing and Communications (EDGE)*. IEEE, 2023, pp. 12–22.

M. Rohith, A. Sunil *et al.*, “Comparative analysis of edge computing and edge devices: key technology in iot and computer vision applications,” in *2021 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*. IEEE, 2021, pp. 722–727.

A. C. Cob-Parro, C. Losada-Gutiérrez, M. Marrón-Romera, A. Gardel-Vicente, and I. Bravo-Muñoz, “Smart video surveillance system based on edge computing,” *Sensors*, vol. 21, no. 9, p. 2958, 2021.

Y. Zhao, Y. Yin, and G. Gui, “Lightweight deep learning based intelligent edge surveillance techniques,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1146–1154, 2020.

X. Luo, K. Jia, P. Liu, D. Xiong, and X. Tian, “Improved three-frame-difference algorithm for infrared moving target," in *2020 IEEE 5th International Conference on Image, Vision and Computing (ICIVC)*. IEEE, 2020, pp. 108–112.

Z. Zhang, H. Zhang, and Z. Zhang, "Using three-frame difference algorithm to detect moving objects," in *Cyber Security Intelligence and Analytics*. Springer, 2020, pp. 923–928.

O. Boufares, M. Boussif, and N. Aloui, "Moving object detection system based on the modified temporal difference and otsu algorithm," in *2021 18th International Multi-Conference on Systems, Signals & Devices (SSD)*. IEEE, 2021, pp. 1378–1382.

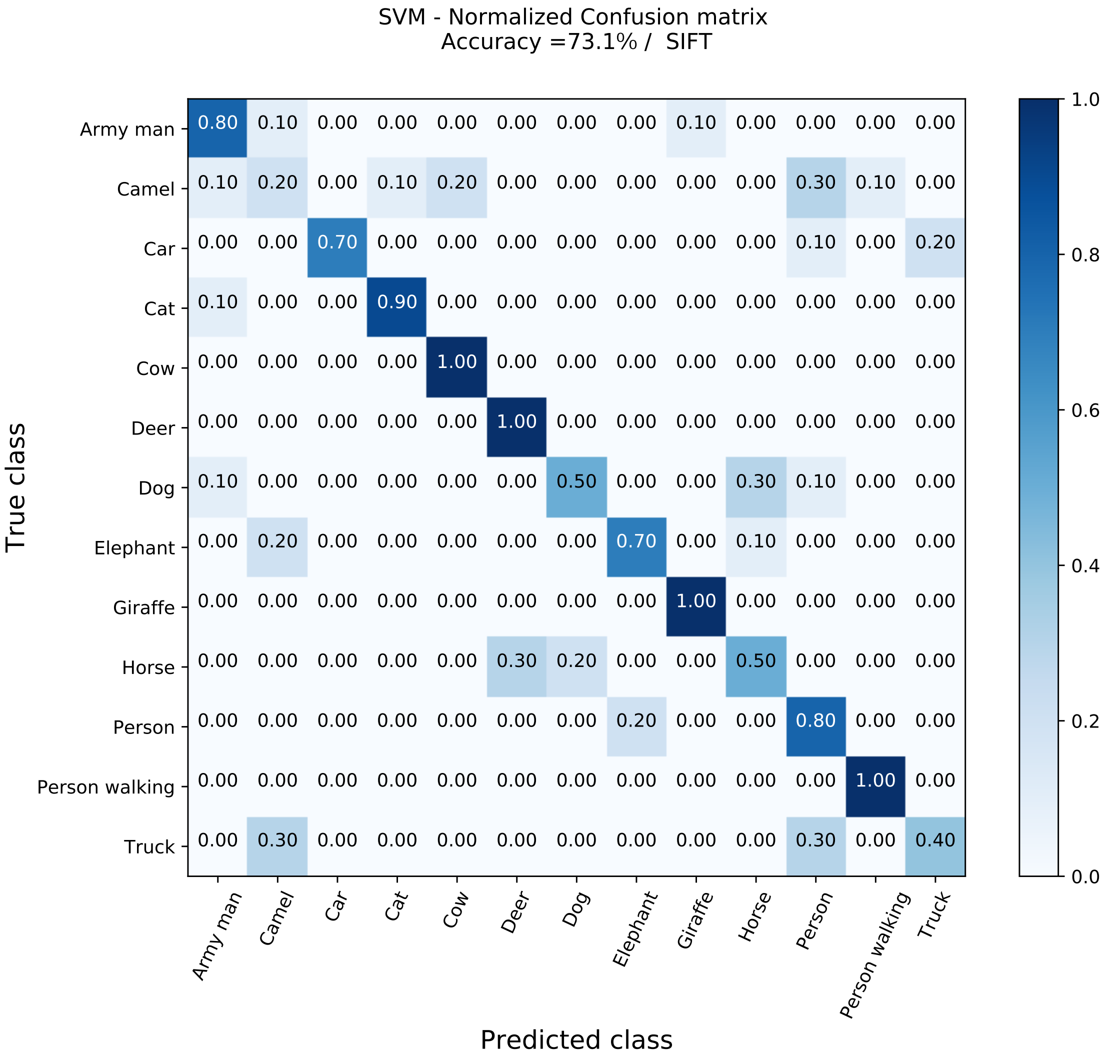
E. Simsek and B. Ozyer, "Selected three frame difference method for moving object detection," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 9, no. 2, pp. 48–54, 2021.

S. Jeannin and M. Bober, "Shape data for the mpeg-7 core experiment ce-shape-1@ online," *URL: http://www. cis. temple. edu/latecki/TestData/mpeg7shapeB. tar. g*, 1999.

J. Wang, H. Jiang, Z. Yuan, M.-M. Cheng, X. Hu, and N. Zheng, "Salient object detection: A discriminative regional feature integration approach," *International Journal of Computer Vision*, vol. 123, no. 2, pp. 251–268, 2017.

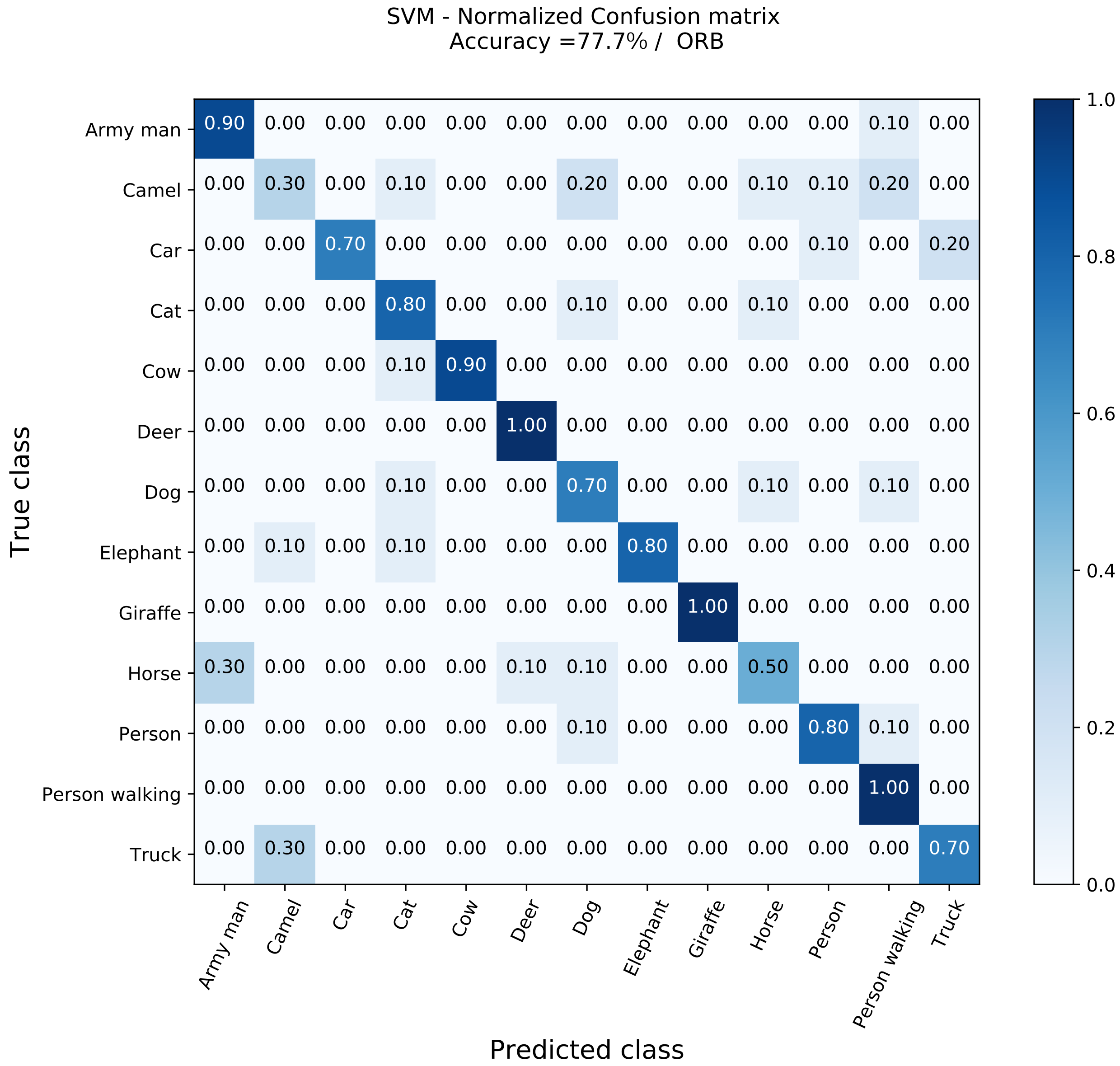
S.-H. Lee, J.-H. Kim, K. P. Choi, J.-Y. Sim, and C.-S. Kim, "Video saliency detection based on spatiotemporal feature learning," in *Image Processing (ICIP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1120–1124.

Appendix I: Confusion Matrices



True class	Army man	Camel	Car	Cat	Cow	Deer	Dog	Elephant	Giraffe	Horse	Person	Person walking	Truck
Army man	0.80	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00
Camel	0.10	0.20	0.00	0.10	0.20	0.00	0.00	0.00	0.00	0.00	0.30	0.10	0.00
Car	0.00	0.00	0.70	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.20
Cat	0.10	0.00	0.00	0.90	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cow	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Deer	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dog	0.10	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.30	0.10	0.00	0.00
Elephant	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.70	0.00	0.10	0.00	0.00	0.00
Giraffe	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Horse	0.00	0.00	0.00	0.00	0.00	0.30	0.20	0.00	0.00	0.50	0.00	0.00	0.00
Person	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.80	0.00	0.00
Person walking	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Truck	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.40

Figure 5.9: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.(SIFT)



True class \ Predicted class	Army man	Camel	Car	Cat	Cow	Deer	Dog	Elephant	Giraffe	Horse	Person	Person walking	Truck
Army man	0.90	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00
Camel	0.00	0.30	0.00	0.10	0.00	0.00	0.20	0.00	0.00	0.10	0.10	0.20	0.00
Car	0.00	0.00	0.70	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.20
Cat	0.00	0.00	0.00	0.80	0.00	0.00	0.10	0.00	0.00	0.10	0.00	0.00	0.00
Cow	0.00	0.00	0.00	0.10	0.90	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Deer	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dog	0.00	0.00	0.00	0.10	0.00	0.00	0.70	0.00	0.00	0.10	0.00	0.10	0.00
Elephant	0.00	0.10	0.00	0.10	0.00	0.00	0.00	0.80	0.00	0.00	0.00	0.00	0.00
Giraffe	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Horse	0.30	0.00	0.00	0.00	0.00	0.10	0.10	0.00	0.00	0.50	0.00	0.00	0.00
Person	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.80	0.10	0.00
Person walking	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Truck	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.70

Figure 5.10: Confusion matrices applying the bounding box prediction method to dt_13_eq_indr(ORB).

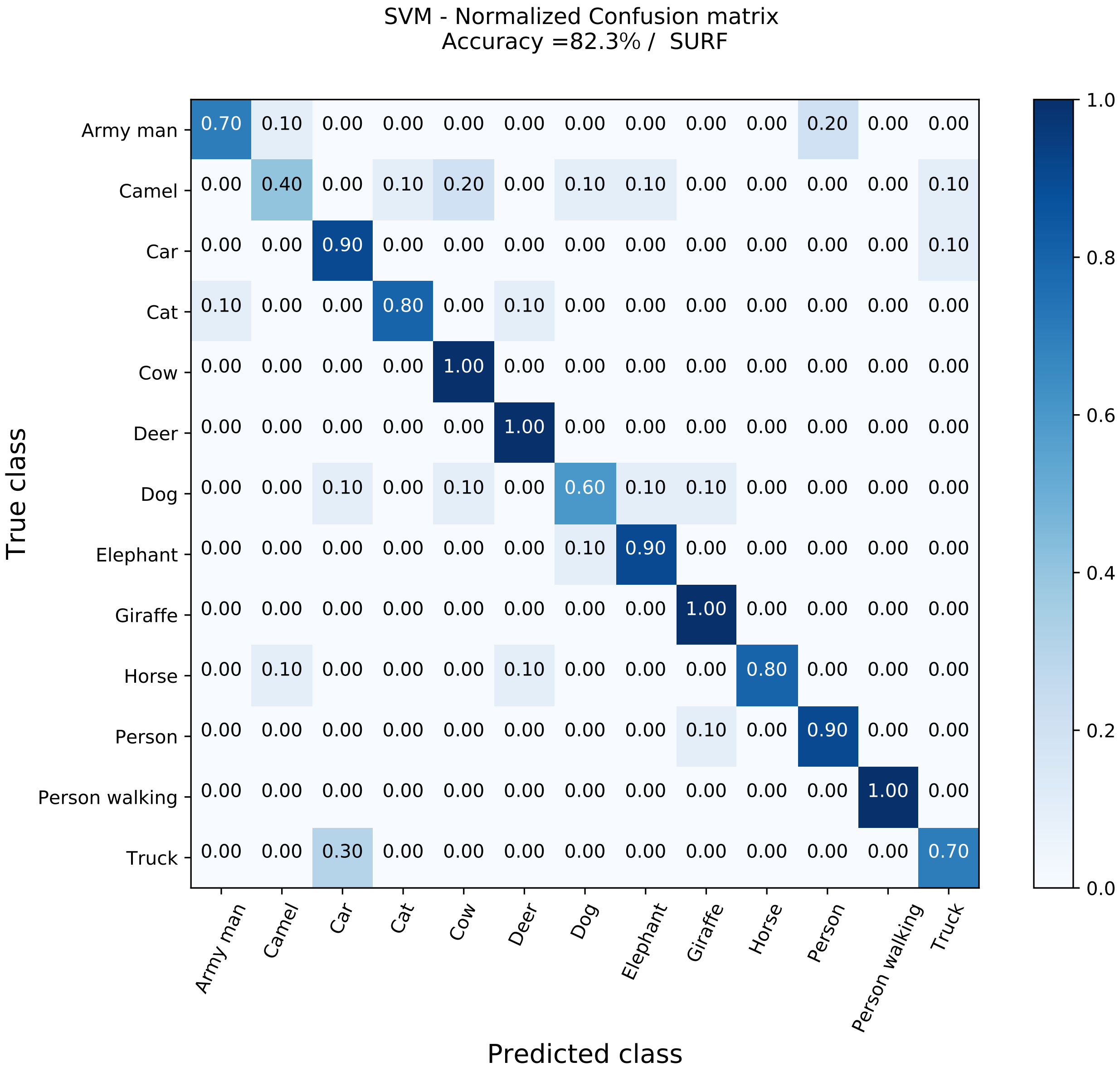


Figure 5.11: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind(SURF).

