Multiple target extraction and identification for automotive radar with A.I.

Colin Decourt

RÉPUBLIQUE FRANÇAISE
Liberté Égalité Fraternité

Université de Toulouse

Doctorat de l'Université de Toulouse

préparé à l'Université Toulouse III - Paul Sabatier

Extraction et identification de cibles multiples pour radar automobile à l'aide d'intelligence artificielle

---

Thèse présentée et soutenue, le 19 décembre 2023 par

Colin DECOURT

École doctorale
EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse

Spécialité
Informatique et Télécommunications

Unité de recherche
CERCO - Centre de Recherche Cerveau et Cognition

Thèse dirigée par
Rufin VANRULLEN et Thomas OBERLIN

Composition du jury
M. Yassine RUICHEK, Rapporteur, Université de Technologie de Belfort Montbéliard
M. Martin BOUCHARD, Rapporteur, University of Ottawa
M. Didier SALLE, Examinateur, NXP Semiconductors, France
M. Rufin VANRULLEN, Directeur de thèse, CNRS Toulouse - CerCo
M. Thomas OBERLIN, Co-directeur de thèse, ISAE-SUPAERO
Mme Veronique BERGE-CHERFAOUI, Présidente, Université de Technologie de Compiègne

Contents

List of Figures

List of Tables

Glossary

ACC Adaptive Cruise Control

ADAS Advanced Driver Assistance Systems

ADC Analog to Digital Converter

AEB Automatic Emergency Braking

ANN Artificial Neural Network

AoA Angle of Arrival

ASPP Atrous Spatial Pyramid Pooling

AUC Area Under the Curve

BEV Bird-Eye-View

BPTT Backpropagation through time

BSD Blind Sport Detection

CCA Canonical Correlation Analysis

CDMA Code Division Multiple Access

CFAR Constant False Alarm Rate

CNN Convolutional Neural Networks

CRF Condional Random Fields

CRNN Convolutional Recurrent Neural Network

CUT Cell Under Test

CW Continuous Wave

DDMA Doppler Division Multiple Access

DML Deep Metric Learning

DoA Direction of Arrival

EM Electromagnetic Waves

EMA Exponential Moving Average

FCMAE Fully Convolutional Masked Auto-Encoder

FCN Fully Convolutional Networks

FFT Fast Fourier Transform

FMCW Frequency Modulated Continuous Wave

FP False Positive

FPN Feature Pyramid Network

GAN Generative Adversarial Networks

GNN Graph Neural Network

GPU Graphical Power Unit

GT Ground-Truth

IoU Intersection Over Union

IR Inverted Residual

LAC Lane Change Assist

LRR Long-Range Radar

mAP Mean Average Precision

MCU MicroController Unit

MIM Masked Image Modeling

MIMO Multiple Input Multiple Output

MLM Masked Language Modelling

MLP Multi-Layer Perceptron

MRR Medium-Range Radar

NAS Neural Architecture Search

NLP Natural Language Processing

OFDM Orthogonal Frequency-Division Multiplexing

OVA One-vs-All

OVO One-vs-One

PMCW Pulse Modulated Continuous Wave

PRF Pulse Repetition Frequency

R-CNN Region Convolutional Neural Network

RA Range-Angle

RAD Range-Angle-Doppler

RADAR RAdio Detection and Ranging

RCS Radar Cross Section

RD Range-Doppler

ReLU REctified Linear Unit

RF Radio Frequency

RNN Recurrent Neural Networks

RoI Regions of Interest

RPN Region Proposal Network

RSP Radar Signal Processing

SD Self-Distillation

SFCW Stepped Modulated Continuous Wave

SIMO Single Input Multiple Output

SNR Signal-to-Noise Ratio

SOTA State-of-the-art

SSD Single Shot Detector

SSL Self-Supervised Learning

SSR Short-Range Radar

SVM Support Vector Machine

TDMA Time Division Multiple Access

TP True Positive

V2X Vehicle-to-Everything

ViT Vision Transformers

YOLO You Only Look Once

Résumé en français

Introduction

Ces dernières années, poussée par le besoin de systèmes de transport plus sûrs et plus autonomes, l'industrie automobile a connu un changement de paradigme vers l'intégration d'un nombre croissant de systèmes avancés d'aide à la conduite (ADAS). À mesure que les niveaux d'aides à la conduite augmentent, allant de l'aide au freinage à des niveaux plus élevés d'automatisation (dits de niveaux 4 et 5), il est désormais primordial de développer des systèmes de perception robustes. Dans cette thèse, on appelle perception la capacité d'un système à modéliser son environnement à l'aide de multiples capteurs.

La plupart des systèmes d'aides à la conduite reposent sur des capteurs de types caméras, LiDAR et radar pour créer une représentation de l'environnement, chacun de ces capteurs présentant des avantages et des inconvénients. Par exemple, la haute résolution des caméras est indispensable pour lire les panneaux de signalisation ou pour reconnaître des objets. D'un autre côté, le LiDAR apparaît comme un capteur adapté pour cartographier en 3D l'environnement de part sa haute résolution angulaire. Cependant, en cas de mauvaises conditions météorologiques (brouillard, pluie) et lumineuses (nuit, contre-jour), l'efficacité des caméras et des LiDAR est limitée. Également, bien qu'il soit possible d'obtenir des informations de vitesse et de distance à l'aide de caméras stéréo par exemple, la capacité des caméras à estimer ces grandeurs reste extrêmement limitée, et il en est de même pour le LiDAR.

D'un autre côté, le radar s'est imposé comme un concurrent de choix pour compléter les caméras et le LiDAR en raison de ses capacités uniques et sa robustesse pour détecter des objets et estimer leur vitesse dans des conditions météorologiques défavorables ou des scénarios à faible luminosité. En émettant des ondes radio et en mesurant leurs réflexions, le radar permet de mesurer la vitesse et la distance avec une grande précision. Alors que les faisceaux laser émis par le LiDAR peuvent être diffractés par des gouttelettes d'eau, créant ainsi des fausses détections, les ondes émises par le radar les traversent et n'entravent pas le fonctionnement du radar.

Combinés, les caméras, le LiDAR et le radar garantissent un cocon de sécurité à 360° autour du véhicule. Si la fusion des capteurs est apparue comme une approche essentielle pour accroître la précision, la sécurité et la redondance des systèmes ADAS, cette efficacité dépend grandement de la capacité de chaque capteur à fournir une représentation adéquate de l'environnement. Grâce à l'émergence de l'apprentissage profond, des algorithmes de vision par ordinateur ainsi que le grande nombre de jeu de données pour des applications de conduites autonomes [Ettinger 2021, Urtasun 2012, Caesar 2019], des progrès considérables ont été faits

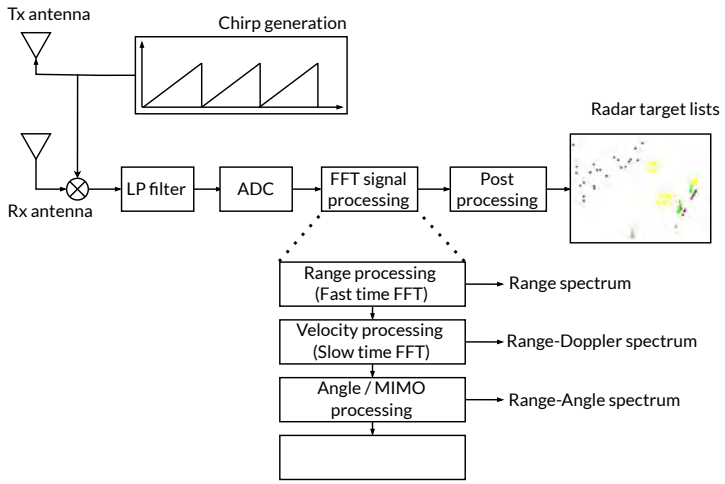


Figure 1: Chaîne de traitement du signal d'un radar FCMW. L'antenne Tx émet des ondes électromagnétiques qui sont réfléchies par les objets environnant le radar et reçues par l'antenne Rx. Une fois reçue, l'onde réfléchie est multipliée par l'onde émise, filtrée et échantillonnée au travers de l'ADC (convertisseur analogique-numérique). Les données de l'ADC sont ensuite analysée pour estimer la distance, la vitesse et l'angle d'arrivée des objets environnant le radar. Une étape de post-traitement (groupement et suivi de cibles) est appliquée pour produire le nuage de point radar.

dans le développement d'algorithmes de perception et de fusion spécifiques aux capteurs caméras et LiDAR. Cependant, malgré les atouts du radar, peu de travaux ont exploré les possibilités d'utiliser de l'intelligence artificielle (IA) sur des données radar. Ce manque d'exploration est attribué à plusieurs facteurs, notamment la disponibilité limitée des jeux d'apprentissage avec des données radar, sa résolution angulaire limitée par rapport aux capteurs caméra et LiDAR mais également les défis inhérents au traitement de ces données à l'aide d'IA.

En se concentrant sur les capteurs radar, cette thèse vise à combler le fossé entre les radars automobiles et les algorithmes de perception basés sur l'IA. Dans leur forme actuelle, les données radar consistent en une liste de cibles (également connues sous le nom de nuages de points radar), qui contiennent des informations sur la position de la cible, sa vitesse et une notion de surface équivalente radar (RCS) caractérisant la cible. Dans la littérature, certains travaux ont déjà essayé d'utiliser des modèles d'IA sur ces données radar pour de la classification ou de la segmentation d'objets [Scheiner 2018, Scheiner 2020, Ulrich 2021, Tatarchenko 2023]. Cependant, comme le montre la Figure 1, bien que permettant le développement d'algorithmes peu coûteux en mémoire et en calcul, les nuages de points radar nécessitent d'importantes étapes de pré et post-traitement (détection, suivi, regroupement) avant d'être utilisés par des modèles d'intelligence artificielle. De plus, ces

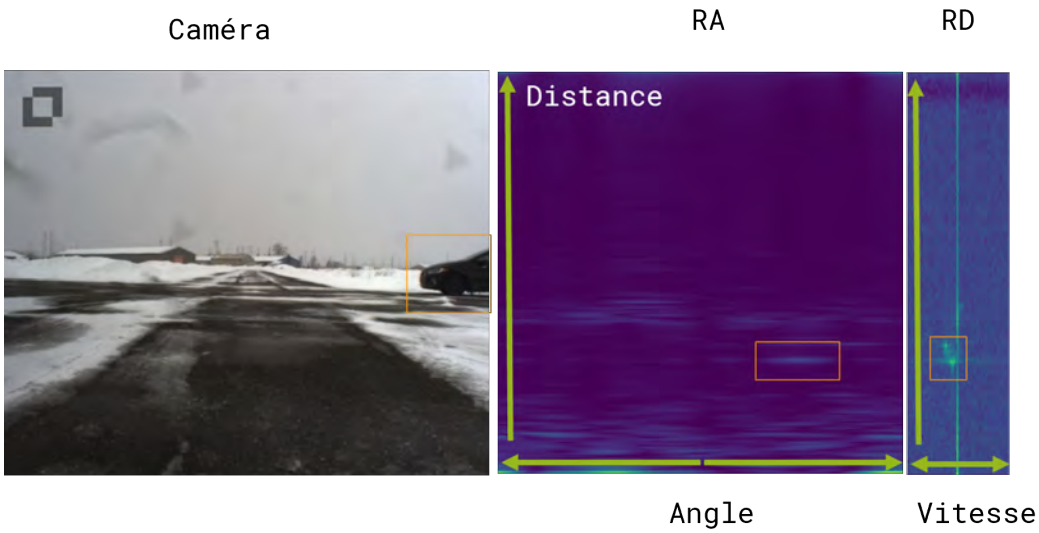


Figure 2: Exemples de spectres RD et RA. De gauche à droite: image caméra, spectre RD et spectre RA

étapes de traitement filtrent le signal brut réfléchi par les objets, ce qui peut affecter les performances des algorithmes d'intelligence artificielle.

Une alternative aux nuages de points consiste à représenter le signal réfléchi par les objets sous la forme d'un spectre (données brutes) qui représente l'environnement en distance et en vitesse (distance-Doppler, RD), en distance et en angle (distance-angle, RA), ou en distance, en angle et en vitesse (distance-angle-Doppler, RAD). La Figure 2 montre un spectre RD et RA avec l'image caméra associée. Ces trois dernières années, la parution de base de données comme CARRADA [Ouaknine 2021b], RADDet [Zhang 2021] ou encore CRUW [Wang 2021c] a permis d'accélérer la recherche vers le développement de modèles d'IA pour la détection et la classification d'objets à partir de données radar brutes. Alors que certains travaux se concentrent sur la classification d'objets à partir de données brutes [Akita 2019, Palffy 2020, Khalid 2019], d'autres aspirent à réduire le nombre d'étapes de post-traitement du radar pour détecter et classifier simultanément des objets [Ouaknine 2021a, Wang 2021b, Gao 2021, Giroux 2023].

Inspirée de ces travaux, cette thèse propose d'exploiter les spectres radar pour détecter et identifier des usagers de la route dans des environnements complexes. Essentiellement, cette thèse vise à proposer des algorithmes d'apprentissage profond conçus explicitement pour les données radar et à étudier si ces algorithmes peuvent se substituer à certaines étapes dans la chaîne de traitement radar. Cette thèse a été réalisée au sein de l'institut d'intelligence artificielle ANITI, en collaboration avec la société NXP, un leader mondial dans le domaine des émetteurs-récepteurs et des micro contrôleurs pour radars automobiles. L'entreprise est activement

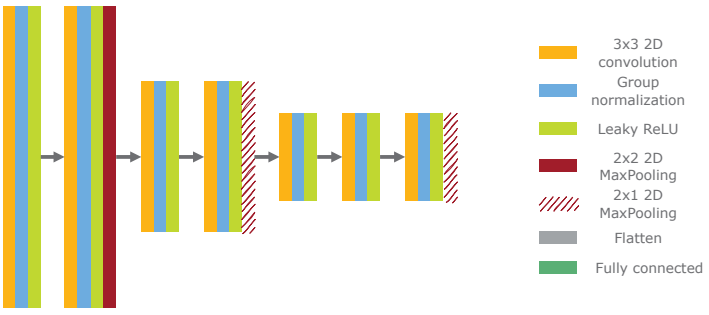
impliquée dans la construction de radars de nouvelle génération pour améliorer la sécurité routière et le confort du conducteur. Dans ce contexte, cette thèse constitue une étape pionnière dans la conception d'émetteurs-récepteurs et de micro contrôleurs de radar basés sur l'IA. Les algorithmes proposés dans cette thèse devront répondre aux contraintes de l'environnement automobile : faible consommation d'énergie, faible complexité et temps de réaction rapide.

Détection et identification d'objets à partir de spectres distance-vitesse

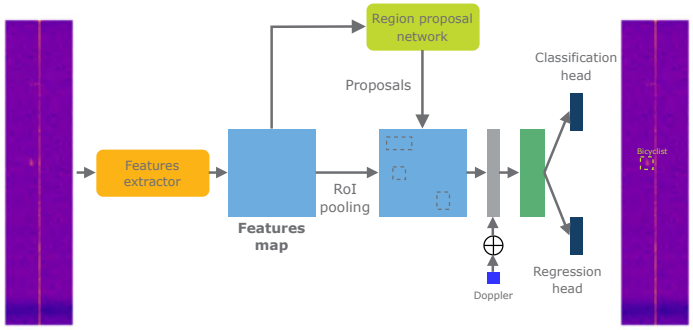
La première étude de cette thèse est dédiée à la détection et l'identification d'objets multiples à partir de spectres RD. Le choix d'utiliser le spectre RD pour des tâches de détection (ou de segmentation) plutôt que RA [Wang 2021b, Ju 2021] et RAD [Ouaknine 2021a, Gao 2021] est principalement motivé par des raisons d'efficacité. Premièrement, le spectre RAD exige beaucoup de calcul pour être produit et est volumineux en mémoire. Deuxièmement, dans la chaîne de traitement radar, les objets sont détectés dans la représentation RD (à l'aide d'algorithmes de type CFAR [Blake 1988], voir Figure 1) puis, pour chaque objet, l'angle auquel se trouve l'objet par rapport au radar est calculé. Ainsi, l'utilisation de ce type de spectre apparaît naturelle pour intégrer un modèle d'IA dans un système radar, de la manière la plus efficace possible.

Le spectre RD pouvant être vu comme une *image* représentant l'environnement en distance et vitesse, cette étude vise à étudier la possibilité d'utiliser des algorithmes de détection pensés pour des images caméras au radar. Plus particulièrement, un modèle de type Faster R-CNN [Ren 2017] est adapté pour les données radar. La Figure 3 illustre l'architecture de Faster R-CNN utilisée dans cette étude. Étant donné un spectre RD, un réseau de neurones convolutionnel (CNN) est d'abord utilisé pour extraire des caractéristiques spécifiques à la scène. Ensuite, un second petit CNN est utilisé pour générer des régions du spectre susceptibles de contenir des objets (le RPN). Pour chacune de ces régions, un autre petit réseau de neurones est utilisé pour déterminer la classe de l'objet présent dans la région ainsi que sa position exacte.

Les spectres RD et les images caméras présentant des différences notables en termes de tailles, de textures et de formes d'objets, il n'est pas optimal d'utiliser des réseaux de neurones pensés pour des images caméras dans l'optique d'extraire des caractéristiques des spectres radar. De plus, les modèles utilisés pour l'extraction de caractéristiques pour des images caméras sont bien trop gros et coûteux en calcul pour être utilisés dans des applications embarquées. Un réseau de neurones



(a) Proposed light backbone architecture



(b) Faster R-CNN architecture

Figure 3: Architecture de Faster R-CNN et du modèle DAROD proposée. **(a)** Architecture de DAROD. Le modèle proposé est inspirée de VGG [Simonyan 2015] **(b)** Exemple du modèle Faster R-CNN. Étant donné un spectre RD, un réseau de neurones convolutionnel (CNN) est d'abord utilisé pour extraire des caractéristiques spécifiques à la scène. Ensuite, un second petit CNN est utilisé pour générer des régions du spectre susceptibles de contenir des objets (le RPN). Pour chacune de ces régions, un autre petit réseau de neurones est utilisé pour déterminer la classe de l'objet présent dans la région ainsi que sa position exacte.

convolutionnel (CNN) spécifique aux données radar, dénommé DAROD, léger en mémoire, nécessitant peu d'opérations est donc proposé et intégré à un modèle Faster R-CNN. Ce réseau est inspiré de l'architecture VGG [Simonyan 2015] et est composé de trois blocs de convolutions. Une pratique courante en apprentissage profond consiste à réduire la taille de la donnée d'entrée à mesure que la profondeur de réseau augmente. Les dimensions du spectre représentant des informations de distance et de vitesse, la résolution spatiale du spectre est réduite d'un facteur deux dans la dimension Doppler pour minimiser la perte d'information de vitesse des objets. Également, pour chaque région proposée par le RPN, la vitesse de l'objet dans la région est estimée et ajoutée comme information additionnelle pour améliorer la classification de l'objet.

Le modèle proposé, est entraîné sur deux jeux de données différents (CARRADA [Ouaknine 2021b] et RADDet [Zhang 2021]). De manière générale, les

expériences montrent qu'un modèle d'apprentissage profond atteint de très bonnes performances de détection en utilisant uniquement des données radar brutes de type spectres distance-vitesse. DAROD se montre plus performant que l'implémentation originale de Faster R-CNN (qui utilise un modèle créé pour des images caméras, ResNet-50 [He 2016]), ainsi que le modèle RADDet [Zhang 2021] spécifiquement développé pour des données radar, tout en ayant un coût calculatoire moindre. Des expériences supplémentaires montrent aussi que l'ajout de l'information de vitesse comme information additionnelle pour aider la classification améliore les performances du modèle. Enfin, les comparaisons avec des approches classiques (détection avec un algorithme CFAR [Blake 1988], regroupement de cibles et suivi de cibles) montrent qu'une approche basée sur de l'apprentissage profond améliore la précision de localisation des objets et diminue le nombre de fausses détections, confirmant la promesse de l'IA pour le radar automobile.

Détection et identification d'objets en temps réel à partir de données radar brutes

La seconde étude de cette thèse est dédiée à la détection d'objets à partir de données radar en temps réel. Cette étude, un peu plus générale, vise à exploiter l'information temporelle pour améliorer les performance de détection des détecteurs d'objet radar basées sur de l'apprentissage profond. Les modèles utilisés dans la première étude ont montré de bonnes performances de détection et de classification. Cependant, leurs capacités à différencier des objets de classes similaires (comme des piétons et des cyclistes) sont limitées. En radar, exploiter l'information temporelle est cruciale car la signature d'un objet évolue au cours du temps et varie selon plusieurs facteurs tels que sa distance par rapport au radar, son orientation et sa classe. Ainsi, l'exploitation de l'information temporelle, c'est-à-dire l'utilisation de plusieurs spectres radar à des pas de temps successifs, pourrait permettre d'apprendre des informations comme la dynamique de l'objet et donc de limiter la confusion entre classes.

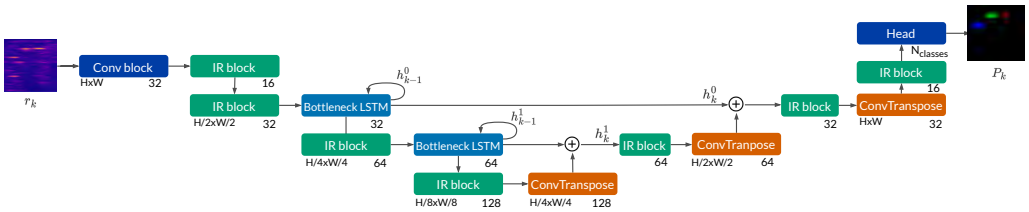
Récemment, différent travaux ont vu le jour dans le but d'apprendre des dépendances temporelles entre différents spectres radar ou entre objets. Principalement, ces approches reposent sur des convolutions temporelles [Ouaknine 2021a, Wang 2021b, Ju 2021], sur des réseaux de neurones récurrents convolutionnels [Major 2019] ou sur des modèles d'attention [Li 2022]. Cependant, la plupart de ces méthodes ont du mal à capturer des dépendances à long terme et sont souvent non-causales (elles utilisent des informations du passés et du futur) et donc impossibles à utiliser en temps réel.

Dans le but d'extraire des dépendances spatio-temporelles entre objets, un nou-

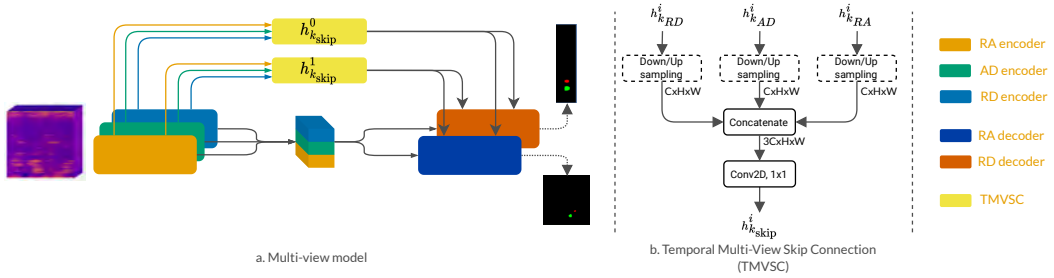
veau modèle mélangeant convolutions et réseaux de neurones récurrents est proposé. Contrairement à la plupart des modèles de détection temporels pour du radar, le modèle proposé, appelé RECORD, est causal, c'est-à-dire qu'il n'utilise que l'information passé pour effectuer une prédiction de la position des objets à un instant donné. Le modèle proposé consiste en un modèle mélangeant convolution et LSTMs convolutionnels [Shi 2015] à différentes résolutions afin d'apprendre à extraire des informations pour des objets de différentes tailles. Pour satisfaire les contraintes d'un modèle temps réel, des convolutions et des LSTMs convolutionnel efficients sont utilisés (Inverted Residual Bottleneck blocs [Sandler 2018] et Bottleneck LSTM [Zhu 2018]). L'architecture étudiée ayant pour but d'être générique, elle peut traiter tout types de données radar brutes (RD, RA, RAD) et apprendre différentes tâches de détection d'objet (détection et segmentation). Deux variantes de l'architecture sont proposées: simple vue (RECORD, voir Figure 4a) et multi vues (MV-RECORD, voir Figure 4b). Alors que l'architecture simple vue est entraînée à détecter des objets dans l'espace distance-vitesse ou distance-angle, la multi vues utilise un encodeur par vue et est entraînée à détecter des objets simultanément dans l'espace distance-vitesse et distance-angle. Deux types d'entraînement sont proposés: *buffer* et *online*. L'approche *buffer* consiste à entraîner le modèle sur des séquences d'environ une seconde et à prédire la position des objets dans la scène uniquement en fonction des N précédents spectres. La *mémoire* (ses états) des ConvLSTMs est réinitialisée toutes les N spectres. L'inconvénient de cette approche est qu'elle nécessite de sauvegarder N spectres en mémoire avant de faire une prédiction. L'approche *online*, la plus efficace, consiste à entraîner le modèle à prédire la position des objets dans la scène pour chaque pas de temps. Le modèle est entraîné sur des séquences plus longues (environ deux secondes) sans jamais réinitialiser la *mémoire* des ConvLSTMs. Le modèle apprend ainsi à garder et supprimer les informations nécessaires selon les situations.

En raison de sa haute fréquence d'image (30 image par seconde) et de sa grande taille comparée à CARRADA [Ouaknine 2021b], RECORD est d'abord prototypé sur la base de données CRUW [Wang 2021c] sur des spectres distance-angle. Les performances de RECORD sont comparées à différents modèles basés sur des convolutions temporelles [Ju 2021], de l'attention [Fare Garnot 2021], des Transformers [Jiang 2023] et des variantes de RECORD n'utilisant pas le temps. Dans la plupart des scénarios de conduite, RECORD est plus performant que ses concurrents tout en étant plus efficace en terme de nombre de paramètres, d'opérations et de temps d'inférence. Les versions *online* et *buffer* montrent des performances similaires, cependant, la version online est bien plus efficace et adaptée pour être embarquée.

Entraîné sur la base de données CARRADA [Ouaknine 2021b], le modèle proposé surpasse le modèle état de l'art TMVA-Net [Ouaknine 2021a] dans plusieurs cas pour les versions *online* et *buffer*: multi-vues (RD et RA), simple vue (RD).



(a) Architecture du modèle RECORD. Les flèches arrondies représentent des couches récurrentes. Le signe plus représente l'opération de concaténation. Le nombre de filtre par couche et la taille de sortie sont représentés à gauche et à droite de chaque couche.



(b) Architecture du modèle multi vues MV-RECORD. Chaque encodeur utilise l'architecture de RECORD présentée en Figue 4a. Les blocs en pointillées correspondent à une opération optionnelle, appliquée seulement si les cartes de caractéristiques ont des tailles différentes.

Figure 4: Architectures des modèles RECORD et MV-RECORD.

Comme pour les expériences sur la base de données CRUW, les modèles RECORD et MV-RECORD sont plus efficients que TMVA-Net. Bien qu'intéressant pour la recherche, les approches multi vues sont longues et difficiles à optimiser et à intégrer dans un système radar. À mesure que la résolution des radar augmente, la quantité de mémoire nécessaire à la production des spectres RA et RAD augmente. En conclusion, cette étude suggère que les modèles simple vue semblent plus appropriés pour traiter des données radar brutes et pour détecter des objets. Appliqués sur des spectres RD et couplés à un algorithme d'estimation d'angle d'arrivée, ils devraient permettre d'améliorer les performances de détection et de classification des radars.

Apprentissage auto-supervisé pour de la détection d'objets radar

La dernière étude de cette thèse présente un travail préliminaire pour apprendre à détecter des objets avec peu de données radar annotées. L'annotation des données est cruciale en apprentissage profond pour apprendre de bonnes représen-

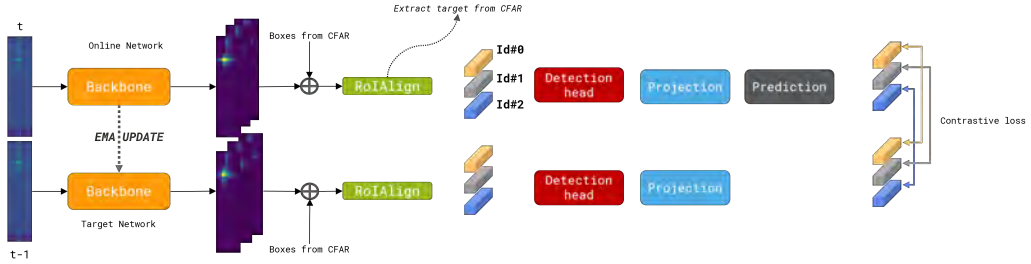


Figure 5: Vue d'ensemble de RICL. Deux réseaux sont utilisés pour encoder les caractéristiques de chaque spectre (un réseau *online* et un réseau *target*). L'opération RoIAlign [He 2017] est utilisée pour extraire les caractéristiques de chaque objet détecté par CFAR. Une fonction de coût contrastive est appliquée pour chaque paire d'objet.

tations pour la détection d'objets. L'annotation des données radar étant compliquée, la plupart des auteurs de base de données radar annotent les jeux de données de manière semi-automatique [Ouaknine 2021b, Wang 2021c, Zhang 2021, Rebut 2022]. Cependant, ces annotations reposent sur la fusion des détections à l'aide d'une caméra [He 2017] et des détections du radar (obtenues avec des méthodes *classiques*). Une telle méthode peut mener à des mauvaises détections ou des objets manqués. Le but de cette dernière étude vise donc à réduire la quantité de labels nécessaires à l'entraînement des modèles de détections d'objets utilisant des données radar, en les pré-entraînant de manière auto supervisée et en les spécialisant à une tâche de détection avec des annotations manuelles.

En utilisant un apprentissage contrastif, une extension de la méthode SoCo [Wei 2021] est proposée pour apprendre des représentations de ce qu'est un objet dans un spectre RD, sans utiliser de labels (appelé RICL). Une vue d'ensemble de la méthode est présentée en Figure 5. L'idée consiste à extraire la position d'un même objet à deux pas de temps successifs dans un spectre RD (à l'aide d'un algorithme de type CFAR), d'encoder cette représentation à l'aide d'un réseaux de neurones convolutionnel et de maximiser la similarité entre ces objets pour en extraire des informations relatives à leurs classes (inconnues au moment de l'entraînement). Une fois le modèle pré-entraîné (les représentations des objets apprises), le modèle est spécialisé sur une tâche donnée. Ici la détection d'objets.

Dans cette étude, le réseau de neurones convolutionnels choisi est un ResNet-50 [He 2016]. Ce modèle est pré-entraîné et spécialisé sur la base de données CARRADA [Ouaknine 2021b], et des spectres RD sont utilisés. Le modèle de détection choisi est le même que pour la première étude, à savoir un Faster R-CNN. Pour tester l'efficacité de la méthode, le modèle est spécialisé pour de la détection d'objets avec différentes quantités de labels, allant de 100% à 5%. Des comparaisons en utilisant un pré-entraînement différent (supervisé) sur

une base de données d'images naturelles (ImageNet [Russakovsky 2015]) sont également faites. En utilisant 10% et 20% de données, pré-entraîner Faster R-CNN avec RICL améliore les performances de détection par rapport à un entraînement avec une initialisation aléatoire. Dans le cas où plus de données labélisées sont utilisées, le pré-entraînement proposé ne semble pas améliorer les performances. En revanche, le pré-entraînement d'un ResNet-50 sur le jeu de données ImageNet améliore les performances de détection peu importe la quantité de labels. Ces travaux étant préliminaires, plusieurs pistes d'amélioration existent. Premièrement, la méthode utilisée pour associer les objets à deux pas de temps successif est simpliste. L'utilisation d'un algorithme de suivi de cibles pour associer les objets ensemble semble plus adéquate et plus précise. Également, l'apprentissage auto-supervisé repose sur la possibilité d'apprendre d'une grande quantité de données non annotées. En radar, les jeux de données sont petits et CARRADA est l'un des plus petit. Pré-entraîner le modèle à partir de jeux de données plus grand comme RADIal [Rebut 2022] est une autre piste de recherche.

Conclusion

Pour conclure, cette thèse montre le potentiel de l'utilisation de modèles d'IA pour améliorer les capacités de perception des radar automobiles en utilisant des données brutes: spectres distance-vitesse, distance-angle et distance-angle-vitesse. Ce travail a montré que des algorithmes d'IA utilisant ces données brutes peuvent se substituer aux traitements basés sur les nuages de points, plus coûteux et nécessitant une chaîne de pré- et post-traitement plus lourde. Il a également permis d'évaluer et de mieux comprendre les avantages et inconvénients des différents modèles, tâches de détection, et types de données, d'un point de vue des performances de détection mais aussi en vue de l'intégration dans une chaîne temps-réelle et embarquée. Ce travail souligne l'évolution constante de la synergie entre IA et radar, ouvrant la voie à des transports plus sûrs et plus intelligents.

Introduction

In recent years, driven by the need for safer and more autonomous transport systems, the automotive industry has undergone a paradigm shift towards the integration of a growing number of advanced driver assistance systems (ADAS, Figure 7). As we navigate the journey from low ADAS levels (driver assistance, partial and conditional automation) toward higher levels of driving automation (levels 4 and 5, see Figure 6), robust perception systems have become paramount. Perception forms the cornerstone of ADAS systems, allowing vehicles to represent their surroundings through multiple sensors, enabling informed decision-making for safer and more efficient driving scenarios.

Among the array of sensors employed in perception, the primary sensing technologies for ADAS systems are cameras, LiDAR and radars. Ultimately, there is no one-size-fits-all sensor solution. Each sensor has unique strengths and weaknesses and can complement or provide redundancy to the other sensor types [Gu 2022]. High-resolution camera sensors appear indispensable for *reading* traffic signs or detecting and classifying objects. The ultra-precise angular and fine resolutions at the range of LiDAR sensors make LiDAR well-suited for high-resolution 3D environment mapping. However, cameras and LiDAR technologies' effectiveness and reliability become compromised in varying lighting and harsh weather conditions. Despite the speed and depth information that can be obtained using stereo cameras, cameras' ability to measure distance and speed remains extremely limited. Also, LiDAR's ability to estimate velocity and detect objects far ahead remains limited.

On the other hand, radar has emerged as a formidable contender due to its

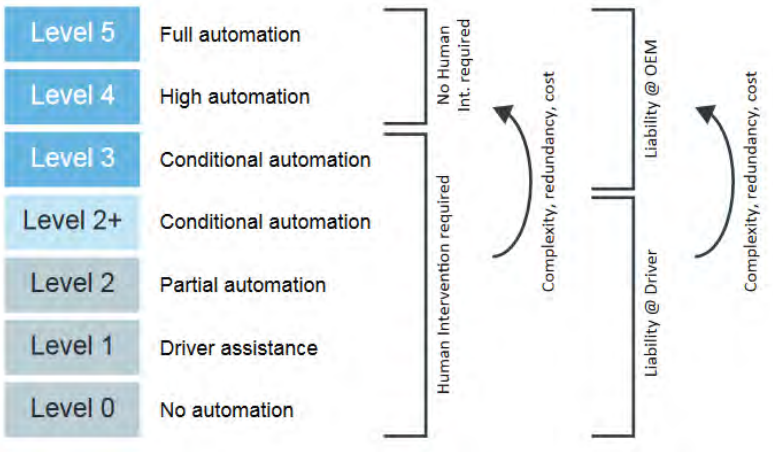


Figure 6: Levels of ADAS and their meaning. Source: [Gu 2022]

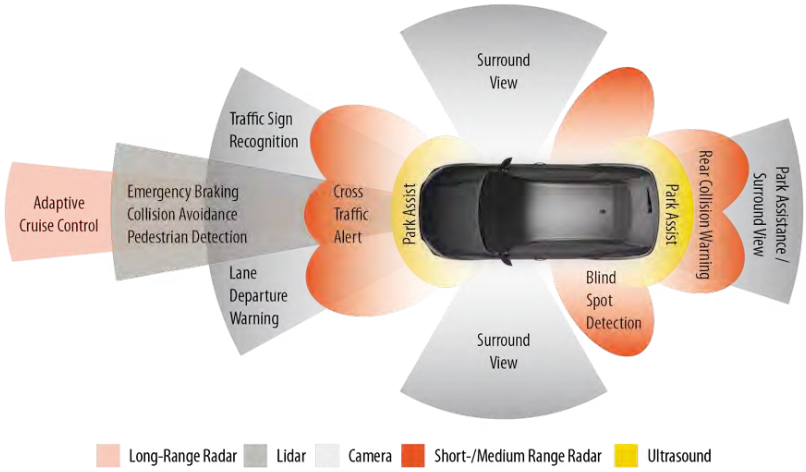


Figure 7: Type of sensors found in ADAS systems. Vehicles with ADAS are equipped with various cameras and sensors for 360-degree visibility. Source: `https://dewesoft.com/blog/types-of-adas-sensors`

unique capabilities in adverse weather conditions or low-light scenarios, and its robustness in maintaining consistent performance across diverse environments. By emitting radio waves and measuring their reflections, radar allows for highly accurate speed and distance measurements. While LiDAR illuminates the target scene with sparsely placed laser beams, radar illuminates the scene seamlessly. LiDAR may miss smaller targets at greater distances if the targets are situated between the sharply defined laser beams. As a result, radar is a much more reliable sensor for longer-range operation [Gu 2022]. Moreover, environmental debris and water drop refraction introduced by adverse weather conditions will not impair radar operations.