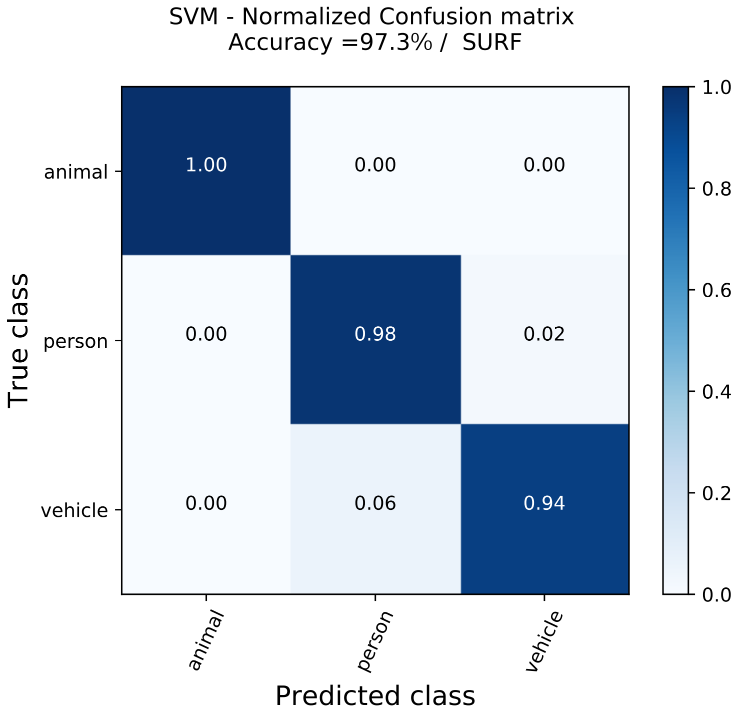


Figure 5.12: Confusion matrices applying the bounding box prediction method with SVM for SURF descriptors to dt_3_eq_ind.

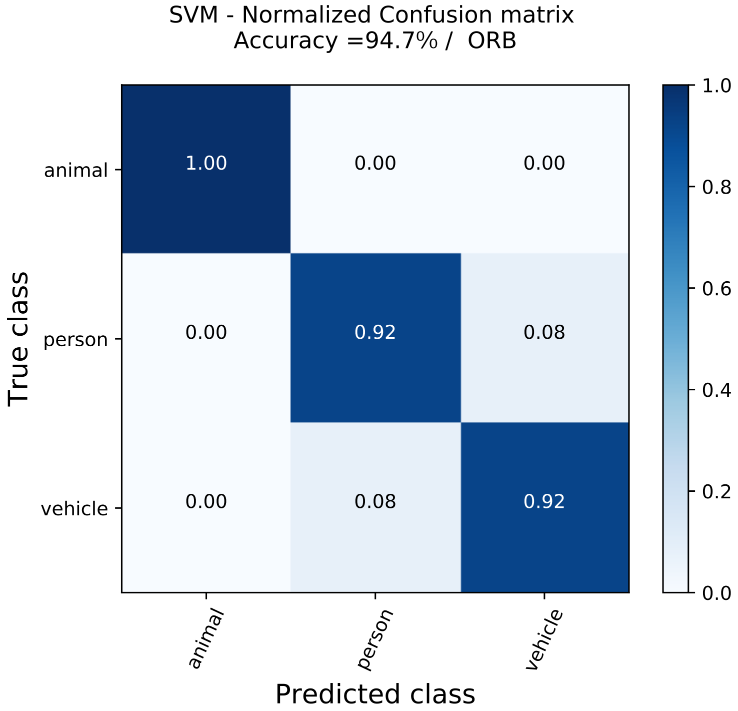


Figure 5.13: Confusion matrices applying the bounding box prediction method with SVM for ORB descriptors to dt_3_eq_ind.

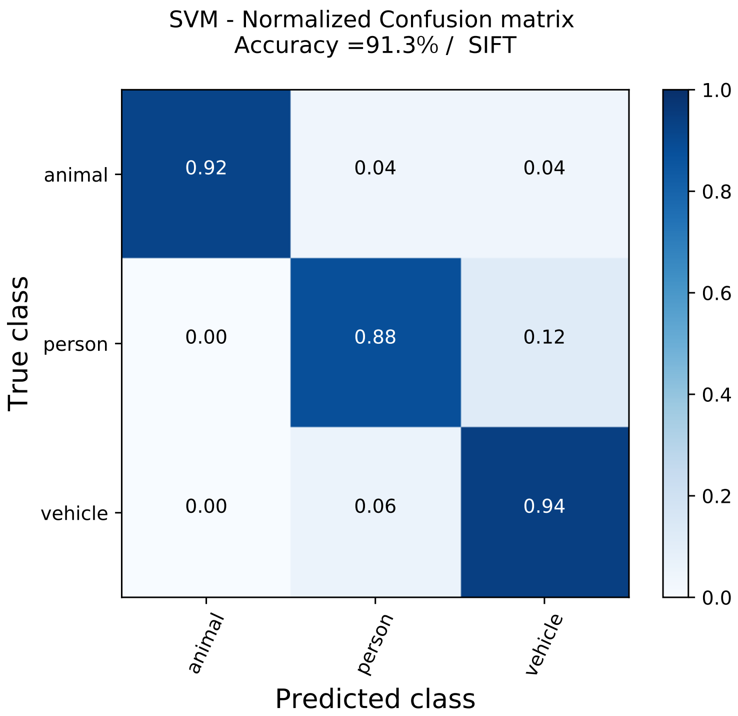


Figure 5.14: Confusion matrices applying the bounding box prediction method with SVM for SIFT descriptors to dt_3_eq_ind.

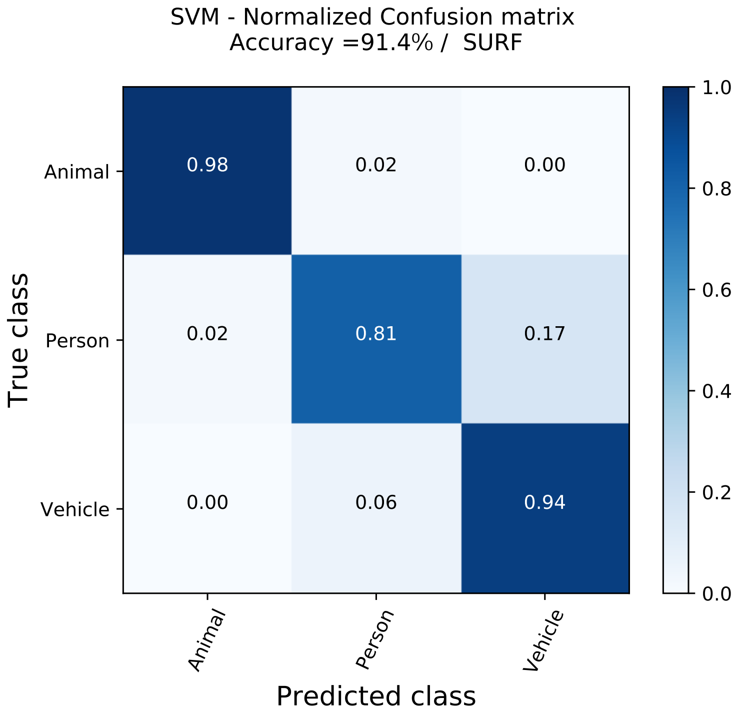


Figure 5.15: Confusion matrices applying the bounding box prediction method with SVM for SURF descriptors to dt_3_diff_ind.

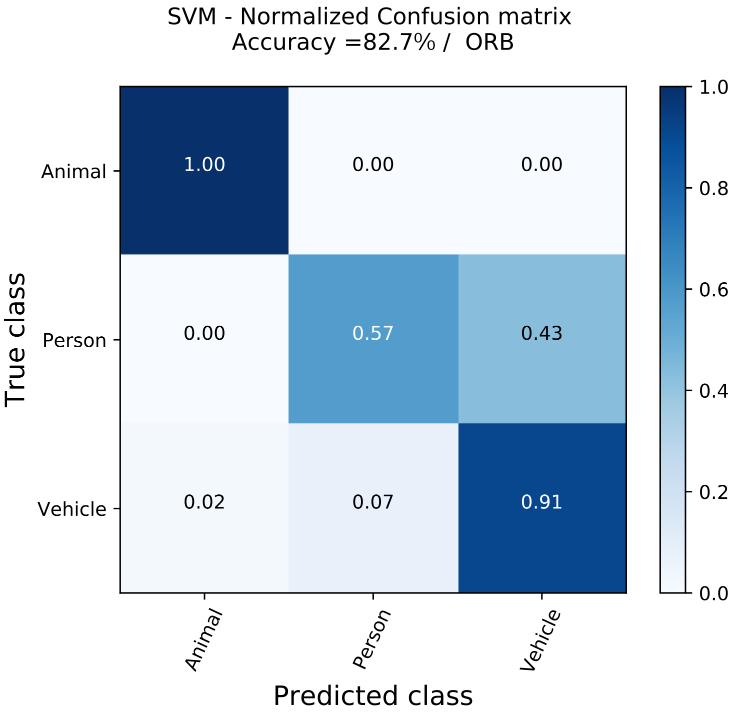


Figure 5.16: Confusion matrices applying the bounding box prediction method with SVM for ORB descriptors to dt_3_diff_ind.

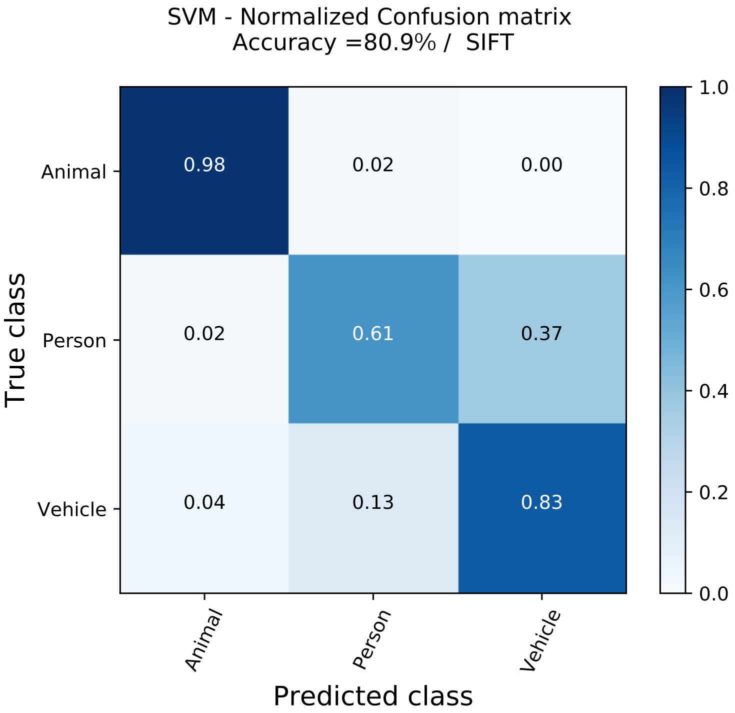
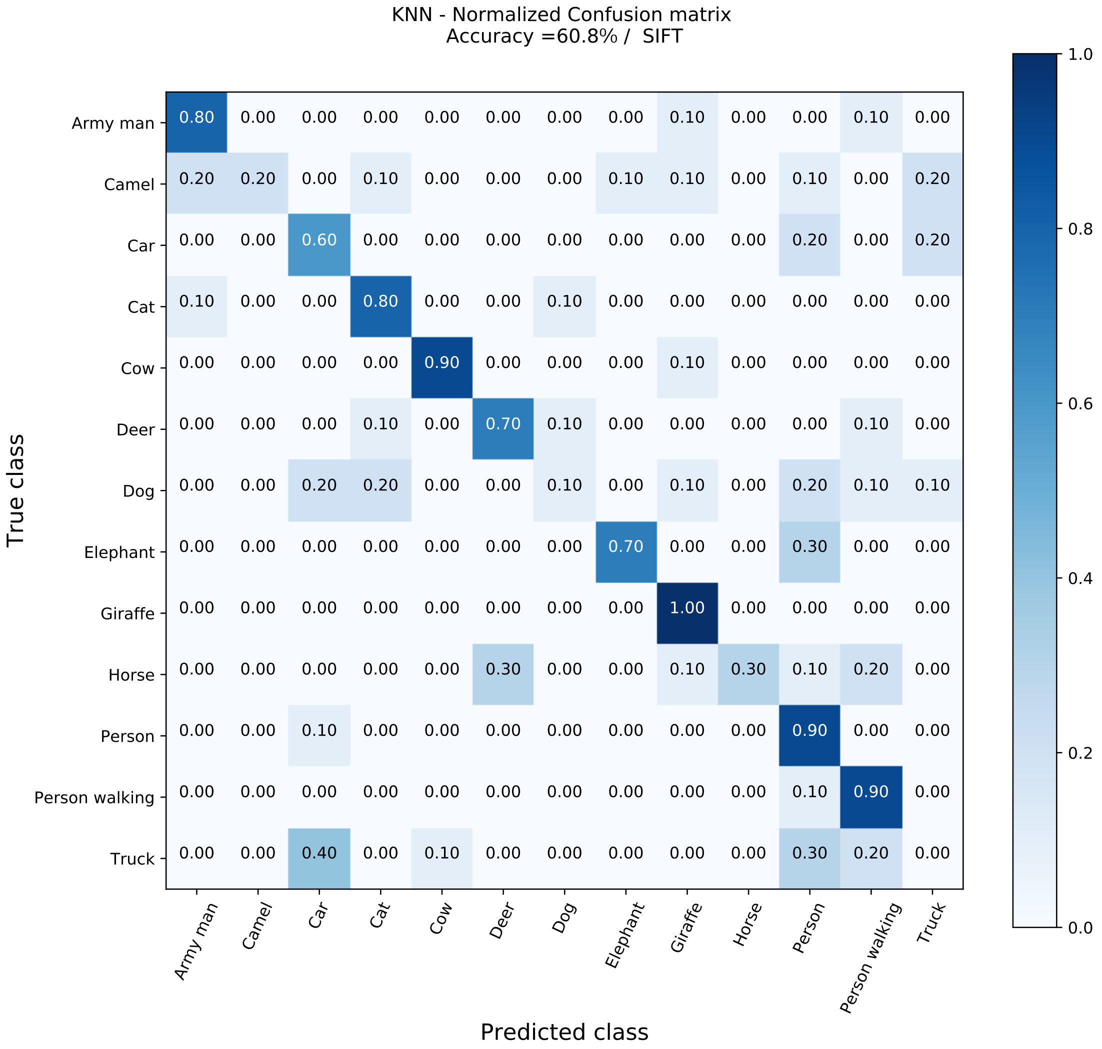


Figure 5.17: Confusion matrices applying the bounding box prediction method with SVM for SIFT descriptors to dt_3_diff_ind.



KNN - Normalized Confusion matrix
Accuracy =60.8% / SIFT

True class \ Predicted class	Army man	Camel	Car	Cat	Cow	Deer	Dog	Elephant	Giraffe	Horse	Person	Person walking	Truck
Army man	0.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.10	0.00
Camel	0.20	0.20	0.00	0.10	0.00	0.00	0.00	0.10	0.10	0.00	0.10	0.00	0.20
Car	0.00	0.00	0.60	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.20
Cat	0.10	0.00	0.00	0.80	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00
Cow	0.00	0.00	0.00	0.00	0.90	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00
Deer	0.00	0.00	0.00	0.10	0.00	0.70	0.10	0.00	0.00	0.00	0.00	0.10	0.00
Dog	0.00	0.00	0.20	0.20	0.00	0.00	0.10	0.00	0.10	0.00	0.20	0.10	0.10
Elephant	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.70	0.00	0.00	0.30	0.00	0.00
Giraffe	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Horse	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.10	0.30	0.10	0.20	0.00
Person	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.90	0.00	0.00
Person walking	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.90	0.00
Truck	0.00	0.00	0.40	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.30	0.20	0.00

Figure 5.18: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.

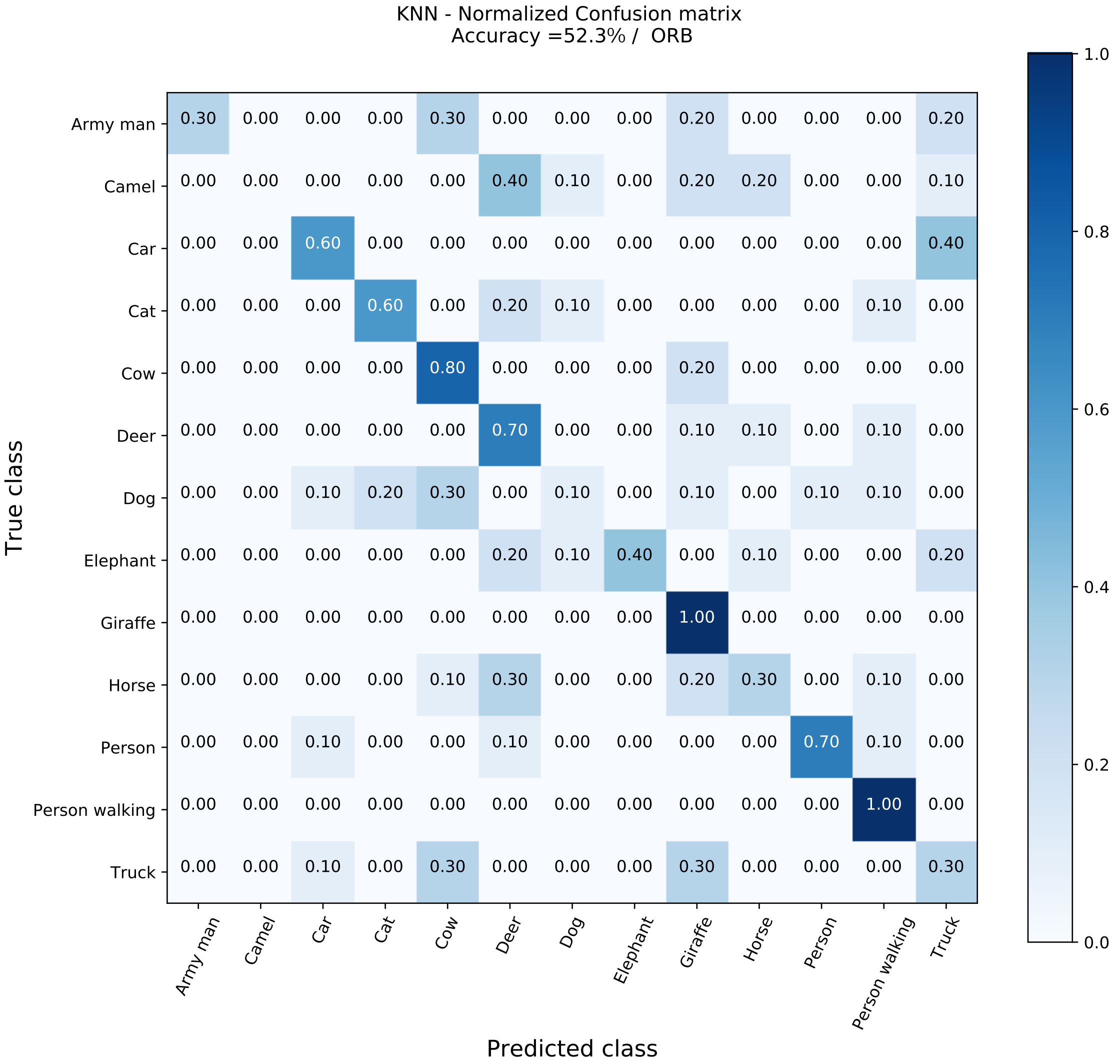


Figure 5.19: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.

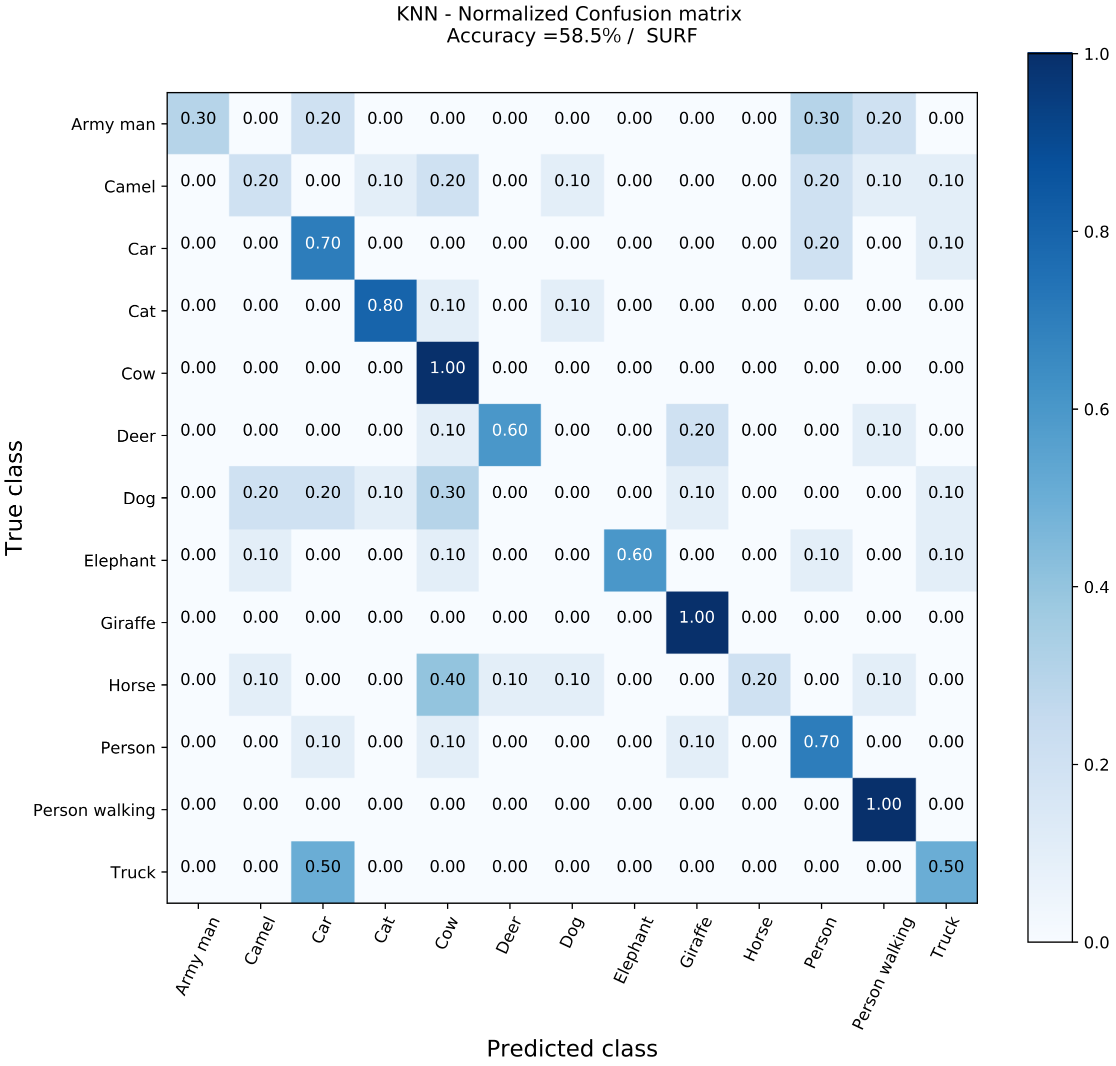


Figure 5.20: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.

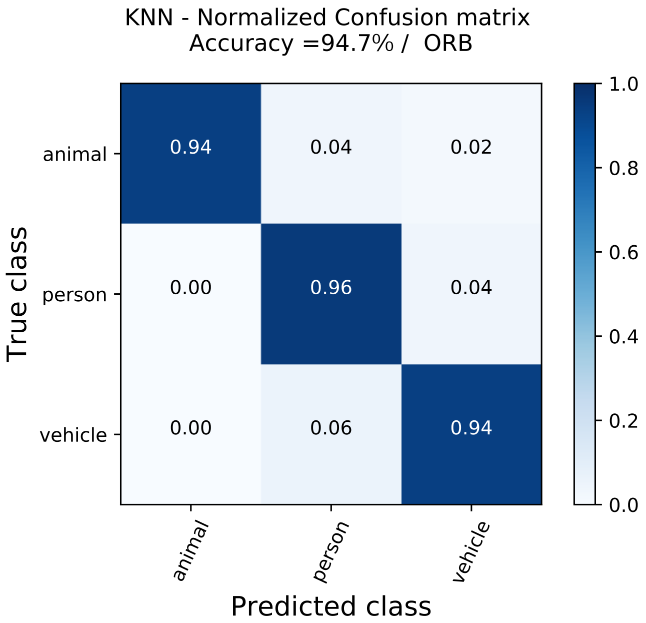


Figure 5.21: Confusion matrices applying the bounding box prediction method with KNN for ORB descriptors to dt_3_eq_ind.

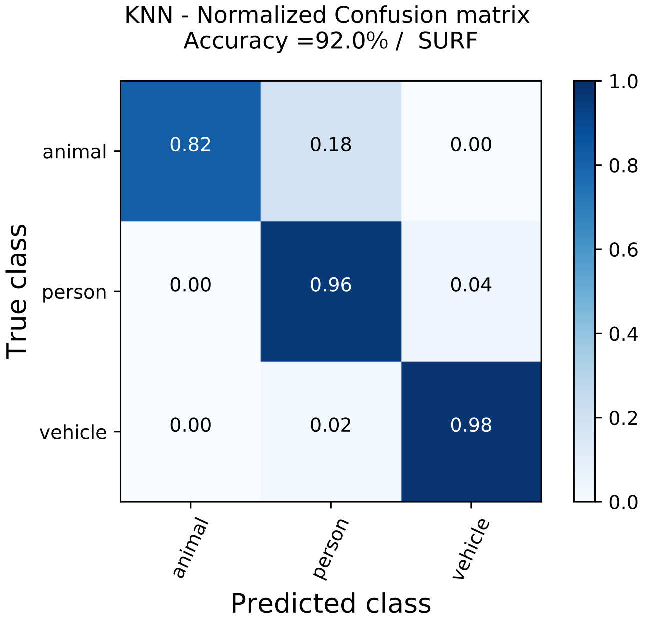


Figure 5.22: Confusion matrices applying the bounding box prediction method with KNN for SURF descriptors to dt_3_eq_ind.

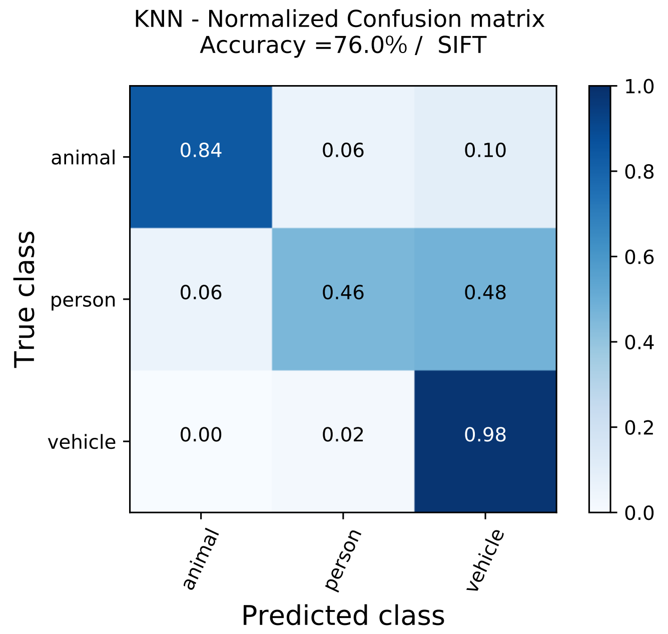


Figure 5.23: Confusion matrices applying the bounding box prediction method with KNN for SIFT descriptors to dt_3_eq_ind.

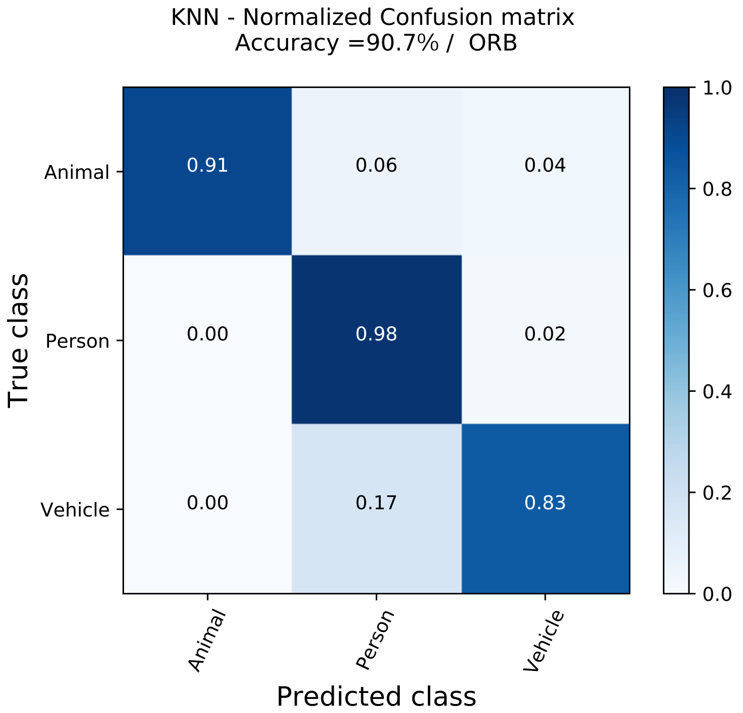


Figure 5.24: Confusion matrices applying the bounding box prediction method with KNN for ORB descriptors to dt_3_eq_ind.

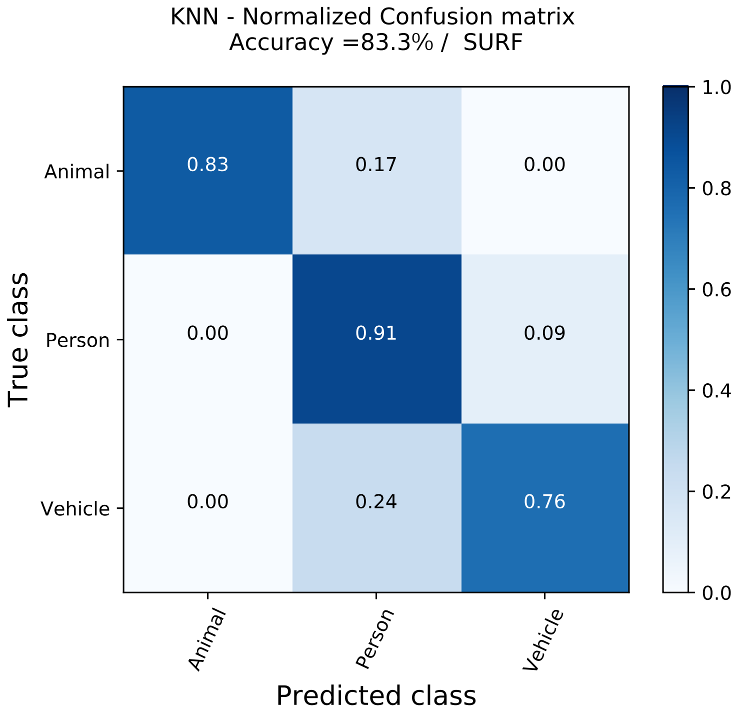


Figure 5.25: Confusion matrices applying the bounding box prediction method with KNN for SIFT descriptors to dt_3_eq_ind.