Combined, cameras, LiDAR and radar guarantee a 360° safety type cocoon around the vehicle as shown in Figure 7. While sensor fusion has emerged as a critical approach for enhancing the perception accuracy and the safety of ADAS systems (by adding sensor redundancy), the efficacy of fusion hinges upon the robustness and the performance of individual sensor processing. Driven by the recent surge in deep learning and computer vision, and the large number of automotive datasets [Ettinger 2021, Urtasun 2012, Caesar 2019] with cameras and LiDAR data, the research community has seen significant strides in the development of sensor-specific perception and sensor fusion involving cameras and LiDARs. However, despite its distinctive strengths, the radar has been sidelined for artificial intelligence (AI)-driven perception tasks. This dearth of exploration is attributed to several factors, including the limited availability of radar datasets, the inability of radar to capture

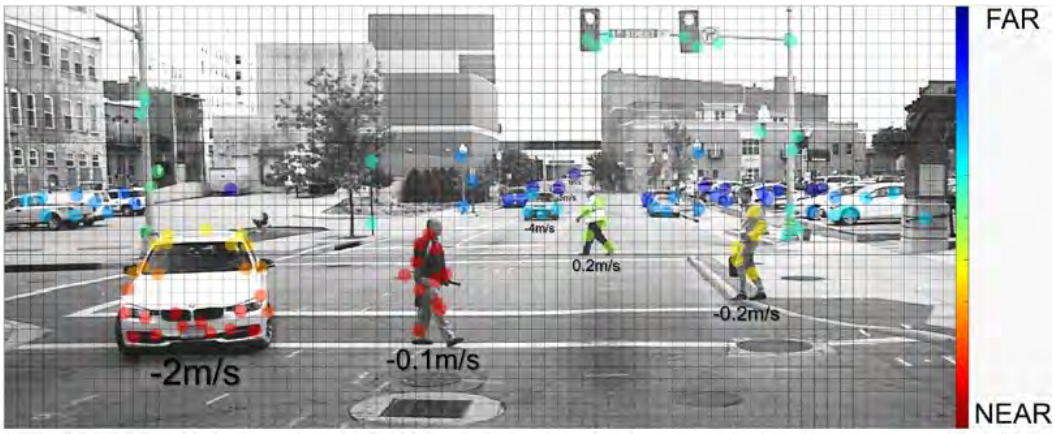


Figure 8: Example of radar point clouds. Source: [Gu 2022]

colour information, its limited angular resolution compared to camera and LiDAR sensors and the inherent challenges of processing such data using AI.

Focusing on the radar sensors, this thesis aims to bridge the gap between automotive radar technology and AI-driven perception. In its current form, radar data consists of a list of targets (also known as radar point clouds), which contains information about the position of the target, its velocity and a notion of radar cross section (RCS) characterising the target (see Figure 8). However, radar point clouds require significant pre and post-processing steps before being used by AI models. Also, these processing steps filter the raw signal reflected by objects, which can affect the performance of artificial intelligence algorithms. One alternative to point clouds consists of representing the signal reflected by the objects as a spectrum which represents the environment in distance and velocity (range-Doppler, RD), distance and angle (range-angle, RA), or range, angle and velocity (range-angle-Doppler, RAD).

This thesis proposes leveraging radar spectrum representations to detect and identify road-user objects in complex environments. In essence, this thesis aims to propose deep learning algorithms tailored explicitly for radar data and study if those algorithms can substitute conventional radar processing steps. This thesis was conducted at the ANITI artificial intelligence institute, in collaboration with the semiconductors company NXP, a world leader in automotive radar transceivers and microcontrollers. The company is actively involved in building next-generation radar for enhancing road safety and increasing driver convenience. In this context, this thesis also serves as a pioneering step in designing AI-enabled radar transceivers and microcontrollers. The algorithms proposed in this thesis will have to meet the constraints of the automotive environment: low energy consumption, low complexity and fast reaction time.

Thesis overview

First, we give an introduction to automotive radar, its role in ADAS systems and its limitations in **Chapter 1**. Then **Chapter 2** gives an overview of prior works to this thesis. Particularly, we review the literature on AI for automotive radar perception and the limits of current methods.

Second, we investigate similarities between an image-based object detection model and radar data in its range-Doppler representation. Radar signal can be transformed into an image-like spectrum containing the objects' position and speed but remains different from camera images. Regarding the data's small size and the raw radar dataset's limited size, it is challenging to use a computer vision model directly on it. We propose in **Chapter 3** to adapt a Faster R-CNN [Ren 2017] model for radar object detection using range-Doppler data. We study the distinctive characteristics of range-Doppler data compared to camera images, and we propose a lightweight radar-specific feature extractor to detect objects in range-Doppler view. We then compare this approach to traditional radar object detectors.

Third, we tackle the problem of online object detection for radar. Time is crucial information for perception. For example, it allows for exploiting correlations between objects in successive frames. In radar, exploiting the time is crucial as an object's signature evolves depending on many factors like the distance, the angle of arrival and the object's class. In **Chapter 4**, we propose a model leveraging convolutions and convolutional recurrent neural networks for online radar object detection. The proposed model learns spatio-temporal features from several types of radar data (range-Doppler, range-azimuth or range-azimuth-Doppler) and can perform different perception tasks, ranging from object detection to semantic segmentation. Finally, as our model aims to operate in a computationally constrained environment, we propose an efficient model with few parameters and operations.

In deep learning, data annotation is a key parameter to succeed in learning meaningful representations for object detection, semantic segmentation or classification. However, radar data is not human-friendly; therefore, it is challenging to label it. In **Chapter 5**, we propose a preliminary work to learn with less data. We leverage self-supervised learning frameworks from computer vision and radar knowledge, and we propose a pre-training strategy to reduce the labelling effort and train models with less data.

For each of the works presented in this thesis, aiming to bridge the gap between automotive radar technology and AI-driven perception, we present the limitations and

the general perspectives to design future generations of radar perception algorithms.

Publications

Decourt, Colin, Rufin VanRullen, Didier Salle, and Thomas Oberlin. **"DAROD: A Deep Automotive Radar Object Detector on Range-Doppler Maps."** In 2022 IEEE Intelligent Vehicles Symposium (IV), pp. 112-118. IEEE, 2022.

Decourt, Colin, Rufin VanRullen, Didier Salle, and Thomas Oberlin. **"A Recurrent CNN for Online Object Detection on Raw Radar Frames."** In arXiv preprint arXiv:2212.11172 (2022). (Under review, IEEE Transactions on Intelligent Transportation Systems)

Chapter 1

Introduction to automotive radar

Contents

1.1 Radar in ADAS systems

In the quest for safer and more autonomous transportation systems, advanced driver assistance systems (ADAS) have become increasingly important in our lives, with a growing number of vehicles being equipped with these advanced technologies. In 2021, approximately 33% of new vehicles in the United States, Europe, Japan and China had ADAS features. In 2030, 50% of new vehicles are expected to be ADAS-enabled [Nagpal 2022].

ADAS refers to a range of technologies and features designed to assist drivers in operating their vehicles [Galvani 2019]. It provides safety and convenience through various functionalities such as collision avoidance, lane departure warning, adaptive cruise control and automated parking. These systems rely on sensors, such as radar, cameras, lidar and V2X (Vehicle-to-Everything) to gather and interpret real-time

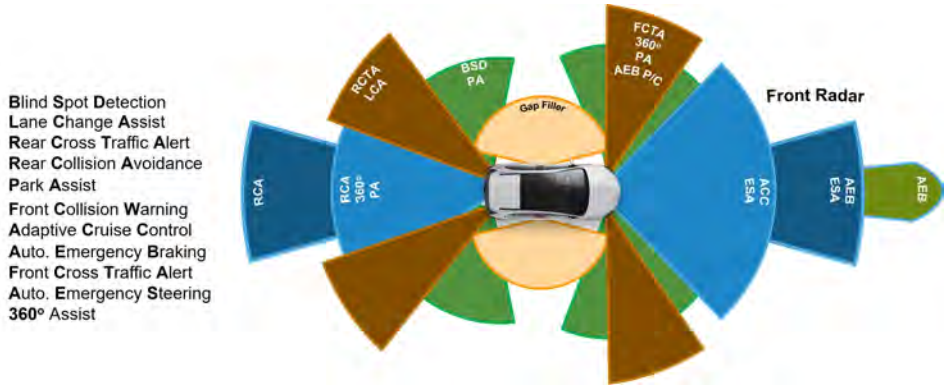


Figure 1.1: Type of radars found in ADAS systems. Different types of radars (short range, long range, corner), with different functions are arranged all around the vehicle. Source: NXP Semiconductors

data about the vehicle's surroundings. As most road crashes come from human error, ADAS enables proactive actions to enhance safety and improve the driving experience [Brookhuis 2001].

Among camera and LiDAR sensors, radar sensors are good candidates for ADAS applications as they bring complementary information to other sensors. In particular, because radar sensors emit electromagnetic waves, they can operate in difficult weather conditions (night, fog, snow, dust, and intense light). Also, they allow more accurate distance and velocity estimation in a single capture compared to camera and LiDAR sensors. Therefore, radar sensors are appropriate for, but are not limited to:

- Blind Spot Detection (BSD)
- Lane Change Assist (LCA)
- Adaptive Cruise Control (ACC)
- Automatic Emergency Braking (AEB)

When arranged all around the vehicle, and combined with other sensors, radars allow creating a 360-degree safety cocoon as shown in Figure 1.1.

1.2 Radar principle

Radar (RAdio Detection and Ranging) is an active sensor that transmits radio frequency (RF) electromagnetic (EM) waves and uses the reflected waves from objects to estimate the distance, velocity, and azimuth and elevation angles of these targets [Patole 2017]. Figure 1.2 illustrates the principle of radar. Radars can operate

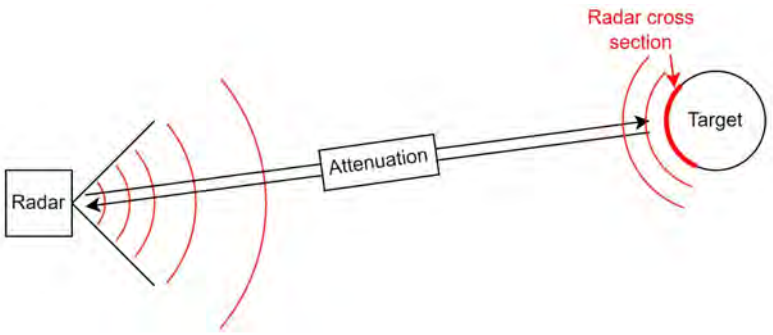


Figure 1.2: Principle of a radar. A radar emits electromagnetic waves and uses the reflected waves from objects to estimate the distance, velocity, and azimuth and elevation angles.

in frequency bands ranging from 3MHz to 300GHz. Automotive radar systems generally use 24GHz or 77GHz bands to achieve high velocity and range resolution.

Distance, speed and direction of arrival are estimated by computing the difference between the emitted and received signals. Targets can be detected depending on their radar cross section (RCS). The RCS characterises the target and measures how detectable an object is by the radar. The RCS is expressed in m^2 or $dB{\cdot}m^2$. For example, the RCS of a pedestrian, a car and a bicycle are around 1 m^2, 10 m^2 and 2 m^2 respectively [Richards 2005]. Nevertheless, this varies greatly depending on the target's orientation or distance from the radar. The radar equation 1.1 shows how the range affects the power reflected by a target.

1.2.1 The radar equation

Considering P_t, the nominal transmit power, and a target at a distance R, then the received power is related to the transmit power of a radar:

$$P_r = \frac{P_t G \sigma \lambda^2}{(4\pi)^3 R^4} \tag{1.1}$$

where G is the antennas gain, σ is the RCS of the target, and λ is the wavelength of the emitted EM wave. Equation 1.1 is known as the **radar equation** [Richards 2005]. For automotive radar, G, P_t and λ vary little. Hence the power received back from a target depends on its RCS σ and decreases proportionally to R^4. The radar equation determines the maximum range R_{max} (in meter) of radar for a given target RCS:

$$R_{max} = \sqrt[4]{\frac{P_t G \sigma \lambda^2}{P_{r,min}(4\pi)^3}} \tag{1.2}$$

	LRR (front radar)	MRR (corner radar)	SRR
Distance range ($R_{min} - R_{max}$)	10-250 m	1-100m	0.15-30m
Range resolution (δ_r)	0.5m	0.5m	0.1m
Range accuracy (Δ_r)	0.1m	0.1m	0.02m
Azimuth field of view	± 15°	± 40°	± 80°
Angular accuracy (Δ_ϕ)	0.1°	0.5°	1°
Bandwidth	600MHz	600MHz	4GHz

Table 1.1: Automotive radar sensors classification and their associated characteristics [Hasch 2012]

Where $P_{r,min}$ (W) is the smallest perceivable power.

1.2.2 Automotive radar classification and waveform

Depending on the application (see Figure 1.1) and the waveform used, automotive radar exhibits different characteristics. These characteristics include:

- Maximum range (R_{max}): the maximum range at which a target can be detected.
- Maximum speed (v_{max}): the maximum non-ambiguous speed the radar can detect. Targets with a relative speed v_r higher than v_{max} will be detected but their speed will be incorrect.
- Range resolution (δ_r): how close in range can two objects of equal strength be and theoretically still be detected as two objects.
- Speed resolution (δ_v): how close in velocity can two objects of equal strength be and theoretically still be detected as two objects.
- Accuracy: how precisely the measurement can be made. Accuracy informs about the uncertainty about the real position of target. We refer the range and the angular accuracy to as Δ_r and Δ_ϕ respectively.

According to the characteristics mentioned above, automotive radars can be classified in three different categories: short-range radar (SSR), medium-range radar (MRR , or **corner radar**), and long-range radar (LRR, or **front radar**). Short-range radars require higher range resolution, accuracy and field of view than long-range radars, as they must detect objects close to the car. However, the different categories of radar are vagueness, evolve and need to be standardised in the radar marketplace. Therefore, Table 1.1 gives a rough estimate of the characteristics of the different types of radar sensors.

Automotive radar classes, summarised in Table 1.1, present diverse specifications in terms of several fundamental radar system performance metrics (range and velocity resolution, angular direction, SNR (signal-to-noise ratio) [Patole 2017]. The type of waveform the radar emits also affects these metrics. Automotive radar waveforms are either continuous waves (CW) or pulsed and modulated in frequency or phase. Modulated radar waveforms include FMCW (Frequency Modulated Continuous Wave), PMCW (Pulsed Modulated Continuous Wave), SFCW (Stepped Frequency Continuous Wave) or OFDM (Orthogonal Frequency-Division Multiplexing). Each waveform type has its advantages and limitations in processing, implementation, and performance.

Today, most automotive radars use FMCW waveforms. FMCW radars transmit periodic wideband frequency-modulated pulses, whose angular frequency increases linearly during the pulse (also known as a **chirp**). While simple CW waveforms can determine velocity but cannot determine target range, FMCW radars allow simultaneous range and velocity estimation with high range and velocity resolution. The ADC (Analog to Digital Converter) is an expensive piece of hardware in radar. To satisfy the Nyquist-Shannon theorem, the sampling rate (f_s) of the ADC must satisfy:

$$B < \frac{f_s}{2} \tag{1.3}$$

where B is the desired detectable frequency band. Because FMCW radars use a narrow band, they require a low sampling rate (around 80 MHz for LLRs) compared to PMCW and OFDM radars. Thus, they are cheaper and are preferred for automotive applications.

1.3 FMCW automotive radar

1.3.1 FMCW radar system

System overview Figure 1.3 illustrates a 1Tx-1Rx, FMCW radar system, *i.e.* a system with one emitting (Tx) and one receiving (Rx) antenna. Chirps are emitted through an Tx antenna. The receiving Rx antenna receives the signal reflected by targets in the radar's field of view. Then, a mixer multiplies the sent and the received signals to produce a low-frequency beat signal, whose frequency gives the target range. A low-pass filter is used to filter out unwanted high frequencies. The ADC digitises the signal at periodic sampling intervals during each chirp. Finally, signal processing is performed to obtain radar point clouds.

Transmitted FMCW signal An FMCW radar periodically transmits P chirps over N_{Tx} transmitting antennas to estimate the range, the velocity and the DoA of

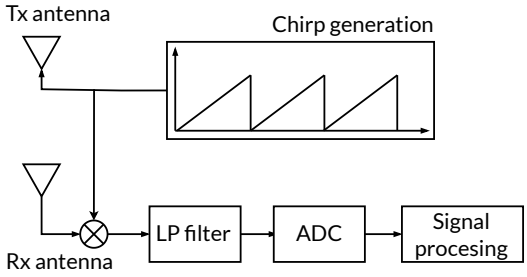


Figure 1.3: FMCW radar system overview. Chirps are emitted through an antenna. The receiving antenna receives the signal reflected by targets in the radar's field of view. Then, a mixer multiplies the sent and the received signals to produce a low-frequency beat signal, whose frequency gives the target range. A low-pass filter is used to filter out unwanted high frequencies. The ADC digitises the signal at periodic sampling intervals during each chirp. Finally, signal processing is performed to obtain radar point clouds.

the targets. For n^{th} antenna, we express the emitted signal as:

$$s(t) = A_t \exp j2\pi(f_c + \frac{B}{2T}t)t \tag{1.4}$$

$$= A_t \exp j(2\pi f_c t + \pi K t^2) \quad 0 \leq t \leq T_0 \tag{1.5}$$

$$= A_t \exp j\Phi(t) \tag{1.6}$$

where $K = \frac{B}{2T}$, f_c is the carrier frequency, B is the bandwidth of the signal, T_0 is the duration of the chirp (fast time), T is the pulse period, and A_r is the amplitude related to the transmit power. Figure 1.4 depicts one chirp profile and its parameters. As shown in Figure 1.4, different parameters define the FMCW signal:

- f_c: the starting frequency of the chirp (76-81 GHz)
- B: the bandwidth of the chirp (hundreds of MHz)
- T_{dwell}: a pause time between chirps (few μs)
- T_{settle}: the time for the ramp to be linear. During this phase, there is no acquisition by the ADC.
- T_{FFT}: the time during which the ADC acquires the data (tens of μs)
- T_{reset}: the time needed for the ramp generator to reset before the next chirp (few μs)
- T_{ramp}: $T_{settle} + T_{FFT}$, the total time of the ramp. Also referred to as T_0 in Equation 1.5.

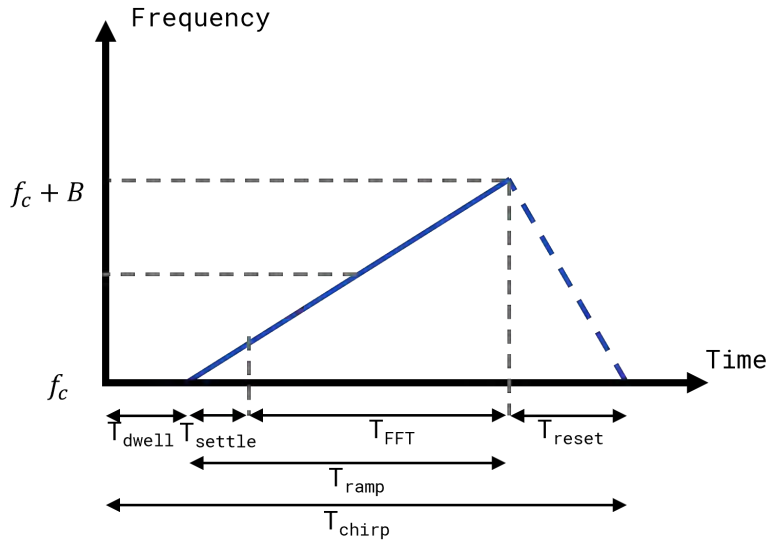


Figure 1.4: FMCW radar chirp and parameters.

- T_{chirp}: $T_{ramp} + T_{reset} + T_{dweel}$, the total time of the chirp. Also referred to as T in Equation 1.5.

- N_{chirp}: the total number of chirps emitted within a frame, or sequence of chirps (usually 256, 512 or 1024).

- T_{frame}: the duration of a radar frame. It comprises the emission of N_{chirp} of duration t_{chirp} and the processing time. $T_{frame} = N_{chirp} * t_{chirp}$

Received signal The signal $r(t)$ received at time t from a single reflector at radar range $r = c\tau$ is related to the signal transmitted at time $t - 2\tau$ earlier as:

$$r(t) = A_r \exp j(2\pi f_c(t - 2\tau) + \pi K(t - 2\tau)^2), \quad 0 \leq t \leq T_0 \tag{1.7}$$

where c is the speed of light, and A_r is the amplitude of the received signal.

After mixing (IF block, Figure 1.3), the mixed signal for a **single chirp** duration (but this can be generalised to all chirps) is:

$$y(t) = s(t)r(t) \tag{1.8}$$

$$= A_t A_r \exp j(\Phi(t) - \Phi(t - 2\tau)), \quad 0 \leq t \leq T_0 \tag{1.9}$$

This mixed signal has components at the sum and difference frequencies of the two signals. After filtering the sum of the frequencies (which are outside the receiver's

bandwidth), the beat signal $x(t)$ can be expressed as follows:

$$x(t) = A \exp j(4\pi\tau K t + 4\pi f_c \tau) \tag{1.10}$$

$$= A \exp j(4\pi(\frac{r}{c})K t + 4\pi f_c(\frac{r}{c})) \tag{1.11}$$

$$= A \exp j(2\pi f_b(r)t + \psi(r)) \tag{1.12}$$

where A is a modulation constant (which is related to A_r and A_t). Equating Equations 1.11 and 1.12, we have the beat frequency f_b of the IF signal:

$$f_b(r) = \frac{2K}{c} r \tag{1.13}$$

and the phase of the IF signal:

$$\psi(r) = \frac{4\pi f_c}{c} r = \frac{4\pi}{\lambda} r \tag{1.14}$$

We can see that the phase and the beat frequency of the IF signal are range dependent. While the beat frequency allows distance measurement, we will see in Section 1.3.3 the phase variation provides an exquisitely sensitive measure of range variation which is used in Doppler processing over multiple chirps in the frame. Finally, we can express the sampled ADC output $x[j]$ at ADC sample j within a chirp from a target at range r by setting $t = \frac{j}{f_{ADC}}$:

$$x[j] = A \exp j(2\pi f_b(r)(\frac{j}{f_{ADC}}) + \psi(r)) \tag{1.15}$$

where f_{ADC} is the sampling frequency of the ADC. For each chirp, the ADC samples the signal at periodic intervals to obtain a grid-like representation of the signal as shown in Figure 1.6.

Figure 1.5: Radar signal processing chain. First, the received signal is converted from the time domain to the frequency domain to extract distance and velocity information. An object detector is applied to find the range and velocity bins where there are objects. Then, the azimuth (and the elevation) of objects is estimated. Finally, some post-processing steps are applied to output the final target list.

1.3.2 Radar signal processing chain

Figure 1.5 gives a simplified view of the signal processing block of Figure 1.3. We refer to this as the *radar signal processing chain* (RSP). After being reflected by the target, the signal is sampled at even intervals during each chirp as illustrated in Figure 1.6(1) and a windowing operation is applied. This corresponds to the signal conditioning block.

Following the signal conditioning, the signal analysis block estimates the distance and the velocity of objects in the radar's field of view. A 2D Fast Fourier Transform (FFT) (range and Doppler FFTs) is applied to the received signal to measure the difference in frequency and phase between the emitted and received waves (therefore, the distance and the velocity of an object). We go into the details of it in Section 1.3.3. Then direction of arrival (DoA) estimation techniques (a 3rd FFT for example) are applied to estimate the angle of arrival (AoA) of the target. In the case of Multiple Input Multiple Output (MIMO) radars, MIMO demodulation is applied before the DoA. We explain DoA estimation and MIMO radar in Section 1.3.5.

In practice, a peak detector such as the Constant False Alarm Rate algorithm (CFAR) [Blake 1988] is applied after the Doppler FFT to detect potential targets. This is because computing the 3rd FFT on all the range and velocity bins, and antennas is computationally demanding. For each potential target, the DoA is estimated to save computation. We detail the CFAR principle in Section 1.3.4.

Once targets are detected, post-processing steps are applied. It can be super-resolution algorithms (MUSIC [Schmidt 1986], ESPRIT [Paulraj 1985]) to estimate better the DoA of targets, clustering (DBSCAN [Ester 1996], K-means [Lloyd 1982]), target tracking (using Kalman filtering [Kalman 1960, Bertozzi 2004]) and classification [Rohling 2010, Yamada 2005, Gavrila 2001].

In the following sections, we give details of range, velocity and angle estimation, target detection and post-processing.

1.3.3 Range and velocity estimation

Range estimation We saw in Section 1.3.1 the range of an object can be easily determined from the beat frequency f_b as:

$$r = \frac{cf_b(r)}{2K} \tag{1.16}$$

The beat frequency is the difference in frequency between the emitted and the received signals. Because of the linearity of the frequency variation of the chirp, this difference in frequency is constant and proportional to the delay τ between the emitted and received signals as shown in Figure 1.6(1). This difference in frequency

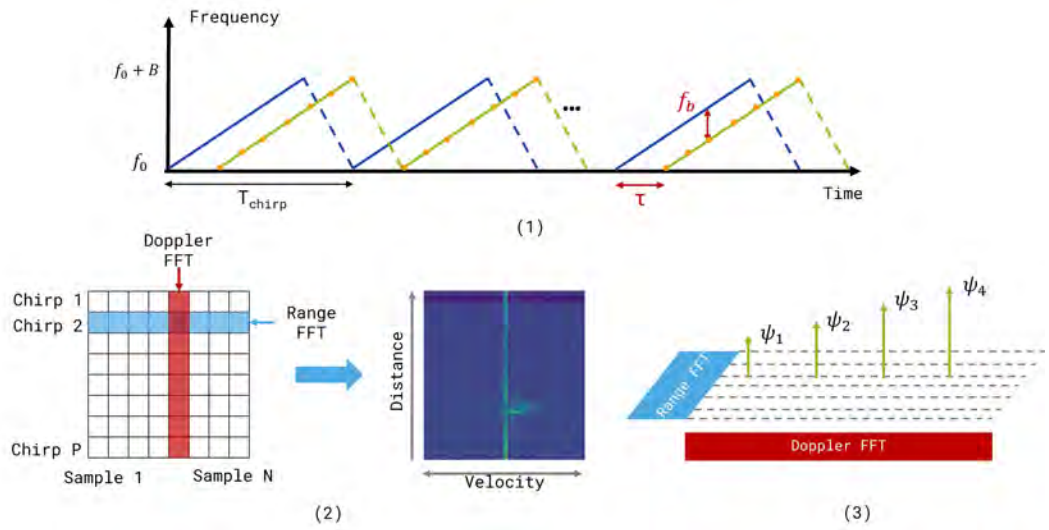


Figure 1.6: Range and Doppler radar processing. **(1)** A spectrogram of an FMCW waveform with carrier frequency f_c and bandwidth B. The emitted signal is in blue, and the received signal is in green. Orange points correspond to the points sampled by the ADC. **(2)** The ADC matrix after sampling and the corresponding Range-Doppler spectrum. Range FFT is first applied for every chirp. Doppler FFT is then applied for each chirp index (or sample). **(3)** Illustration of the phase of the range FFT that evolves according to the relative velocity.

can be precisely measured using an FFT for every chirp (fast-time index) to obtain the range spectrum. We call this operation the range FFT (Figure 1.6(2)). The beat frequency $f_b[m]$ (in Hz) corresponding to a peak in bin m in the M point range FFT sampled at f_{ADC} is:

$$f_b[m] = \frac{m f_{ADC}}{M}, \text{ for } 0 \leq m \leq \frac{M}{2} \tag{1.17}$$

The range of individual objects $r[m]$ can be computed from the beat frequencies $f_b[m]$ present in peaks of the range spectrum:

$$r[m] = \frac{c f_b[m]}{2K} = \frac{cm f_{ADC}}{2MK}, \text{ for } 0 \leq m \leq \frac{M}{2} \tag{1.18}$$

The constraint $0 \leq m \leq \frac{M}{2}$ restricts the range calculation to positive frequencies and ranges. Negative ranges are meaningless and cannot be computed since the ADC data is real.

Range limit and resolution For a given ADC sampling frequency f_{ADC}, the maximum range of the radar is inversely proportional to the slope of the chirp and

appears at the Nyquist bin $m = \frac{M}{2}$:

$$R_{max,ADC} = \frac{cT_{ramp}f_{s,ADC}}{4B} = \frac{cf_{ADC}}{4K} \tag{1.19}$$

The bin spacing in a M point range FFT is an estimate of the range resolution δ_r:

$$\delta_r = \frac{cT_{ramp}}{2BT_{chirp}} = \frac{cf_{ADC}}{2MK} \tag{1.20}$$

Velocity estimation The velocity estimation is based on a phenomenon called the Doppler effect. Suppose a target is moving ahead from the radar with a velocity v. With a relative motion between the target and the radar, the reflected waves are now delayed by time $\tau = \frac{R \pm vt}{c}$. The time dependant delay term causes a frequency shift in the received wave known as the *Doppler shift*:

$$f_d = \frac{\pm 2v}{c} \tag{1.21}$$

The beat signal $x(t)$ in Equation 1.12 now includes the Doppler shift:

$$x(t) = A \exp j(2\pi(f_b(r) + f_d) + \psi(r)) \tag{1.22}$$

The beat frequency f_b now depends on the target range r and the target's relative speed $v = \frac{\partial r}{\partial t}$. The two cannot be separated using the beat frequency of a single pulse. Instead, multiple chirps are used to estimate the velocity. For the same sample, the small target's variation in distance will slightly change the phase $\psi(r)$ between two chirps. This phase appears in the phase of the range FFT bin as shown in Figure 1.6(3). Differentiating Equation 1.14 with respect to time gives:

$$\frac{d\psi(r)}{dt} = \frac{4\pi f_c}{c}\frac{dr}{dt} = \frac{4\pi f_c v}{c} \tag{1.23}$$

where v is the reflector's radial velocity. Therefore, by applying a FFT on P points (the number of emitted chirps) for each chirp index (the fast time), we can estimate the velocity of a target at range r. We call this operation the Doppler FFT, and this operation results in a **range-Doppler (RD) spectrum** (or map). The target velocity $v[p]$ corresponding to a peak in the Doppler FFT bin p is:

$$\begin{cases} v[p] & = \frac{pcf_{PRF}}{2Nf_c}, \text{ for } 0 \leq p \leq \frac{P}{2} \\ v[p] & = \frac{(p-P)cf_{PRF}}{2Nf_c}, \text{ for } \frac{P}{2} \leq p \leq N \end{cases} \tag{1.24}$$

where $f_{PRF} = \frac{1}{T_{chirp}}$ is the pulse repetition frequency (PRF). If there is a target at range bin m and Doppler bin pn then, a peak in the RD map at position (m, p) can be seen, as shown in Figure 1.6.1

Velocity limit and resolution The maximum target velocity (in m/s) is a function of the PRF:

$$v_{max,PRF} = \frac{\lambda}{4T_{chirp}} = \frac{cf_{PRF}}{4f_c} \tag{1.25}$$

The bin spacing in a N point Doppler FFT is an estimate of the velocity resolution δ_v:

$$\delta_v = \frac{c}{2f_c N_{chirp} T_{chirp}} = \frac{cf_{PRF}}{2f_c N_{chirp}} \tag{1.26}$$

1.3.4 Target detection

Figure 1.8 shows the procedure to compute the DoA of targets after range and velocity estimation. Before estimating the DoA of targets, one must first detect potential targets in the radar's field of view. Recall that a potential target corresponds to a peak in the range-Doppler spectrum. Most FMCW radars apply a Constant False Alarm Rate [Blake 1988] algorithm to detect peaks in the RD spectrum. CFAR automatically adapts the threshold to keep the false alarm rate at the desired level. Therefore, it will also adapt the probability of detection.

The most common CFAR detector is the cell-averaging detector (CA-CFAR) [Rohling 1983]. First, the RD map is divided into a grid of cells; each cell contains information about the radar reflections at a specific range and velocity. For each cell in the map (cell under test, CUT), the noise is estimated using a 2D sliding window:

$$P_n = \frac{1}{N+M} \sum_{j=1}^{N} \sum_{k=1}^{M} x_{jk} \tag{1.27}$$

where $N+M$ are the number of training cells in the 2D window and x_{jk} are the cells in the window. Figure 1.7 shows the 2D window of a CFAR algorithm. Generally, guard cells are placed adjacent to the CUT to prevent signal components from leaking into the training cells, which could affect the noise estimate. Then, the threshold factor can be written as [Richards 2005]:

$$\alpha = (N+M)(P_{fa}^{-\frac{1}{N+M}} - 1) \tag{1.28}$$

where P_{fa} is the desired alarm rate (set empirically). The detection threshold is set as:

$$T = \alpha P_n \tag{1.29}$$

If the value of the CUT is higher than T, there is a potential object at the cell coordinate. The coordinate is kept in memory for further processing. It is also possible to represent the output of the threshold operation as a binary detection mask.

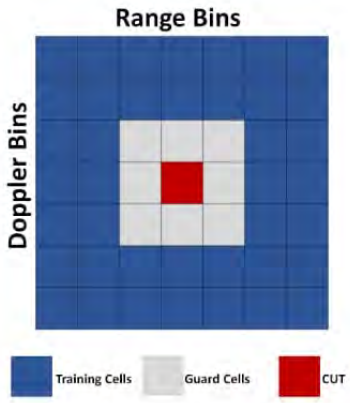


Figure 1.7: 2D CFAR window. Source: [Hameed 2022]

1.3.5 Direction of arrival estimation

Basic angle estimation Estimating the angle of arrival of an object requires at least two Rx antennas. Figure 1.9 shows a radar that has one Tx and four Rx antennas separated by a distance d (Single Input Multiple Output, SIMO). The signal emitted by the Tx antennas is reflected and received by all the Rx antennas.

However, the signal from the object must travel an additional distance of $d \sin\theta$ to reach the second R_x antenna. This corresponds to a phase difference of $\omega = \frac{2\pi}{\lambda} d \sin\theta$ between the signals received between the first and the second Rx antennas. For each subsequent antenna, an additional phase-shift ω with respect to the preceding antennas is added. This results in a linear progression in the phase of the signal across the N antennas [Rao 2018] (for example, $[0, \omega, 2\omega, 3\omega]$). Similarly to the range FFT, the differential distance from the object to each antenna results in a phase change in the Doppler FFT peak. Therefore, ω (so the angle) can be estimated by performing an FFT across the N_{Rx} antennas (thus, the N_{Rx} RD maps). We refer to this operation as the angle FFT. This results in a novel radar representation, namely the **Range-Angle-Doppler** (**RAD**) **map**. We sum-

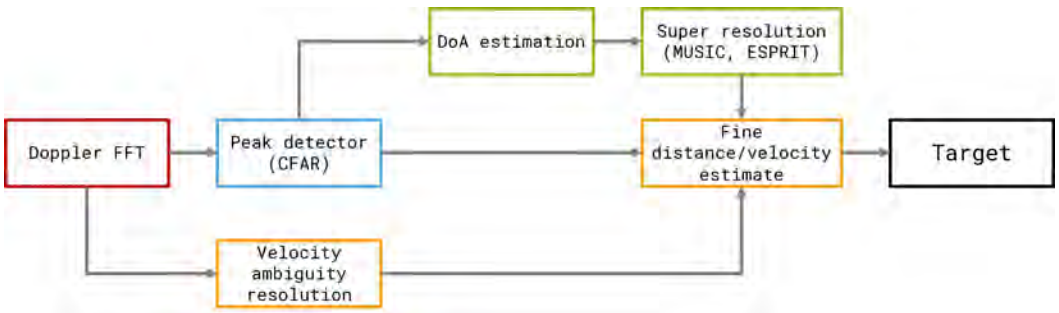


Figure 1.8: DoA estimation in radar.

marise the processing in Figure 1.10. Summing along the Doppler axis results in a **Range-Angle (or range-azimuth, RA) spectrum**.