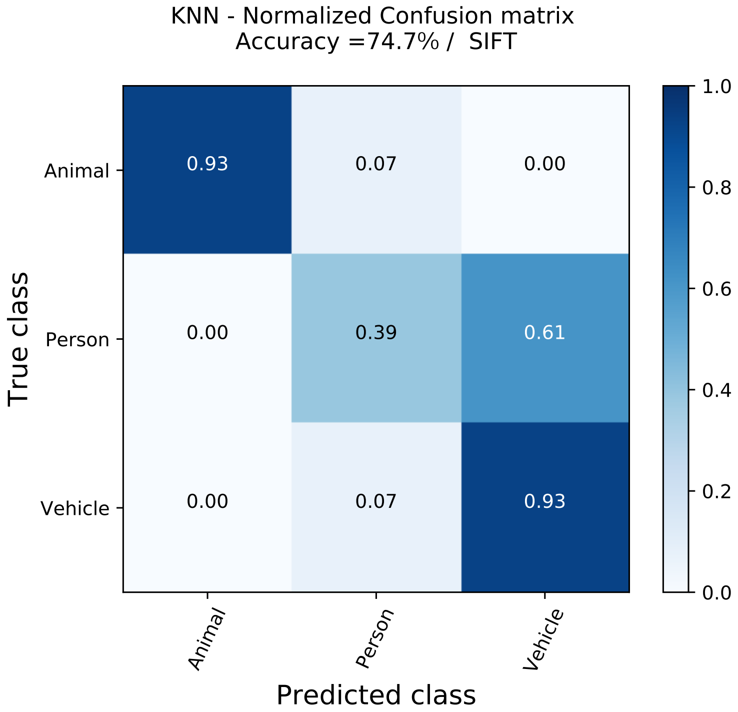
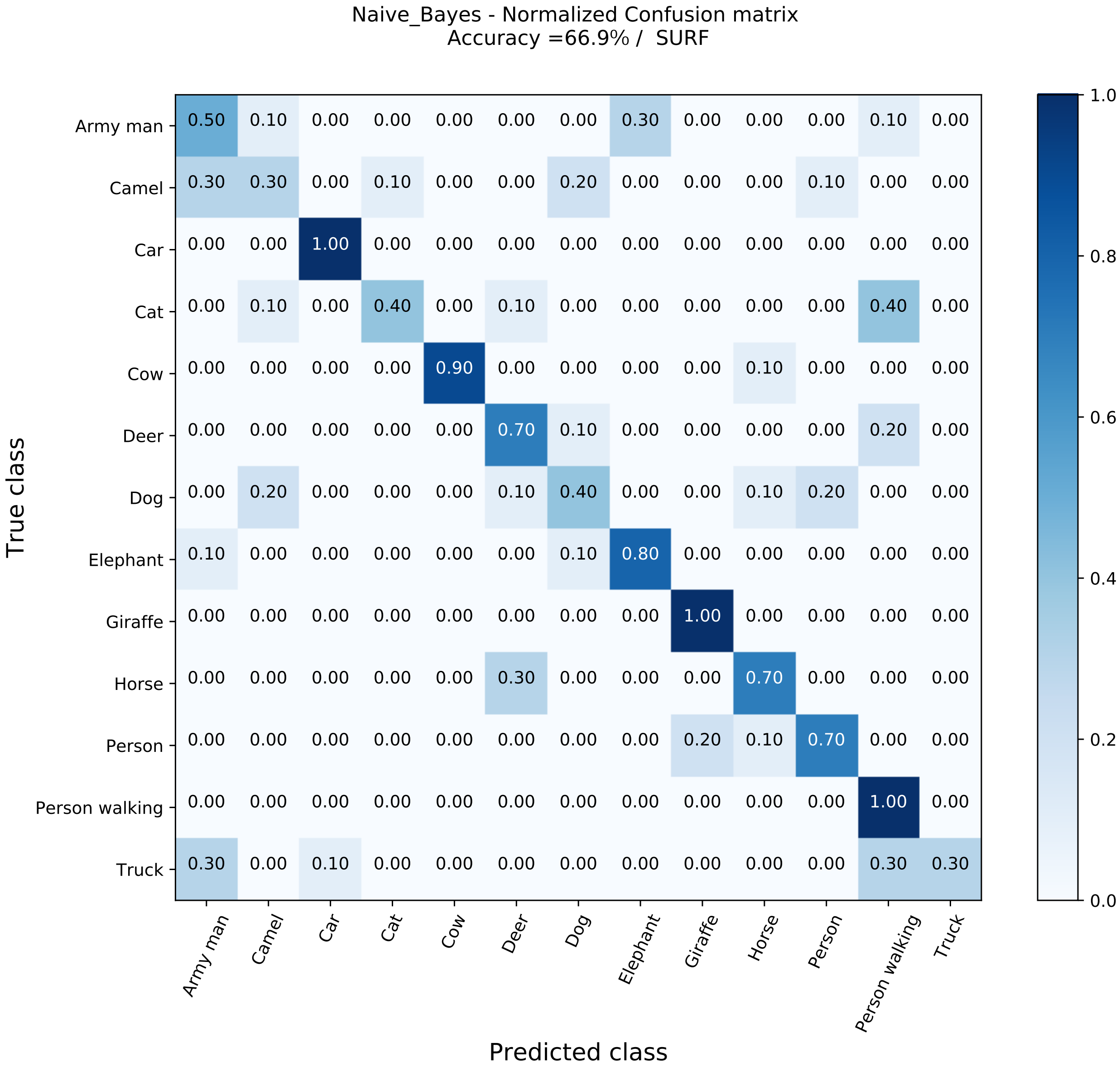


Figure 5.26: Confusion matrices applying the bounding box prediction method with KNN for SIFT descriptors to dt_3_eq_ind.



	Army man	Camel	Car	Cat	Cow	Deer	Dog	Elephant	Giraffe	Horse	Person	Person walking	Truck
Army man	0.50	0.10	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.10	0.00
Camel	0.30	0.30	0.00	0.10	0.00	0.00	0.20	0.00	0.00	0.00	0.10	0.00	0.00
Car	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cat	0.00	0.10	0.00	0.40	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.40	0.00
Cow	0.00	0.00	0.00	0.00	0.90	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00
Deer	0.00	0.00	0.00	0.00	0.00	0.70	0.10	0.00	0.00	0.00	0.00	0.20	0.00
Dog	0.00	0.20	0.00	0.00	0.00	0.10	0.40	0.00	0.00	0.10	0.20	0.00	0.00
Elephant	0.10	0.00	0.00	0.00	0.00	0.00	0.10	0.80	0.00	0.00	0.00	0.00	0.00
Giraffe	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Horse	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.70	0.00	0.00	0.00
Person	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.10	0.70	0.00	0.00
Person walking	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Truck	0.30	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.30

Figure 5.27: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.

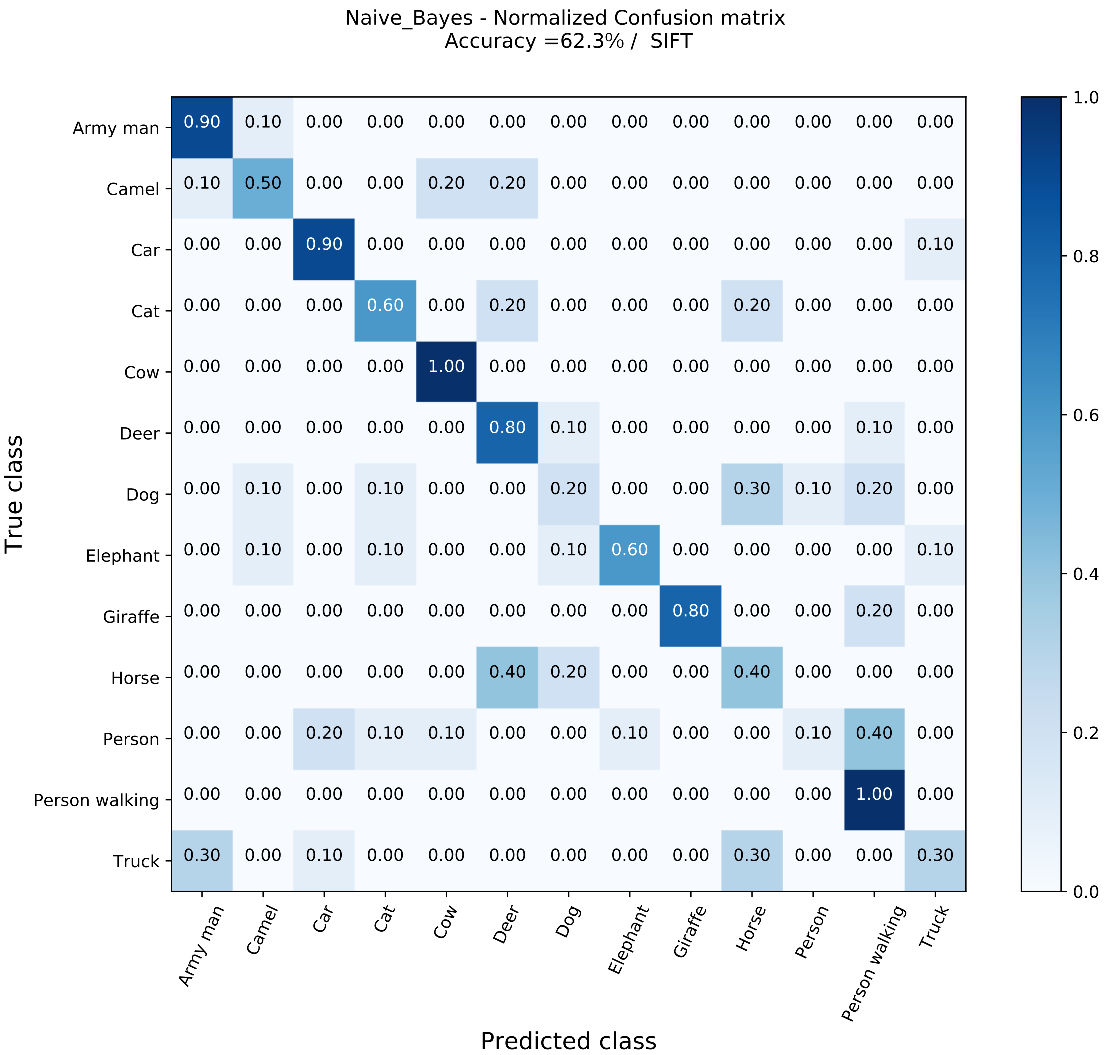
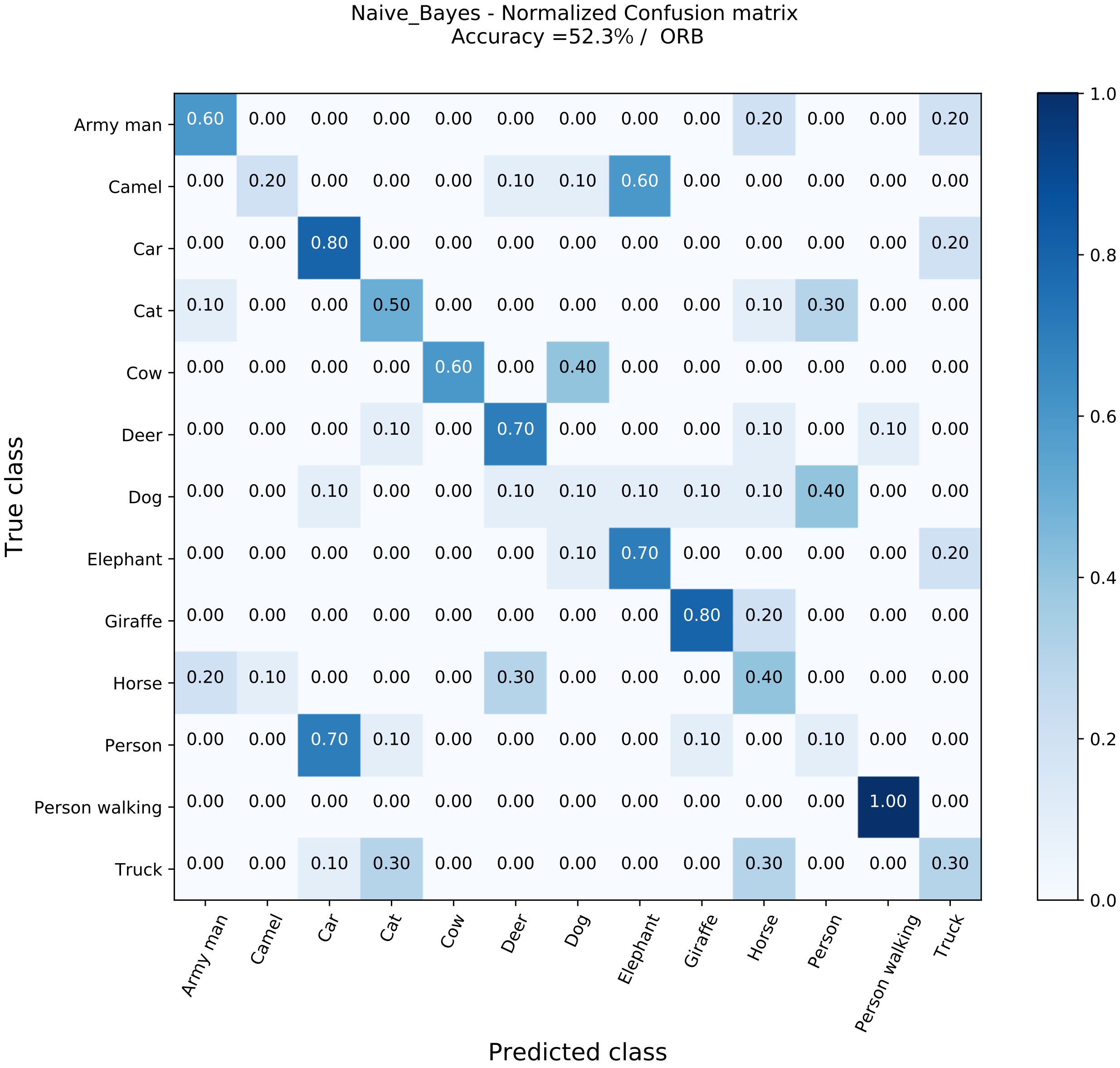


Figure 5.28: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.



	Army man	Camel	Car	Cat	Cow	Deer	Dog	Elephant	Giraffe	Horse	Person	Person walking	Truck
Army man	0.60	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.20
Camel	0.00	0.20	0.00	0.00	0.00	0.10	0.10	0.60	0.00	0.00	0.00	0.00	0.00
Car	0.00	0.00	0.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20
Cat	0.10	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.10	0.30	0.00	0.00
Cow	0.00	0.00	0.00	0.00	0.60	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00
Deer	0.00	0.00	0.00	0.10	0.00	0.70	0.00	0.00	0.00	0.10	0.00	0.10	0.00
Dog	0.00	0.00	0.10	0.00	0.00	0.10	0.10	0.10	0.10	0.10	0.40	0.00	0.00
Elephant	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.70	0.00	0.00	0.00	0.00	0.20
Giraffe	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.80	0.20	0.00	0.00	0.00
Horse	0.20	0.10	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.40	0.00	0.00	0.00
Person	0.00	0.00	0.70	0.10	0.00	0.00	0.00	0.00	0.10	0.00	0.10	0.00	0.00
Person walking	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Truck	0.00	0.00	0.10	0.30	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.30

Figure 5.29: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.

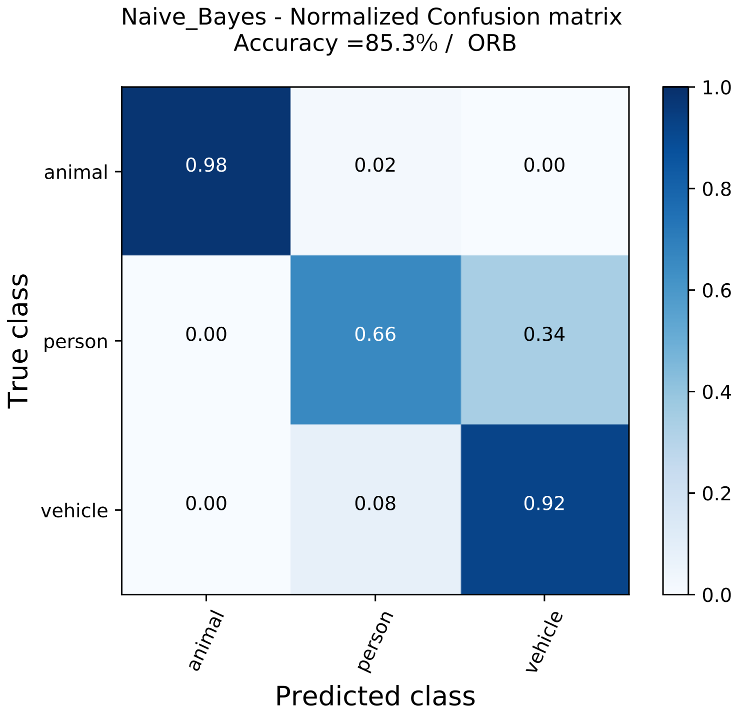


Figure 5.30: Confusion matrices applying the bounding box prediction method with KNN for ORB descriptors to dt_3_eq_ind.

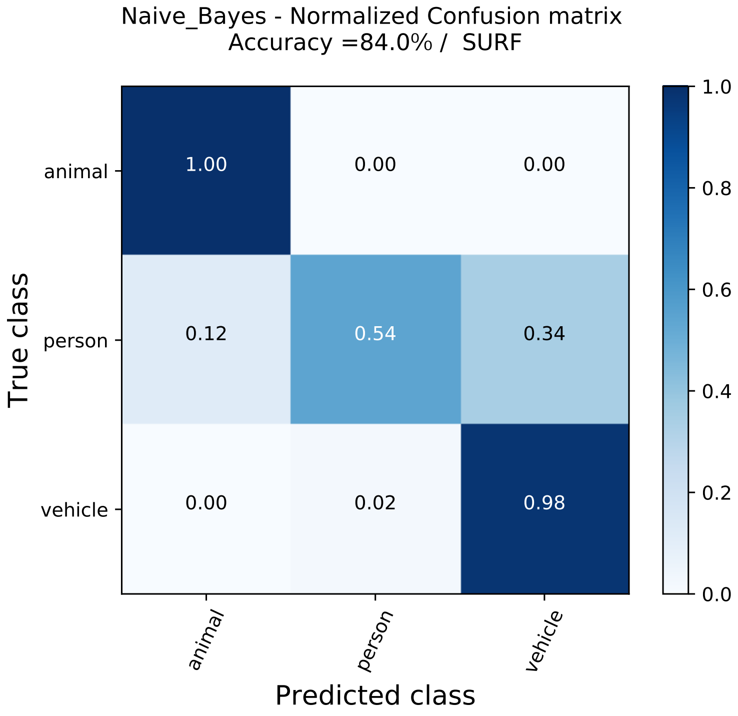


Figure 5.31: Confusion matrices applying the bounding box prediction method with KNN for SURF descriptors to dt_3_eq_ind.

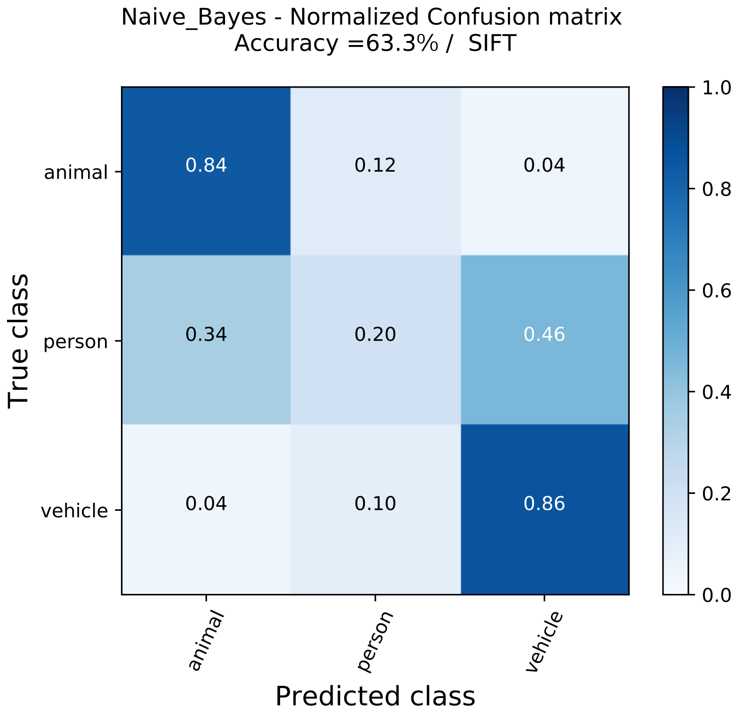


Figure 5.32: Confusion matrices applying the bounding box prediction method with KNN for SIFT descriptors to dt_3_eq_ind.

Figure 5.33: Confusion matrices applying the bounding box prediction method with NB for SURF descriptors to dt_3_eq_diff.

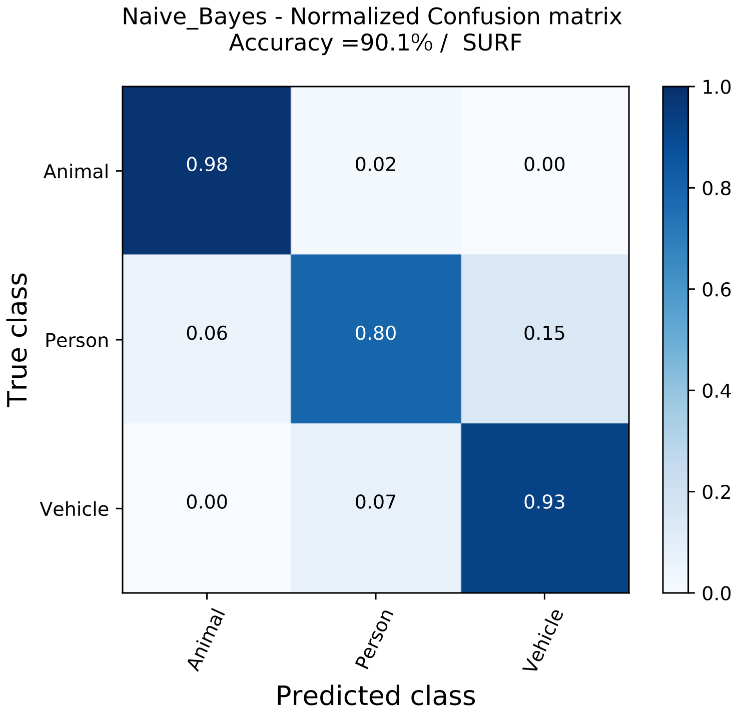


Figure 5.34: Confusion matrices applying the bounding box prediction method with NB for ORB descriptors to dt_3_eq_diff.

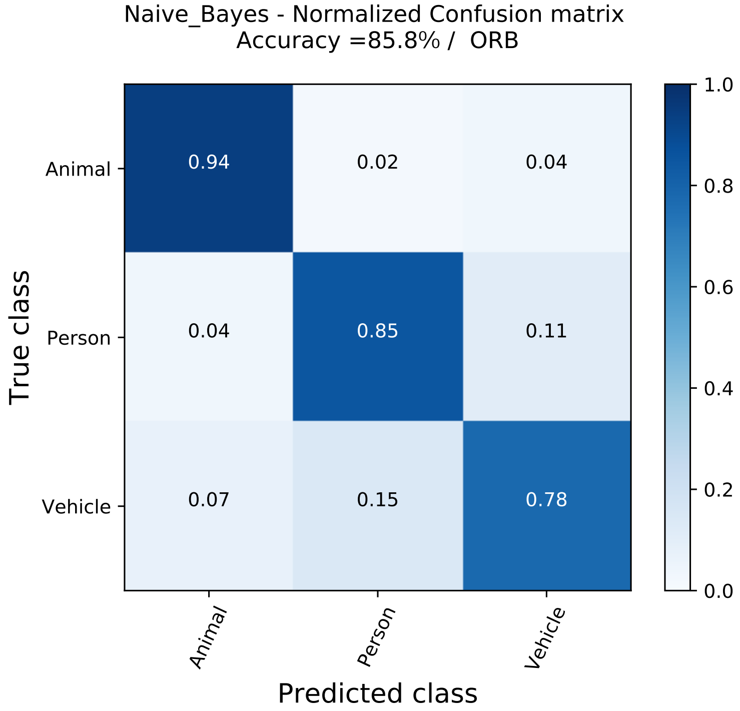
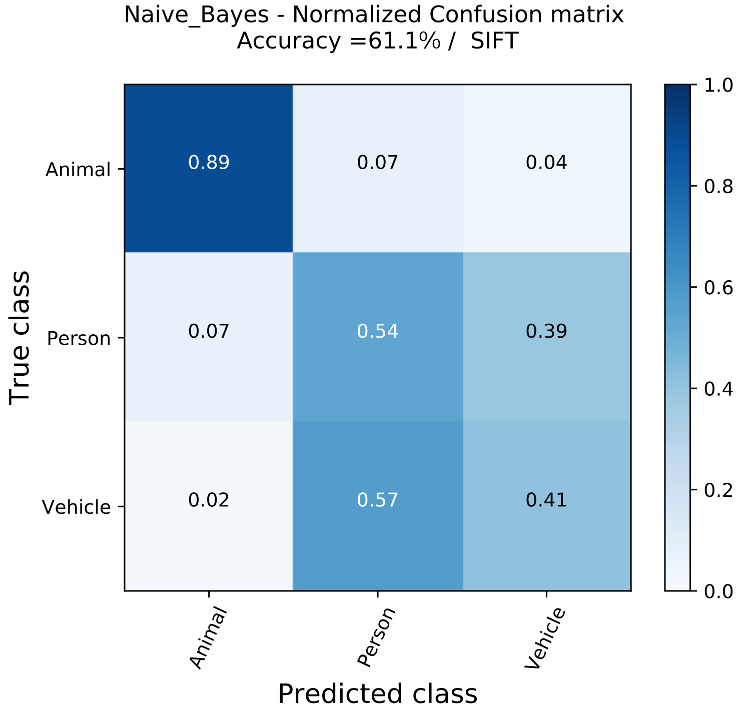
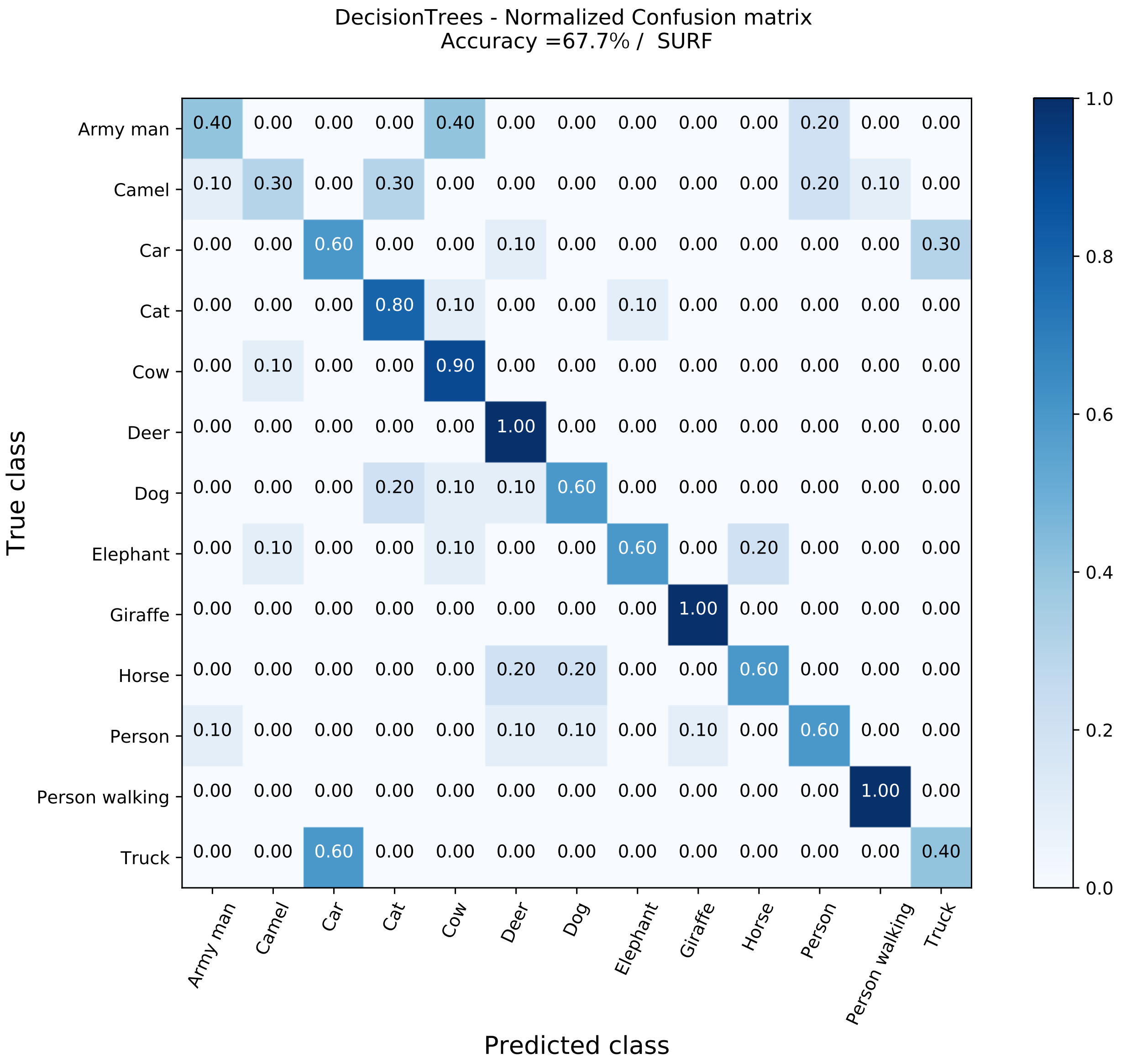


Figure 5.35: Confusion matrices applying the bounding box prediction method with NB for SIFT descriptors to dt_3_eq_diff.





	Army man	Camel	Car	Cat	Cow	Deer	Dog	Elephant	Giraffe	Horse	Person	Person walking	Truck
Army man	0.40	0.00	0.00	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00
Camel	0.10	0.30	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.10	0.00
Car	0.00	0.00	0.60	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.30
Cat	0.00	0.00	0.00	0.80	0.10	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00
Cow	0.00	0.10	0.00	0.00	0.90	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Deer	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dog	0.00	0.00	0.00	0.20	0.10	0.10	0.60	0.00	0.00	0.00	0.00	0.00	0.00
Elephant	0.00	0.10	0.00	0.00	0.10	0.00	0.00	0.60	0.00	0.20	0.00	0.00	0.00
Giraffe	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Horse	0.00	0.00	0.00	0.00	0.00	0.20	0.20	0.00	0.00	0.60	0.00	0.00	0.00
Person	0.10	0.00	0.00	0.00	0.00	0.10	0.10	0.00	0.10	0.00	0.60	0.00	0.00
Person walking	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Truck	0.00	0.00	0.60	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40

Figure 5.36: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.