In practice, the angle FFT is computed across the N_{Rx} antennas only on the range and the Doppler bins where an object is detected, as shown in Figure 1.8. To enhance the spatial resolution of the radar, super-resolution algorithms (MUSIC [Schmidt 1986], ESPRIT [Paulraj 1985]) can be used. Super-resolution algorithms use multiple radar measurements of an object and fuse them to estimate its position accurately.

Introduction to MIMO radars Increasing the number of antennas results in an angle FFT with more points, improving the angle estimation accuracy and enhancing the angle resolution. Nevertheless, adding an infinite number of receiving antennas is not feasible in practice.

Imagine we want to double the angle resolution of our radar in Figure 1.9. One way to do this would be doubling the number of RX antennas, which would bring us to add four antennas. Instead, another way to achieve the same configuration is to add one Tx, as shown in Figure 1.11.

As for the SIMO case, transmission from Tx_1 results in a phase shift of $[0, \omega, 2\omega, 3\omega]$ at the four Rx antennas. Because the space between Tx_1 and Tx_2 equals $4d$, any signal from Tx_2 travels an additional distance of $4d \sin\theta$. Consequently, transmission from Tx_2 results in a phase shift of $[4\omega, 5\omega, 6\omega, 7\omega]$. Gathering the RD maps from all the virtual antennas and applying an FFT across the virtual antennas allows us to estimate ω, hence the angle of arrival.

Generally, with N_{Tx} and N_{Rx} antennas, it is possible to generate a virtual antenna array of $N_{Tx} \times N_{Rx}$. Employing MIMO techniques results in a multiplicative increase in the number of (virtual) antennas [Rao 2018].

MIMO modulation All Rx antennas must be able to separate the signals corresponding to different Tx antennas. One way of doing it is to have the T_x trans-

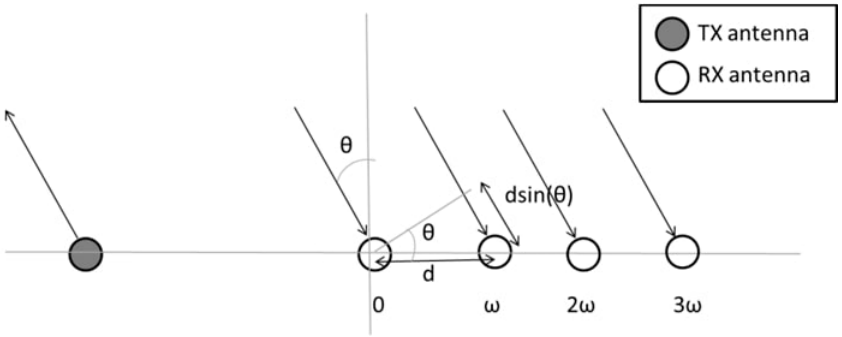


Figure 1.9: $1Tx$-$4Rx$ radar. Source: [Rao 2018]

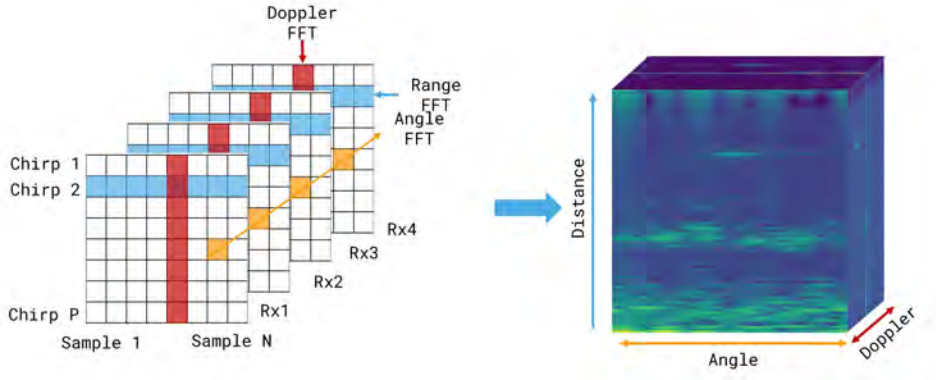


Figure 1.10: Range-Angle-Doppler FFT processing for 4 *Rx* antennas. ADC data from all the receiving antennas are sampled to create a cube of data. Then three FFTs are successively applied on the cube to obtain the RAD cube. In practice, the RAD cube is not used.

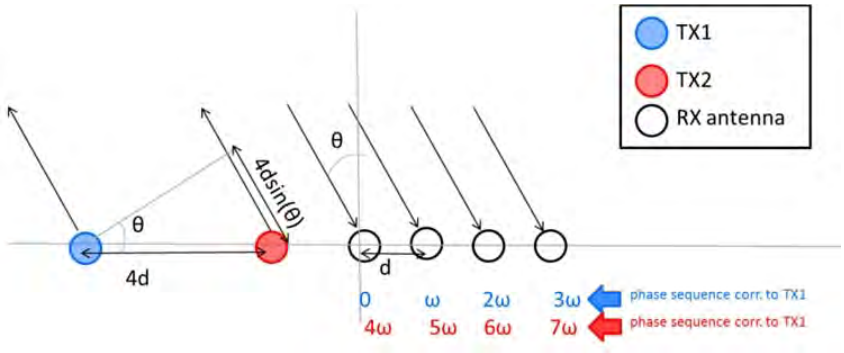


Figure 1.11: Principle of MIMO radar. Source: [Rao 2018]

mitting on orthogonal channels (see Figure 1.12(a)). From communication theory, there are three known ways to make signals orthogonal:

- Time: Each TX transmits signals one at time, using the same spectrum
- Frequency: TXs transmit at the same time, but using a central frequency shift big enough so that all their spectra don't overlap.
- Coding: TXs transmit at the same time and frequency, using orthogonal sequences.

Figure 1.12 show examples of MIMO modulated radar waveforms.

1.3.6 Post-processing steps

Figure 1.13 shows the post-processing steps required to obtain the final target lists. For safety, automotive radars require multiple measurements to confirm detection.

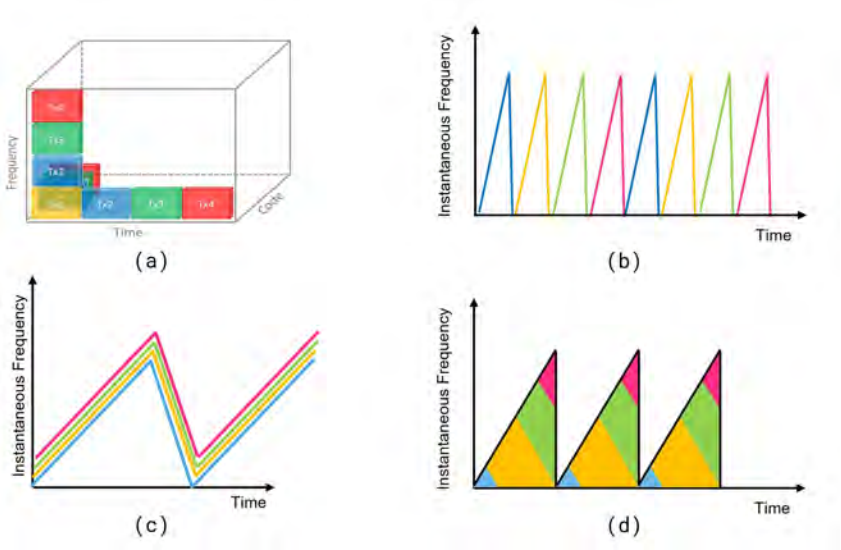


Figure 1.12: MIMO modulation strategies. (a) Orthogonality domains for Tx signals. (b) Time Division Multiple Access (TDMA) modulation. (c) Doppler Division Multiple Access (DDMA) modulation. (d) Code Division Multiple Access (CDMA) modulation.

Following the target detection (range and velocity estimation, peak detection and DoA estimation), targets are ego-motion compensated, clustered and tracked. For tracking, Kalman filters [Kalman 1960] are typically used. The tracking might be constrained to moving targets only; this is why targets are first classified as moving or static. Ego-velocity of the radar can be estimated based on the detections and used as a filter to separate still-standing and moving targets. Finally, the final target list consists of the target's position in cartesian coordinates (x, y), the radial velocity v_r, the RCS σ and the angle of arrival θ of the target.

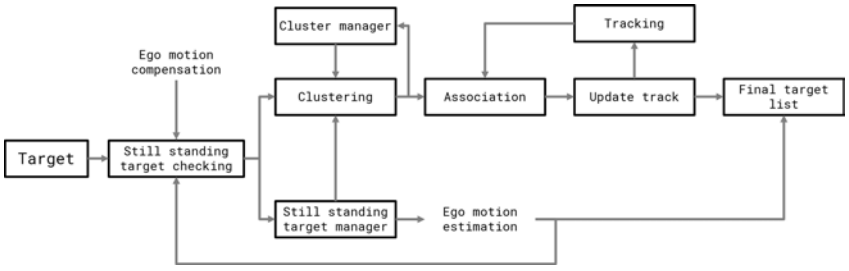


Figure 1.13: Simplified post-processing operations for target detection.

1.4 Limits of current radar systems

The target identification problem Although current radar systems work and are efficient, they present some limitations. First of all, current radar systems do not have classification ability. They can detect objects but cannot differentiate them (*e.g.*, is it a car or a pedestrian?). Thus, radar systems sometimes output undesirable targets that are unnecessary for ADAS tasks. In a future where vehicles are expected to be autonomous, identifying targets is mandatory to make decisions. Suppose the camera or the LiDAR are deficient, and the system must use the radar as a backup sensor. In such specific cases, the radar is expected to output reliable predictions about the position and the class of the object to help the system to make a decision. This thesis explores the possibility of using radar data for object classification using AI.

The complexity of the radar signal processing chain Second, as shown in Figures 1.8 and 1.13, there are many post-processing steps before obtaining the target list. However, CFAR, Kalman filtering, or clustering algorithms depend on many hyper-parameters that affect their performance. Moreover, all these filtering techniques produce very sparse target lists. Post-processing operations ranging from CFAR to Kalman filtering often reduce a target to a few points. At the same time, the spectrum (RD, RA or RAD) contains much more valuable information about the environment. Indeed, the RAD cube is a kind of 3D representation of the radar's field of view. For that reason, this thesis aims to leverage the potential of deep learning to reduce the complexity of the radar signal processing chain and add classification ability to radar. Considering the raw data (RD, RA or RAD maps), we study deep learning-based object detection and segmentation algorithm to detect and identify targets using only raw radar data. This work has been made possible thanks to the release of raw data datasets such as CARRADA [Ouaknine 2021b]. As an example, prior to our work, Gao *et al.* [Gao 2019b] and Akita *et al.* [Akita 2019] show the potential of deep learning for target recognition. Also, Fatseas and Bekooij [Fatseas 2019] use object detection algorithm and Kalman filtering to detect, identify and track targets using only the RD spectrum.

Chapter 2

Related work

Contents

This chapter gives an overview of the prior work to this thesis. First, we present deep learning and computer vision fundamentals. Second, we review the literature for automotive radar perception. In this thesis, we use the term *perception* for computer vision applied to radar because it acknowledges the broader process of acquiring and interpreting sensory information, regardless of the sensing modality involved, while computer vision traditionally focuses on visual data analysis.

In Sections 2.1 and 2.2, we present deep learning background and some models for object detection and segmentation in computer vision that are useful for this thesis. In Section 2.3, we present available automotive datasets for radar perception. In Section 2.4 and 2.5, we give an overview about the literature on radar perception for radar point clouds and raw data, respectively.

2.1 Deep learning background

Machine learning and deep learning Deep learning refers to a subset of machine learning which is a subset of artificial intelligence. A machine learning al-

gorithm is an algorithm that can learn from data. According to [Mitchell 1997]: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.". While traditional machine learning learns a mapping from hand-crafted representation (features) to a specific output, deep learning learns not only a mapping from representation to output but also the representation itself.

In this thesis, task T consists of image classification, segmentation, and object detection. The performance measure P is specific to the task T. It is used to measure how well our algorithm performs on unseen data. The experience E is a set of data points with associated features, also known as a dataset. Each data point is associated with a label or target in a supervised setting. The target is used to teach the deep learning algorithm what to do.

Artificial neural networks An artificial neural network (ANN) is a computing system inspired by the biological neural networks that constitutes animal brains. The model can be represented with a directed acyclic graph of artificial *neurons* connected to each other [Goodfellow 2016]. A neuron is a function that takes a set of input signal $x = (x_1, x_2, ..., x_n)$ and outputs a single value y:

$$y = \sigma(\sum_{i=1}^{n} w_i x_i + b) \tag{2.1}$$

with w_i and b are the weights and the bias of the neuron and σ is an activation function. Typically σ is non-linear to allow the neuron to learn complex non-linear functions. The most popular activation functions are the sigmoid function $\sigma(a) = \frac{1}{1+\exp -a}$, the rectified linear unit (ReLU) $\sigma(a) = max(0, a)$ and the hyperbolic tangent function $\sigma(a) = tanh(a)$. Figure 2.1 depicts the computation of a neuron.

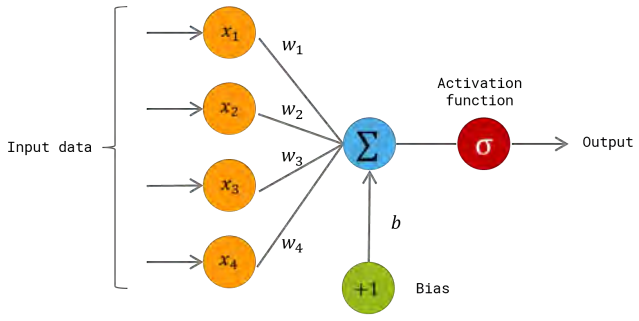


Figure 2.1: Artificial neural neuron. A neuron is a function that takes a set of input signal $x = (x_1, x_2, ..., x_n)$, computes a weighted sum of the input and apply an activation function to it to produce the output.

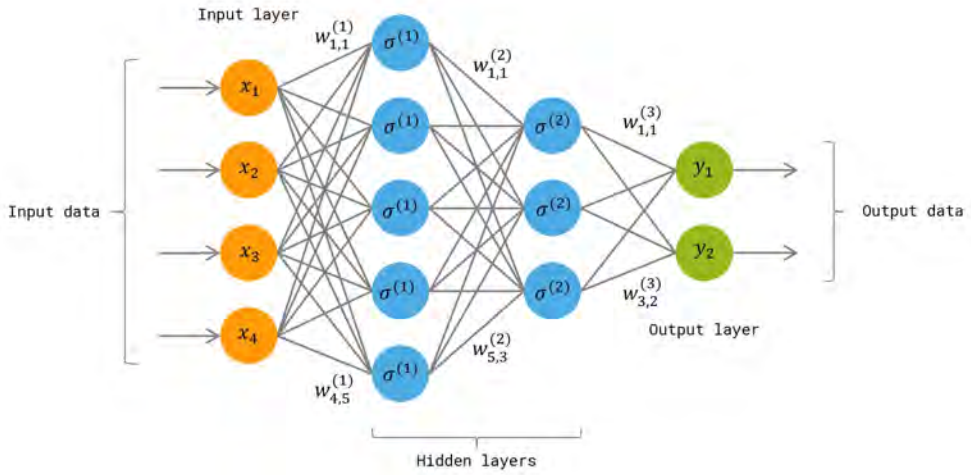


Figure 2.2: The multi-layer perceptron. Neurons are stacked together to form *layers*. The first layer is called the input layer. The last layer is called the output layer. Other layers are called hidden layers.

In an ANN, neurons are stacked together to form *layers*, and the network becomes a composition of layers. Since an artificial neural network is a composition of functions, it is a function. Because the information flows through the function evaluated from the inputs, intermediate functions and finally through the output, we also refer to ANN as feed-forward neural networks.

A common ANN is the multi-layer perceptron (MLP, Figure 2.2). The MLP stacks multiple layers of neurons together, where neurons of each layer are connected to other neurons belonging to different layers through edges with associated weights. The output of the i^{th} layer of an MLP is computed as:

$$y^{(i)} = \sigma^{(l)}(W^{(l)}y^{(i-1)} + b^{(i)}) \tag{2.2}$$

where $W^{(l)}$, $b^{(i)}$ and $\sigma^{(i)}$ are the weight matrix, the bias vector and the activation of the i^{th} layer respectively. Thus, the MLP is θ parameterised function defining a mapping $\mathbf{y} = f(\mathbf{x};\theta)$ where $\theta = ((W^{(i)}, b^{(i)}), ..., (W^{(1)}, b^{(1)}))$ are the parameters of the network, $\mathbf{x} = (x_1, x_2, ..., x_n)$ are the input values and $\mathbf{y} = (y_1, y_2, ..., y_m)$ are the output values of the network.

MLPs are not the only type of ANNs. In the following paragraphs we choose to present two types of neural networks, namely the convolutional neural networks and the recurrent neural networks. As an example we choose to leave aside the well known Transformer architecture [Vaswani 2017a].

Training a neural network Training a neural network consists of iteratively updating the weights θ of the network to minimise a cost function $J(\theta)$. In the

supervised training setting, for a given loss function $\mathcal{L}$, we minimise the empirical risk:

$$J(\theta) = \mathbb{E}_{x,y\sim\hat{p}_{data}(x,y)}[\mathcal{L}(f(x;\theta),y] = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(f(x^{(i)};\theta),y^{(i)}) \tag{2.3}$$

Where N is the number of training examples, and y is the target output. The minimisation process, the optimisation step, uses backpropagation and a gradient descent algorithm such as the Stochastic Gradient Descent (SGD) [Kiefer 1952, Robbins 1951].

Recurrent neural networks (RNN) Recurrent neural networks (or RNNs) [Rumelhart 1986] are a family of neural networks for processing sequential data (*e.g.* music, video, text) [Goodfellow 2016]. While a traditional feed-forward neural network would have separate parameters for each input feature, a RNN shares the same weights across several time steps. RNNs are called recurrent because they perform the same task for every sequence element. As show in Figure 2.3, a RNN takes as input an element of the sequence and outputs a hidden state and an activation vector. The hidden state acts as the *memory* of the RNN. It is updated based on the current input and the previous time step's hidden state. Let U, V, W be the shared weights of the RNN, b, c, two bias vectors, and σ_1 and σ_2 some activation functions. The traditional RNN is defined as:

$$h^{(t)} = \sigma_1(b + Vh^{(t-1)} + Ux^{(t)}) \tag{2.4}$$

$$\hat{y}^{(t)} = \sigma_2(c + Wh^{(t)}) \tag{2.5}$$

Where $\hat{y}^{(t)}$ is the prediction of the RNN at timestep t and $h^{(t)}$ is the hidden state at timestep t. RNNs are trained using backpropagation through time (BPTT). BPTT is a computational technique that allows for the efficient calculation of gradients by unfolding the network across time and propagating the errors backwards. However, traditional RNNs cannot handle long-term dependencies and suffer from the van-

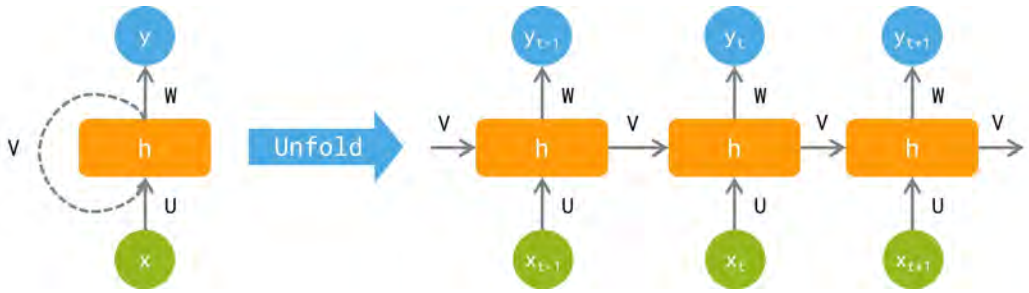


Figure 2.3: Left: a recurrent neural network. Right: Unfolded recurrent neural network.

ishing gradient problem. The vanishing gradient problem refers to the issue where the gradients become very small during backpropagation, hindering the training process and resulting in slow or ineffective convergence. Variants of RNNs have been proposed to solve this, such as Long-Short-Term-Memory networks (LSTMs) [Hochreiter 1997] and Gated Recurrent Unit (GRU) [Cho 2014]. LSTMs and GRUs add additional cells (called *gates*) to allow the gradient to flow through the network without vanishing.

Convolutional neural networks (CNN) Considering an input image with size $H \times W \times C$, we want to process with an MLP. In an MLP, every output unit interacts with every input unit (see Figure 2.2). This means a single MLP would require at least $H \times W \times C$ parameters to process each image pixel. Thus, MLPs cannot scale for high-dimensional data. Moreover, they cannot learn spatial features as the data is flattened. Convolutional neural networks [Lecun 1989] tackle this issue by replacing matrix multiplication with convolutions. For the 2D case, the convolution operation consists of applying a set of C 2D kernels $K \in \mathbb{R}^{k_1 \times k_2 \times C}$ over an input image (or a feature maps) $I \in \mathbb{R}^{H \times W \times C}$ such as:

$$(K * I)(h, w) = \sum_i \sum_j \sum_c I(i, j, c) K(h - i, w - j, c) \tag{2.6}$$

Where $h, w \in \mathbb{N}$ defines the coordinates in the image (or the feature maps). CNNs show interesting properties for grid-like data. The kernel weights are shared for the entire image, reducing the number of parameters of the network. Also, convolutions are equivariant to translation. This means the representation will be the same if we move an object in the input image I and apply a convolution on the shifted object.

2.2 Computer vision background

This section presents well-known deep learning algorithms for computer vision for image classification, object detection and semantic segmentation. Figure 2.4 summarises the different computer vision existing tasks we describe in this section.

2.2.1 Image classification

Image classification is one domain for which deep learning has achieved breakthroughs in the past few years. Image classification is the task of identifying an object in an image and assigning a class to it. Yann LeCun [Lecun 1989] pioneered the use of neural networks on image classification using convolutional neural networks and backpropagation. The increase in computational power and the release of large-scale datasets such as ImageNet (ImageNet Large Scale Visual Recognition

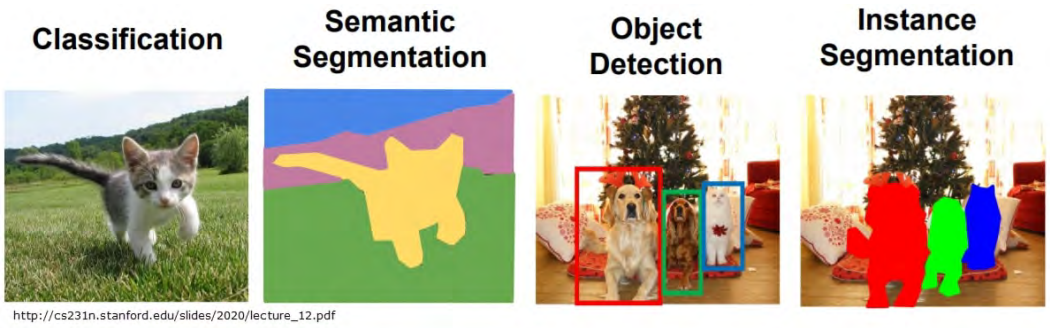


Figure 2.4: Computer vision tasks. Image classification consists in identifying an object in an image and assigning a class to it. Semantic segmentation (or image segmentation) is the tasks in which the goal is to categorise each pixel in an image into a class or object. Object detection aims to detect and locate objects of interest in an image. Instance segmentation involves identifying and separating individual objects withing an image.

Challenge, ILSVRC) [Russakovsky 2015] revived the interest of researchers in deep learning. ImageNet is an object recognition dataset containing over a million images across 1k of object classes.

AlexNet

Krizhevsky *et al.* [Krizhevsky 2012] leverage the power of GPUs (Graphical Power Unit) and introduced AlexNet, a CNN with a similar architecture as LeNet-5 [Lecun 1989]. AlexNet introduced better non-linearity in the network with the ReLU activation function and proves ReLU is more efficient for gradient propagation. Moreover, the paper introduced two major deep learning concepts: the dropout as a regularisation method and the concept of data augmentation to reduce overfitting. Finally, Krizhevsky *et al.* show that deeper networks are better. The more convolutional layers there are, the more fine-grained features the network learns for classification. Although now outdated, AlexNet was the forerunner of the current use and craze for deep learning.

Deeper is better (VGG)

Krizhevsky *et al.* suggest that the depth of CNNs allows finer features extraction. Simonyan and Zisserman [Simonyan 2015] explored this point by stacking several convolutional layers with small kernels (3×3)together. VGG networks build upon the following configuration: a stack of convolutional layers (which have different depths in different architectures), three fully connected layers and a softmax layer. The depth of the networks ranges from 11 layers to 19 layers. The deepest architecture (VGG19) reaches 7.5% top-5 validation errors, outperforming AlexNet. The

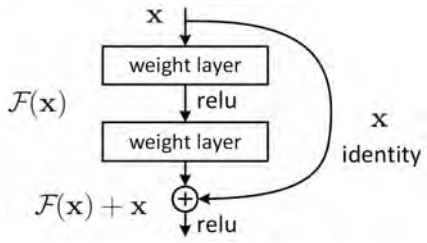


Figure 2.5: ResNet building block [He 2016]

model we present in Chapter 3 is inspired by this architecture.

Connecting the layers (ResNet)

AlexNet [Krizhevsky 2012], VGG [Simonyan 2015] or GoogLeNet [Szegedy 2015] all follow the same trend: going deeper. However, stacking more and more layers does not necessarily lead to better accuracy. When the depth of the network increases, a degradation problem appears. Accuracy gets saturated, and adding layers leads to higher training errors. In other words, the networks are more challenging to train when they are deeper because it is more difficult to backpropagate the gradient. ResNet networks family [He 2016] introduces the concept of *residual connections*. As shown in Figure 2.5, identity mapping is added via an element-wise addition between the input and the output of the layer. This helps the gradient propagation and avoids the problem of vanishing gradient. Also, residual connections help to combine different levels of features at each network step. He *et al.* [He 2016] were able to stack up to 152 layers, thus reaching a top-5 validation error of 5.71%.

Efficient CNNs

The general trend in deep learning is to build bigger and deeper networks to extract more fine-grained features. Despite increasing the accuracy, these networks could be more computationally efficient in size and speed. In many real work applications, including automotive applications in this thesis, the recognition and detection tasks must be carried out on the edge of computationally limited accelerators. MobileNet [Howard 2017, Sandler 2018, Howard 2019] family introduces a new kind of efficient architecture in order *"to build very small and low latency models that can be easily matched to the design requirements for mobile and embedded vision application"* [Howard 2017]. In this thesis, we build an efficient network upon these requirements; the contribution is presented in Chapter 4.

MobileNetV1 [Howard 2017] is one of the first CNN architectures built for mobile and embedded vision applications. MobileNetV1 is based on a simple architecture (similar to VGG [Simonyan 2015]) and uses depthwise separable convolutions instead of plain convolutions to build a lightweight deep neural network. Depthwise

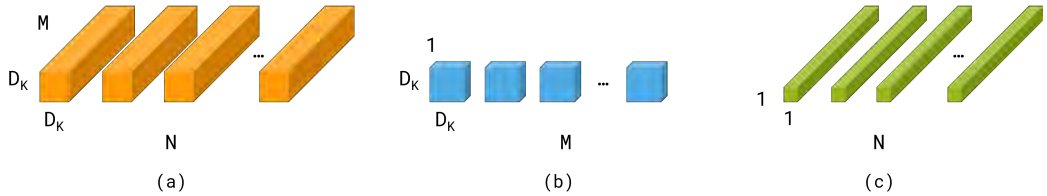


Figure 2.6: In MobileNetV1, the standard convolutional filters in (a) are replaced by two layers: depthwise convolution in (b) and pointwise convolution in (c) to build a depthwise separable filter with M input channels, N output channels and a kernel size D_K.

separable convolutions use 8 to 9 times less computation than standard convolutions. Figure 2.6 illustrates this module.

MobileNetV2 [Sandler 2018] was proposed by Sandler *et al.* as an improvement of MobileNetV1. MobileNetV2 is based on a new type of layer called inverted residual (IR). An IR block is a residual block [He 2016] that uses an inverted structure for efficiency reasons. A first 1×1 convolution is used to widen the number of input channels by an expansion rate γ. Then, a 3×3 depthwise convolution is used on the output of the first convolution. Finally, a second 1×1 convolution is used to reduce the number of channels to apply a residual connection. The 3×3 depthwise convolution drastically reduces the number of parameters of the block. We show the structure of an IR block in Figure 2.7.

MobileNetV3 [Howard 2019] is the last generation of the MobileNets family. Compared to MobileNets V1 and V2, MobileNetV3 is tuned through a combination of Neural Architecture Search (NAS) and novel architecture modules. MobileNetV3 mixes the IR block proposed in MobileNetV2 with Squeeze-and-Excite modules [Hu 2018]. Additionally, the authors propose to replace the ReLU activation func-

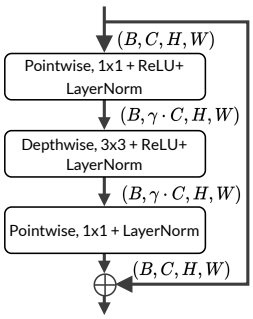


Figure 2.7: Inverted Residual block. The + symbol corresponds to the addition operation.

tion with a Hard Swish activation function:

$$\text{h-swish}(x) = x\frac{ReLU6(x+3)}{6} \tag{2.7}$$

2.2.2 Object detection

Object detection is a computer vision task that aims to detect and locate objects of interest in an image or video. The task involves identifying the position and the boundaries (bounding boxes) of objects and classifying them into different categories (see Figure 2.4).

Datasets and benchmarks

The most popular benchmarks and datasets for object detection are the Pascal VOC (Visual Object Classes) [Everingham 2012] and the MSCOCO [Lin 2014] datasets. For autonomous driving applications, other benchmarks exist, such as the nuScenes dataset [Caesar 2019], the KITTI Vision Benchmark [Urtasun 2012], or the Waymo Open Dataset challenge [Ettinger 2021].

Pascal VOC dataset [Everingham 2012] is one of the first object detection and segmentation dataset. The first version was released in 2005, but the 2012 version is the most popular. Pascal VOC dataset contains around 10K images over 20 object categories (vehicles, animals, bicycles). Each image has pixel-level segmentation, bounding box, and object class annotations. The Pascal VOC has been widely used for object detection, semantic segmentation and classification tasks but remains a small dataset.

MSCOCO dataset [Lin 2014] is a large-scale object detection, segmentation, keypoint recognition, and captioning dataset. To this day, this is the benchmark of reference for computer vision tasks. The dataset comprises 328k images over 80 categories (91 for the latest version). The dataset has various annotations, including bounding boxes, semantic and panoptic segmentation, and key points detection annotations.

Metrics

Object detection can be seen as a regression (locate objects) and a classification task formulated as a multi-task problem. In order to evaluate object detectors, one needs to evaluate the localisation and classification performance of the model. For object detection, the mean average precision (mAP) is commonly used. The mAP is built upon the following metrics: Intersection Over Union (IoU), precision and recall.

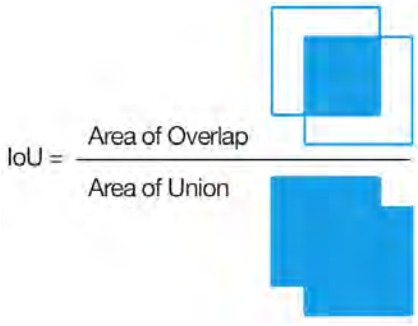


Figure 2.8: Intersection over Union

Intersection Over Union IoU is a measure based on Jaccard Index that evaluates the overlap between two bounding boxes (or two segmentation masks for image segmentation). It requires a ground-truth box B_{gt} and a predicted bounding box B_p. The IoU ranges from 0 to 1. A perfect object localisation would have an IoU of 1. By setting a threshold, we can tell if a detection is valid (true positive, TP) or not (false positive, FP). The IoU is given by the overlapping area between B_{gt} and B_p, divided by the area of union of them:

$$\text{IoU} = \frac{\text{area}(B_p \cap B_{gt})}{\text{area}(B_p \cup B_{gt})} \tag{2.8}$$

Figure 2.8 shows the IoU operation for bounding boxes.

Precision and Recall The precision is the ability of the model to identify **only** the relevant objects. It is the percentage of correct positive predictions (IoU > threshold):

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{TP}{\text{all detection}} \tag{2.9}$$

The recall is the ability of a model to find all the relevant cases (all ground truth bounding boxes). It is the percentage of true positives detected among all relevant ground truths. For critical automotive scenarios, a high recall is desirable, indicating we do not miss any objects.

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{TP}{\text{all ground truth}} \tag{2.10}$$

Mean Average Precision The mAP evaluates the average precision for all classes in the dataset. In practice, the AP is the area under the curve (AUC) of the precision vs recall curve. The COCO mAP [Lin 2014] consists of computing the AP for each class with different IoU thresholds, ranging from 0.5 to 0.95 with a 0.05 step, and average them.

Models

The state-of-the-art (SOTA) methods for object detection can be categorised into two categories: one-stage [Redmon 2016, Liu 2016, Lin 2017b, Zhou 2019, Tian 2019] and two-stage methods [Ren 2017, He 2017, Lin 2017a].

R-CNN (Region-CNN) R-CNN object detectors are a family of two-stage object detectors. Two-stage detectors aim first to find the location of potential objects (proposals), then extract a feature vector at the position of the detected object to classify it and improve the localisation precision.

Using a selective search algorithm, the R-CNN model [Girshick 2014] groups objects based on their colours, texture, and shape. Then, each proposal is resized to match the input of a CNN pre-trained on ImageNet[1]. The feature vector of the CNN is then used as input of an SVM (Support Vector Machine) for classification, and a linear regression model is used to predict objects' location. In practice, the regression model outputs δ-values representing the difference between the proposal boxes and the actual location of the objects. This remains true for all object detection models.

Fast R-CNN In R-CNN, each proposal goes through a CNN for classification and regression, which is inefficient and not adapted for real-time applications. In [Girshick 2015], Girshik *et al.* try to reduce the computational cost of R-CNN by performing a single CNN forward pass[2] on the image. Then, they extract Regions Of Interest (RoI) using the selective search algorithm on the produced feature maps. Each RoI is reduced to a fixed size using a pooling layer and a small shared two-layer MLP extracts features. Finally, the vector created by the two-layers MLP is used to predict the object's class with a softmax classifier and its position with a linear regressor. Fast R-CNN allows nine times faster training speed and processes images 146 times faster than R-CNN.

Faster R-CNN The R-CNN [Girshick 2014] and the Fast R-CNN [Girshick 2015] depend on heuristic-based region proposal algorithms (selective search) to hypothesise object locations. However, region proposal algorithms are slow compared to neural networks on GPUs. For example, in Fast R-CNN, the selective search algorithm takes up to 2 seconds in inference to produce proposals. Therefore, Ren *et al.* [Ren 2017] proposes to use GPUs to compute the proposals with deep convolutional neural networks. They introduce a novel network in the Fast R-CNN framework, the region proposal network (RPN). The RPN takes a set of feature maps (produced

[1] R-CNN uses the AlexNet [Krizhevsky 2012] architecture to extract features
[2] Fast R-CNN uses VGG [Simonyan 2015] backbone.