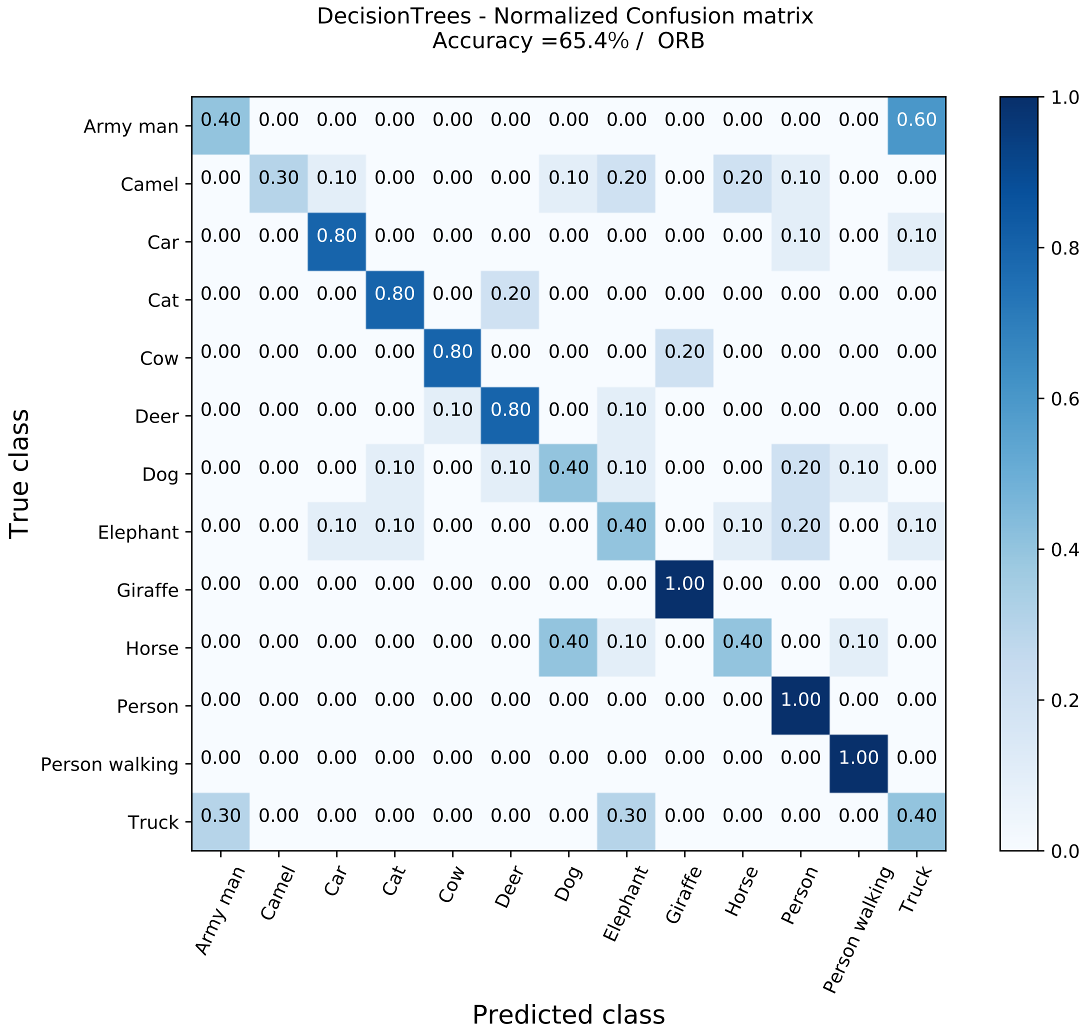


Figure 5.37: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.

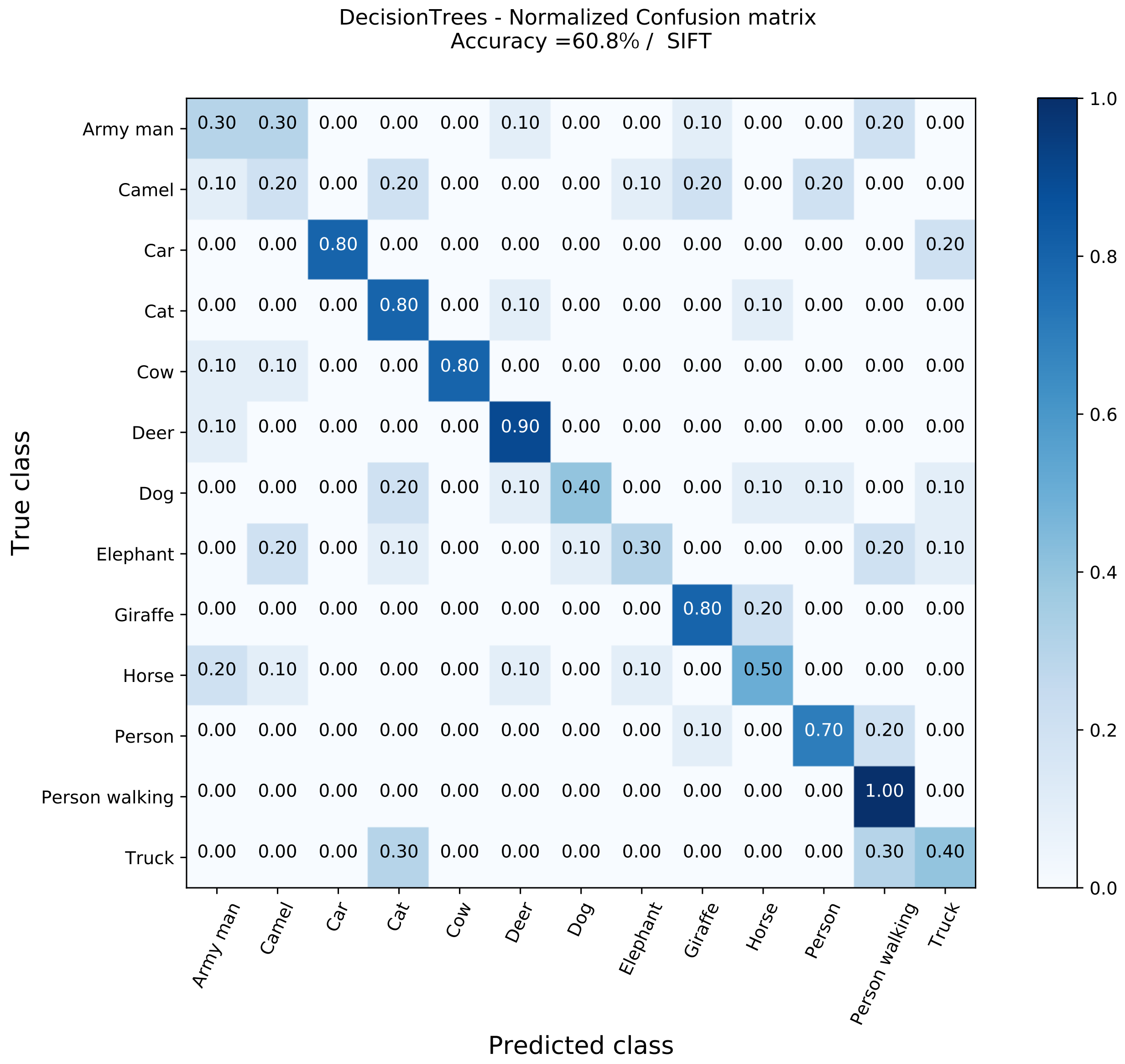


Figure 5.38: Confusion matrices applying the bounding box prediction method to dt_13_eq_ind.

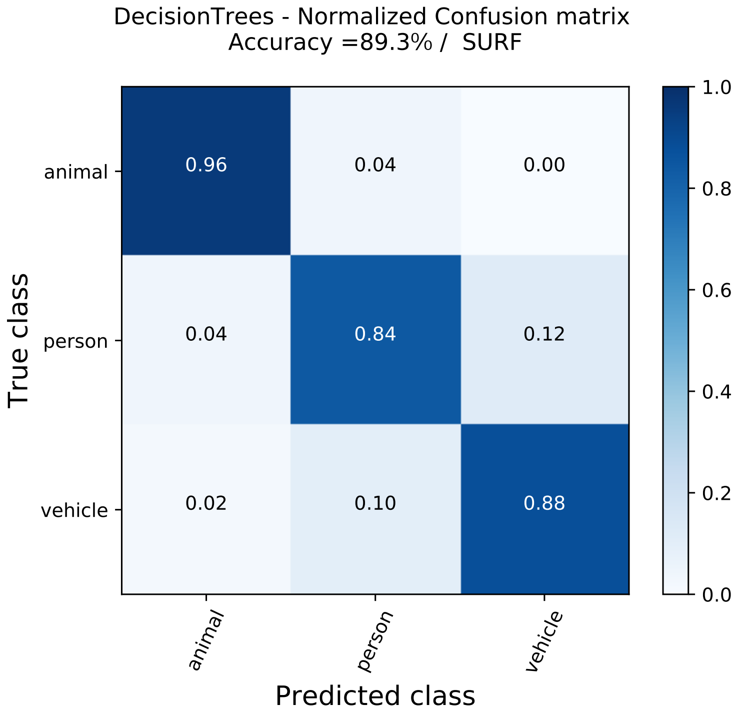


Figure 5.39: Confusion matrices applying the bounding box prediction method with DT for SURF descriptors to dt_3_eq_ind.

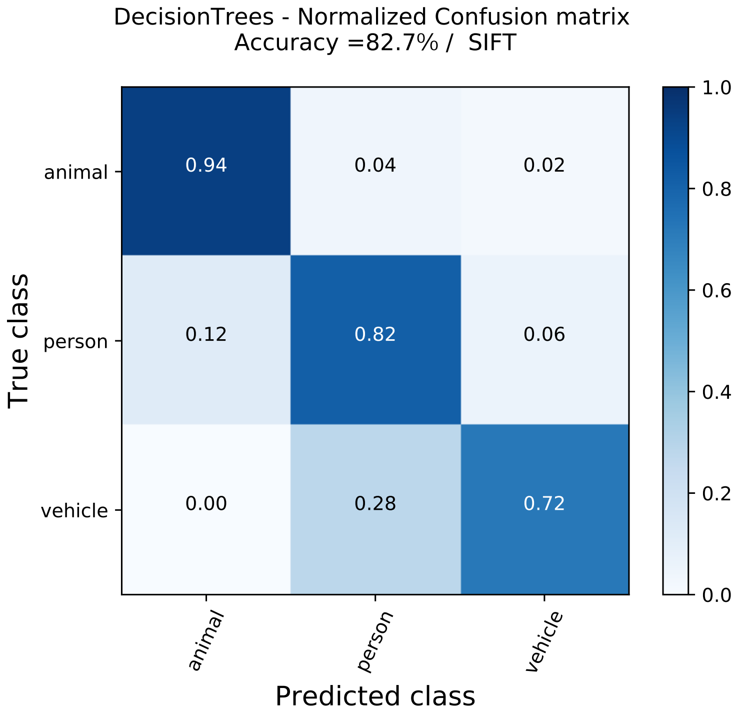


Figure 5.40: Confusion matrices applying the bounding box prediction method with DT for SIFT descriptors to dt_3_eq_ind.

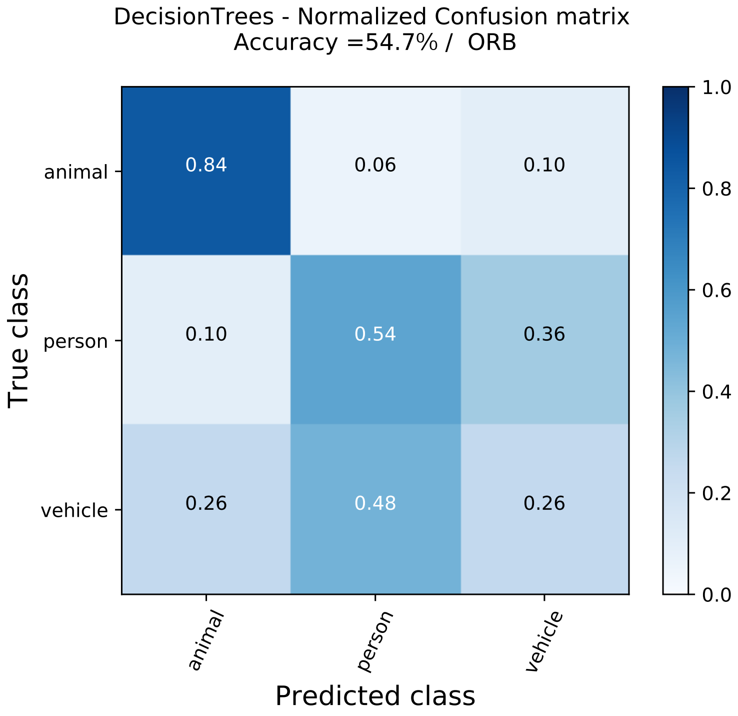


Figure 5.41: Confusion matrices applying the bounding box prediction method with DT for SURF, SIFT, and ORB descriptors to dt_3_eq_ind.

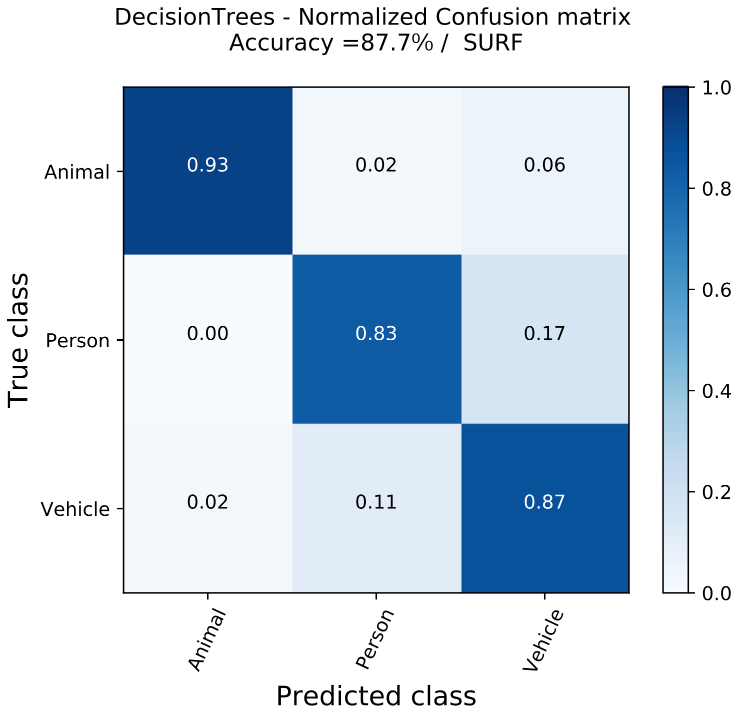


Figure 5.42: Confusion matrices applying the bounding box prediction method with DT for SURF descriptors to dt_3_diff_ind.

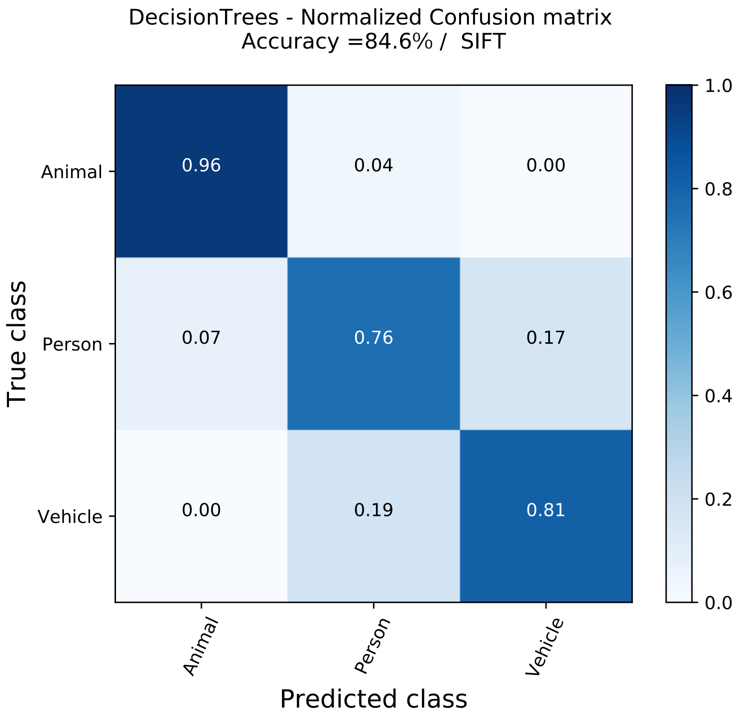


Figure 5.43: Confusion matrices applying the bounding box prediction method with DT for SIFT descriptors to dt_3_diff_ind.

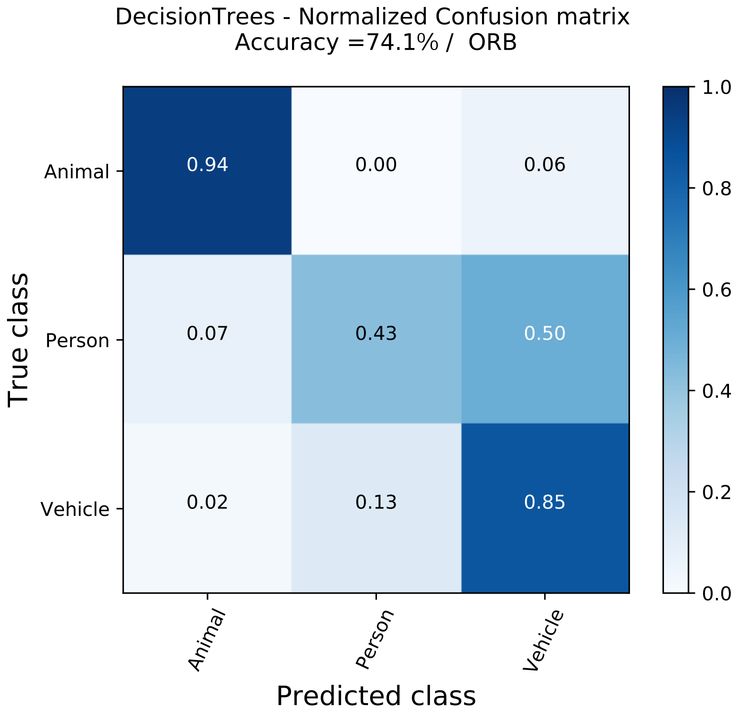


Figure 5.44: Confusion matrices applying the bounding box prediction method with DT for ORB descriptors to dt_3_diff_ind.

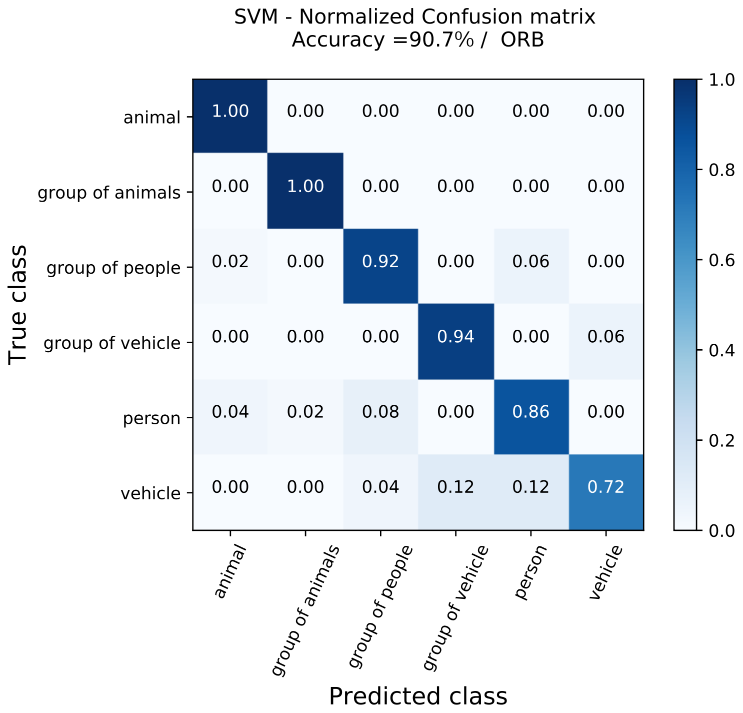


Figure 5.45: Confusion matrices applying the bounding box prediction method with SVM for ORB descriptors to dt_6_eq_gr.

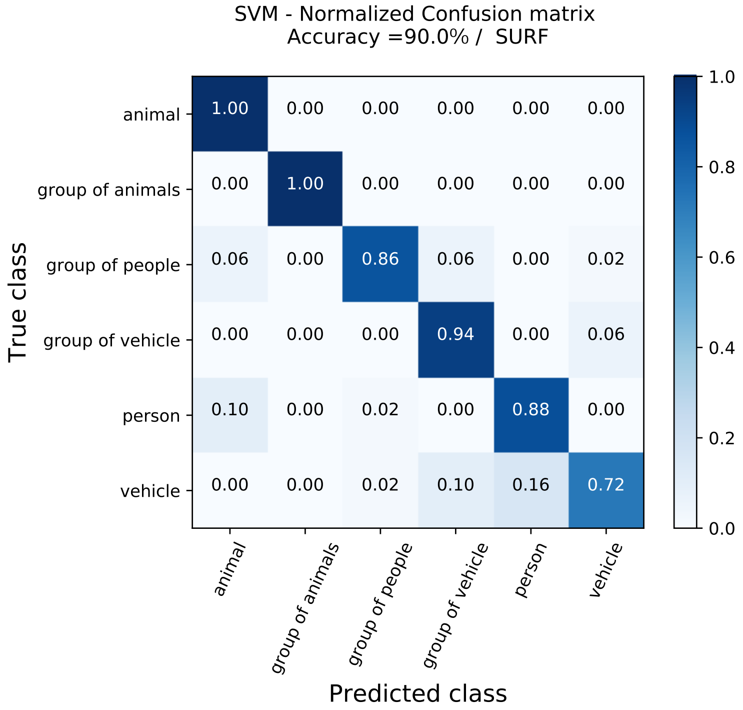


Figure 5.46: Confusion matrices applying the bounding box prediction method with SVM for SURF descriptors to dt_6_eq_gr.

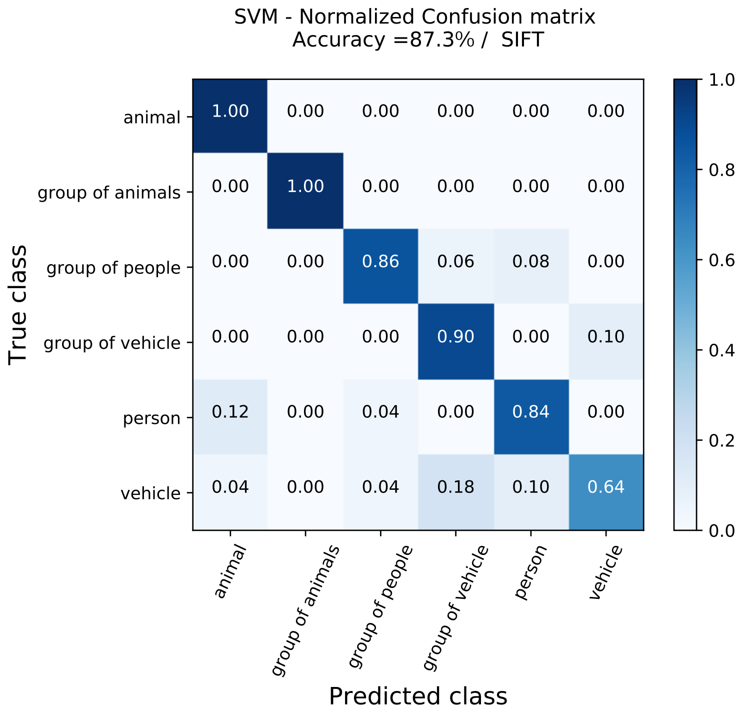


Figure 5.47: Confusion matrices applying the bounding box prediction method with SIFT, and ORB descriptors to dt_6_eq_gr.

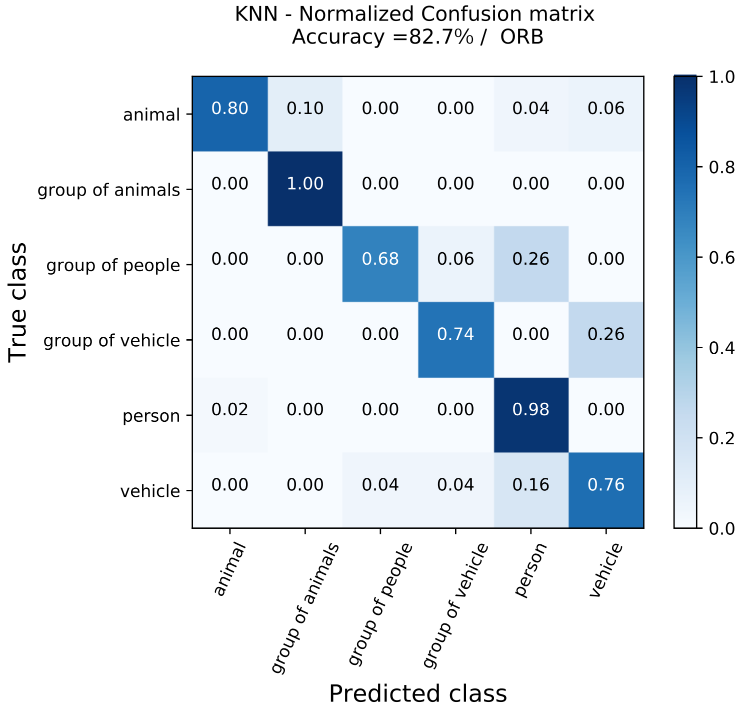


Figure 5.48: Confusion matrices applying the bounding box prediction method with KNN for ORB descriptors to dt_6_eq_gr.

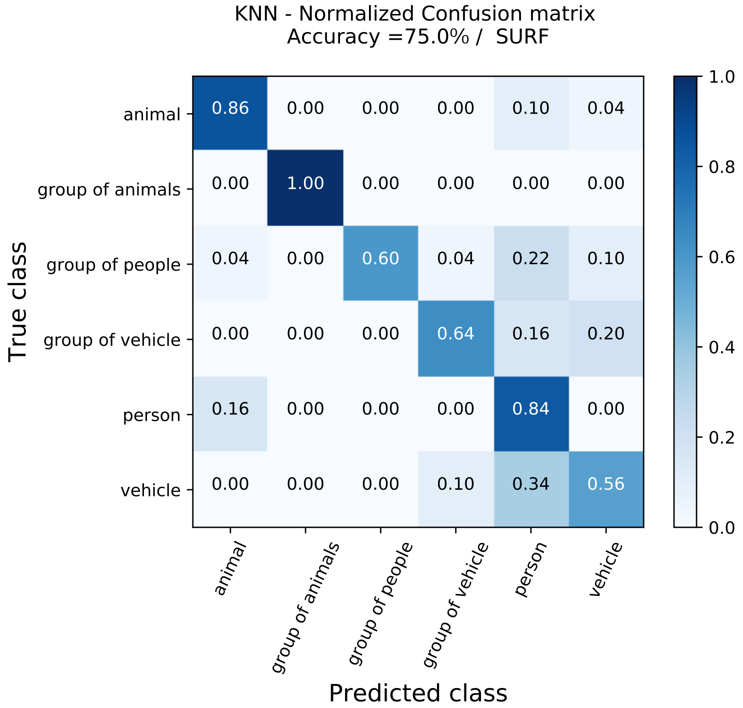


Figure 5.49: Confusion matrices applying the bounding box prediction method with KNN for SURF descriptors to dt_6_eq_gr.

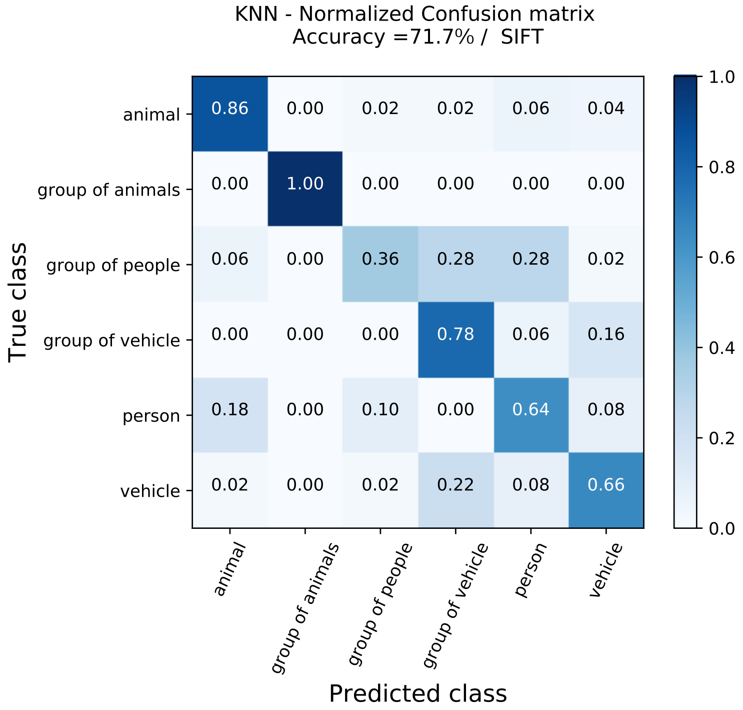


Figure 5.50: Confusion matrices applying the bounding box prediction method with KNN for SIFT descriptors to dt_6_eq_gr.

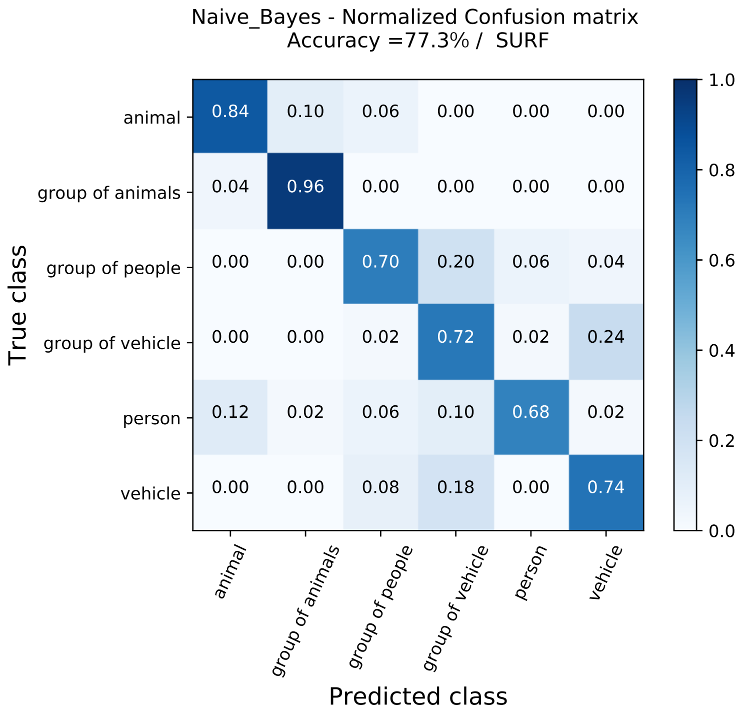


Figure 5.51: Confusion matrices applying the bounding box prediction method with NB for SURF descriptors to dt_6_eq_gr.