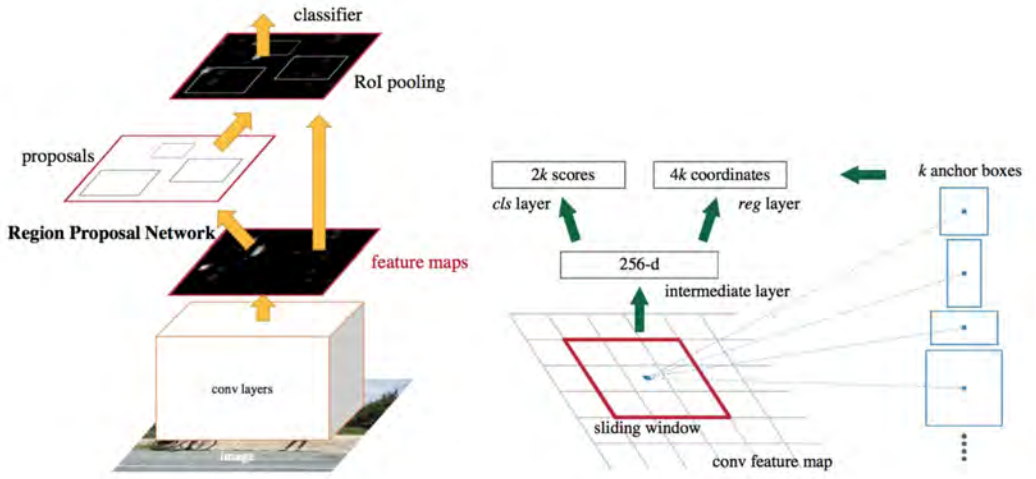


Figure 2.9: **Left**: Faster R-CNN overview. First a CNN extracts features from an input image to produce the features maps. Then the RPN proposes proposals that are used as input of a Fast R-CNN head to predict the position and the class of objects in the input image. **Right**: The RPN and its anchors at a single location. For each position in the feature map, a 3×3 convolution is applied. Then two MLPs are used to predict a set of k proposals relative to k reference boxes called the *anchors*. Source: [Ren 2017]

using a CNN) as input and learns to generate proposals. To generate proposals, a 3×3 sliding window is applied over the feature maps[3]. Then, each sliding window is mapped to an n-dimensional feature space. The RPN predicts a set of k proposals at each sliding window location. So the classification layer outputs $2k$ objectness score, and the regression layer outputs $4k$ outputs encoding the coordinates of k boxes. The predictions of the RPN are relative to k reference boxes with different scale and aspect ratios, which we refer to as *anchors* (see Figure 2.9). Anchors are selected according to their confidence score to keep only relevant anchors corresponding to a potential object. Finally, the proposals and the feature maps are used as input of a Fast R-CNN model. Learning to propose regions and sharing the features between the RPN[4] allows faster inference speed (0.2 seconds) than Fast R-CNN.

Mask R-CNN Finally, in [He 2017], He *et al.* proposed an extension of Faster R-CNN to perform the object detection and segmentation task simultaneously. Mask R-CNN follows the Faster R-CNN framework (*i.e.* a shared backbone and a RPN) and adds a third segmentation head to a Fast R-CNN model. Mask R-

[3]In practice, this is implemented using the convolution operation.

[4]The RPN can be seen as the combination of a feature extractor (VGG16 [Simonyan 2015] or ZF-net [Zeiler 2014]) and a small CNN.

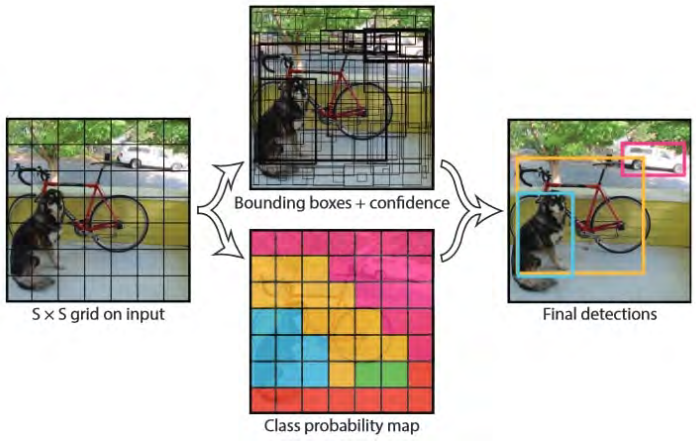


Figure 2.10: YOLO overview. YOLO divides the input image into a $S \times S$ grid, and for each cell predicts B bounding boxes, confidence for those boxes and C class probabilities. The predictions are then encoded as an $S \times S \times (B*5+C)$ tensor. The $S \times S$ grid corresponds to the output feature maps of a CNN, and the B bounding boxes are similar to the anchors in Faster R-CNN. Source: [Redmon 2016]

CNN replaces the classic RoI pooling operation with a new pooling operation called RoIAlign (Region of Interest Align). RoIAlign removes the quantisation of RoI pooling and computes the exact coordinates of objects. Additionally, Mask R-CNN adopts a ResNeXt-101 backbone [Xie 2017] with Feature Pyramid Networks (FPN) [Lin 2017a]. FPN uses a top-down architecture with lateral connections to extract features according to their scale.

You Only Look Once (YOLO) Compared to RPN-based object detectors, which perform detection on region proposals and thus end up performing prediction multiple times for various regions in an image, YOLO [Redmon 2016] architecture (and more generally single-stage architectures) aims to perform the detection and the classification in a single forward pass. YOLO[5] divides the input image into a $S \times S$ grid, and for each cell predicts B bounding boxes, confidence for those boxes and C class probabilities. The predictions are then encoded as an $S \times S \times (B*5+C)$ tensor. The $S \times S$ grid corresponds to the output feature maps of a CNN, and the B bounding boxes are similar to the anchors in Faster R-CNN. YOLO predicts offsets between anchors and bounding box coordinates for each grid cell. However, because YOLO predicts few objects per location after several downsampling steps, it struggles with small objects and makes localisation errors. YOLOv2 (or YOLO9000) [Redmon 2017] reduces the number of localisation errors by using batch normalisation and pre-training the backbone on high-resolution images to learn from

[5]We refer to YOLO for the first version of YOLO. We use YOLOv* for subsequent versions of it.

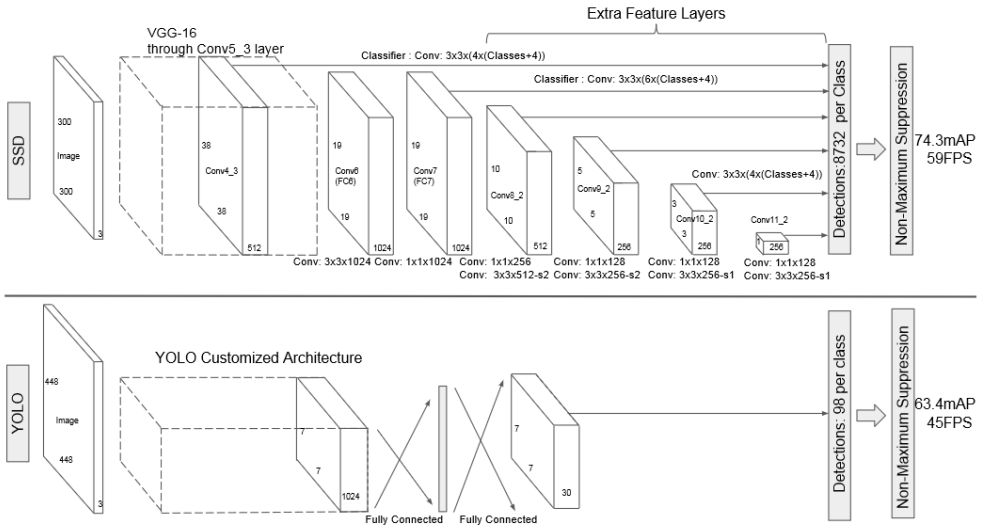


Figure 2.11: SSD model vs. YOLO model. SSD detects objects at multiple scales while YOLO uses a single scale feature maps [Liu 2016]

high-resolution features. Moreover, the authors add pass-through layers which concatenate high-resolution features with low-resolution features to obtain fine-grained feature maps.

Single Shot MultiBox Detector (SSD) SSD is another single-stage object detection approach that predicts object locations and classes in a single forward pass. Compared to YOLO [Redmon 2016] and YOLOv2 [Redmon 2017], SSD uses multi-scale feature maps for detection. Similarly to YOLO, SSD defines a set of anchors at each feature maps cell for multiple feature maps (extra-feature layers). SSD computes c class scores and four offsets relative to the default anchor boxes for each extra-feature layer and each box at each location. This yields $(c+4)*k*m*n$ predictions, where m and n are the size of the feature maps, and k is the number of anchors.

Conclusion In this section we presented the main frameworks for object detection in computer vision. Most of object detectors are built upon YOLO, SSD or Faster R-CNN. To further improve the performance of these object detectors, researchers focus their works on improving features extraction [Liu 2022, Liu 2021], training strategy [Caron 2021] or new paradigms such as anchor-free object detectors [Tian 2019, Carion 2020a, Zhou 2019]. In this thesis, we will use the Faster R-CNN framework and study how much this architecture is suited for radar object detection. Two-stage detectors are generally more accurate but slower than single stage detectors. Thus we will optimise the features extraction stage to extract

meaningful radar features while being computationally efficient.

2.2.3 Image segmentation

Segmentation is the process of partitioning an image into multiple regions. There are three groups of segmentation: semantic segmentation, instance segmentation and panoptic segmentation.

Semantic segmentation is the task in which the goal is to categorise each pixel in an image into a class or object. Semantic segmentation aims to produce an image's dense pixel-wise segmentation map, also known as a segmentation mask. Instance segmentation involves identifying and separating individual objects within an image, including detecting the boundaries and assigning a unique label to each object. Instance segmentation is a type of object detection. Finally, panoptic segmentation combines semantic segmentation and instance segmentation to provide a comprehensive understanding of the scene. In this thesis, we will focus on semantic segmentation only.

Datasets and benchmarks

As for object detection, the PASCAL VOC [Everingham 2012] and the MSCOCO [Lin 2014] datasets are famous benchmarks for semantic segmentation. Additionally, the Cityscapes [Cordts 2016] dataset is widely used for semantic segmentation. We refer the reader to Section 2.2.2 for details about Pascal VOC and MSCOCO.

The Cityscapes dataset [Cordts 2016] is a large-scale dataset for semantic understanding of urban street scenes. It provides semantic, instance-wise, and dense pixel annotations for 30 classes grouped into eight categories (flat surfaces, humans, vehicles, constructions, objects, nature, sky, and void). The dataset is small compared to MSCOCO. Indeed, Cityscape contains only 5000 annotated images and 20000 coarse annotated ones.

Metrics

Because semantic segmentation models predict masks, the mIoU metric we defined in Section 2.2.2 and the pixel accuracy is used. The IoU is computed between a ground truth mask and the prediction for each class. Then, by averaging the IoU of each class, we compute the mIoU.

The pixel accuracy is the percentage of pixels in the image which are correctly classified. Generally, pixel accuracy is reported for each class separately and by averaging across classes. One issue with pixel accuracy is that it can provide misleading results when the class representation is small within the image (*e.g.* mostly background).

Models

Semantic segmentation models aim to label each pixel of an image with a corresponding class. Thus, such models require the same input and output size. A naive approach is to design an architecture with convolutional layers without decreasing the input size. Then, apply a softmax function to the last feature maps. Nevertheless, this is computationally expensive. Deep CNNs for image classification generally downsample the size of the input multiple times to learn deeper representations. However, we must produce a full-resolution segmentation mask the same size as the input image for semantic segmentation. One popular image segmentation approach follows an encoder-decoder architecture, where the encoder downsamples the spatial resolution, developing lower-resolution feature maps, and where the decoder upsamples the feature representations learned by the encoder into a segmentation mask.

Fully Convolutional Networks (FCN) Long *et al.* [Long 2015] were the first to propose a fully convolutional network trained end-to-end for image semantic segmentation. The authors proposed to adapt existing image classification networks (*e.g.* AlexNet [Krizhevsky 2012]) as an encoder and use transpose convolution (or deconvolution) layers on the top of the feature maps to upsample low-resolution features into a full-resolution segmentation map. However, FCN struggles to produce fine-grained segmentation masks. Indeed, the input's resolution is reduced by 32, and the authors use a single deconvolution layer. To address this issue, the authors propose slowly upsampling the encoded representation at different stages, adding skip connections from earlier layers and summing feature maps together. It allows fine layers (*where*) to be combined with coarse layers (*what*), improving the segmentation of object boundaries.

U-Net Later on, Ronneberger *et al.* [Ronneberger 2015] improved the FCN architecture by expanding the capacity of the decoder. Instead of using a single deconvolution, they propose a symmetric encoder-decoder architecture for image semantic segmentation. The encoder (referred to as *contracting path* in the original paper) captures context. The decoder (referred to as *expanding path*) is symmetric to the encoder and enables precise localisation. Also, U-Net adds *skip connections* between the encoder and the decoder to combine low-level features (*where*) with high-level features (*what*). U-Net architecture has become popular, modified, and adapted for various segmentation problems. Today, we can consider this architecture as the reference encoder-decoder architecture. Figure 2.12 depicts the U-Net architecture.

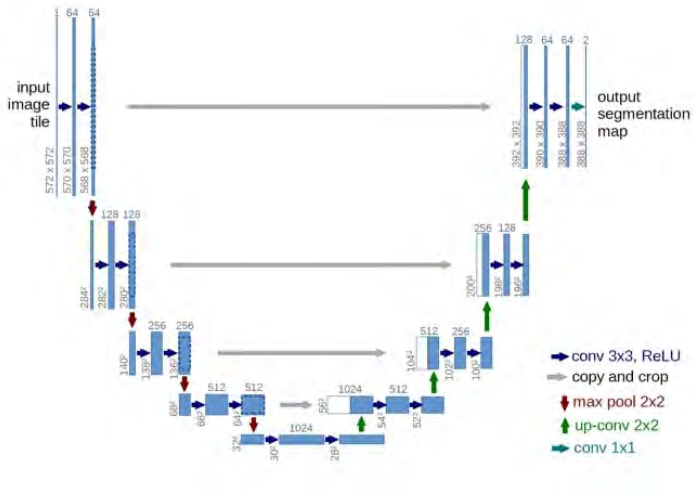


Figure 2.12: U-Net architecture [Ronneberger 2015]

The DeepLab family One problem in image segmentation or object detection is classifying, segmenting or detecting the same objects at different scales. This has been addressed for object detection using FPN [Lin 2017a], multi-scale feature maps [Liu 2016], or multi-scale anchors [Ren 2017]. A second problem is the coarse resolution of the feature maps caused by the repeated downsampling operations performed at consecutive layers in the backbone network. Encoder-decoder with skip connections [Ronneberger 2015] has been shown to help to solve these issues. Chen *et al.* proposed DeepLabV1 [Chen 2015] and then DeepLabV2 [Chen 2018a] to solve the problems above. DeepLabV2 architecture combines atrous convolutions, spatial pyramid pooling and Conditional Random Fields (CRF). To increase the size of the feature maps, Chen *et al.* propose to remove the downsampling operator from the last few max-pooling of deep convolutional neural networks[6] and instead upsample the feature maps using *atrous* (or dilated) convolutions. Atrous convolution consists of convolution with a sparse kernel. A fixed number of zeros separates elements in the kernel, called the dilation rate. It allows to enlarge the receptive field of a network to incorporate a larger context without increasing the number of parameters. This way, the authors can increase the size (and the resolution) of the computed feature maps by a factor of 4. Inspired by Spatial Pyramid Pooling [He 2014] and Feature Pyramid Network [Lin 2017a], the authors also introduce a new module called Atrous Spatial Pyramid Pooling (ASPP) to deal with multiple scale objects. ASPP applies multiple parallel filters with different dilation rates on the feature maps, thus allowing learning patterns at different scales. The feature maps are processed with different receptive fields and then concatenated. Finally,

[6]The deep convolutional neural networks used for DeepLabV2 are VGG16 [Simonyan 2015] or ResNet101 [He 2016].

to improve the segmentation of object boundaries, the concatenated feature maps are processed by a fully connected conditional random field [Krähenbühl 2011].

Later on, Chen *et al.* revisited their work and proposed DeepLabV3 [Chen 2017]. DeepLabV3 employs atrous convolution in cascade or parallel to better capture long-range information in the deeper blocks. The ASPP module is also modified with lower dilation rates, and a 1×1 convolution and batch normalisation are added. The concatenated feature maps of the ASPP module are directly processed to predict the segmentation mask, and the dense CRF is no longer required.

Last, DeepLabV3+ [Chen 2018b] extends DeepLabV3 by opting for an encoder-decoder structure. DeepLabV3+ uses the DeepLabV3 as an encoder, and a simple decoder processes the low-level features from the encoder and the feature maps from the ASPP module. For efficiency, Chen *et al.* use depthwise-separable convolutions [Howard 2017] in the decoder and the ASPP module. Finally, they adapt the Xception model [Chollet 2017] in the encoder to extract features.

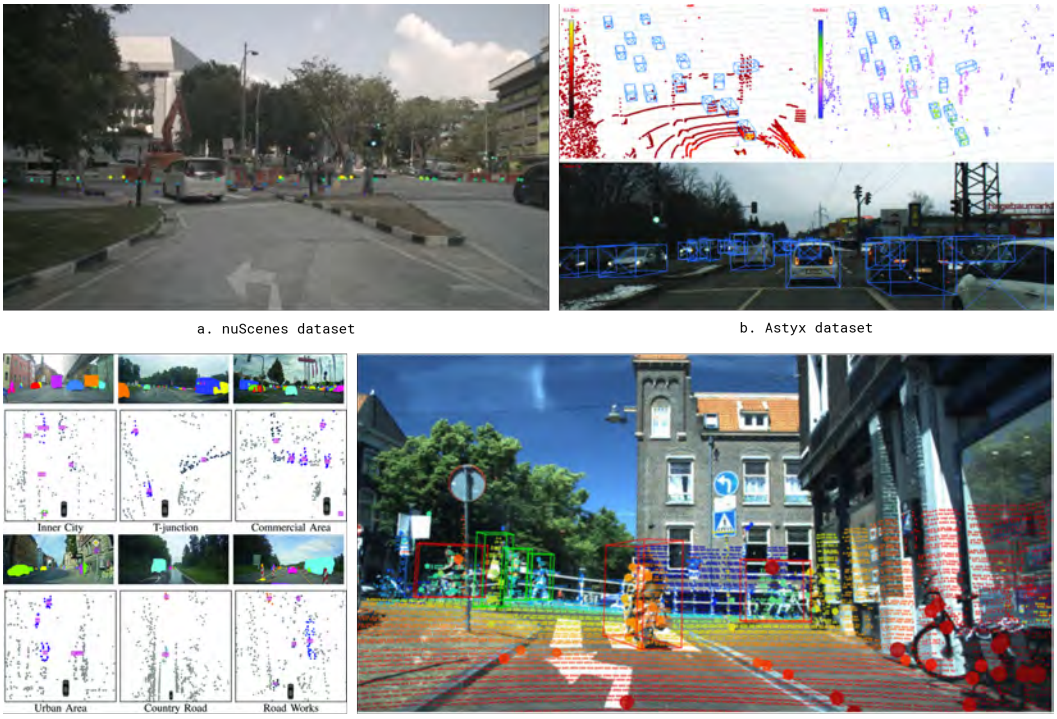
Conclusion In this section, we presented well-known models for semantic segmentation. Today, most image segmentation networks are built following an encoder-decoder architecture and skip connections. To this day, this is the more natural way to process images. Using these principles, we will build the semantic segmentation and detection models we present in this thesis.

2.3 Automotive radar datasets

One fundamental principle of deep learning is the need for data. The best computer vision models generally learn from vast amounts of annotated data. Hence, datasets with annotated radar data are required to apply deep learning algorithms for radar perception. When starting this thesis, very few radar datasets with raw radar frames were available [Gao 2019b, Ouaknine 2021b]. From 2020 to now, more and more datasets have been released for different tasks (classification, detection, segmentation, tracking) and with different radar representations. This section presents available radar datasets we can use for automotive radar perception. We split these datasets into point cloud datasets and raw data datasets.

2.3.1 Point clouds datasets

Point clouds are the standard representation of radar data. Point clouds represent the signal at the *target-level*. Similarly to Palffy *et al.* [Palffy 2022], we refer to 2+1D radar for radars that output a sparse point cloud of reflection. Each point contains the range r of the target, the angle of arrival θ and the radial velocity v_r. We refer to 3+1D radar for radars having three spatial dimensions: the range r, the



a. nuScenes dataset b. Astyx dataset

c. RadarScenes dataset d. View-of-Delft dataset

Figure 2.13: Overview of point clouds radar datasets. (a) nuScenes [Caesar 2019] dataset sample. We notice a few points per object, and the radar has no elevation capabilities. (b) Astyx [Meyer 2019] dataset sample. Compared to nuScenes data, the radar has elevation capabilities, and more points per object are available. (c) RadarScenes [Schumann 2021] dataset sample. The point cloud is denser than the nuScenes point cloud, but no elevation measurement exists. (d) VoD dataset [Palffy 2022] sample. We can see that radar (big points) has a lower resolution than LiDAR but has a better resolution than nuScenes data and elevation data.

azimuth θ, the elevation α and the radial velocity v_r. Point clouds can be used for object classification, segmentation (clustering objects of the same class) and sensor fusion. Because radar point clouds are sparse, they are particularly appropriate for embedded applications. This sparsity is also a drawback because models need more information to generalise well.

nuScenes [Caesar 2019], released in 2019, is the first public large-scale dataset for autonomous driving. It contains 2+1D radar point clouds from 5 radars alongside cameras, LiDARs and IMUs. Although data from several radars is available, these radars have low resolution, resulting in very sparse (few points per object) point clouds as shown in Figure 2.13. Some work tried using this dataset for object detection [Niederlöhner 2022, Svenningsson 2021], but the resolution is too low to obtain enough detection accuracy. As a result, nuScenes' radar data is mainly used for sensor-fusion applications rather than radar-only perception.

To address the sparsity of low-resolution radar, 2+1D or 3+1D high-resolution radar datasets were made public such as Astyx [Meyer 2019], RadarScenes [Schumann 2021] or View-of-Delft (VoD) [Palffy 2022] datasets.

Astyx dataset [Meyer 2019] (no longer available) is the first 3+1D high-resolution radar dataset. It contains data from one radar, camera and LiDAR with 3D bounding box annotations. Compared to the nuScenes dataset, one radar frame contains around 1000 3D points instead of 100 2D points. However, the Astyx dataset remains very small as the dataset consists of only 500 synchronised frames, containing around 3000 manually annotated objects divided into seven classes (bus, car, cyclist, motorcyclist, person, trailer and truck). Moreover, the dataset mainly comprised non-consecutive measurements, and few driving scenarios are available.

Schumann *et al.* try to alleviate that and propose the RadarScenes dataset [Schumann 2021], which consists of four 2+1D high-resolution radars for automotive perception. It contains 4.3 hours of driving in real automotive scenarios, resulting in 118.9 million radar points. The point clouds of the dataset are manually annotated and are divided between 11 object classes with four large main categories and a total of over 7000 road users. Unfortunately, only moving objects are annotated.

The VoD dataset [Palffy 2022] then appears as a good candidate for 3+1D radar perception. VoD dataset brings around 9000 synchronised and calibrated camera, LiDAR and radar frames together. Data were acquired in complex, urban traffic. The dataset consists of more than 120000 manually annotated 3D bounding boxes of moving and static objects, including pedestrians, cyclists and car labels. The VoD dataset is the biggest and the most realistic radar point clouds dataset yet.

2.3.2 Raw datasets

The sparsity of radar point clouds is both an advantage and a disadvantage. On the one hand, this sparsity allows the development of very efficient algorithms for automotive radar perception running on edge devices. On the other hand, radar point clouds contain low-level information (position of targets, velocity, radar cross section) due to many filtering techniques (Figure 1.5). The information available in the raw signal could help increase radar perception algorithms' accuracy. Thus, filtering techniques might be removed from the conventional signal processing chain, and raw data could be used as input of deep neural networks for object detection and classification that we detail thereafter.

Low-resolution radar datasets

Ouaknine *et al.* observed a need for annotated datasets with range-angle or range-Doppler raw radar data and proposed in 2020 the CARRADA dataset [Ouaknine 2021b]. CARRADA is the first raw radar data dataset made public.

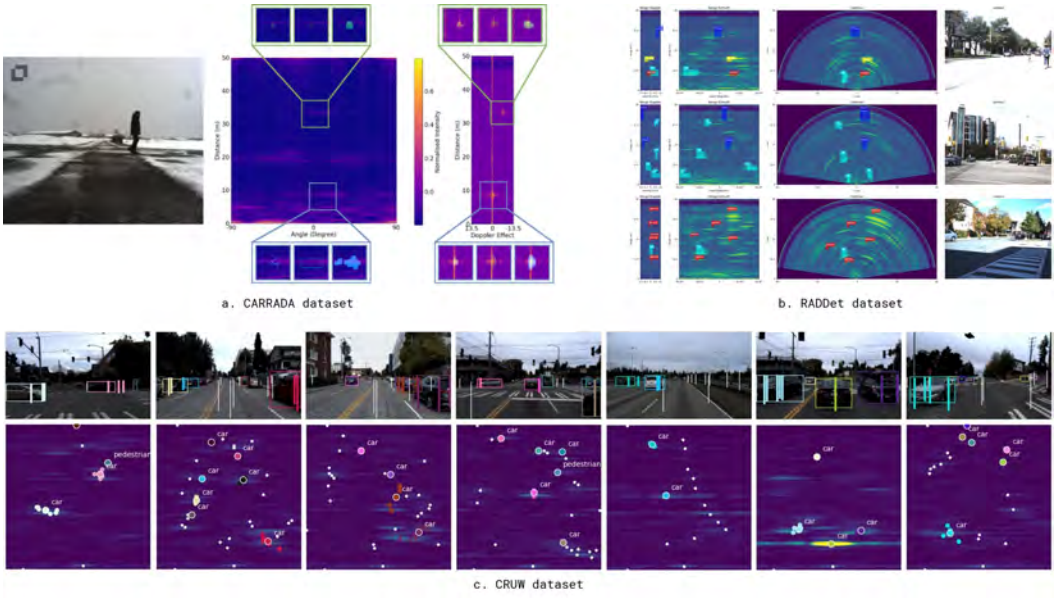


Figure 2.14: Overview of low-resolution raw radar datasets. (a) CARRADA [Ouaknine 2021b] dataset sample. RA and RD maps are available with corresponding annotations in the following format: bounding boxes, points, and segmentation masks. (b) RADDet [Zhang 2021] dataset sample. RAD tensors are available in polar and cartesian coordinates. Annotations are 3D bounding boxes in range, angle and Dopper dimensions. (c) CRUW [Wang 2021c] dataset sample in city road driving scenario. The upper row shows the RGB images with the detected bounding boxes from Mask R-CNN and the projected CFAR detections (vertical lines). The lower row shows the RF images with the CFAR detections (dots) and the final object annotations [Wang 2021c].

It contains around 12000 synchronised camera-FMCW radar frames with range-angle-Doppler annotations. The dataset has been recorded on a test track to reduce environmental noise. Simple scenarios with cars, pedestrians and cyclists moving with various trajectories have been recorded to simulate urban driving scenarios. Zhang *et al.* propose a similar dataset with RAD tensors and 3D RAD bounding boxes annotations.

The RADDet dataset comprises around 10k frames collected in sunny weather at several random locations. The sensors were set up on the sidewalks and facing the main roads. Such a sensor setup makes RADDet more realistic than CARRADA. Similarly to CARRADA, the sensor of the RADDet dataset is static, thus reducing noise effects due to the vehicles' movements.

The CRUW dataset [Wang 2021c] is another large camera-radar dataset. It contains about 400k frames of two stereo cameras and two FMCW radars collected in four scenarios: campus road, parking lot, city street and highway. Contrary to CARRADA or RADDet datasets, only RA maps (no Doppler) are available in the

CRUW dataset, and annotations are in the form of points to remain compliant with conventional radar outputs. Unfortunately, the CRUW dataset is not publicly available, but authors release a subset of it, named the ROD2021[7] dataset. The ROD2021 dataset contains about 50k frames from a single RGB camera and FCMW radar.

In this thesis, we use these datasets as they were the only ones available during the major part of the thesis. We also chose to work on these datasets as the low-resolution aspect of the radars used in CARRADA, RADDet or CRUW is challenging. We hypothesise that with enough data and good annotations, a deep learning algorithm could overstep the low resolution of the radar. We show samples of these datasets in Figure 2.14.

High-resolution radar datasets

Low-resolution radar datasets are very useful for detecting objects in range and velocity. However, the low-angle resolution makes detecting and classifying objects difficult in the azimuth domain. More recently, high-resolution radar datasets have started emerging. The goals of high-resolution datasets are multiple: enabling accurate 3D detection and classification for radar sensors [Paek 2022, Rebut 2022, Madani 2022] and avoiding the computationally expensive generation of RA radar maps [Rebut 2022].

RADIal (for Radar, LiDAR *et al.*) is a raw high-resolution radar dataset including other sensor modalities like cameras and LiDAR. RADIal aims to motivate research on automotive high-resolution radar and camera-lidar-radar sensor fusion. RADIal contains around 25k synchronised frames, out of which more or less 8k are labelled with vehicles and free-space driving masks. Data are provided in ADC data to be used directly to detect and classify vehicles and avoid time-consuming RA map generations. However, the RADIal dataset mainly contains data recorded on highways or countryside and only car labels, making it challenging to use for more general applications (city roads).

The K-Radar [Paek 2022] is a 3+1D radar dataset with a similar size as RADIal [Rebut 2022] collected under various scenarios (*e.g.* urban, suburban, highways), time (*e.g.* day, night), and weather conditions (*e.g.* clear, fog, rain, snow). It contains around 35k manually annotated 4D radar tensors (range, Doppler, azimuth, elevation). Unlike the ADC data of the RADIal dataset, K-Radar tensors are heavy, and the dataset cannot be downloaded because of its massive size (16TB).

Finally, Madani *et al.* introduce the Radatron dataset [Madani 2022]. The Radatron dataset is a high-resolution radar dataset using a cascaded MIMO radar. As for the K-Radar dataset [Paek 2022], radar data is in the form of 4D tensors. The

[7] https://www.cruwdataset.org/rod2021

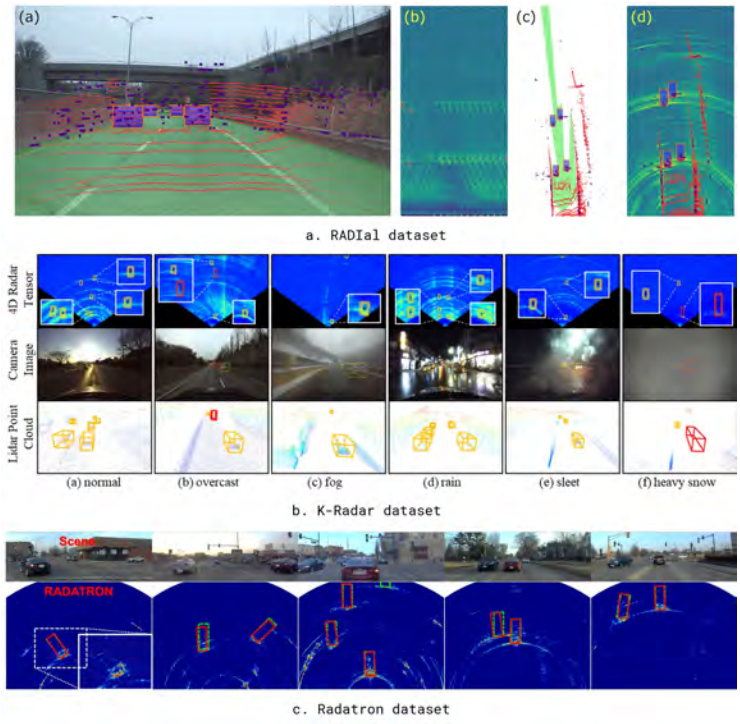


Figure 2.15: (a) RADIal dataset sample. From left to right: camera image with projected laser point cloud in red and radar point cloud in indigo, vehicle annotation in orange and free-driving space annotation in green ; radar power spectrum (MIMO RD) with bounding box annotation ; free-driving space annotation in bird-eye view, with vehicles annotated with orange bounding boxes, radar point cloud in yellow and LiDAR point cloud in red ; range-azimuth map in Cartesian coordiates, overlayed with radar point cloud and LiDAR point cloud [Rebut 2022]. (b) K-Radar dataset sample in various weather conditions. (c) Radatron dataset sample. Ground truth are marked in green.

dataset is collected under clear weather conditions, and out of the 152k frames, 16k vehicles were annotated with 2D bounding boxes on RA maps. Radatron presents several limitations:

1. The radar's maximum range is 25 meters, while RADIal or K-Radar can detect objects up to 100 meters.

2. Radatron does not leverage the 4D nature of the data because annotations are only provided for 2D RA maps.

3. As RADIal, Radatron only provides annotations for vehicles.

However, learning to detect vulnerable road users like pedestrians is essential for automotive applications.

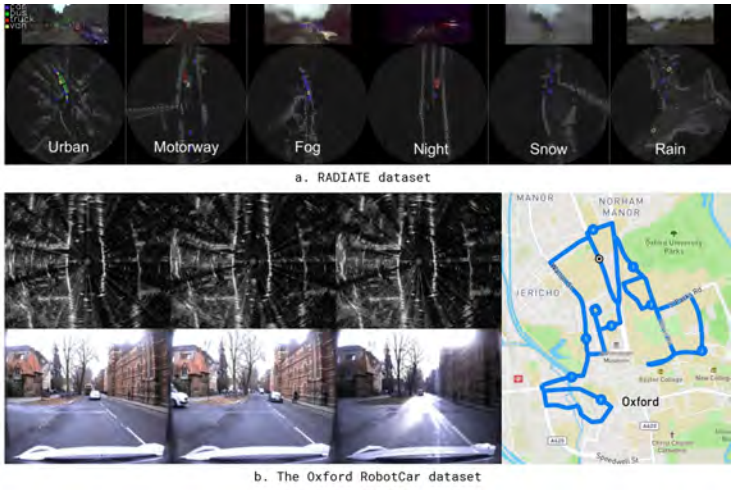


Figure 2.16: Overview of scanning radar datasets. Scanning radar allows a 360° view around the car. (a) RADIATE dataset sample with different driving scenarios under several weather conditions. (b) Oxford RobotCar dataset sample.

To conclude, the main challenge of high-resolution radar datasets is the storage and sharing of these datasets. Indeed, 4D radar tensors are very cumbersome in memory. The choice of RADial's authors to release the ADC data alongside a signal processing pipeline is the best choice to allow research on high-resolution radar datasets. It also enables the development of new processing techniques on ADC data to replace some parts of the conventional radar signal processing chain (see Figure 1.5). We show samples of high-resolution datasets in Figure 2.15

Scanning radar datasets

Scanning radar data (see Figure 2.16) is another radar data available. One advantage of scanning radars is that they allow a 360° representation of the environment. Because they measure each azimuth using a moving antenna, scanning radar provides better azimuth resolution than low-resolution radars (around 0.9° azimuth resolution). However, they do not provide Doppler information, a significant advantage of radar sensors and crucial for automotive applications. During this thesis, we will not use these datasets because we consider the Doppler information a core radar component and because scanning radar is not used in practice.

The Oxford RobotCar [Barnes 2020], MulRan and RADIATE [Sheeny 2021] datasets provide radar data using scanning radars. The Oxford RobotCar dataset contains around 240k radar scans collected in various traffic, weather and lightning conditions in Oxford, UK. Data from sensors such as LiDAR, GPS or cameras are also available. However, the authors of the Oxford RobotCar dataset do not provide annotations.

The RADIATE [Sheeny 2021] dataset is an annotated dataset collected in a mixture of weather (sun, fog, rain, snow, day, night) and driving scenarios. The dataset includes 5 hours of radar images, 3 hours of which are fully annotated with eight categories (*i.e.* van, car, bus, truck, motorbike, bicycle, pedestrian and group of pedestrians). RADIATE dataset includes other modalities such as cameras, LiDAR and GPS data. Likewise, K-Radar [Paek 2022], the strength of RADIATE lies in the various weather driving scenarios, showing the relevance of using radar on top of other sensors for perception in automotive scenarios.

2.4 Automotive radar perception on radar point clouds

Radar point cloud (or radar reflection) is the default output of radar. Radar reflections are sparse and light in memory. Thus, they are the most commonly used representation in the industry for efficiency and embedded object classification, detection and segmentation [Ulrich 2021].

Object classification is the task of assigning a class to clusters of reflections. The clusters are generally obtained after applying clustering algorithms like DBSCAN on the radar point cloud [Scheiner 2018, Scheiner 2019, Scheiner 2020, Tatarchenko 2023]. Scheiner *et al.* [Scheiner 2018, Scheiner 2019] propose to use One-vs-All (OVA) and One-vs-One (OVO) binary classifiers for multi-class road users classification. Binary classifiers overcome data imbalance. They extract a set of features from clusters of point clouds for the target recognition process. [Scheiner 2019] show that ensemble classifiers and recurrent neural networks allow accurate feature selection and improved classification accuracy. Building upon [Scheiner 2018, Scheiner 2019], [Scheiner 2020] study if high-resolution radar point clouds allow better accuracy than conventional radars. However, these methods are features-based and require radar data knowledge to build a good set of features.

The progress of deep learning algorithms has led to the development of data-driven approaches [Tatarchenko 2023, Ulrich 2021] for object recognition. Ulrich *et al.* [Ulrich 2021] propose DeepReflecs, a deep learning approach using a cluster of points which contains the position, the radar cross-section, the range and the radial velocity. They propose to process the data using a simple 1D CNN of 1284 parameters. They show superior performance than features-based approaches at a lower computational cost. In order to explain the classification process, Tatarchenko and Rambach [Tatarchenko 2023] format radar point clouds as a histogram. They use the histogram vector as input of a simple MLP and improve the classification accuracy of DeepReflecs [Ulrich 2021].

Nonetheless, clustering algorithms require setting hyper-parameters that can affect classification performance. Thus, object detection aims to combine clustering and classification methods at once [Dreher 2020, Ulrich 2022, Niederlöhner 2022,

Danzer 2019]. In contrast, object segmentation methods attempt to classify each reflection to create clusters automatically [Schumann 2018, Danzer 2019, Feng 2019].

Object detection and segmentation models for radar point clouds use all the reflections available in the scene (or accumulated over a short period to increase the resolution) as input. The most common approaches are grid-based or point-based.

Grid-based approaches usually render the radar point cloud to a 2D bird-eye-view (BEV) representation or 3D cartesian grid and apply a CNN on it [Dreher 2020, Niederlöhner 2022, Xu 2021]. In [Xu 2021], the author renders the point cloud onto a pillar and uses a self-attention mechanism to solve the problem of orientation estimation in a grid-based approach. [Dreher 2020] exploits the YOLOv3 architecture on a grid-map representation of the point cloud. However, the sparsity of the data does not lead to encouraging results. Niederlöhner *et al.* [Niederlöhner 2022] accumulate point clouds over time to reduce the sparsity of the data and apply an FPN architecture on a rendered point clouds for object detection and cartesian velocity estimation. As in [Dreher 2020], the results were not encouraging due to the high sparsity of the radar point cloud.

Point-based approaches are appropriate for sparse point-cloud object detection. Indeed, they do not pad the data with zeros when there is no measurement. Instead, they learn the relationship between each point in a local neighbourhood. Point-based CNNs create a pseudo-image of the point cloud in the object detection model. Well known architectures are PointNet [Charles 2017], PointNet++ [Qi 2017], VoxelNet [Zhou 2018] or PointPilars [Lang 2019]. In the literature, researchers successfully modify these architectures for object detection or segmentation on radar point clouds [Schumann 2018, Feng 2019, Tilly 2020, Palffy 2022, Xiong 2022]. In particular, Xiong *et al.* [Xiong 2022] show contrastive learning on radar clusters helps improve overall detection performance using fewer training data. Ulrich *et al.* [Ulrich 2022] take advantage of both methods. They mix point-based and grid-based approaches to improve object detection and orientation estimation on radar point clouds. [Fent 2023] employs a graph neural network (GNN) instead of a CNN for object detection and segmentation on radar reflections.

Finally, other works on radar point clouds exist for ghost target detection [Kraus 2020] or scene-flow estimation [Ding 2022, Ding 2023].

2.5 Automotive radar perception on raw data

As explained in Chapter 1, radar sensors usually output point clouds representing the detected targets. Section 2.4 shows we can use radar point clouds for tasks such as object recognition [Ulrich 2021], segmentation [Feng 2019] or ghost target detection [Kraus 2020]. However, the low resolution of radar sensors and the numerous

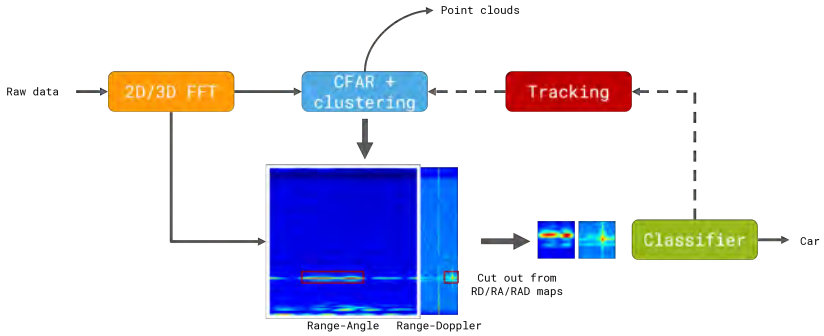


Figure 2.17: Raw data object classification data flow. The dotted lines represent an optional operation.

filtering techniques (thresholding, clustering, tracking) applied to the signal result in a loss of valuable information compared to the raw radar signal (see Chapter 1). Moreover, point clouds contain little information about the target and are sparse no matter the resolution of the radar, lowering the accuracy of point clouds radar object detectors [Niederlöhner 2022, Ulrich 2022]. Therefore, several works consider lower-level representation such as RD, RA or RAD spectra to perform tasks like object classification, detection, segmentation or tracking to exploit the information available in the radar signal fully.

2.5.1 Object classification

In order to classify objects using raw radar data, the vast majority of approaches we present in this section rely on CFAR [Blake 1988] and clustering (DBSCAN) algorithms to first detect targets before classifying them using machine learning. Figure 2.17 shows typical data flow to classify targets using raw radar data. We can split object classification on raw radar data methods into features-based and spectrum-based methods.

Features-based methods

Features-based methods are in between point clouds-based and raw data-based methods. They consist of using handcrafted features from the range and the Doppler profile to recognise objects such as pedestrians or vehicles [Heuel 2011, Prophet 2018b, Prophet 2018a, Lee 2017] before tracking them and producing point clouds. SVM classifiers are commonly used for the classification process. [Prophet 2018b] and [Heuel 2011] also propose to add a tracker after the classifier to improve classification accuracy. In [Prophet 2018a], Prophet *et al.* propose to directly use regions of interest (RoI) from the range-Doppler maps and the image-based feature descriptor SURF [Bay 2006] to learn scale, rotation and skew invari-

ant features from the image directly. They show that using image-based features as input of an SVM classifier achieves an 88% accuracy on pedestrians.

Spectrum-based methods

The work of Prophet *et al.* [Prophet 2018a] shows the potential of using the spectrum to classify objects in radar. This work lead to many studies for object classification using of convolutional neural networks [Kim 2018, Pérez 2018, Patel 2019, Khalid 2019, Cai 2019, Akita 2019, Lee 2019, Palffy 2020, Gao 2019b, Gao 2019a, Patel 2022, Saini 2023, Angelov 2018].

Micro-Doppler object classification Some methods such as [Angelov 2018], [Lee 2019] or [Gao 2019b] use the micro-Doppler signatures of detected objects to recognise moving targets using VGG [Gao 2019b, Lee 2019], ResNet50 [Angelov 2018], or AlexNet [Lee 2019] models. The micro-Doppler effect is a phenomenon that appears when, in addition to the constant Doppler frequency shift induced by the motion of a radar target, the target or any structure on the target (the wheels of a car, arms of a human body) undergoes micro-motion dynamics that induce additional Doppler modulations on the returned signal [Chen 2006]. However, using the micro-Doppler signature is unsuitable for real-time applications because it requires accumulating the signal over a short period (from 0.5 seconds to 2 seconds). Furthermore, for reliable detection of multiple objects, range and Doppler features must be considered [Khalid 2019]. As a result, we focus our research on models based on the range-Doppler or the range-azimuth spectrum.

Classification with RA, RD or RAD tensors Pérez *et al.* [Pérez 2018] use tiny two layers CNN to classify moving targets such as pedestrians, cyclists or cars based on their RoI in the range-angle-Doppler power spectrum. They show that such a model can achieve 97.3% accuracy in classifying objects in single-target scenarios. Kim *et al.* [Kim 2018], Khalid *et al.* [Khalid 2019] and Akita *et al.* [Akita 2019] propose to learn temporal dynamics of moving objects using recurrent and convolutional neural networks. [Kim 2018] classifies sequences of range-Doppler spectra with single moving objects, while [Khalid 2019] and [Akita 2019] extract RoIs from the RD and RA spectrum respectively. The authors of [Akita 2019] show that using raw data benefits object classification in this study. They compare the performance of their classifier on raw data (raw reflection intensity RoI) versus radar features such as the maximum intensity of the reflection or a set of features (average reflection intensity, maximum reflection intensity, roundness). Patel *et al.* [Patel 2022] notice that deep radar classifiers maintain high confidence for ambiguous, complex samples under domain shift and signal corruptions. Indeed, according to the radar equation (Equation 1.1), the same target at different ranges produces

different measurements. As a result, they introduce a specific label-smoothing strategy for radar during training to improve the uncertainty of the classifier.

Hybrid approaches Other methods propose to mix radar knowledge (*target-level*) and the radar cube (*low-level*) to address the direction of arrival estimation problem, which is not used by spectrum-based classifiers. Patel *et al.* [Patel 2019] addresses the lack of position information in object classification by adding preliminary information to RoIs as a channel. It can be the distance between the highest intensity pixel and other pixels of the RoI or a decayed RoI spectrum. The decayed RoI spectrum consists of a spectrum where peripheral reflections are attenuated and important reflections are pronounced. They show a 3% to 6% improvement in the accuracy by adding this target-level knowledge to the input using a three-layer CNN. Gao *et al.* [Gao 2019a] build an SVM-CNN hybrid model to classify moving objects. Palffy *et al.* [Palffy 2020] propose an approach that fully exploits *target-level* and *low-level* information. First, they detect targets using conventional radar signal processing. A region of interest from the RAD cube is extracted for each detected object, and range, azimuth and speed features are learned using two small CNNs. Then, target-level features (range, azimuth, RCS, absolute speed) are concatenated with low-level features to be classified by One-vs-All and One-vs-One binary classifiers. An overview of RTC-Net is shown in Figure 2.18. RTC-Net is more accurate than a features-based method like [Prophet 2018b]. More recently, [Saini 2023] introduced a similar hybrid approach mixing *target-level* (point clouds) and *low-level* (spectra). The authors use two graph neural networks to learn embeddings from the radar range-Doppler RoI and the reflections. Then, the spectral embedding is concatenated with the reflections embedding before being used for target recognition.

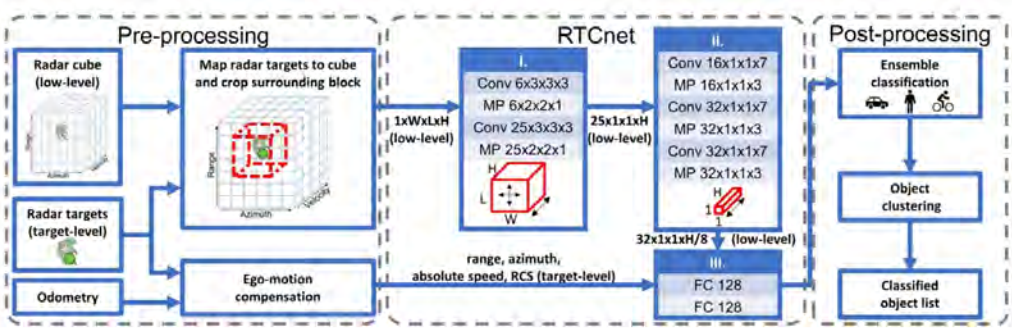


Figure 2.18: RTC-Net model overview. RTC-Net extracts RoIs from the radar cube using a list of detections. A combination of CNNs is used to extract features for each extracted RoI. Ensemble classifiers use the features to perform the target classification. Source: [Palffy 2020]

Features-based, spectrum-based or hybrid classification models rely on conventional radar object detectors and clustering methods to detect targets before classification. In addition, most of the methods above work for single-moving object scenarios and report results on proprietary datasets. It has been shown in [Gao 2019b] that, despite good classification accuracy, when tested in more complex scenarios (multiple objects, city roads), these models exhibit a lot of missing detection, which might be critical for safety. Adding a tracking algorithm on top of classification algorithms helps to reduce unwanted detection. However, we may wonder about the possibility of an all-in-one system to detect and classify objects simultaneously from raw data.

2.5.2 Object detection and segmentation

Deep learning-based object classification models rest upon thresholding algorithms to detect targets before identification and tracking. Nevertheless, thresholding, clustering and tracking algorithms are heuristic-based and sometimes produce false positives. Object detection and segmentation is one of the main tasks in computer vision, for which deep neural networks have achieved a significant breakthrough in the past decade. Such approaches have been successfully applied to LiDAR and cameras. However, as stated in Section 2.3, before 2021, the need for more publicly available radar datasets has slowed down research in object detection for radar data. Given the performance of object detection algorithms for computer vision, several researchers have attempted to exploit these models for radar object detection on different modalities (RA, RD, RAD) [Fatseas 2019, Kaul 2020, Zhang 2021, Li 2022]. Deep learning algorithms can leverage all the information in the raw signal to improve the detection and the classification of road users. Moreover, directly applying deep learning models on raw data allows to reduce the number of post-processing steps before detecting objects.

Object detection and segmentation on RAD tensors

In order to exploit all the available information in the radar signal [Gao 2021, Ouaknine 2021a, Franceschi 2022, Brodeski 2019, Major 2019, Zhang 2021, Giroux 2023, Fang 2022] conducted work on RAD tensors. Indeed, RAD tensors aggregate distance, velocity and angle information together. Brodeski *et al.* [Brodeski 2019] first introduced this kind of approach in 2019. They propose a two-stage detector on the RAD cube for detection and DoA estimation. To remain compliant with the conventional signal processing chain, they first detect objects in range and velocity using a U-Net-like [Ronneberger 2015] segmentation model. Then, for each detected object, they crop a RoI from the RAD cube and the detection network latent space. Finally, they pass it to a small network for

elevation and azimuth prediction. However, their data comes from an anechoic chamber with some corner reflectors inside, which is unrealistic. Franceschi and Rachkov [Franceschi 2022] extend this work to simulated radar data. They use the same network as [Brodeski 2019] and show a higher accuracy, recall and dice score than conventional methods. However, this work highlights the difficulty of deep neural networks to estimate azimuth and elevation in complex scenarios despite good generalisation results on real data for object detection. Moreover, the simulated data is unrealistic and looks closer to LiDAR data than radar data.

Similarly, Fang *et al.* [Fang 2022] introduce ERASE-Net, a two-stage detector called *detect-then-segment*. From a RAD tensor, they first detect object centres in RAD space, then extract and separate regions of interest from the background to form sparse point clouds. Lastly, they segment objects in RA and RD views using a sparse segmentation network for efficiency. In [Zhang 2021] Zhang *et al.* adapt the famous YOLO architecture [Redmon 2016] for 3D object detection on the RAD cube. They propose a backbone named RadarResNet that learns to extract velocity information in the channel dimension without 3D convolutions. Their model predicts object position in polar and cartesian coordinates, the latter providing the best detection result. However, the Doppler information is encoded as an extra channel. In computer vision, increasing the number of channels as we go deeper into the network is a good practice. Encoding the velocity in such a way might lead to a wrong estimation of the object's velocity.

To avoid this, multi-view models were proposed [Major 2019, Ouaknine 2021a, Gao 2021]. They use one encoder per view to extract information separately before

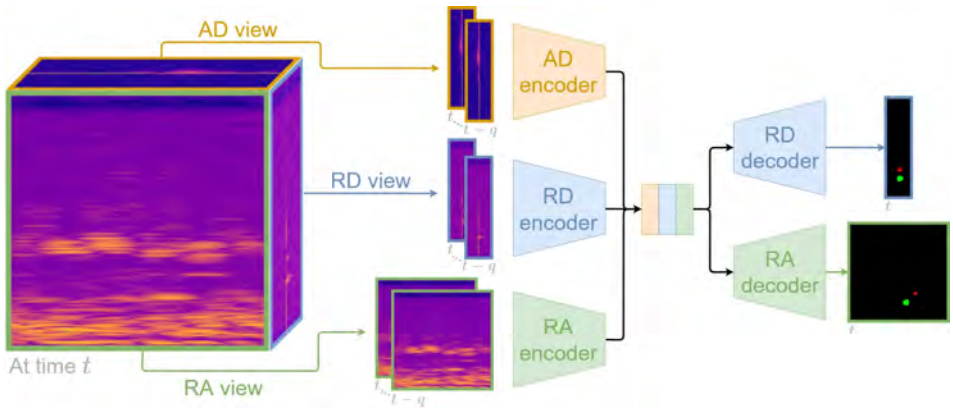


Figure 2.19: MVRSS framework. At a given instant, radar signals take the form of a range-angle-Doppler (RAD) tensor. Sequences of q + 1 2D views of this data cube are formed and mapped to a common latent space by the proposed multi-view architectures. Two heads with distinct decoders produce a semantic segmentation of the range-angle (RA) and range-Doppler (RD) views respectively. Source: [Ouaknine 2021a]

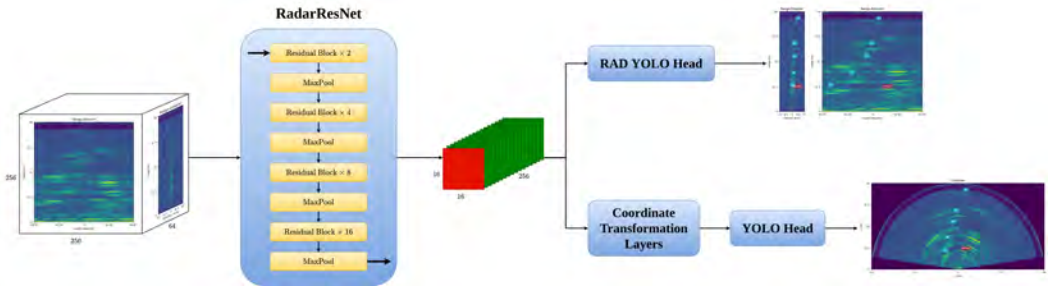


Figure 2.20: RADDet model. Features are extracted from the RAD cube with a custom ResNet model adapter to radar. Two YOLO heads are used to detect objects in the RAD and in the RA cartesian views. Source: [Zhang 2021]

merging it into a single latent space. For example, RAMP-CNN [Gao 2021] predicts barycentres in the RA domain using multiple views (RA, RD, AD) as input. The model comprises three different 3D convolutional autoencoders learning across multiple timesteps and domains. However, RAMP-CNN is huge (around 104 million parameters) and cannot be considered for real-time applications. Ouaknine *et al.* [Ouaknine 2021a] also introduce multi-view radar semantic segmentation (MVRSS) architectures to detect and classify objects in range-azimuth and range-Doppler domains (TMVA-Net and MVA-Net). As RAMP-CNN, they use one encoder per view and concatenate features from each view in a latent space. In order to handle the variability of radar objects' signature, Atrous Spatial Pyramid Pooling module [Chen 2018a] is used. They use the latent space to feed two decoders in charge of segmenting objects in RD and RA view. The models learn from past frames using 3D convolutions but only predict the positions of objects for the last timestep, making it more efficient than RAMP-CNN. Finally, Major *et al.* [Major 2019] perform bird-eye-view object detection in the RA domain using a multi-view model. Instead of using 3D convolutions to learn from time, they propose to add an LSTM cell on top of a detection head. One takeaway of their work is that predicting the position in cartesian coordinates instead of polar coordinates leads to higher detection accuracy. Indeed, it considers the increase in distance between adjacent bins when the range increases.

Nevertheless, RAD tensors are computationally demanding to produce and cumbersome in memory (especially for high-resolution radar). Multi-view models are hard and long to train and do not necessarily lead to better performance as shown by [Major 2019]. Since RA maps provide range angle information, thus allowing detection targets around the car, it has been explored extensively for object detection. RA maps are smaller than RAD cube, hence they are more efficient.

Object detection and segmentation on RA maps

Alongside the CRUW dataset [Wang 2021c], Wang *et al.* launch the ROD2021 challenge. The ROD2021 challenge came with the ROD2021 dataset, a subset of CRUW. This competition motivates research on new models for object detection using the RA modality. RODNet [Wang 2021b] has paved the way for a new radar object detection paradigm. To overcome the low resolution of radar, they propose to detect objects as points in RA view instead of using bounding boxes [Major 2019, Zhang 2021] or segmentation masks [Ouaknine 2021a, Kaul 2020]. That makes the detection task easier and well-posed when the boxes are not well defined, but reduces objects to a single point which is not always true. RODNet [Wang 2021b] consists of an hourglass [Newell 2016] 3D encoder-decoder model that predicts object location at multiple successive timesteps. However, as RAMP-CNN [Gao 2021], the models proposed by Wang *et al.* are huge (more than 100 million parameters). Ju *et al.* [Ju 2021] then introduce a lightweight module called Dimension Apart Module (DAM), which separately learns range, azimuth and time information to save computations. Zheng *et al.* [Zheng 2021] replace the 3D convolutions of RODNet with (2+1)D convolutions [Tran 2018]. They use ensemble learning to detect objects either in static or moving scenarios. Lately, Dalbah *et al.* [Dalbah 2023] exploit the power of the Transformer architecture [Vaswani 2017b] to solve the ROD2021 challenge. However, all these models process and predict data by batch of N frames and require a buffer of N frames to store in memory. 3D convolutions are used to learn spatio-temporal information, therefore the learned temporal context is not reused by the network from one batch to another. Moreover, because frames are treated and predicted by batch the methods presented above can be seen as non-causal because the convolutional kernel is applied on past and future frames. In real-time scenarios, one only needs to predict the position of objects for the last timestep, not for all the frames. In Chapter 4, we propose an alternative by predicting only the object position for the last frame based on the previous ones with recurrent neural networks to handle long-term dependencies.

Apart from the ROD2021 challenge, Dong *et al.* [Dong 2020] propose a probabilistic and class-agnostic object detector. Based upon the CenterNet [Zhou 2019] architecture, they model the uncertainty by predicting variances for bounding boxes orientation, size and offset. They also experiment with different types of RA inputs: polar or cartesian coordinates, with or without MUSIC [Schmidt 1986] super-resolution algorithm. Kaul *et al.* [Kaul 2020] present a weakly-supervised method using camera and LiDAR supervision semantic segmentation using scanning radar data. As in many works [Wang 2021b, Major 2019, Ouaknine 2021a], they use the time information and store it in the channel dimension. Using the same type of data, Li *et al.* use a Transformer-like module and computer vision backbones to

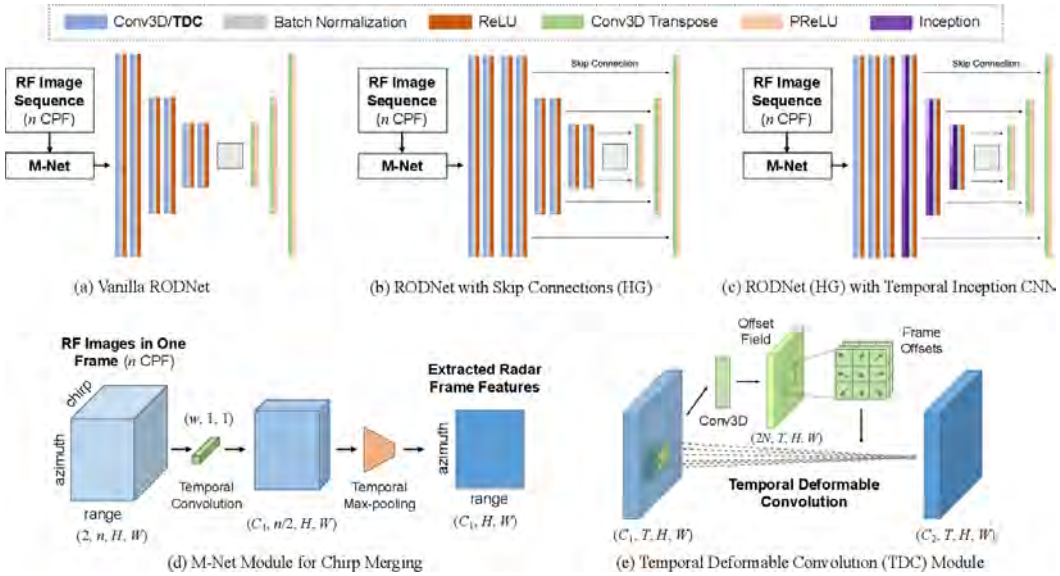


Figure 2.21: RODNet models. The authors propose three encoder-decoder architecture to predict the positions of objets in radar snippets. The M-Net model allows to merge RA maps from multiple chirps. Source: [Wang 2021b]

learn temporal dependencies between objects' features (size, position, shape) at two successive frames; then predict the class and position of objects. Madani *et al.* introduce a two-stream FPN-based car detection algorithm for cascaded MIMO radar using low-resolution and high-resolution RA maps to solve the misdetection problem due to high-speed vehicles in large cascaded MIMO arrays. In contrast, Meyer *et al.* [Meyer 2021] investigate if the information on the cartesian distance between data points in the radar signal can be used in graph neural networks to improve the performance of object detection tasks. Their model only detects cars but performs better than tensor-based approaches.

Object detection and segmentation on RD maps

The release of high-resolution radar datasets has marked the last year [Rebut 2022, Madani 2022, Paek 2022]. The higher the resolution, the more data to process and to store. As a result, it becomes unfeasible to use RA or RAD data for object detection. The range-Doppler spectrum is one of the most efficient representations available in radar. Indeed, it contains information about the distance and the velocity, and last but not least, it contains angle information through the antenna's dimensions. For efficiency reasons and before high-resolution radars, some prior works on RD maps using low-resolution radar have been done [Fatseas 2019, Dubey 2020, Guo 2022, Fatseas 2022]. In [Fatseas 2019], the authors use YOLO [Redmon 2016] object detector and Kalman filtering to detect and track

pedestrians and bicyclists in the range-Doppler domain. Dubey *et al.* [Dubey 2020] propose to use generative adversarial networks (GAN) [Goodfellow 2014] to detect the presence of targets in a scene. The generator is a U-Net [Ronneberger 2015] model, taking as input a RD spectrum, and the discriminator is an autoencoder which predicts whether the input is a detection mask or the ground truth mask. Using computer vision object detectors (YOLOX, D-DETR, SSD, RetinaNet, Faster R-CNN), Guo *et al.* [Guo 2022] first detect objects in RD view using a single frame. Then, they use Kalman filtering and Deep SORT algorithms to fix wrong detection made by the detection model based on historical information. For low-resolution radars, because of the lack of annotated datasets and the low resolution in angle, the methods mentioned above were mainly used as alternatives to CFAR algorithms.

High-resolution radar object detection and segmentation

Thanks to high-resolution radars and large-scale annotated datasets [Rebut 2022, Paek 2022], more and more researchers have started proposing new architectures for radar object detection using ADC data or complex MIMO RD spectra [Rebut 2022, Yang 2023, Giroux 2023]. All these works use ADC data or complex MIMO RD spectra to predict the position of objects in cartesian coordinates, with no need for complex DoA processing techniques. Rebut *et al.* try to replace conventional signal processing with deep neural networks to cope with the computationally demanding resource and memory footprint of high-resolution radar data. They propose a multi-task architecture composed of five blocks: a pre-encoder reorganising and compressing RD tensor into meaningful and compact representation, a shared FPN encoder learning semantic information, a range-angle decoder building a range-azimuth latent representation from the feature pyramid, a detection head localising vehicles in range-azimuth coordinates and a segmentation head predicting the free driving space. Giroux *et al.* [Giroux 2023] replace the backbone of [Rebut 2022] with a SwinTransformer backbone and use the ADC data instead of the complex MIMO RD spectrum. Signal processing is replaced by complex-valued linear layers, exploiting the prior knowledge of the Fourier transform, as in [Zhao 2023]. Similarly, Yang *et al.* [Yang 2023] learn to transform ADC data to RD latent space from the Fourier transformer algorithm. They build a dataset with ADC data and complex MIMO spectra pairs and learn semi-supervised to build complex MIMO spectra from ADC data. Then, they use the FFT-RADNet [Rebut 2022] model for object detection and free-space driving segmentation.

We saw in this section that there needs to be a consensus in the community about the type of data to use for automotive radar perception (RA, RD, RAD) and the formulation of the problem to detect and identify objects (detection with bounding boxes, segmentation, point-based detection). All of the methods, as mentioned

earlier, have advantages and drawbacks. Regarding the type of data, because of the size of high-resolution radar data, the use of complex MIMO RD spectra or ADC data seems to be the most promising and realistic research direction. Some works [Major 2019, Dong 2020, Rebut 2022] prefer to learn or to predict the position of objects in cartesian coordinates directly and show a slight performance improvement when using this representation. Other works [Wang 2021b, Ouaknine 2021a] directly predict objects' position in polar coordinates and map the prediction in cartesian coordinates afterwards without losing accuracy.

2.5.3 Data augmentation for radar

In computer vision, one technique to artificially augment a dataset, avoid overfitting and increase generalisation capability of a model is to use data augmentation. Basic transforms are image manipulation, image erasing or image mix. Image manipulation methods are basic image transformations. They include random flipping (horizontal, vertical), rotation, scaling or Gaussian noise addition. Image erasing methods idea is to delete one or more sub-regions in the image, then replace the pixel values of these sub-regions with constant or random values. The CutOut [Xu 2022] is one of them. ImageMix data augmentation receive increasing interest in the recent year. These methods are mainly completed by mixing two or more images or images sub-regions into one. MixUp [Zhang 2018], CutMix [Yun 2019] or AugMix [Hendrycks* 2020] are the main mixing data augmentation methods.

As mentioned in [Gao 2021], most of the existing data augmentation techniques algorithms we mentioned above cannot be applied to the radar data. Indeed, radar data differs from cameras images. Raw radar data has complex input, energy loss with range (a same object differs from one range, velocity or angle to another, see Equation 1.1), and non uniform resolution in the angular domain. In order to take into account these different factors, some works tried to exploit radar specific information to increase diversity of radar data [Gao 2021, Zheng 2021, Brodeski 2019, Sheeny 2020].

Similarly to camera images, we can horizontally and vertically flip radar spectrum without altering the data because radar has symmetric property in the Doppler and in the angular domains. Most the methods we mentioned in the previous section use flipping. Given the received power of an object varies with range and viewing angle, Sheeny *et al.* propose three data augmentation methods for radar classification: attenuation in range, change of resolution and background shift. Gao *et al.* also translate targets in range and angle by shifting cells in the polar-coordinates. A detailed view of the process is given in Figure 2.22.

Finally, Zheng *et al.* derive CutMix family algorithms for radar. Their SceneMix algorithm mixes RA snippets from different scenes. SceneMix comprises three

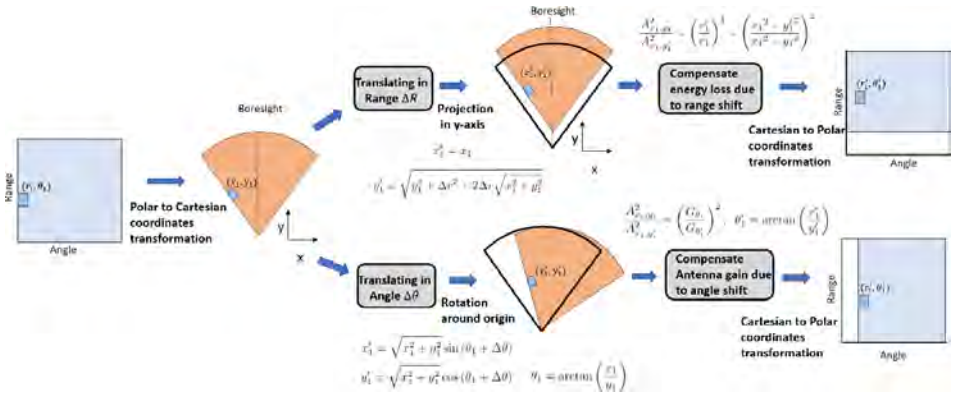


Figure 2.22: Translation in range and angle data augmentation. Source: [Gao 2021]

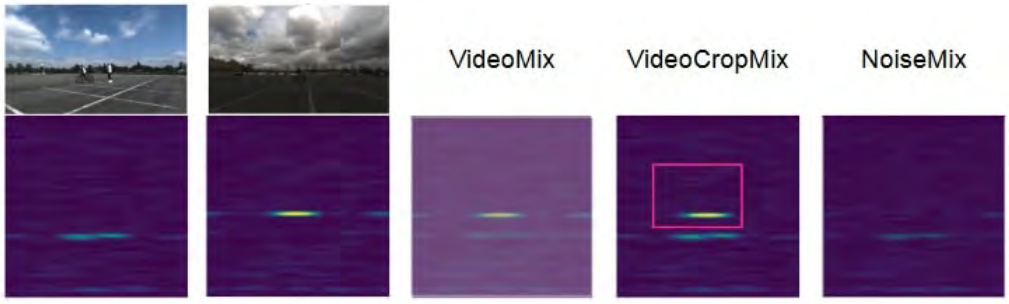


Figure 2.23: SceneMix augmentations. Source: [Zheng 2021]

mixing strategies: VideoMix [Yun 2020], VideoCropMix and NoiseMix. VideoMix mixes two RA snippets with random proportion. VideoCropMix randomly crop on a radar snippet and replace the cropped area with the corresponding area in another video. NoiseMix extracts noise from one radar snippet and add it to another snippet. Figure 2.23 shows an example of the SceneMix augmentation.

Chapter 3

Multiple road-users detection on Range-Doppler maps

Contents

3.1 Motivation

This chapter is dedicated to multiple road-users detection on range-Doppler spectra. Based upon the Faster R-CNN architecture [Ren 2017], we propose a new object detection and classification model to resolve road-users targets in distance and velocity. This chapter introduces a lightweight backbone for Faster R-CNN adapted to RD data. We design our model to handle the complexity of the RD maps and the small size of radar objects while trying to keep the processing pipeline as efficient as possible.

Chapter 2 indicates that radar point clouds are sparse, and the filtering techniques applied to the radar signal to obtain those reduce the information for target classification. Hence, the reflections list might hamper classification performance. Radar data can also be represented as raw data tensors (RD, RA or

RAD maps). Contrary to radar point clouds, such tensors benefit radar object detection because they represent the unfiltered signal. Prior works to this contribution show that deep learning models, and particularly CNNs, enable accurate object classification [Akita 2019], segmentation [Ouaknine 2021a] or detection [Meyer 2021, Wang 2021b, Zhang 2021] on raw data tensors. However, most of these works exploit the RA or the RAD views. Instead, we propose to build a model for object detection on RD maps in this chapter.

The use of RD maps instead of RAD tensors is motivated by the fact that RAD tensors are more computationally demanding to produce for radar microcontroller units (MCUs) and heavy in memory. Also, the RA view might not be an adequate representation for object detection since it does not account for Doppler, which is crucial information, as we will see in this chapter. Besides, the RA map usually suffers from a poor angular resolution caused by a few antennas in the FMCW radar. In this chapter, we hypothesise that the RD spectrum contains enough information for detection and classification tasks in automotive radar. Angular information can be computed for each target afterwards in a post-processing step, either using standard techniques or with AI as done by Brodeski et al. in [Brodeski 2019].