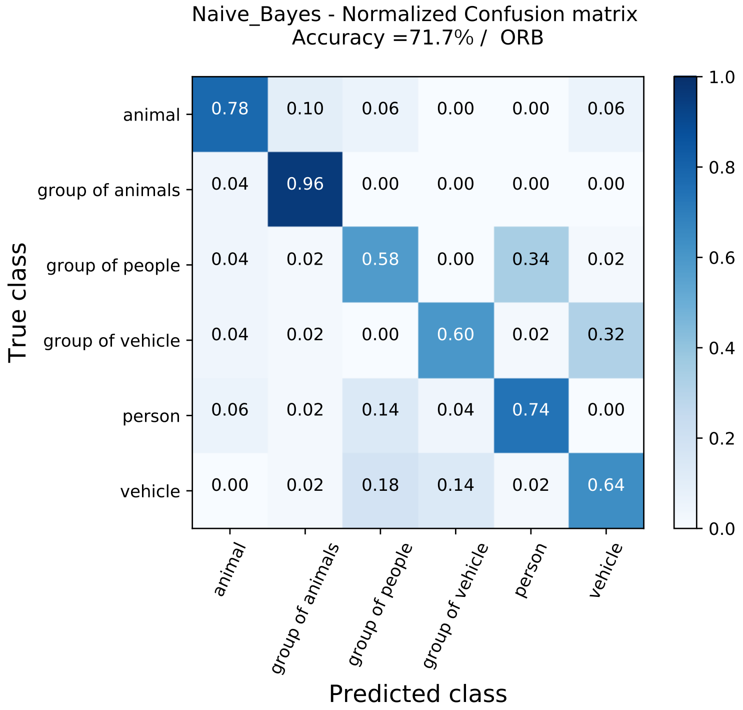


Figure 5.52: Confusion matrices applying the bounding box prediction method with NB for ORB descriptors to dt_6_eq_gr.

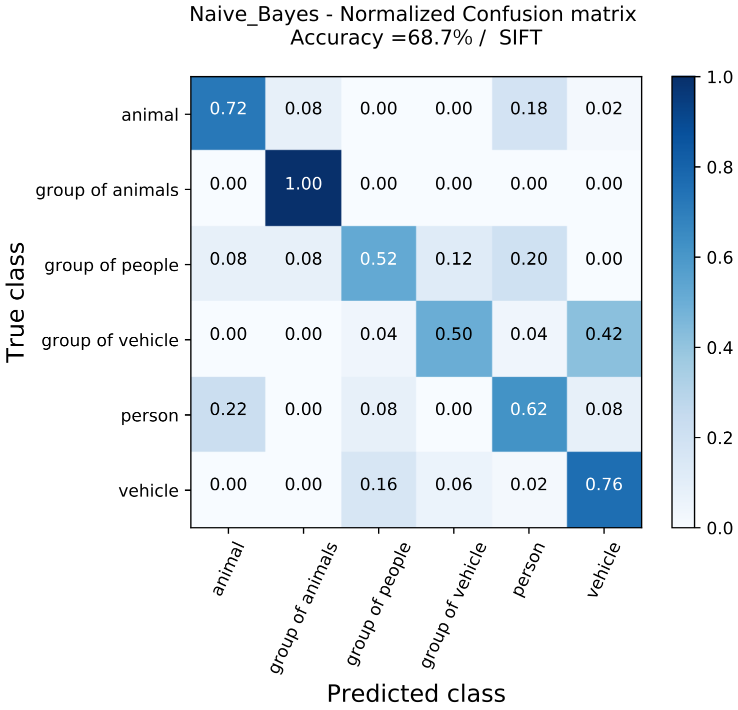


Figure 5.53: Confusion matrices applying the bounding box prediction method with NB for SIFT descriptors to dt_6_eq_gr.

Figure 5.54: Confusion matrices applying the bounding box prediction method with DT for ORB descriptors to dt_6_eq_gr.

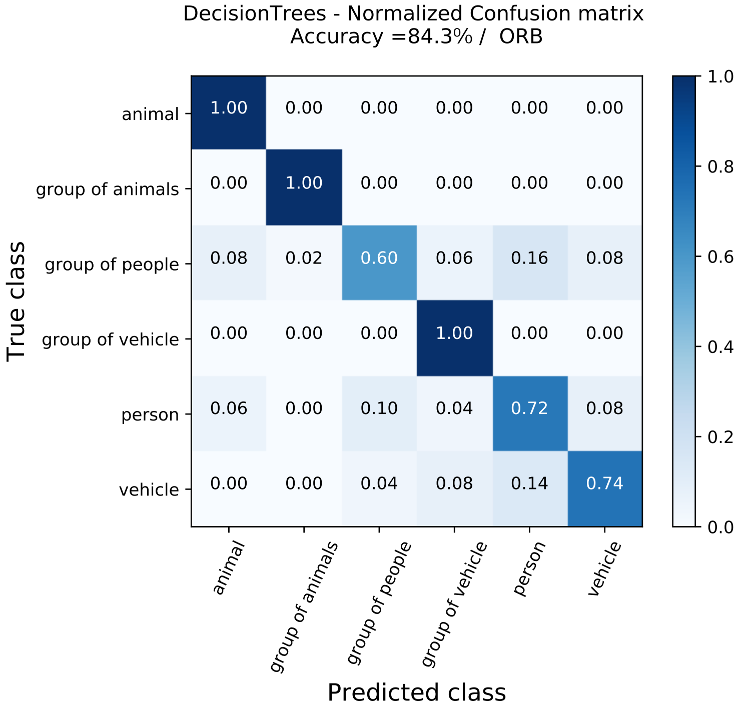


Figure 5.55: Confusion matrices applying the bounding box prediction method with DT for SURF, SIFT, and ORB descriptors to dt_6_eq_gr.

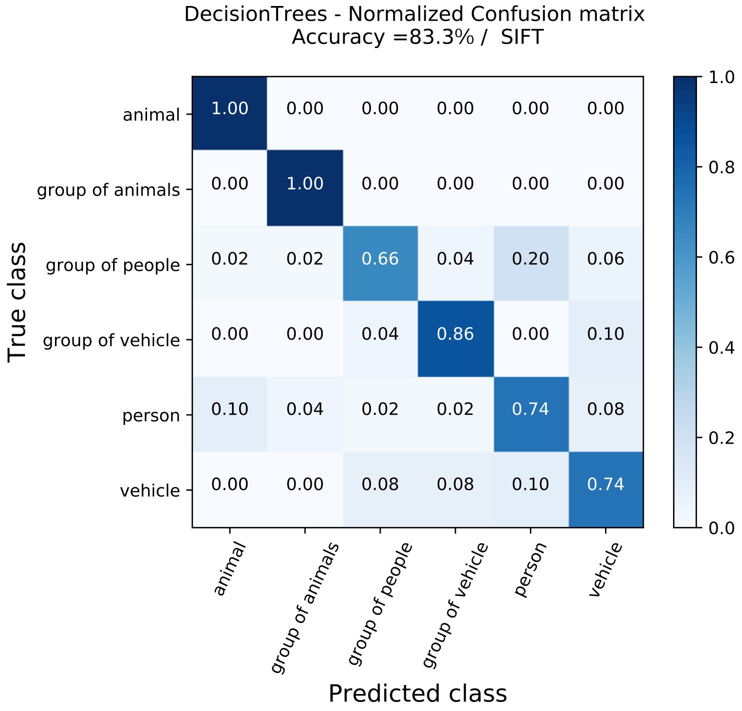
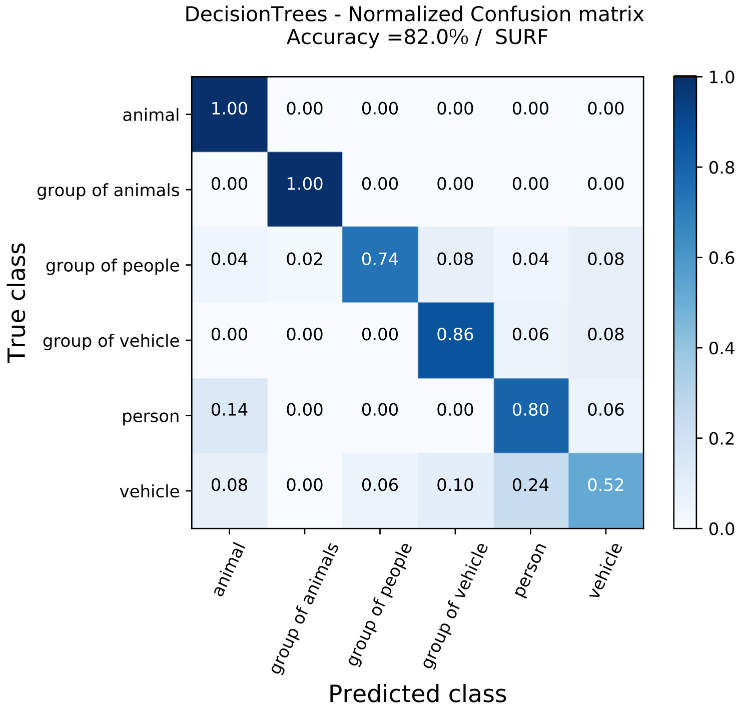


Figure 5.56: Confusion matrices applying the bounding box prediction method with DT for SIFT descriptors to dt_6_eq_gr.



Appendix II: Source Code

```python
import sys
import os
import cv2
import imutils
import numpy as np
from sklearn.svm import LinearSVC, SVC
import joblib
from scipy.cluster.vq import kmeans,vq
from sklearn.preprocessing import StandardScaler

def mylistdir(directory):
    filelist = os.listdir(directory)
    return [x for x in filelist
            if not (x.startswith(’.’))]

detector_number = 1
if detector_number == 0:
    detector = cv2.xfeatures2d.SIFT_create()
    detector_name = "SIFT"
elif detector_number == 1:
    detector = cv2.xfeatures2d.SURF_create()
    detector_name = "SURF"
elif detector_number == 2:
    detector = cv2.ORB_create()
    detector_name = "ORB"

# Get the path of the training set
trainingSet = ’Dataset_Set3/train/’
train_path = "Dataset_Set3/train/"
training_names = mylistdir(train_path)

# Get all the paths to the images and save them in a list
# image_paths and the corresponding label in image_paths
image_paths = []
image_classes = []
class_id = 0
for training_name in training_names:
    dir = os.path.join(train_path, training_name)
    class_path = imutils.imlist(dir)
    image_paths+=class_path
    image_classes+=[class_id]*len(class_path)
    print (len(class_path),training_name)
    class_id+=1

# List where all the descriptors are stored
des_list = []

for image_path in image_paths:
    im = cv2.imread(image_path)
    gray = cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    kpts, des = detector.detectAndCompute(gray, None)
    des_list.append((image_path, des))

descriptors = des_list[0][1]
for image_path, descriptor in des_list[1:]:
    descriptors = np.vstack((descriptors, descriptor))

# Perform k-means clustering
k = 3000
voc, variance = kmeans(descriptors.astype(np.float),  k, 1)

# Calculate the histogram of features
im_features = np.zeros((len(image_paths), k), "float32")
for i in range(len(image_paths)):
    words, distance = vq(des_list[i][1],voc)
    for w in words:
        im_features[i][w] += 1

# Perform Tf-Idf vectorization
nbr_occurences = np.sum( (im_features > 0) * 1, axis = 0)
idf = np.array(np.log((1.0*len(image_paths)+1) / (1.0*nbr_occurences + 1)), 'float32')

# Scaling the words
stdSlr = StandardScaler().fit(im_features)
im_features = stdSlr.transform(im_features)

# Train the Linear SVM
clf = SVC(kernel='linear')
#clf = LinearSVC()
clf.fit(im_features, np.array(image_classes))

# Save the SVM
joblib.dump((clf, training_names, stdSlr, k, voc), (detector_name+"bof.pkl"), compress = 3)
```

Listings 4.1: Python code of Train the Model

```

import cv2
import imutils
import numpy as np
import numpy
import os
import joblib
from scipy.cluster.vq import vq
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report,cohen_kappa_score
import matplotlib.pyplot as plt
import sys
import itertools
import logging as log

np.set_printoptions(threshold=sys.maxsize)

def listdir_nohidden(path):
    for f in os.listdir(path):
        if not f.startswith(’.’):
            yield f

detector_number = 2
if detector_number == 0:
    detector = cv2.xfeatures2d.SIFT_create()
    detector_name = "SIFT"
elif detector_number == 1:
    detector = cv2.xfeatures2d.SURF_create()
    detector_name = "SURF"
elif detector_number == 2:
    detector = cv2.ORB_create()
    detector_name = "ORB"


# Load the classifier, class names, scaler, number of clusters, and vocabulary.
clf, classes_names, stdSlr, k, voc = joblib.load((detector_name+"bof.pkl"))



# Obtain the pathway to the image(s) used for testing and accumulate them within a list.
image_paths = []
test_arr = []
test_path = "Dataset_Set3/test/"
testing_names = listdir_nohidden(test_path)
for testing_name in testing_names:
    dir = os.path.join(test_path, testing_name)
    class_path = imutils.imlist(dir)
    image_paths+=class_path
    #print (testing_name)
    list = os.listdir(dir) # dir is your directory path
    number_files = len(list)
    #print (number_files)
    for iter in range(0 , number_files):
        test_arr.append(testing_name)


# List where all the descriptors are stored.
des_list = []
for image_path in image_paths:
    im = cv2.imread(image_path)
    gray = cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    kpts, des = detector.detectAndCompute(gray, None)
    des_list.append((image_path, des))

# Stack all the descriptors vertically in a numpy array

descriptors = des_list[0][1]
for image_path, descriptor in des_list[0:]:
    descriptors = np.vstack((descriptors, descriptor))

test_features = np.zeros((len(image_paths), k), "float32")
for i in range(len(image_paths)):
    words, distance = vq(des_list[i][1],voc)
    for w in words:
        test_features[i][w] += 1


nbr_occurences = np.sum( (test_features > 0) * 1, axis = 0)
idf = np.array(np.log((1.0*len(image_paths)+1) / (1.0*nbr_occurences + 1)), ’float32’)

# Scale the features
test_features = stdSlr.transform(test_features)


# Perform the predictions
predictions =  [classes_names[i] for i in clf.predict(test_features)]
#print (predictions)



predictions =  [classes_names[i] for i in clf.predict(test_features)]
print("Calculating Accuracy.....")
total = 0
hits = 0
errors = 0
for image_path, prediction in zip(image_paths, predictions):
    classe=os.path.split(image_path)[-2].split("/")[-1]
    total += 1
    if prediction == classe:
        hits += 1
    else:
        errors += 1
avg = (hits/total) * 100
```

```


acc = accuracy_score(test_arr, predictions)

rpt = classification_report(test_arr, predictions)
rpt = rpt.replace('avg / total', 'avg / total')
rpt = rpt.replace('   support', '   support')
plt.annotate(rpt, xy = (0,0), xytext = (0, -280),
     xycoords='axes fraction', textcoords='offset points',
     fontsize=12, ha='right')

conf = confusion_matrix(test_arr, predictions)
conf = conf.astype('float') / conf.sum(axis=1)[:, np.newaxis]
plt.matshow(conf, fignum=False, cmap='Blues', vmin=0, vmax=1.0)
```

Listings 4.2: Python code of Evaluating the Classifier Accuracy