In the computer vision literature, one can detect an object by drawing bounding boxes around it (object detection) or attributing a class to every pixel in the image (image segmentation). Today, there has yet to be a consensus in the radar community about which task to use for radar object detection. Many automotive radar datasets (CARRADA [Ouaknine 2021b], RADDet [Zhang 2021], CRUW [Wang 2021c]) are annotated semi-automatically because radar data is difficult to annotate. Usually, an object detection model (Mask R-CNN [He 2017]) first detects objects on the camera. Then, the detection from the radar (the target list) and the object detection model are merged together to keep objects of interest. Finally, valid points are projected onto the radar view. Bounding boxes or segmentation masks are then created from those points. However, this process can lead to miss targets if the object detection models miss objects. Also, the points projected on the radar might not truly represent the targets because of the filtering operation in the radar signal processing chain.

This is why this work focuses on learning to represent targets as boxes instead of segmenting the RD map. According to the radar equation 1.1, the power received by the radar, thus the signature in the RD spectra, decreases proportionality to the distance to the power of four. The same car at five meters will have a different signature at 40 meters. While an image segmentation model learns regular shapes and pixel values, an object detection approach might be more robust to shape and intensities variation and less prone to overfitting. Indeed, the RD spectrum contains mostly noise, creating imbalance in the dataset. In contrast, object detection operates on a higher level by identifying and localising specific targets, allowing it

to be less affected by noise present in individual range-Doppler bins.

Section 3.2 introduces our model. Section 3.3 presents the settings and results of the experiment. In Section 3.4, we compare the results of our deep learning-based model with traditional radar object detectors. Finally, Section 3.5 discusses and concludes the chapter.

This chapter is mainly inspired by our article "DAROD: A Deep Automotive Radar Object Detect on Range-Doppler maps" published at IEEE Intelligence Vehicle Symposium (IV) [Decourt 2022a] [1].

3.2 Methodology

This section presents a lightweight Faster R-CNN architecture for object detection on Range-Doppler spectra. Given an RD map as input, we use a convolutional neural network to learn relevant features, as in Faster R-CNN. Following the feature extraction, we use a region proposal network (RPN) to propose spectrum regions containing potential targets. A small network is slid over the learned convolutional feature map to generate region proposals. For each point in the feature maps, the RPN learns whether an object is present in the input image at its corresponding location and estimates its size. A set of *anchors* is placed on each location's output feature maps' input image. These anchors indicate possible objects in various sizes and aspect ratios at this location. We refer to Section 2.2.2 for more detailed information about RPN and anchors. Next, the bounding box proposals from the

[1] The code of this work was made publicly available here: `https://github.com/colindecourt/darod/`

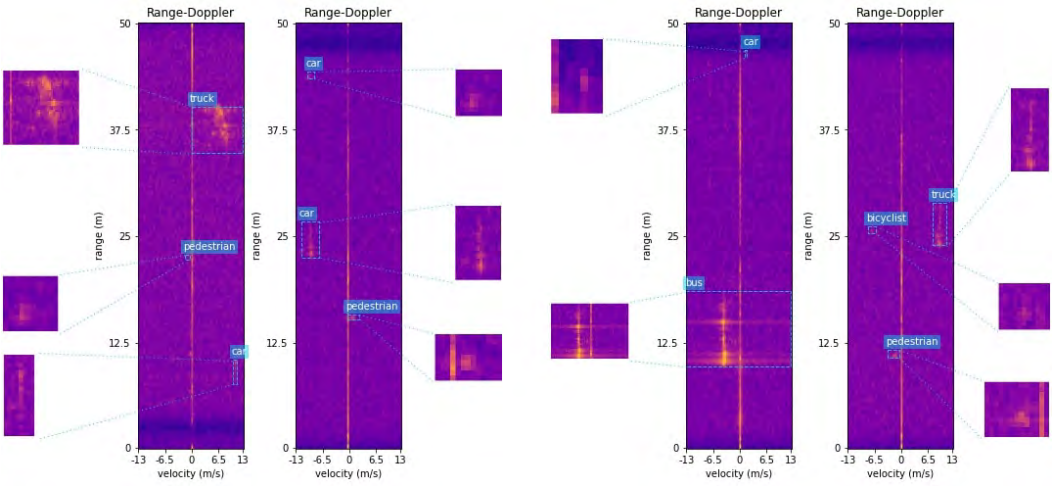
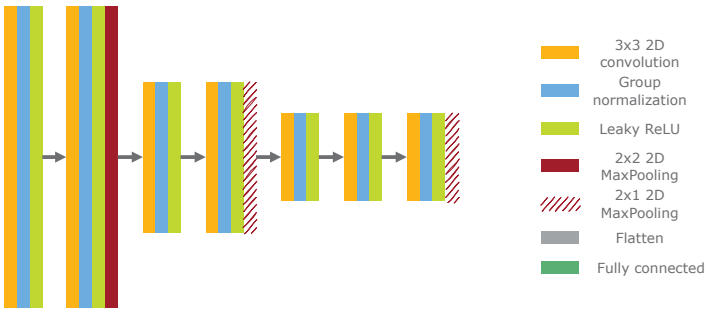


Figure 3.1: Road users signature in range-Doppler view. We show two RD maps of RADDet dataset, along with the bounding boxes around objects and their zoom.

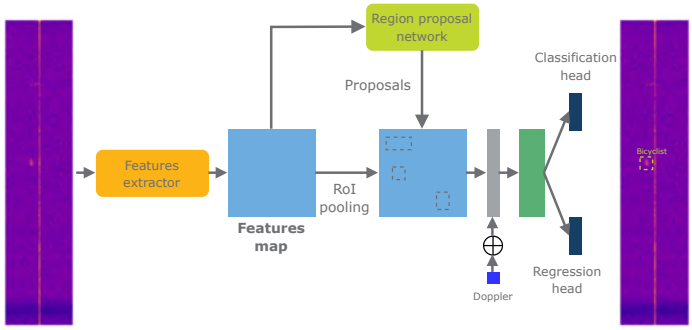
RPN are used to pool features from the backbone feature maps. This study uses a pooling size of 4×4. These features are used to classify the proposals as background or object and to predict a bounding box using two sibling fully connected layers. This second part is named Fast-RCNN [Girshick 2015]. We depict this pipeline in Figure 3.2b.

We show in Figure 3.1 two RD maps with radar signatures of some objects in the captured scene of the RADDet dataset. Even though those RD maps seem complex, their information remains of low complexity, contrary to camera images which are bigger and more diverse in textures, orientations, geometry, and lighting. Although noisier, RD maps have fixed orientation, and their objects exhibit more similar patterns and shapes.

To account for those differences, we modify Faster R-CNN to include a lighter backbone and a modified RPN. Our backbone is derived from the VGG architecture [Simonyan 2015] and contains seven convolutional layers. Figure 3.2a depicts this lightweight backbone architecture. To keep the processing pipeline as simple and



(a) Proposed light backbone architecture



(b) Faster R-CNN architecture

Figure 3.2: DAROD overview. **(a)** DAROD backbone. We propose a simple feature extractor derived from the VGG architecture which contains seven convolutional layers. **(b)** Overview of the Faster R-CNN architecture. First we extract feature from a RD map. Then, the RPN make proposals using DAROD's feature maps. For each proposal, we extract a RoI from the feature maps and we classify it as an object or not.

efficient as possible, we decide not to resize the spectrum and to process it as it is, resulting in an input of size 256×64. Indeed, our goal is to study if an object detection model performs better than conventional detection algorithms like CFAR. Therefore, to add such an algorithm in the radar processing chain (see Figure 1.5), we prefer to adapt the backbone to the output of the signal analysis block. The backbone comprises two blocks with two 2D convolutions and one with three 2D convolutions. Following each convolutional block, we apply a 2D max pooling operation to down-sample the input size. Because the dimension is smaller than camera images and to minimise the loss of Doppler information which is helpful for classification, we down-sample the Doppler dimension only by a factor of two after the first block. Then, we obtain a set of 32×32 feature maps, which led to the best performance. The number of channels for each block of convolutions is respectively set to 64, 128 and 256.

The next step is to define the anchors used by the RPN to capture the objects' diversity of shape and size. In this work, we use three scales and three aspect ratios to generate anchors, yielding nine anchors at each position in the feature map. The mean size of objects in RD maps is 8×8. We use this size as a reference for the anchor's scale. We use scales of 4 and 16 for smaller and bigger objects, resulting in anchors scales of sizes $[4, 8, 16]$. Additionally, we set aspect ratios to $[\frac{1}{4}, \frac{1}{2}, \frac{1}{8}]$. Contrary to Faster R-CNN where aspect ratios are set to $[1, 2, \frac{1}{2}]$, we choose aspect ratios where denominators are multiples of 4 to account for the $\frac{1}{4}$ ratio between input height and width. To reduce the computational complexity of the model, we do not consider all the combinations of scales and ratios. We only consider combinations containing scale eight and ratio $\frac{1}{4}$ resulting in fewer anchors generated per image (5 at each position). We find these settings provide the best performances.

Since the RD spectrum is not translation invariant (the velocity is a characteristic of the target), we decide to add this information to the feature vector used for classification and bounding box regression. This feature vector corresponds to the flattened region proposed by the RPN. We compute the velocity by extracting the top-k (k is set to 3) pixel positions with the highest intensities in the proposed RoI. Knowing the velocity resolution of the radar δ_v and the position of the i^{th} highest pixel in the RoI p_i, we compute the i^{th} velocity using the following formula: $v_i = \delta_v \cdot p_i$. We notice a slight improvement in the performances using the Doppler values as extra features.

We optimise the model using the loss functions described in [Ren 2017]. For training RPN, a binary class label is assigned to each anchor. To take into account the uncertainty of the annotations, we assign a positive label to anchors having a high IoU overlap with a ground-truth (GT) box or having an IoU overlap higher than 0.5 (instead of 0.7 in the original paper [Ren 2017]). We assign a negative

label to an anchor if it has an IoU overlap lower than 0.3 for all GT boxes. Anchors that are neither positive nor negative do not contribute to the training objective. According to [Ren 2017], we minimise the following multi-task loss to train the RPN:

$$\mathcal{L}_{rpn} = \frac{1}{N_{cls}} \sum_i \mathcal{L}_{cls}(p_i, p_i^*) + \frac{1}{N_{reg}} \sum_i p_i^* \mathcal{L}_{reg}(t_i, t_i^*), \tag{3.1}$$

where i is the index of an anchor in a mini-batch, and p_i is the predicted probability of anchor i being an object. p_i^* is set to 1 if the anchor is positive and 0 if the anchor is negative. t_i is a vector representing the coordinates transformation between the predicted box and an anchor. t_i^* represents the coordinates transformation between an anchor and the GT box. The classification loss $\mathcal{L}_{cls}$ is a binary cross-entropy. The regression loss $\mathcal{L}_{reg}$ is the Huber loss defined in [Huber 1964]. N_{cls} is the mini-batch size, representing the number of proposals (positive and negative) to use for training the RPN. Here we set N_{cls} to 32 as there are few objects in our RD data. N_{reg} is the number of positive anchor locations.

For training the second part of the network (the detection head $\mathcal{L}_{det}$), we use a similar multi-task loss as for the RPN. We train the regression head using the same loss function as the RPN and replace the binary cross entropy with multi-class cross entropy. As a result, we optimise the following loss function:

$$\mathcal{L} = \mathcal{L}_{rpn} + \mathcal{L}_{det} \tag{3.2}$$

3.3 Experiments and results

3.3.1 Datasets and competing methods

We train our model on the two publicly available radar datasets CARRADA [Ouaknine 2021b] and RADDet [Zhang 2021]. For the CARRADA dataset, we use the segmentation masks as a reference to create our bounding boxes by drawing a box around masks. For each instance, we take the minimum and the maximum (x, y) coordinates of the segmentation masks to create the bounding boxes. Regarding the RADDet dataset, we extract the RD maps by summing the values of the RAD tensors over the angle dimension. We use the same bounding boxes provided by the authors of the RADDet dataset by only taking coordinates along the range and the Doppler dimension. We use the default train/val/test distribution of the CARRADA dataset. For RADDet, we randomly split the train into training and validation sets with a 9:1 ratio. For testing, we use the provided test set.

We compare our model DAROD, made of the lightweight backbone and the simplified Faster R-CNN architecture displayed in Figure 3.2, with the RADDet model [Zhang 2021]. At the time of this study, it was the only published object detector designed for radar data. We modify the RADDet model to train it only with

Model	IoU 0.3			IoU 0.5			# params (M)	Runtime (ms)
	mAP	Precision	Recall	mAP	Precision	Recall		
DAROD (ours)	68.26 ± 0.08	**79.84**	48.37	58.20 ± 0.03	**74.31**	44.19	**3.4**	**25.31**
Faster R-CNN (pretrained)	**71.08 ± 0.12**	51.70	72.97	**64.56 ± 0.09**	47.86	67.21	41.3	37.19
Faster R-CNN	64.21 ± 0.07	45.90	**74.17**	52.93 ± 0.06	41.59	**67.40**	41.3	37.19
RADDet RD	48.59 ± 0.05	61.31	42.56	18.57 ± 0.08	36.73	25.50	7.8	74.03

Table 3.1: Results of different models on CARRADA dataset.

RD maps as input instead of RAD tensors. We also consider the variant RADDet RAD, corresponding to the original RADDet model (train on RAD tensors) evaluated only on the range and the Doppler dimensions, using the pre-trained weights provided in [Zhang 2021]. As a second baseline, we consider the state of the art in computer vision by selecting the Torchvision[2] Faster R-CNN implementation using the default hyper-parameters, namely a resizing of the input from 256×64 to 800×800 and a ResNet50+FPN backbone pre-trained on ImageNet. In addition, we train the Torchvision Faster R-CNN without the pre-training on ImageNet to evaluate the impact of this pre-training on the results.

3.3.2 Training setting and evaluation metrics

We use the Adam optimiser with the recommended parameters and a learning rate of 1×10^{-4} for all our experiments. We set the batch size to 32 for both datasets. Our model has trained over 100 and 150 epochs for CARRADA and RADDet datasets. As the Faster R-CNN object detector contains several hyper-parameters, we perform a grid search over some carefully chosen parameters to improve the performance of our model. We randomly use horizontal and vertical flipping as data augmentation strategies.

We evaluate our model using the mean average precision (mAP), a well-known metric for evaluating object detectors. We consider mAP at IoU thresholds 0.3 and 0.5 to consider the uncertainty of the annotations, which are generated semi-automatically for both datasets as discussed in Section 3.1. In addition, we provide precision and recall at IoU thresholds 0.3 and 0.5. All the experiments are conducted using the Tensorflow[3] deep learning framework and an Nvidia RTX 2080Ti GPU.

3.3.3 Results

Tables 3.1 and 3.2 show the performance of our model on CARRADA and RADDet datasets[4]. Our DAROD model outperforms the RADDet method on both datasets while it remains competitive with Faster R-CNN. When pre-trained on ImageNet,

[2] `https://github.com/pytorch/vision`

[3] `https://www.tensorflow.org/`

[4] We train all the models ten times, and we show the mean results for each in Table 3.1 and 3.2

Model	IoU 0.3			IoU 0.5			# params (M)	Runtime (ms)
	mAP	Precision	Recall	mAP	Precision	Recall		
DAROD (ours)	**65.56 ± 0.83**	**82.31**	47.78	46.57 ± 0.7	**68.23**	38.74	**3.4**	**25.31**
Faster R-CNN (pretrained)	58.47 ± 0.67	52.17	56.92	**49.55 ± 0.72**	47.78	51.77	41.3	37.19
Faster R-CNN	49.16 ± 0.56	32.33	**61.46**	40.84 ± 0.61	29.37	**55.29**	41.3	37.19
RADDet RD	38.42 ± 1.12	78.20	29.77	22.87 ± 1.45	60.41	20.55	7.8	74.03
RADDet RAD [Zhang 2021]	38.32	68.80	26.83	17.13	46.55	16.99	8	75.2

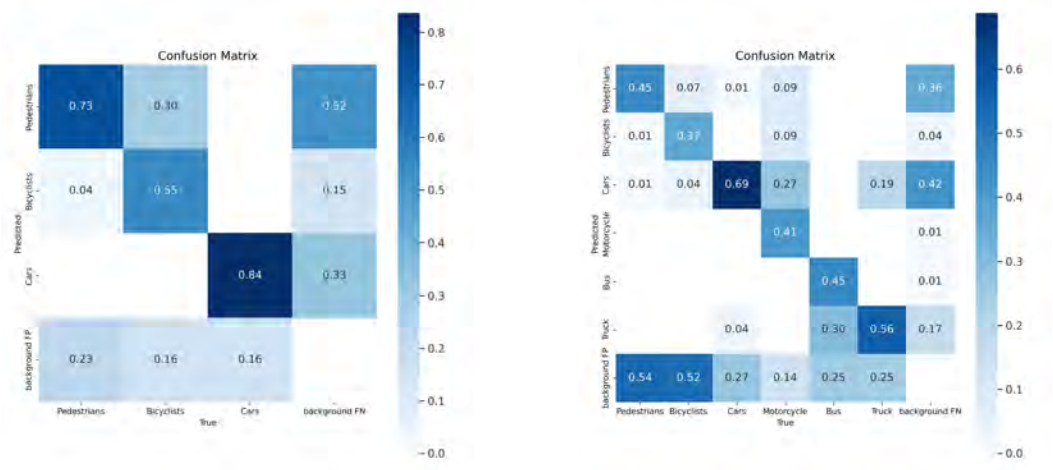
Table 3.2: Results of different models on RADDet dataset. We do not report mean and standard deviation for RADDet RAD as we report the results from the paper.

Faster R-CNN leads to the best mAP in 3 cases, with DAROD being second best, the positions being inverted in the last experiment (RADDet dataset and IoU at 0.3).

Generally, we observe that DAROD achieves good precision scores but medium recall. This suggests that our model accurately classifies targets when there are detected but misses some objects present in the scene. The confusion matrices in Figure 3.3a confirms this interpretation. For each class, we notice that 20% of the time, targets are detected while there are no objects in the image (last line of the confusion matrix). The confusion matrix's last column also shows that DAROD tends to miss objects in the scene, which can be problematic for critical applications. We explain this behaviour because we aimed to optimise mAP, which measures the global performance of object detectors. We might be able to improve the recall by reducing the selectivity of our model during training and in the post-processing step or by decreasing the penalty of classification errors. Finally, because of their similarities (velocity, RCS), we notice confusion between pedestrians and bicyclists. Mainly, pedestrians are classified as bicyclists. On the contrary, cars are either correctly classified or missed. Examples in Appendix A show some failure cases of DAROD (missed targets and confusion between similar classes).

We draw the same conclusion for the RADDet model, which obtains decent precision scores but low recall, impacting mAP@0.3 and mAP@0.5. The confusion matrix in Figure 3.3b shows many pedestrians and bicyclists false positives and mostly missed cars and pedestrians. Under-represented classes (bicyclists, motorcycles, buses) are rarely missed. As for the CARRADA dataset, we notice confusion between similar classes (bus and truck here), which raises the question about the necessity of labelling such classes.

The original version of the Faster R-CNN model achieves sufficient precision scores and good recall, resulting in more false positives but fewer missed targets, which may be better for critical applications. In this implementation, because the input spectrum is upsampled targets are bigger, therefore they match more anchors than in our implementation. The number of positive labels to train the RPN and the Fast R-CNN part is also higher. This is why the recall of Faster R-CNN is better than ours. However, upsampling the input might change the radar signature which



(a) DAROD confusion matrix (CARRADA dataset).

(b) DAROD confusion matrix (RADDet dataset).

Figure 3.3: DAROD confusion matrices on CARRADA and RADDet datasets. DAROD tends to miss objects in the scene, and struggle to classify similar classes correctly, which can be problematic for critical applications

can affect the classification results. Also, because of its size Faster R-CNN is more subject to overfitting than DAROD. Finally, the pretraining of the Faster R-CNN backbone on the ImageNet dataset helps to improve the detection performance. It drastically improves the precision score but does not impact the recall score. This is interesting because the features in ImageNet are highly different from radar data. This suggests the network uses the shapes and the patterns learned on ImageNet to find objects in the spectrum. We discuss further pretraining strategies in Section 3.5.

A critical point in automotive radar is the computational load of the different models. We compute the FLOPS (floating point operations per second) of the different models and represent it as a function of the performance in Figure 3.4. Not surprisingly, radar based approaches are far more efficient than Faster R-CNN that uses up-sampling and deeper backbones. RADDet model is the model with the lowest number of FLOPS as it is inspired from the single stage detector YOLO [Redmon 2016]. The number of FLOPS required by DAROD is slightly bigger than RADDet, but stays reasonable to run on microcontrollers.

3.3.4 Ablations studies

Impact of additional features

We add the velocity of each detected target to the feature vector used for the classification. We try to add this information in different ways:

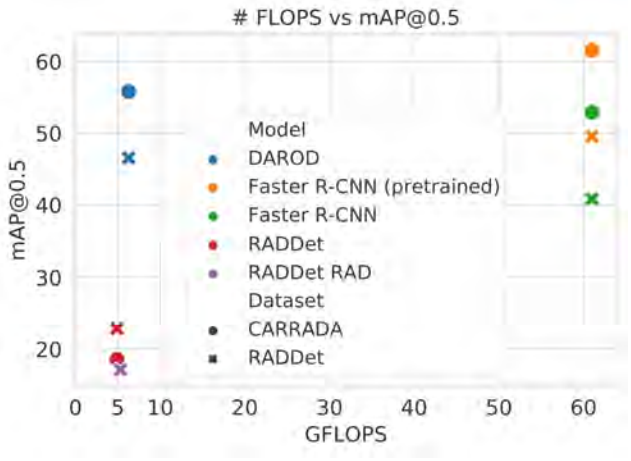


Figure 3.4: Number of FLOPS vs. mAP@0.5 for DAROD, Faster R-CNN (pre-trained or trained from scratch), RADDet and RADDet RAD.

1. We extract the range and the velocity values from the centre of the RoI.
2. We extract the top-k maximum intensities from the RoI and extract the top-k range and velocity values from it, with $k = 3$.
3. We compute a range and velocity grids, then use these grids as additional input channels.

Figure 3.5 summarises the different methods.

We report in Table 3.3 the detection results on the CARRADA dataset when adding the following features: the range, the velocity and the range and the velocity values. Overall, adding features about the position and the velocity of targets boost the mAP. We notice that extracting the top-3 maximum intensities from the RoI lead to the best results. Indeed, the successive downsampling stages and the RoI pooling operation result in an approximate location of the target. Selecting the centre of the RoI as a reference might extract features from the background (*i.e.* noise) instead of from the target.

On the contrary, choosing the top-k maximum intensity will help to correct localisation error by putting more weights on high-intensity values (*i.e.* foreground). Finally, using the range and Doppler grids as additional input channels does not improve the results. We imagine the distance and Doppler information are lost and do not flow through successive layers. However, we see a slight improvement when using a Doppler grid, which suggests the targets' velocity is helpful.

Size of the feature map

We study the impact of the size of the feature maps on the mAP. We do not consider feature maps bigger than 32×32 to save computations. We experiment with square

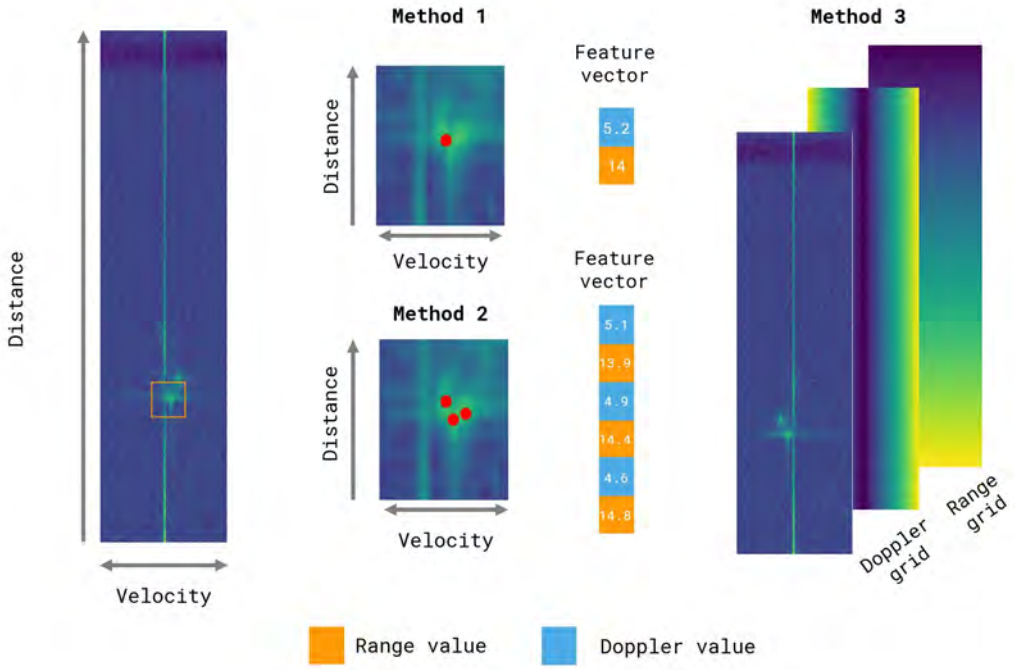


Figure 3.5: DAROD additional features extraction process summary.

features maps (16×16 and 8×8) and rectangular features maps to keep the ratios of the input (32×8 and 16×8). Table 3.4 shows a 32×32 feature maps provides the best results. The smaller the feature maps, the less accurate the model. While IoU thresholds increase, the mAP of models using larger feature maps increase. We experimented with smaller pooling sizes in Faster R-CNN for smaller feature maps and did not notice any improvements. We use the CARRADA dataset for these experiments.

Features	Maximum (top-3)	Centre	Grid	No features
Range	55.87	55.52	55.42	
Doppler	58.10	55.73	56.19	
Range & Doppler	**58.20**	53.13	52.34	
No features				54.85

Table 3.3: mAP@0.5 for DAROD when adding additional features (range, Doppler or range and Doppler) to the feature vector of detected targets. The last line reports the mAP@0.5 for a model without additional features.

Feature maps size	mAP@0.1	mAP@0.3	mAP@0.5	mAP@0.7
32×32 (baseline)	73.33	68.26	58.20	29.28
16×16	62.11	57.76	47.50	20.15
8×8	50.22	45.37	33.22	9.11
32×16	59.92	55.58	44.83	20.32
16×8	52.49	47.17	38.01	15.35

Table 3.4: DAROD mAP for different feature maps size. We report the mAP at different thresholds to show how the size of the feature maps affects location accuracy. Experiments are on the CARRADA dataset.

3.4 Comparison with conventional object detectors

In this section, we aim to compare our DAROD model with conventional object detectors. The goal of this thesis being to reduce the complexity of the radar signal processing chain (see Figures 1.5 and 1.13). This chapter shows that deep learning-based object detectors can be considered alternatives to CFAR detectors and their associated post-processing steps. In the CARRADA dataset, DoA points from the radar are available. They are obtained using the processing chain presented in Figure 1.5. Because these points are potential objects, we consider two baselines.

In Section 3.4.1 we evaluate DAROD for detection only (objects/no objects) and we compare it with the DoA points. However, whereas the CFAR algorithm detects objects based on their intensity, our model was trained to detect specific objects like cars or pedestrians explicitly. Therefore, in Section 3.4.2 we consider a second baseline with an AI-based classifier which learns to remove background detections (binary) or to classify them (multi-class).

3.4.1 Binary object detection

To compare DAROD with conventional object detectors, we project the DoA points on the RD spectrum as our model detects objects in the RD view. Unfortunately, we can not compute the DoA for the CARRADA dataset because we do not have the ADC data. As our model outputs boxes, we cluster the radar detection using DBSCAN and draw a bounding box around each cluster following the same method as in [Ouaknine 2021b]. Figure 3.6 summarises the process, and Table 3.5 shows the mAP at different IoU thresholds.

Table 3.5 shows that our approach outperforms conventional radar object detectors. We consider another baseline with an AI-based classification stage (binary and multi-class). We describe this approach in detail in Section 3.4.2. We note that using a classifier which learns to remove background objects increases the overall detection performance. The results highlight the relevance of using deep-learning-

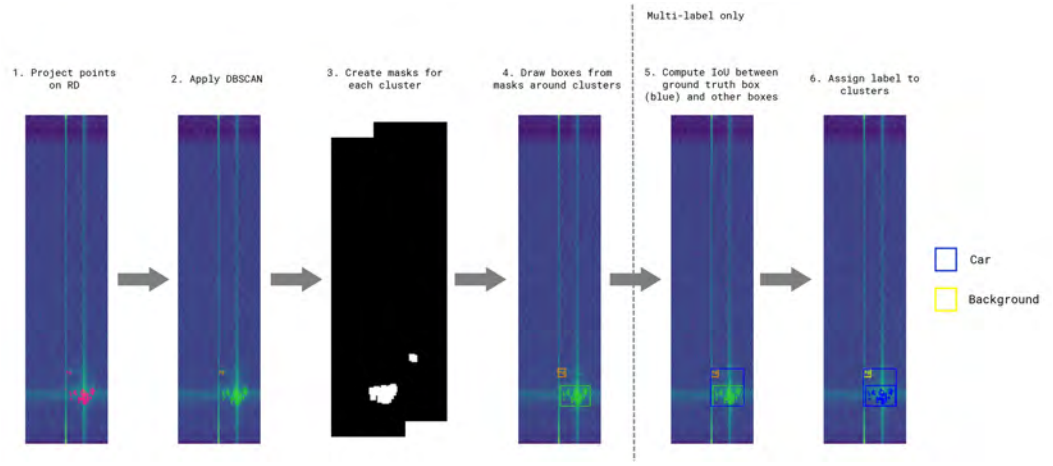


Figure 3.6: CARRADA Point Clouds dataset generation. After projecting point clouds on the RD map (pink points), we cluster points using the DBSCAN algorithm (green and orange points). For each cluster, we create a segmentation mask using the same procedure as [Ouaknine 2021b]. From the segmentation masks, we draw bounding boxes around each cluster to obtain box for binary detection (green and orange boxes). For the multi-class detection, we then compute the IoU between ground truth box (blue box) and boxes from CFAR+DBSCAN. If the IoU is greater than a threshold, we label the box from CFAR+DBSCAN using the ground truth label (a *car* here) and set the label of the last box to *background*.

based object detectors for the detection tasks. Indeed, DAROD exhibits better localisation precision, resulting in a better AP for object detection.

3.4.2 Multi-class object detection

Dataset Following the clustering stage, we use the DeepReflecs model [Ulrich 2021] to classify the detected point clouds as background or objects (cars, pedestrians, bicyclists). We create the CARRADA Point Clouds dataset by mapping the point clouds with the labels provided in [Ouaknine 2021b]. To build the dataset, we map the ground truth labels available in the CARRADA dataset on the DoA points. For each detected cluster, we draw a bounding box around it. We then compute the IoU between the cluster bounding box and all the ground truth boxes.

	mAP@0.1	mAP@0.3	mAP@0.5
CFAR+DBSCAN	46.88	39.23	26.34
CFAR+DBSCAN+DeepReflecs (binary)	69.90	60.10	37.26
DAROD (detection only)	86.10	77.85	66.66

Table 3.5: Object/no object detection performance comparison between DAROD and a traditional radar object detection.

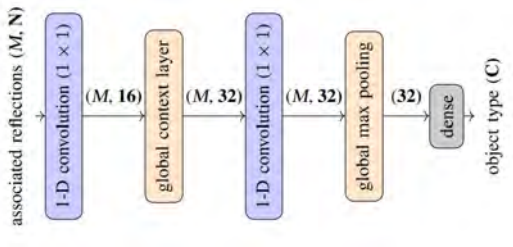
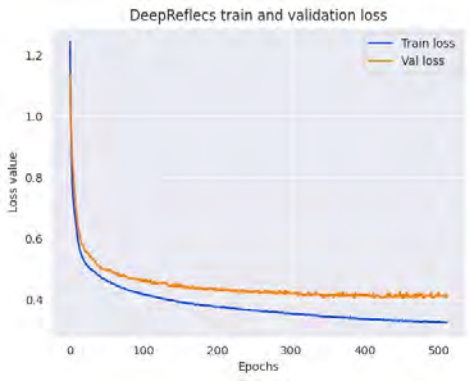


Figure 3.7: DeepReflecs model architecture [Ulrich 2021]

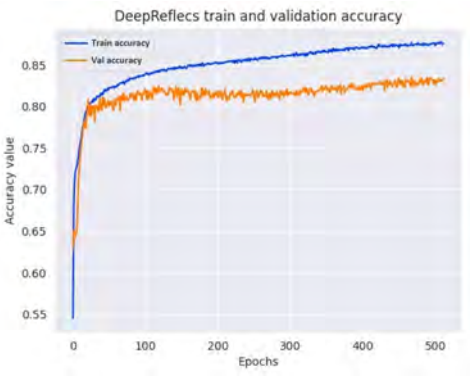
If the IoU between the cluster bounding box and a ground truth box is higher than a threshold, we label the cluster with the ground truth bounding box label. The ground truth box is removed from the label list, and we repeat the operation for all the clusters. Non-matched clusters are then labelled as background. Figure 3.6 illustrates the process.

Model Using our CARRADA Point Clouds dataset, we train the DeepReflecs model for object classification. DeepReflecs [Ulrich 2021] takes as input a list of M reflections with five features as input. In the original paper, the authors use the final detection list, which contains the position of the object in cartesian coordinates, the velocity, the RCS and the range of the target. In this study, we work in the RD domain. Thus we do not aim to estimate the object's DoA. We slightly change the features for the classification accordingly, and we use the following features: the range, the velocity and the RCS of the reflection, and the mean range and velocity value of the cluster.

We train the model over 500 epochs using Adam optimiser and a learning rate



(a) DeepReflecs train and validation loss



(b) DeepReflecs train and validation accuracy

Figure 3.8: Train and validation metrics (loss and accuracy) for the DeepReflecs model on the CARRADA Point Clouds dataset.

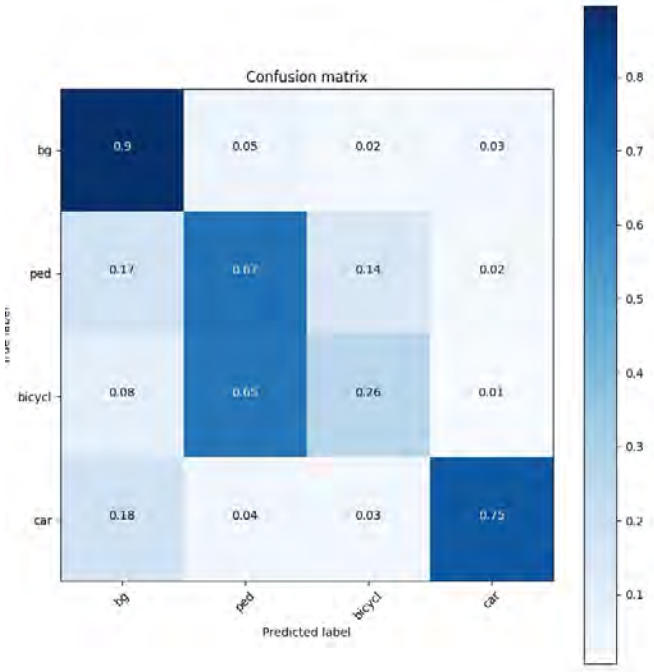


Figure 3.9: DeepReflecs confusion matrix on the CARRADA Point Clouds dataset

	mAP@0.1	mAP@0.3	mAP@0.5
DAROD (ours)	73.33	68.26	58.20
RADDet RD [Zhang 2019]	59.63	48.59	18.57
CFAR+DBSCAN+DeepReflecs	39.81	34.69	19.57

Table 3.6: Comparison between deep learning object detectors and a conventional approach with DeepReflecs [Ulrich 2021] model.

of 1×10^{-4}. The batch size is set to 512. Figure 3.7 shows the DeepReflecs model.

Classification results For classification only, DeepReflecs reaches an accuracy of 80.33%. Performance can be further improved by using re-sampling strategies to decrease the imbalance in the dataset. Figure 3.8 shows the train and validation loss and accuracy on the CARRADA Point Clouds dataset. Also, we report in Figure 3.9 the confusion matrix of DeepReflecs on the test set. As for the detection task, bicyclists are often classified as pedestrians. Generally, the model correctly classifies background objects.

Object detection results We report in Table 3.6 the mAP of deep learning radar object detectors and a hybrid radar object detector (CFAR+DBSCAN+DeepReflecs). Section 3.4.1 shows that combining CFAR and

DBSCAN approaches lead to lower localisation accuracy than deep learning object detectors. Table 3.6 confirms that deep learning models (DAROD or RADDet) achieve higher mAP than conventional approaches with point cloud classification networks. Nevertheless, such an approach reaches better mAP@0.5 than RADDet RD [Zhang 2019]. We think the mAP of the CFAR+DBSCAN+DeepReflecs model could be improved by improving the accuracy of DeepReflecs (thus trying to reduce the confusion between cyclists and pedestrians).

3.4.3 Discussion

Although DAROD outperforms traditional methods (with and withou AI) in binary and multi-class object detection tasks, the comparison could be unfair. First, the CARRADA dataset is made to detect **only** pedestrians, bicyclists and cars. However, many other targets or reflections in the scene affect the detection results (binary task). We noticed that using a classification network helps to increase the overall performance of the approach, but the mAP at high IoU remains low. Second, because we formulate the detection problem as bounding boxes, we evaluate the conventional approach using the mAP metric. We created the bounding boxes from the point clouds of each cluster using the same code as the authors of the CARRADA dataset. However, in the CARRADA dataset, the authors correct DBSCAN's clusters using the prediction of a Mask R-CNN model [He 2017] they project on the RD spectra. As a result, IoU between boxes from CFAR+DBSCAN and the ground truth might be low for objects corrected in the annotations process, hence decreasing IoU at higher thresholds (greater than 0.5). Figure 3.6, which shows how we build the dataset, illustrates this issue. We think the mAP@0.3 is a good starting point to compare different deep-learning and non-deep-learning methods. Although less accurate than our approach, the CFAR+DBSCAN+DeepReflecs approach achieves decent results at this threshold. For the multi-class object detection, this method has low mAP@0.3 compared to DAROD and RADDet [Zhang 2021]. Decreasing the class imbalance in the dataset and performing grid-search over hyper-parameters could improve the accuracy of the classifier and the overall mAP. We did not conduct such experiments as this is not the purpose of this thesis.

3.5 Conclusion and perspectives

In this chapter, we presented an adaptation of the Faster R-CNN model for object detection on range-Doppler maps. First, we show that a deep learning model can achieve good detection performance compared to traditional radar object detectors. Second, we demonstrated that our DAROD backbone attains higher mAP than the original implementation of Faster R-CNN when no pre-training of the backbone is

done, using ten times fewer parameters. Also, we show that our model outperforms the radar-based model RADDet [Zhang 2021] on the CARRADA and the RADDet dataset.

We designed our backbone to deal with range-Doppler data specifically. We experimented with different pooling and feature map sizes, proving that the distance and, mainly, the velocity information are crucial to improving detection and classification results. We introduced three methods to add the distance and the velocity information into the Faster R-CNN framework. We found sampling the top-k highest intensity values from the detected RoI to provide the best results.

Finally, we compared our model with conventional radar object detectors (*i.e.* CFAR + DBSCAN) with and without object classification. Experiments show that deep-learning object detectors improve localisation precision and yield fewer false positives. This comparison confirms the promise of deep learning applied to automotive radar.

Comparison between deep-learning object detectors and conventional object detectors In Section 3.4, we studied CFAR detectors' performance and analysed their performance against DAROD. For a fair comparison, we trained a point cloud classifier, DeepReflecs [Ulrich 2021], to classify targets as background or road users (pedestrians, bicyclists and cars). We found that adding a classifier increases the performance of conventional object detectors. Nevertheless, the DeepReflecs performance could be improved by making the association between points and label better (see Figure 3.6), reducing the imbalance in the dataset and searching for better hyper-parameters. Moreover, we compared our model with the detections projected on the RD spectrum. A better comparison would be to estimate the DoA of objects detected by our method and then compare our point clouds with those from the radar. Such comparison could be done using the RADDet [Zhang 2021] or the RADIal [Rebut 2022] datasets as they contain the ADC data required for running the entire processing chain. However, the whole processing chain must be implemented to obtain the default point clouds.

Comparison with the RADDet model In section 3.3.3, we show that our model achieves much better mAP than the RADDet model on CARRADA and RADDet datasets. We want to emphasise that the RADDet model was specifically designed to process RAD tensors instead of RD spectra. Therefore, it might be inefficient on RD spectra, and the results might suffer from this difference. For this reason, we evaluated the RADDet model on the range and the Doppler dimensions only, using the pre-trained model provided by the authors. Results are given in Table 3.2 as the RADDet RAD model. Results show that an RD-only model achieves better mAP than the RAD model. Because RADDet uses the Doppler information

in the channel dimension, we conclude that the model is better at detecting objects in the RA or cartesian space than in the RD view.

Model efficiency We conducted experiments to compare our lightweight radar-based model with the original implementation of Faster R-CNN. At a similar feature map resolution as DAROD, the Faster R-CNN model we compare with achieves good performance. However, regarding the large number of parameters and FLOPS of this model, the gain in performance we observe does not improve by far the results. We concluded that using very deep convolutional neural networks to extract meaningful information from radar data is not essential. Naturally, this is true for the CARRADA and the RADDet datasets as they are small. With larger datasets, we might enlarge the size of the models to increase their learning capacities. Even though our model is the lightest in terms of parameters, the number of floating point operations it requires remains big to be embedded. Replacing the 2D convolutions with depth-wise separable convolutions or transforming our model into a single-stage detector could be a solution to improve the efficiency of our backbone.

Pre-training the backbone Although camera images are very different from RD maps, pre-training the weights of Faster R-CNN lead to an improvement of 7 to 9 points in mAP, which outperformed DAROD in 3 of the 4 cases. Pre-training the backbone of DAROD might also led to a significant increase in performance. However, this is not trivial since it requires a well-suited dataset in terms of shape and complexity. Experiments have been conducted to pre-train the DAROD backbone using the ImageNet [Russakovsky 2015] dataset, but we found DAROD to be undersized to be pre-trained adequately on ImageNet. A good pre-training of object detectors' backbones could help reduce the annotation process of radar data and the number of required labels to train models. Exploring self-supervised pre-training methods such as SimCLR [Chen 2020a], MoCo [He 2020], BYOL [Grill 2020] or DINO [Caron 2021] while exploiting the specificity of radar data could help to reduce the number of required labels for radar object detectors. We start conducting research on pre-training radar networks in Chapter 5.

Exploiting the temporal information We demonstrated that a simple and light backbone performed well for object detection and classification tasks, contrary to deeper image-based backbones. However, our model needs to consider the temporal information of radar data, which could help build a more accurate radar object detector. Indeed, the confusion matrices in Figures 3.3b, 3.3a and 3.9 revealed some confusion between similar objects (pedestrians and bicyclists, trucks and buses). This is due mainly to the intra-class variations in the shape of objects. In radar, the same object at two different distances, angle of arrival or velocity can

be very different in the RD view (this remains true in the RA and RAD views). Exploiting the time information (*e.g.* multiple frames) has been shown to help to capture better the dynamics of objects, and therefore, the intra-class variation of objects [Ouaknine 2021a, Major 2019, Wang 2021b]. The next chapter introduces a new approach to learning features from the time for radar. It shows the relevance of such an approach to improving radar object detectors' detection and classification accuracy.

Chapter 4

Online object detection from radar raw data

Contents

4.1 Motivation

For automotive applications, time is key information which can be exploited to learn temporal patterns between successive frames in videos for example. The models tested in Chapter 3 worked independently on the different frames. While they showed decent detection and classification results, their ability to distinguish between close classes (*e.g.* pedestrians and bikes) was limited (see Figures 3.3a, 3.3b and 3.9). Indeed, radar object signatures vary a lot for a same object located at different distance, angle of arrival and with a different speed, resulting in a lot of variance in the class distribution. According to the radar equation (Equation 1.1), the signature of an object vary with the distance between the object and the

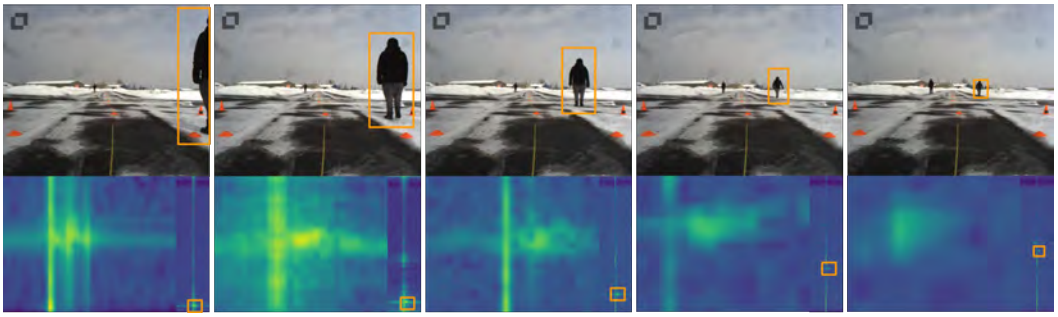


Figure 4.1: Fluctuation of the radar signature of a pedestrian over time. The power reflected by the pedestrian decrease with the distance, according to the radar equation.

radar. This variation, also referred to as the dynamic of the object, is characteristic. Therefore, the use of time in radar makes it possible to learn the dynamics of the objects held in the radar signal, handle the variation in the shape of the object over time, and reduce the noise between successive frames (induced by the movement of the surrounding object and the vehicle itself). We show in Figure 4.1 the fluctuation over time of the signature of a pedestrian which moves away from the radar. In this example we can see that the farther the pedestrian, the lower the reflected power. Also, the figure illustrates the micro-Doppler effects that appears when the pedestrian is close from to radar (due to an higher velocity of the arms compared to the body of the pedestrian).

Recent efforts have been made to exploit temporal relationships between raw radar frames using multiple frames for detection or segmentation tasks. The most common approaches such as [Ouaknine 2021a, Wang 2021b, Ju 2021] is to use temporal convolutions (or 3D convolutions). Conversely, in [Major 2019], Major *et al.* use a ConvLSTM to detect cars in RA view and [Li 2022] processes sequences of two successive radar frames to learn the temporal relationship between objects. Most temporal radar object detectors use temporal convolutions to learn spatial and temporal information. However, these methods are often non-causal, struggle to capture long-term dependencies and are unsuitable for real-time applications. Indeed, temporal convolutions require large kernel, therefore more parameters and computations, to capture long-term dependencies. Moreover, because the convolutional kernel is applied over past and future frames, some models based on 3D convolutions are not causal [Ju 2021, Wang 2021b].

This chapter presents a new convolutional and recurrent neural network (CRNN) for radar spectra. Unlike most multi-frame radar object detectors, our model is causal, which means we only use past frames to detect objects. This characteristic is crucial for real-time ADAS applications because such systems do not have access to future frames. To learn spatial and temporal dependencies, we introduce a

model consisting of 2D convolutions and convolutional recurrent neural networks with ConvLSTMs at different scales. Additionally, we use efficient convolutions and efficient ConvRNNs (inverted residual blocks [Sandler 2018] and Bottleneck LSTMs [Zhu 2018]) to reduce the computational cost of our approach. Our model is end-to-end trainable and does not require pretraining or multiple training steps. We present a generic method that can process either RA, RD or RAD spectra and outperforms state-of-the-art architectures on different tasks (point-based detection and semantic segmentation). To our knowledge, this is the first fully convolutional recurrent network for radar spectra and also the first with LSTMs at different scales.

Section 4.2 presents prior art on sequential object detection in computer vision and radar. Sections 4.3 and 4.4 describe our approach and the proposed model. We present the results of our experiments on the ROD2021 dataset [Wang 2021c] and the CARRADA [Ouaknine 2021b] dataset in Section 4.5. Finally, Section 4.6 discusses and concludes the chapter.

This chapter is mainly inspired from our article "A recurrent CNN for online object detection on raw radar frames" submitted at IEEE Transaction on Intelligent Transportation Systems (T-ITS)[1] [Decourt 2022b]. A patent has also been filed for this work.

4.2 Related work

4.2.1 Sequential object detection in computer vision

Object detection and segmentation are fundamental problems in computer vision. However, the majority of object detection and segmentation algorithms have been developed on static images. For some applications (robotics, autonomous driving, earth observation), processing sequences of images is desirable. Due to motion blur or object occlusion and correlations between frames, it is suboptimal to directly apply classical object detectors, or segmentation algorithms [Chen 2018b] on successive frames. To exploit the temporal information in sequences, optical flows [Zhu 2017, Yu 2022], recurrent networks with or without convolutions [Zhu 2018, Li 2018, Pfeuffer 2019, Ventura 2019, Zhang 2019], attention [Fare Garnot 2021, Yu 2022], transformers [Yu 2022, Duke 2021], aggregation methods [Chen 2020b] or convolutions (temporal or spatial) [Xiao 2018, Bertasius 2018] were widely used.

In [Li 2018] and [Zhu 2018], authors propose to transform the SSD object detector in a recurrent model. In [Li 2018], the authors add ConvLSTM cells on the top of the detection head. In [Zhu 2018], an LSTM cell is added between the last feature map of the feature extractor and the detection head. However, these models only

[1]The code of this contribution is available here: `https://github.com/colindecourt/record/`

use single scale feature maps to learn temporal relationships. In [Ventura 2019], authors present a recurrent model for one-shot and zero-shot video object instance segmentation. Contrary to previous methods and ours, they use a fully recurrent decoder composed of upsampling ConvLSTM layers to predict instance segmentation masks. Another approach proposed by Sainte Fare Garnot and Landrieu in [Fare Garnot 2021] consists in using temporal self-attention to extract multi-scale spatio-temporal features for panoptic segmentation of satellite image time series.

4.2.2 Sequential object detection in radar

In radar, object detection or segmentation algorithms [Schumann 2018, Gao 2021, Franceschi 2022, Zhang 2021, Decourt 2022a] that do not use time suffer from low performances for similar classes such as pedestrians and bicyclists [Ouaknine 2021a, Decourt 2022a]. According to the Doppler principle, motion information is held in the radar signal and should help to differentiate between a pedestrian (non-rigid body, motion information widely distributed), and a car (rigid body, more consistent motion information) [Wang 2021b]. 3D convolutions are primarily used in radar to learn spatio-temporal dependencies between frames. Methods such as [Wang 2021b, Gao 2021, Ju 2021, Zheng 2021] adopt 3D encoder-decoder architectures where they predict the position of objects for N successive frames. Despite their performances, these methods require a buffer of N frames in memory to work and are not really online methods in inference, as the convolutional kernel is applied over past and future frames. Additionally, one may encounter border effects due to the padding applied on the first and the last frame in the buffer. On the contrary, our approach doesn't access future frames neither in training nor in inference. Additionally, the number of parameters of models using 3D convolutions is huge for real-time applications (34.5M for RODNet-CDC [Wang 2021b], 104M for RAMP-CNN [Gao 2021]). Consequently, Ju *et al.* [Ju 2021] introduced Dimension Apart Module (DAM), a lightweight module for spatio-temporal feature extraction on RA maps that can be integrated into U-Net style network architecture. Alternatively, Ouaknine *et al.* propose in [Ouaknine 2021a] to use 3D convolutions to encode the spatial information of the N past frames in an online setting. Similarly to other 3D convolutions-based methods, TMVA-Net [Ouaknine 2021a] has a lot of parameters compared to our approach.

Kaul *et al.* [Kaul 2020] propose a model without 3D convolutions where the time information is stacked in the channel dimension. [Niederlöhner 2022] aggregates point clouds of different time steps to increase the resolution of the radar point cloud. More recently, Li *et al.* [Li 2022] propose an approach inspired by the transformer architecture and based on computer vision-based feature extractors to exploit temporal dependencies between objects in two successive frames. Finally,

Major *et al.* [Major 2019] follow [Jones 2018] by using a ConvLSTM over the features of a multi-view convolutional encoder. Even though this model is similar to ours, the LSTM cell is applied only to the learned cartesian output before the detection head, and the proposed model only detects cars. Additionally, this model is not end-to-end trainable and requires pre-training of a non-recurrent version of it before.

4.3 Problem formulation

We aim to design a model to learn the implicit relationship between frames at different spatial and temporal levels recurrently. This section describes the architecture of our single-view spatio-temporal encoder and decoder. We also introduce a multi-view version of our model designed to learn spatial and temporal features in different views (*e.g.* RD, AD, and RA) simultaneously.

Let us consider a sequence R of N radar frames ranging from time $k-N+1$ to k such as: $R = \{r_{k-N+1}, \ldots, r_k\}$. We aim to find the locations of every object in the scene at time k, P_k, based on the N past frames. We define our model with two functions, the encoder E and the decoder D.

We consider a causal model, which means it uses only the past to predict the next time step. For each time step k of the sequence, we consider a recurrent convolutional encoder taking as input the frame at time k and a set of I previous hidden states $H_{k-1} = \{h^0_{k-1}, \ldots, h^i_{k-1}, \ldots, h^{I-1}_{k-1}\}$ with i the index of the recurrent unit if more than one are used. The encoder returns a set of features maps F_k and a set of updated hidden states $H_k = \{h^0_k, \ldots, h^i_k, \ldots, h^{I-1}_k\}$ such that:

$$E(r_k, H_{k-1}) = (F_k, H_k). \tag{4.1}$$

Because our encoder encodes the past N frames recurrently to predict the position of objects at time step k, our decoder is a fully convolutional decoder that takes as input the encoder's updated hidden states H_k (the memory) and the set of feature maps F_k (spatio-temporal feature maps) such that:

$$D(F_k, H_k) = P_k. \tag{4.2}$$

As we want to improve the classification accuracy more than the localisation accuracy, we use recurrent layers in the encoding phase only. In encoder-decoder architectures, the encoder learns to extract an abstract representation of the radar frame relative to the class while the decoder is used for localisation. Using recurrent layers only in the encoding phase allows the encoder to encode spatio-temporal relationships at the object level to improve the objects' representation.

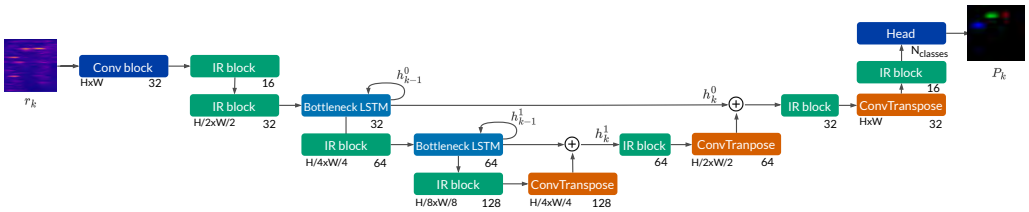


Figure 4.2: Model architecture (RECORD). Our encoder mixes efficient 2D convolution (IR block) and efficient ConvLSTMs (Bottleneck LSTMs) to learn spatio-temporal dependencies at multiple scales. The decoder is a 2D convolutional decoder, taking as input the last feature maps of the encoder and a set of two hidden states. It predicts either a confidence map or a segmentation mask. Rounded arrows on *Bottleneck LSTMs* stand for a recurrent layer. Plus sign stands for the concatenation operation. We report the output size (left) and the number of output channels (right) for each layer.

4.4 Model description

4.4.1 Encoder

An overview of our single-view architecture is shown in Figure 4.2. We propose a fully convolutional recurrent encoder (left part of Figure 4.2). In other words, our encoder mixes 2D convolutions and ConvRNNs. We use 2D convolutions to learn spatial information and reduce the size of inputs. To reduce the number of parameters of the model and its computation time, we use inverted residual (IR) bottleneck blocks from the MobileNetV2 [Sandler 2018] architecture instead of classic 2D convolutions for most of the convolutional layers of the model. IR bottleneck is a residual block based on depthwise separable convolutions that use an inverted structure for efficiency reasons. Then, we propose inserting ConvLSTM [Shi 2015] cells between convolutional layers to learn the temporal relationship between frames. Similarly to the convolutions, we replace the classic ConvLSTM with an efficient one proposed in [Zhu 2018] by Liu and Zhu called Bottleneck LSTM. Contrary to a classic ConvLSTM, authors replace convolutions with depthwise-separable convolutions, which reduces the required computation by a factor of eight to nine. Additionally, *tanh* activation functions are replaced by *ReLU* activation functions.

Bottleneck LSTM's equations are given by:

$$
\begin{aligned}
f_t &= \sigma({}^{(M+N)}W_f^N \star [x_t, h_{t-1}]) \\
i_t &= \sigma({}^{(M+N)}W_i^N \star [x_t, h_{t-1}]) \\
o_t &= \sigma({}^{(M+N)}W_o^N \star [x_t, h_{t-1}]) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \phi({}^{(M+N)}W_c^N \star [x_t, h_{t-1}]) \\
h_t &= o_t \circ \phi(c_t) \\
b_t &= \phi({}^{(M+N)}W_b^N \star [x_t, h_{t-1}]),
\end{aligned}
\tag{4.3}
$$

where M and N are the numbers of input and output channels, ${}^{j}W^{k} \star X$ denotes a depthwise-separable convolution with weights W, input X, j input channels, k output channels and, $\circ$ denotes the Hadamard product. Contrary to [Zhu 2018], we do not use the proposed bottleneck gate b_t authors propose. This gate aims to reduce the number of input channels of the LSTM to reduce the computational cost of the recurrent layer. This gate is not valuable for this work because we use few channels for our ConvLSTM. As a result, $M = N$. σ and ϕ denote the sigmoid and the LeakyReLU activation function, respectively. In this work, we use two bottleneck LSTMs, as a result, $I = 2$. Such a layout enhances spatial features with temporal features and vice versa.

We follow the MobileNetV2 [Sandler 2018] structure by first applying a full convolution to increase the number of channels followed by a single IR bottleneck block. Except for the first IR bottleneck block, we set the expansion rate γ to four. Next, we apply two blocks composed of three IR bottleneck blocks followed by a bottleneck LSTM to learn spatio-temporal dependencies. Because the computational cost of bottleneck LSTMs is proportional to the input size, we use a stride of two in the first IR bottleneck block to reduce the input dimension. Finally, we refine the spatio-temporal feature maps obtained from the bottleneck LSTMs by adding three additional IR bottleneck blocks.

Because we treat data sequences, it is desirable to calculate normalisation statistics across all features and all elements for each instance independently instead of a batch of data (a batch can be composed of sequences from different scenes). As a result, we add layer normalisation before sigmoid activation on gates o_t, i_t and f_t in the bottleneck LSTM, and we adopt layer normalisation for all the layers in the model.

4.4.2 Decoder

As described in Section 4.3, our decoder is a 2D convolutional decoder which takes as input the last feature maps of the encoder (denoted F_k) and a set of two hidden states $H_k = \{h_k^0, h_k^1\}$. Our decoder is composed of three 2D transposed convolutions

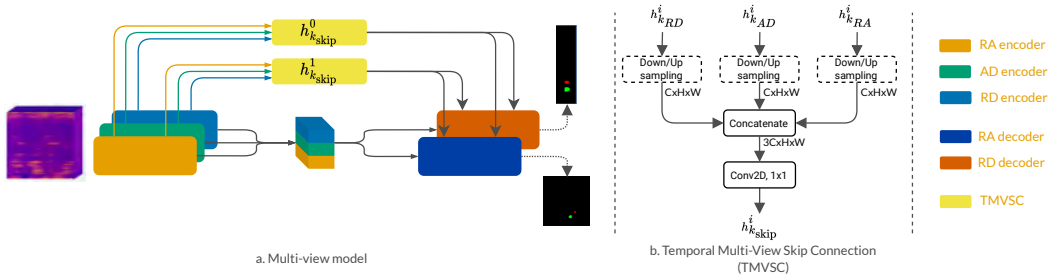


Figure 4.3: Multi-view model architecture (MV-RECORD). We use the encoder described in Figure 4.4.1 for each view. We use the decoder from RECORD as decoder in the multi-view model. Dashed boxes denote an optional operation applied only if the feature maps have different shapes. Gray arrows denote the same output.

followed by a single IR block with an expansion factor γ set to one, and a layer normalisation layer. Each transposed convolution block upsamples the input feature map by two. Finally, we use two 2D convolutions as a classification/segmentation head (depending on the task) which projects the upsampled feature map onto the desired output.

The U-Net architecture [Ronneberger 2015] has popularised skip connections between the encoder and decoder. It allows precise localisation by combining high-resolution and low-resolution features. We, therefore, adopt skip connections between our encoder and our decoder to improve the localisation precision. To prevent the loss of temporal information in the decoding stage, we use the hidden states of each bottleneck LSTM (denoted by h^0_k and h^1_k in Figure 4.2) and concatenate them with the output of a transposed convolution operation to propagate in the decoder the temporal relationship learned by the encoder.

4.4.3 Multi-view spatio-temporal object detector

The preceding sections described a spatio-temporal radar object detection architecture for single view inputs (*i.e.* RA or RD). However, using more than one view to represent targets in their entirety might be desirable. In other words, to simultaneously find the position (distance, angle), the velocity and the class of targets. In this section, we propose to extend the previous architecture to a multi-view approach. We follow the paradigm of Ouakine *et al.* [Ouaknine 2021a] by replicating three times the encoder proposed in Section 4.4.1 (one for RA view, one for RD view and one for AD view, see Figure 4.2). Then, the latent space of each view is concatenated to create a multi-view latent space. We use two decoders to predict objects' positions in all dimensions (RA and RD). One for the RA view and one for the RD view. The multi-view latent space is the input of these decoders.

In Section 4.4.2, we use the hidden states of each bottleneck LSTMs for the skip connection to add the temporal information in the decoding part. For the multi-view approach, we want to take advantage of the multi-view and the spatio-temporal approaches in the skip connections to supplement decoders with data from other views (*e.g.* add velocity information in the RA view). Similarly to the multi-view latent space, we concatenate the hidden states from RD, RA and AD views. This concatenation results in a set of concatenated hidden states $H_k = \{h^0_{k_{skip}}, h^1_{k_{skip}}\}$. We describe the operation to obtain H_k in Figure 4.3b. We concatenate H_k in the same way as in the single view approach. We call this operation Temporal Multi-View Skip Connections (TMVSC). Figure 4.3 illustrates the multi-view architecture we propose.

4.4.4 Training procedure

We propose two training methods to train RECORD and MV-RECORD: *online* and *buffer*, summarised in Figure 4.4. Let us denote by $R = \{r_{k-N+1}, \ldots, r_k\}$ a sequence of N radar frames ranging from time $k-N+1$ to k, $P = \{p_{k-N+1}, \ldots, p_k\}$ the objects' position in the sequence (the ground truth) and $\mathcal{L}$ the loss function we aim to minimise.

Buffer training We adopt a many-to-one paradigm when training using the *buffer* approach. We train the model to predict only the position of the objects in the last frame r_k as shown in Figure 4.4a. Therefore, given a sequence of N radar frames, we minimise the following loss function:

$$\mathcal{L}(\hat{p}, p) = \mathcal{L}(\hat{p}_k, p_k) \tag{4.4}$$

where k is the last time step of the sequence. *Buffer* training forces the model to focus on a specific time window and to learn a global representation of the scene. However, in inference, the model must process N frames sequentially to make a prediction. Therefore, we propose to train the model differently using a many-to-many paradigm to improve the model's efficiency in inference.

Online training We adopt a many-to-many paradigm when training using the *online* approach. We train the model to predict the position of the objects for every frame in the sequence R as shown in Figure 4.4b. Therefore, given a sequence of N radar frames, we minimise the following loss function:

$$\mathcal{L}(\hat{p}, p) = \sum_{k=1}^{N} \mathcal{L}(\hat{p}_k, p_k) \tag{4.5}$$

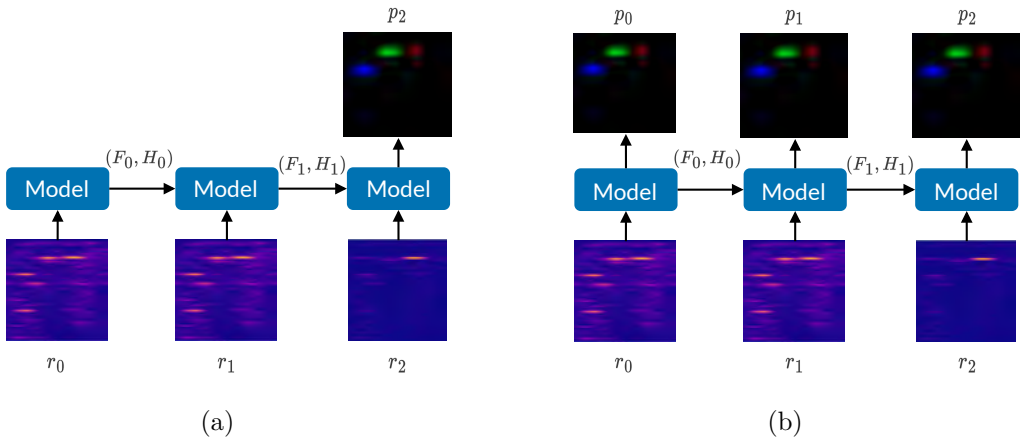


Figure 4.4: Training procedures with $N = 3$. (a) Buffer training procedure (many-to-one). (b) Online training procedure (many-to-many).

Online training pushes the model to use previous objects' positions to make a new prediction. It encourages the model to keep only relevant information from the previous frames. *Online* training requires training with longer sequences but allows data processing one by one (no buffer) in inference. In contrast to the *buffer* approach, the hidden states are not reset in inference.

4.5 Experiments

4.5.1 Single-view object detection

Dataset. We prototype and train our model, RECORD, on the ROD2021 challenge dataset[2], a subset of the CRUW dataset [Wang 2021c]. Due to its high frame rate (30 fps), this dataset is well-suited to evaluate the temporal models. Also, this dataset use a different detection paradigm than the CARRADA dataset. In the ROD2021 datasets, objects are represented as points (as like the output of a radar). We think this representation is more adapted to radar than bounding boxes or segmentation masks. Therefore, we prototype our model to deal with such outputs. The ROD2021 dataset contains 50 sequences (40 for training and 10 for testing) of synchronised cameras and raw radar frames. Each sequence contains around 800-1700 frames in four different driving scenarios (parking lot (PL), campus road (CR), city street (CS), and highway (HW)).

The provided data of the ROD2021 challenge dataset are pre-processed sequences of RA spectra (or maps). Annotations are confidence maps (ConfMaps) in range-azimuth coordinates that represent object locations (see Figure 4.2). According to [Wang 2021b] one set of ConfMaps has multiple channels, each representing

[2] `https://www.cruwdataset.org/rod2021`

one specific class label (car, pedestrian, and cyclist). The pixel value in the cls-th channel represents the probability of an object with class cls occurring at that range-azimuth location. We refer the reader to [Wang 2021b] for more information about ConfMaps generation and post-processing. RA spectra and ConfMaps have dimensions 128×128.

Evaluation metrics We use the metric proposed in [Wang 2021b] to evaluate the models on the ROD2021 challenge datasets. In image-based object detection, intersection over union (IoU) is mostly used to estimate how close the prediction and the ground truth (GT) are. For our single-view approach, as we predict the location of objects, we utilise the object location similarity (OLS) to match detection and GT. The OLS is defined as:

$$OLS = \exp \frac{-d^2}{2(s\kappa_{cls})^2} \tag{4.6}$$

where d is the distance (in meters) between the two points in the RA spectrum, s is the object distance from the radar sensor (representing object scale information) and κ_{cls} is a per-class constant that describes the error tolerance for class cls (average object size of the corresponding class). First, OLS is computed between GT and detection. Then the average precision (AP) and the average recall (AR) are calculated using different OLS thresholds ranging from 0.5 to 0.9 with a step of 0.05, representing different localisation error tolerance for the detection results. In the rest of this section, AP and AR denote the average precision and recall for all the thresholds.

Evaluation procedure In the ROD2021 dataset, annotations of test sequences are unavailable.To train and evaluate our models and the baselines similarly, we selected 36 sequences out of 40 to train the model and four for validation. We use the validation set for early stopping. Once the models are trained, we test them on the test set. Finally, we update the prediction on the ROD2021 evaluation platform to evaluate the performance of each model. As for the ROD2021 challenge, the evaluation is done for 70% of the test set.

Competing methods We compare our approach with several radar-based and image-based methods using sequences of multiple radar frames. For the radar-based approach, we first benchmark our model against DANet[3] [Ju 2021], a 3D convolutional model which won the ROD2021 challenge. Because image-based models are too heavy for our application, we finally contrast our recurrent approach against

[3]The original implementation is not available so we implement it according to author's guidelines. We do not use test augmentation and ensemble learning in this paper.

the attention-based model UTAE [Fare Garnot 2021], which is lighter than image-based approaches and which we can use causally. We found that decreasing the number of channels of UTAE and changing the positional encoding improved the performances (see Table B.1). We also consider two variants of our model without LSTMs, one using the time along the channel dimension (*no lstm, multi)* and one using a single frame (*no lstm, single).*

Experimental settings We use sequences of 32 frames for the *online* training. For validation and testing, frames are processed one by one. For the *buffer* training, we use sequences of 12 frames in both training and evaluation. Here we reset the hidden states every 12 frames. We use this buffer approach for fair comparison with other baselines that also use a buffer, although it is less efficient than the online approach. We optimise our model using the Adam optimiser with learning rate (1×10^{-3}) for the buffer method and 3×10^{-4} for the online method. We decay the learning rate exponentially by a factor of 0.9 every ten epochs. We train all the models using a binary cross-entropy loss.

We use an early stopping strategy to stop training if the model does not improve for seven epochs. To avoid overfitting, we use a stride of four for the buffer model and eight for the online model (i.e., how many frames the model skips between each training iteration) in the training dataset. The stride is set to one in validation and testing as we process data on the fly. We apply different data augmentation techniques during training, such as horizontal, vertical and temporal flipping. We use these settings for all the baselines, except for DANet where we use the settings recommended by the authors. All the models were implemented using the Pytorch Lightning[4] framework and trained on an NVIDIA Quadro RTX 8000 GPU. We run

[4]https://www.pytorchlightning.ai/

Model	AP					AR					Params (M)
	Mean	PL	CR	CS	HW	Mean	PL	CR	CS	HW	
RECORD (buffer, ours)	72.8 ± 2.2	95.0	67.7	48.3	**77.4**	**82.8 ± 1.5**	96.7	73.9	**72.8**	**81.7**	0.69
RECORD (online, ours)	**73.5 ± 3.5**	**96.4**	**72.5**	49.9	72.5	81.2 ± 2.0	96.4	78.1	68.8	77.6	0.69
RECORD (no lstm, multi)	65.5 ± 4.6	89.9	57.3	43.1	68.9	78.9 ± 1.4	93.1	68.2	71.5	75.7	0.47
RECORD (no lstm, single)	59.5 ± 2.9	85.7	48.5	39.11	64.4	75.1 ± 2.1	90.8	62.4	68.9	69.6	0.44
DANet [Ju 2021]	71.9 ± 2.3	94.7	65.7	**51.9**	70.0	80.7 ± 2.3	96.2	75.1	72.8	73.0	0.74
UTAE [Fare Garnot 2021]	68.4 ± 4.6	92.1	67.4	51.4	65.5	78.4 ± 2.2	94.6	74.0	69.7	70.0	0.79
T-RODNet [Jiang 2023]	69.9 ± 3.4	95.6	**72.5**	48.2	63.7	79.5 ± 1.9	**97.2**	**79.1**	70.2	67.2	159.7

Table 4.1: Results obtained on the test set of the ROD2021 challenge for different driving scenarios (PL: Parking Lot, CR: Campus Road, CS: City Street and HW: Highway). Overall, our recurrent models outperform baselines. The model that does not use time gets the worst performance. We report the best results over five different seeds with standard deviation. The best results are in bold, and the second bests are underlined.

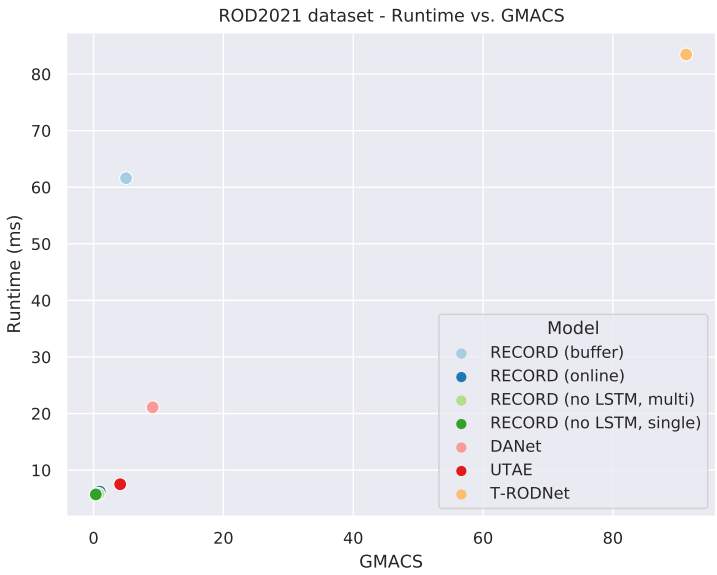
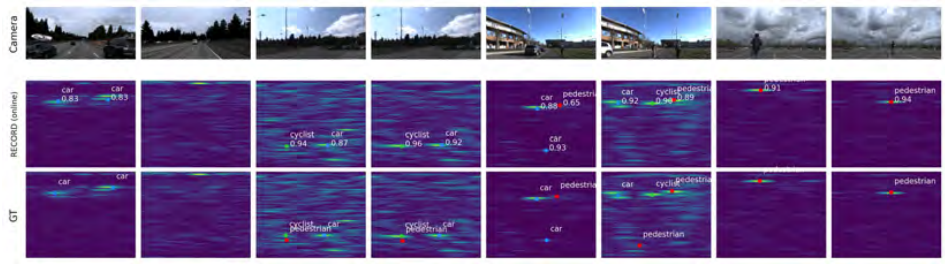


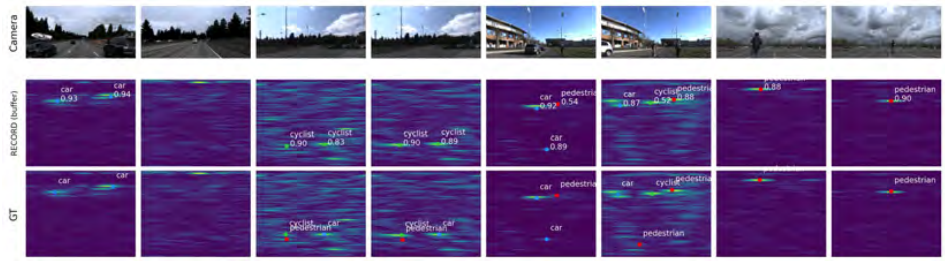
Figure 4.5: Runtime vs. GMACS on the ROD2021 dataset. RECORD (online) is one of the most efficient model among the baselines (low GMACS, low runtime). The T-RODNet model that uses Transformers is the slowest and the one requiring the more operations. Overall, most of the models have a runtime lower than 20 ms (on GPU).

all the models with five different seeds and report the mean and standard deviation results in the next paragraph.

Results Table 4.1 presents the results of our model and the baselines on the test set of the ROD2021 challenge. Our recurrent approaches generally outperform baselines for both AP and AR metrics; this remains true for most scenarios. Overall, most of the targets are detected, as shown in Figure 4.6, which confirms the high recall of the proposed models. Despite some mis-classified targets, the online version of RECORD obtains the best trade-off between performances and computational complexity (parameters, number of multiplications and additions and runtime, see Figure 4.5). Despite having less GMACs than UTAE and DANet, the buffer version of RECORD is the slowest one among all the models. Indeed, for each new frame we need to process the 11 previous ones, which is inefficient. Results show that the online version should be preferred for real-time applications. Additionally, RECORD methods exceed 3D and attention-based methods on static scenarios such as parking lot (PL) and campus road (CR). In PL and CR scenarios, the radar is static and the velocity of targets varies a lot, our recurrent models seem to learn variations of the target's speed better than other approaches. Surprisingly the attention-based method UTAE, initially designed for the segmentation of satellite



(a) RECORD (online).



(b) RECORD (buffer).

Figure 4.6: RECORD (online) and RECORD (buffer) qualitative results on the ROD2021 dataset. We show two samples per scenarios. From left to right: highway, city street, camprus road and parking lot. As shown in Table 4.1, the parking lot scenarios is the easiest one. More results are available in Appendix B.

images, obtains very competitive results with our method and the DANet model. The T-RODNet model shows good results on static scenarios but is unsuitable for real-time application. As shown in Figure 4.5, it is the slowest and the one requiring the more operations. We notice that the approach using the time in the channel dimension reaches lower AP and AR than their counterpart, which explicitly uses time as a new dimension. Finally, training our model without the time and using only a 2D backbone (*no lstm, single)* obtains the lowest performance on the test set. Qualitative results in Appendix B confirm the conclusion we draw here.

Ablation studies We demonstrate the relevance of using bottleneck LSTMs instead of classic ConvGRUs or ConvLSTMs in Table 4.2. Bottleneck LSTMs reduce the number of parameters and GMACS and achieve higher AP and AR than classic ConvRNNs. Additionally, we show in Table 4.3 the AP and the AR of our model with different skip connections. We show that concatenating temporal features with spatial features of the decoder (i.e., our RECORD model) reaches better AP and AR than a method without skip connections, or one where we add the temporal features to the spatial features of the decoder. Nevertheless, the concatenation of features increases the number of parameters and the number of GMACS of the model compared to other approaches.

ConvRNN type	AP	AR	Params (M)	GMACS
Bottleneck LSTM [Zhu 2018]	**69.8 ± 2.2**	80.2 ± 1.5	0.69	5.0
ConvLSTM [Shi 2015]	66.63 ± 3.28	79.36 ± 2.39	1.0	11.6
ConvGRU [Ballas 2016]	69.7 ± 2.4	**81.2 ± 1.4**	0.94	9.8

Table 4.2: Comparison of different types of ConvRNN. We train all the models with the same loss and hyperparameters. Bottleneck LSTM achieves the best AP while having fewer parameters and GMACS.

Skip connection	AP	AR	Params (M)
Concatenation	**69.8 ± 2.2**	80.2 ± 1.5	0.69
Addition	64.4 ± 5.3	**80.5 ± 1.1**	0.58
No skip connections	63.7 ± 6.2	78.7 ± 3.5	0.58

Table 4.3: Comparison of different types of skip connections. Results are averaged over 5 different seeds on the ROD2021 test set. Concatenation is the RECORD model, addition stand for a model where we add the output of transposed convolutions to h_k^i, and no skip connection stands for a model without skip connections.

Data augmentation study Table 4.4 shows the impact of different types of data augmentation and their combination on the performance. The experiments were conducted on the validation set. Among all the data augmentation available, horizontal flipping and Gaussian noise appear to be the most useful one. Temporal flipping reduces the overall performance when used alone or with Gaussian noise. However, when combined with horizontal flipping, or Gaussian noise this

Horizontal Flipping $p = 0.5$	Temporal Flipping $p = 0.5$	Gaussian Noise $p = 0.3$	AP	AR
✗	✗	✗	72.89	79.94
✓	✗	✗	74.66	83.67
✗	✓	✗	70.61	78.73
✗	✗	✓	72.02	81.78
✓	✓	✓	77.37	83.14
✗	✓	✓	70.84	79.52
✓	✓	✗	78.30	84.63
✓	✗	✓	75.59	83.57

Table 4.4: Impact of different types of data augmentation and their combination on the performance. Experiments were done on the validation set using the same seed.

data augmentation helps to increase the overall performance.

4.5.2 Multi-view semantic segmentation

Dataset To demonstrate the relevance of our method, we train our model for multi-view object segmentation on the CARRADA dataset [Ouaknine 2021b]. The CARRADA dataset contains 30 sequences of synchronised cameras and raw radar frames recorded in various scenarios with one or two moving objects. The CARRADA dataset provides RAD tensors and semantic segmentation masks for both RD and RA views. Contrary to the CRUW dataset, the CARRADA dataset only contains simple driving scenarios (static radar on an airport runway). The frame rate is 10Hz. The objects are separated into four categories: pedestrian, cyclist, car and background. The RAD tensors have dimensions $256 \times 256 \times 64$ and the semantic segmentation masks have respectively dimensions 256×256 and 256×64 for the RA and the RD spectra. For training, validation and testing, we use the dataset splits provided by the authors.

Evaluation metrics We evaluate our multi-view model using the intersection over union (IoU). IoU is a common evaluation metric for semantic image segmentation, which quantifies the overlap between the target mask T and the predicted segmentation mask P. For a single class, IoU is defined as:

$$IoU = |\frac{T \cap P}{T \cup P}|. \tag{4.7}$$

We then average this metric over all classes to compute the mean IoU (mIoU).

Competing methods We compare our multi-view model with state-of-the-art multi-view radar semantic segmentation models, namely MV-Net and TMVA-Net [Ouaknine 2021a]. Additionally, we train a buffer and an online single-view variant of our RECORD model. We train two different models, one for the RA view and one for the RD view.

Experimental settings As for the ROD2021 dataset, we use two evaluation settings for MV-RECORD: online and buffer.The CARRADA dataset has a significantly lower frame rate than the ROD2021 dataset. In order to match the same time as a single view model, we set the number of input frames to five for the buffer variant and ten for the online one, which corresponds to a time of respectively 0.5 and 1 second. We set the batch size to eight and optimise the model using Adam optimiser with a learning rate of 1×10^{-3} for both buffer and online methods except for the online multi-view model where the learning rate is set to 3×10^{-4}.

We decay exponentially the learning rate every 20 epochs with a factor of 0.9. We use a combination of a weighted cross-entropy loss and a dice loss with the recommended parameters described in [Ouaknine 2021a] to train our model as we find it provides the best results. To avoid overfitting, we apply horizontal and vertical flipping data augmentation. We also use an early stopping strategy to stop training if the model's performance does not improve for 15 epochs. Training multi-view models is computationally expensive (around six days for TMVA-Net and five days for ours). As a result, we train models using the same seed as the baseline for a fair comparison. We use the pre-trained weights of TMVA-Net and MV-Net to evaluate baselines.

Results Table 4.5 shows the results we obtain on the CARRADA dataset. Our multi-view approaches beat the state-of-the-art model TMVA-Net on the multi-view radar semantic segmentation task while using two times fewer parameters and requiring significantly fewer GMACS (see Figure 4.7). More, as shown in Figures 4.8 and 4.9, both approaches succeed in detecting a car which was not annotated. Our approach seems to correctly learn the variety of objects' shapes without complex operations such as the atrous spatial pyramid pooling (ASPP) used in TMVA-Net. We notice that using recurrent units instead of 3D convolutions in a multi-view approach significantly helps to improve the classification of bicyclists and cars, especially on the RA view, where we double the IoU for bicyclists compared to TMVA-Net. However, bicyclists and pedestrians are very similar classes, and

	Model	IoU					Params (M)
		mIoU	Bg	Ped	Cycl	Car	
RA	MV-RECORD (buffer, ours)	**44.5**	99.8	<u>24.2</u>	**20.1**	<u>34.1</u>	1.9
	MV-RECORD (online, ours)	<u>42.4</u>	99.8	22.1	<u>11.1</u>	**36.4**	1.9
	RECORD* (buffer, ours)	34.8	99.7	10.3	1.4	27.7	0.69
	RECORD* (online, ours)	36.3	99.8	12.1	3.1	30.4	0.69
	TMVA-Net [Ouaknine 2021a]	41.3	99.8	**26.0**	8.6	30.7	5.6
	MV-Net [Ouaknine 2021a]	26.8	99.8	0.1	1.1	6.2	2.4
RD	MV-RECORD (buffer, ours)	**63.2**	99.6	**54.9**	**39.3**	<u>58.9</u>	1.9
	MV-RECORD (online, ours)	58.5	99.7	49.4	26.3	58.6	1.9
	RECORD* (buffer, ours)	58.1	99.6	46.6	28.6	57.5	0.69
	RECORD* (online, ours)	<u>61.7</u>	99.7	52.1	<u>33.6</u>	**61.4**	0.69
	TMVA-Net [Ouaknine 2021a]	58.7	99.7	<u>52.6</u>	29.0	53.4	5.6
	MV-Net [Ouaknine 2021a]	29.0	98.0	0.0	3.8	14.1	2.4

Table 4.5: Results on the multi-view approach on CARRADA dataset. MV-RECORD stands for our multi-view approach. RECORD* stands for a single-view approach. The best results are in bold, and the second bests are underlined.

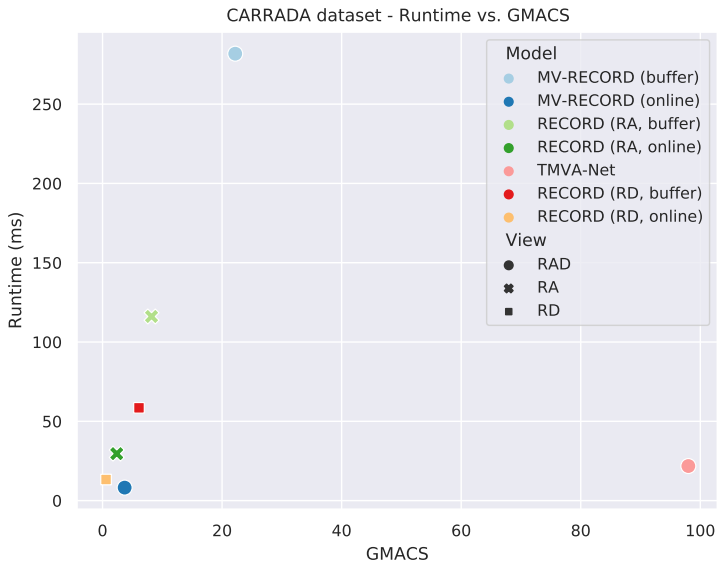


Figure 4.7: Runtime vs. GMACS on the CARRADA dataset. Multi-view methods have higher runtime and require more GMACS than single-view models. Our MV-RECORD (buffer) has few GMACS compared to TMVA-Net but requires more than 250ms for a single forward pass, while the online one can perform a single forward pass in 20 ms. RAD stands for the multi-view approach.

improving the detection performance of bicyclists leads to a loss in the detection performance of pedestrians for the RA view. In the RD view, MV-RECORD models outperform the TMVA-Net approach for all classes. We notice a huge gap in RA view performances between the CARRADA dataset and the CRUW dataset, as well as between the two views of the CARRADA dataset. We hypothesise that the small frame rate of the CARRADA dataset might cause these differences. Indeed, the RD view contains Doppler information, enabling one to learn the dynamics of targets. However, the RA view might not contain as much motion information as in the ROD2021 dataset, where the frame rate is higher, allowing the network to learn the dynamics of targets even in the RA view. Unfortunately, we cannot share the same analysis for the online multi-view approach. Compared to the results on the ROD2021 dataset, where the online approach performs better than the buffer one, we could not find proper training settings for the online multi-view model. Despite MV-RECORD online reaching higher IoU than TMVA-Net on the RA view, this model performs similarly with TMVA-Net on the RD view but has significantly lower IoU than the MV-RECORD buffer approach. Figures 4.8 and 4.9 confirm that the online version of MV-RECORD performs worse than the buffer one. In Figure 4.9 (right column), we notice the pedestrian is detected but misclassified by the model. We think these differences are mostly optimisation problems. Indeed, we show the online training outperforms the buffer training

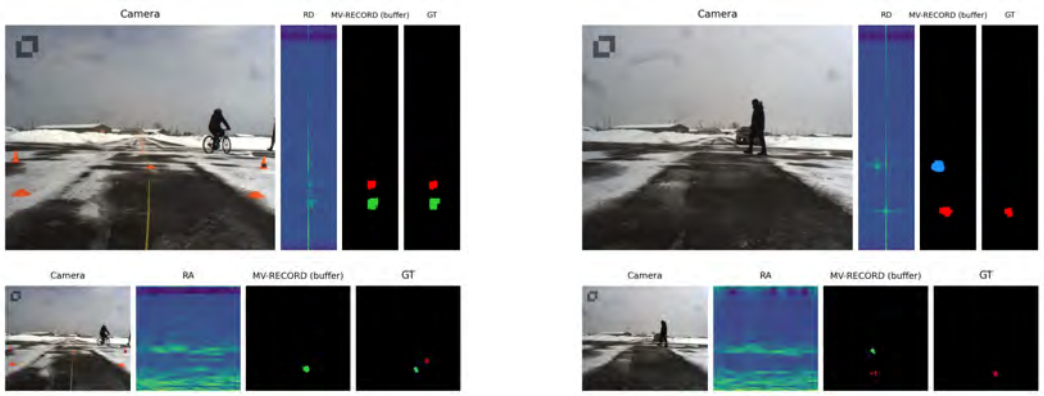


Figure 4.8: Qualitative results for **MV-RECORD (buffer)**. Despite missing annotations for some objects, our model can detect and classify all objects in the scene in some cases. From left to right: camera image, radar spectrum (top row is RD and bottom row is RA), predicted mask and ground truth mask. More results are available in Appendix B. Legend: pedestrians, bicyclists, cars.

when using a single view on both the ROD2021 (Table 4.1) and the CARRADA dataset. Especially on the RD view, our single view and online model outperforms TMVA-Net without using the angle information, with eight times fewer parameters and less computations. This confirms that the low frame rate of the CARRADA dataset limits the motion information that the recurrent layers can learn. Finally, despite having fewer GMACS and parameters than TMVA-Net, our multi-view model (buffer) is much slower in inference than TMVA-Net and is unsuitable for real-time applications. The online version is faster and should be preferred for real-time applications. Decreasing the size of the feature maps in the early layer of the network might help to increase the inference speed of the model. Also, we notice using a profiler that the LayerNorm operation takes up to 90% of the inference time for the multi-view models and up to 70% of the inference time for the single-view models. Replacing layer normalisation with batch normalisation should speed up the runtime of our approaches. Given the good results of the single-view approach (especially for the RD view), we recommend using our model for single-view inputs, as RECORD was originally designed for single-view object detection.

4.6 Conclusion and perspectives

In this chapter, we tackled the problem of online object detection for radar using recurrent neural networks. Contrary to the detectors studies in Chapter 3, which use a single frame to detect objects in different radar representations, we learn spatial and temporal relationships between frames by leveraging characteristics of FMCW radar signal. We propose a new architecture type that iteratively learns spatial and

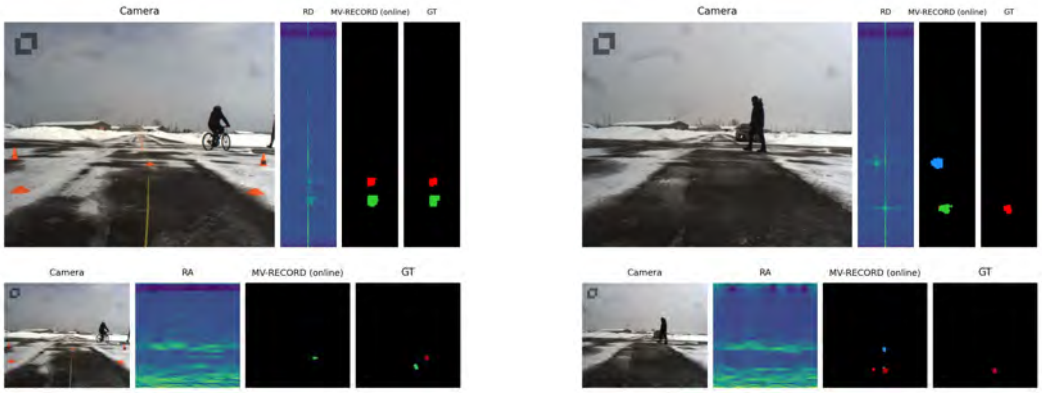


Figure 4.9: Qualitative results for **MV-RECORD (online)**. Despite missing annotations for some objects, our model can detect and classify all objects in the scene in some cases. However, we noticed more misclassified objects than the *buffer* version of MV-RECORD. From left to right: camera image, radar spectrum (top row is RD and bottom row is RA), predicted mask and ground truth mask. More results are available in Appendix B. Legend: pedestrians, bicyclists, cars.

temporal features throughout a recurrent convolutional encoder. We designed an end-to-end, efficient, causal and generic framework that can process different types of radar data and perform various detection tasks (key point detection, semantic segmentation). Our methods outperform competing methods on both CARRADA and ROD2021 datasets. Notably, our models help distinguish pedestrians and cyclists better and learn the target variations better than 3D approaches.

The difference with the results in the DANet paper Experiments in Section 4.5.1 show that DANet produces a 71.9 AP and 79.5 AR which is different from the results announced in the original paper. The code of DANet being unavailable, we implemented it according to the author's guidelines. Although we obtained the same number of parameters announced for the DAM blocks, our implementation has 740k parameters instead of the 460k announced in the paper. Beyond the implementation, the training and evaluation procedure in our paper is different from the one in the DANet [Ju 2021] paper. While DANet is trained on the entire training set, we trained it on 36 carefully chosen sequences for a fair comparison with other models. Also, DANet authors use the following techniques when testing the model to improve the performance: test-time augmentation (TTA), ensemble models and frame averaging. Because DANet predicts frames by a batch of 16 with a stride of four, the authors average the overlapping frames (12 in total) in inference. Together, those techniques boost the performance of DANet around ten points, according to the ablation studies in DANet's paper, which is coherent with the gap between our scores and the ones from DANet paper. While applying TTA,

ensemble models and training all the models using all the sequences would certainly also improve the global performance of all the models in Table 4.1, we preferred comparing the architectures on a simpler but fair evaluation.

Why recurrent neural networks are suitable for radar Radar data differs from LiDAR and images. The most critical differences being 1) the data is simpler in terms of variety, size, and complexity of the patterns; and 2) the datasets are smaller. We thus believe that our lighter architectures are flexible enough, while being less sensitive than huge backbones and less prone to overfitting for radar data. This mainly explains why Bottleneck LSTMs perform better than ConvLSTMs/ConvGRUs (see Table 4.2). Also, we think convolutional LSTMs are more adapted to radar sequences because 1) convLSTMs learn long-term spatio-temporal dependencies at multiple scales, which 3D convolution cannot do because of the limited size of the temporal kernel; 2) LSTMs can learn to weigh contributions of different frames which can be seen as an adaptive frame rate depending on the scenarios and the speed of vehicles; 3) Hidden states keep the position/velocity of objects in previous frames in memory and use it to predict the position in the next timesteps. Indeed, we show that, except for MV-RECORD, which is hard to optimise, online methods generally perform better than buffer ones while having lower computational cost (GMACs and inference time).

Vision Transformers and radar One alternative to RNN is the use of Vision Transformers (ViT)[Dosovitskiy 2021]. ViT has been proven to be a solid alternative to CNN and to solve some challenges of CNNs (pixel weighting, shared concepts across images, spatially distant concepts). In [Naseer 2021], Naseer *et al.* show Transformers are more robust to occlusions, perturbations and domain shift and less biased towards local texture. The work of Caron *et al.* [Caron 2021] shows that self-supervised ViT can automatically segment foreground objects, which is an interesting property for radar. Additionally, ViT mainly utilises dense connections and embeddings, lowering the number of FLOPS compared to CNN but increasing the number of parameters. However, training ViT needs large-scale datasets and sometimes self-supervised pre-training. As large-scale radar datasets become available, it is worth considering ViT as a backbone to learn spatial, temporal or spatio-temporal radar features. Giroux *et al.* [Giroux 2023] and Jiang *et al.* [Jiang 2023] propose to use Transformers blocks (SwinTransformers) as a backbone. Both works show that ViTs perform similarly to CNNs for radar object detection. While [Giroux 2023] confirms the ViT can lower the number of GFLOPS compared to CNNs, the authors of [Jiang 2023] mix 3D convolutions. As shown in Table 4.1 and Figure 4.5, such combination is too computationally expensive and does not help to reduce the FLOPS compared to RECORD, DANet [Ju 2021] or UTAE

[Fare Garnot 2021]. For multi-frame object detection, one future work could be to change the embedding of [Giroux 2023] with the tubelet embedding proposed in [Arnab 2021]. In this way, the Transformer can learn global information in different parts of the spectrum at different time steps.

Multi-view vs. single-view Although multi-view methods are interesting for research purposes, we find them challenging and long to optimise. Multi-view models allow us to find the distance, arrival angle and targets' relative velocity in a single forward pass. However, it requires the use of the RAD cube as input. This poses some problems for real-time applications. First, it requires computing the direction of arrival (a FFT) **for every point** in range and Doppler, resulting in 256×64 FFT to compute. Second, using the RAD cube necessitates to store it. For low-resolution radars (CARRADA, RADDet, CRUW datasets), the size of the RAD cube is generally $256 \times 256 \times 64$, around 1.64MB in memory. For comparison, the size of our RECORD model is about 2.6MB.

In this way, using the RAD cube is not the most efficient approach for real-time applications. Radars generally detect targets in the range-Doppler view and compute the direction of arrival for each detected target to save computation time. Soon, high-resolution radars will replace low-resolution radar. Regarding the size of the ADC data of high-resolution radar, using the RAD tensors for perception tasks is difficult. Indeed, high-resolution radars use more chirps, samples, and antennas to increase the resolution. In the RADIal dataset [Rebut 2022], the RD spectrum has a $512 \times 256 \times 32$ size. After DOA estimation, the RA spectrum has a 512×730 size, resulting in a RAD cube of $512 \times 730 \times 256$ (around 380MB).

For future work and efficiency, we recommend using our RECORD model on single-view inputs, particularly the RD spectrum. We could use our backbone on high-resolution radar data to learn a spatio-temporal RA latent space as in [Rebut 2022, Giroux 2023].

Chapter 5

Self-supervised learning for radar object detection

Contents

5.1 Motivation

In deep learning, the data annotation is a crucial parameter to succeed in learning meaningful representations for object detection, semantic segmentation or classification. Since radar raw data is complex and time-consuming to annotate, generally, the authors of radar datasets generate the annotations semi-automatically [Ouaknine 2021b, Zhang 2021, Rebut 2022, Wang 2021c]. The authors of the CARRADA, the RADDet, and the CRUW datasets first detect objects in the camera (using a pre-trained Mask-RCNN [He 2017]) and in the radar views (using CFAR, DBSCAN and DoA estimation techniques). Then, they project the Mask R-CNN detection on the radar view and combine them to create the label. The authors of the RADIal dataset adopt a similar approach, but they also use the detection from a LiDAR.

However, using a semi-automatic approach to generate the radar data annotations has limitations. For example, an object detected in radar but not by the Mask-RCNN model is removed and reciprocally, which leads to unannotated valid targets. Such examples can be found in Appendix A and B. Also, the annotations are constructed using the traditional operations presented in Chapter 1 (CFAR, DBSCAN, target tracking). This raises the question of developing more accurate deep learning-based detectors than traditional ones. Indeed, if the annotations originated from conventional radar detection techniques, the deep learning algorithms cannot outperform them.

Chapter 3 shows that pre-training the backbone of a Faster-RCNN model on the ImageNet [Russakovsky 2015] dataset drastically improves the detection performance. However, our DAROD model was not large enough to learn from ImageNet. Given the difficulty of annotating radar data, this chapter investigates a method to pre-train radar object detectors using unlabelled radar data and self-supervised learning (SSL). While SSL techniques aim to improve the overall performance of computer vision models, we aim to use SLL to learn with fewer labelled data (*i.e.* reduce the number of annotations required). The long-term goal of this study is to reduce the number of annotations required to train radar object detectors by pre-training models on large radar datasets and fine-tuning them on a small amount of manually annotated data.

This chapter presents an early work to pre-train radar object detectors. We propose an extension of the work of [Wei 2021] by using radar knowledge to learn object-level representations using contrastive learning. Our **preliminary** results show that such pre-training improves the performance of object detectors when trained with only 50% to 10% of labels.

Section 5.2 presents what is self-supervised learning. Section 5.3 reviews the most popular computer vision SSL frameworks. Sections 5.4 and 5.5 raise the limitations of the frameworks presented in Section 5.3 for object localisation and radar applications, respectively. In Section 5.6, we present a self-supervised pre-training method for radar object detection and preliminary results. Finally, Section 5.7 concludes the chapter and proposes further research directions.

5.2 What is self-supervised learning?

In SSL, a model learns to obtain a supervisory signal from the data itself by leveraging the structure of the data. SSL has been particularly impactful in natural language processing (NLP), allowing to pre-train large language models such as GPT-3 [Brown 2020] or BERT [Devlin 2019] on large unlabelled text datasets. More recently, in [Goyal 2021], researchers show that pre-training models with self-supervised techniques on large image datasets (one billion images) enable compet-

itive results with models trained in a supervised manner.

The global goal of SSL (in computer vision) is to learn generic representations of the data across tasks (classification, detection, segmentation). To do so, SSL defines a *pretext task* that allows to learn representations from the dataset without labels [Goodfellow 2016, Balestriero 2023]. Then, the learned representation can be reused for fine-tuning a model on *downstream tasks* (classification, detection or segmentation). A common pretext task in NLP is to mask a word in a text and predict it from the surrounding words, to force the model to capture relationships among words without labels. In computer vision, researchers aim to encourage two views of the same image to be mapped to similar representations [Grill 2020, Chen 2020a] or to predict missing patches of an image [He 2022].

SSL has several advantages over traditional supervised learning. First, it does not require any labelled data, which are complex and expensive to obtain, especially for radar data. Second, self-supervised learning can be used to learn more robust features against adversarial examples or label corruption than supervised learning [Goyal 2022] because it does not rely on the labels to learn features. Finally, self-supervised learning can be used to pre-train models with more data, improving models' performance on downstream tasks.

5.3 A review of SSL frameworks for computer vision

There has been a growing interest in SSL since 2020, thanks to the availability of large datasets and high-memory GPUs. This section gives a non-exhaustive overview of the most popular recent SSL frameworks. According to [Balestriero 2023][1], we categorise SSL into four families: Deep Metric Learning (DML), Self-Distillation (SD), Canonical Correlation Analysis (CCA), and Masked Image Modelling (MIM). However, these methods, built upon the knowledge of early experiments, are only some of the existing approaches for SSL. As an example, early attempts to pre-train models in a self-supervised manner include but are not limited to information restoration [Pathak 2016], learning spatio-temporal relationships in videos [Wang 2015, Pathak 2017], grouping similar objects [Caron 2018], or using denoising auto-encoder (generative approaches) [Vincent 2008].

5.3.1 Deep Metric Learning

SSL's Deep Metric Learning family originated from the idea of *contrastive learning* [Bromley 1993, Hadsell 2006]. The idea of DML is to encourage the similarity between semantically transformed versions of an input (or *views*). In DML, one

[1] We refer the reader to [Balestriero 2023] for a deeper overview of SSL frameworks for computer vision and their challenges.

trains a network to make the embedding of two samples close or far from each other. Generally, because labels are unavailable, different views of the same image are created using image transformations. These views refer to as *positive pairs* are expected to be made similar. The dissimilar samples we want to make are called *negatives*. To make negatives far from positive, a distance m is imposed so that images from different classes must have a distance larger than m. A variant to the contrastive loss is the Triplet loss [Weinberger 2009, Schroff 2015], which consists of a query, a positive and a negative sample. In the Triplet loss, we aim to minimise the distance between the embedding of the query and the positive sample and to maximise the distance between the query and the negative sample.

SimCLR We now present one of the most prominent DML approaches termed SimCLR [Chen 2020a]. The idea of SimCLR is simple. Two views of the same image are created using a combination of image transformations (random resizing, cropping, colour jittering) and are encoded using a CNN. After the views are encoded, a MLP is used to map the features from the CNN to another space where the contrastive loss is applied to encourage the similarity between the two views. In SimCLR, negative samples are other images in the batch; thereby, SimCLR requires large batches to work. Figure 5.1 summarises the SimCLR method.

Apart from SimCLR, other DML approaches exist in the literature. For example, Sermanet *et al.* [Sermanet 2018] use a triplet loss in video frames where positive pairs come from nearby frames.

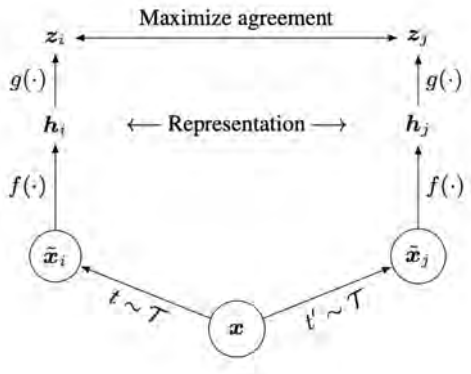


Figure 5.1: SimCLR overview. Two views of the same image are created using a combination of image transformations and are encoded using a CNN. After the views are encoded, a MLP is used to map the features from the CNN to another space where the contrastive loss is applied to encourage the similarity between the two views. Source: [Chen 2020a]

5.3.2 Self-Distillation

As for the DML family, the self-distillation family relies on the following mechanism: feeding two views of the same image to two encoders (a CNN or a ViT) and mapping one view to the other using a predictor (a MLP). However, this approach (using the identical two encoders) can lead to *dimensional collapse*. Dimensional collapse is a phenomenon that appears in SSL when the information encoded across different dimensions of the representation is redundant [Balestriero 2023]. One example of dimensional collapse is that the two encoders consistently predict a constant value for any input. A solution to dimensional collapse is to update one of the two encoder weights with a running average of the other encoder's weights. One advantage of the self-distillation method is that they do not necessarily require negative samples compared to DML approaches. The most famous SD approaches are BYOL [Grill 2020] and DINO [Caron 2021], which we will explain later.

BYOL (Bootstrap Your Own Latent) BYOL [Grill 2020] first introduced self-distillation as a mean to avoid dimensional collapse [Balestriero 2023]. BYOL uses two networks (the *online* or *student* and the *target* or *teacher*) along with a predictor to map the outputs of one network to the other. The online and the target networks are two identical CNNs with different weights. The student network predicts the output, while the teacher network produces the target. As for most SSL methods, each network receives a different view of the same image. BYOL uses image transformations, including random resizing, cropping, colour jittering, and brightness alterations. Each view is encoded using a CNN and then projected in a new space using a MLP. The particularity of BYOL is that the student network uses an additional MLP (the *predictor*) to map the student network's outputs to the target network's output. The student network is updated using SGD, and the teacher is slowly updated using an exponential moving average (EMA) of the student's weights. Figure 5.2 illustrates the BYOL method.

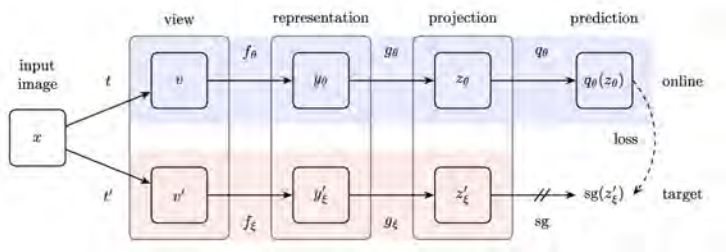


Figure 5.2: BYOL overview. BYOL uses two networks (CNNs), the teacher and the student networks, along with a predictor (a MLP) to map the outputs of one network to the other. The student network is updated using SGD, and the teacher is slowly updated using an EMA of the student's weights. Source: [Grill 2020]

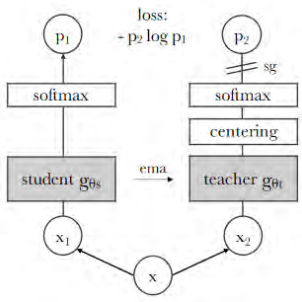


Figure 5.3: DINO overview. Compared to BYOL, DINO does not use a predictor but performs a centring of the student network output and applies a softmax function. [Caron 2021]

DINO Another self-distillation method that builds upon BYOL is DINO [Caron 2021]. DINO follows the BYOL idea but performs a centring of the output of the teacher network instead of using a predictor to avoid the model's sensitivity to mini-batch size. The centring operations consists to subtract the exponential moving average from the teacher's raw activation. The network is trained similarly to BYOL but uses ViT encoders instead of CNN. One interesting property of DINO is that the model can automatically discover and segment objects in images or videos without supervision.

Many other methods belong to the SD family. MoCo [He 2020] builds a dictionary look-up to sample positive and negative pairs, removing the need for large batch size in SimCLR. SiamSiam [Chen 2021] replaces the BYOL moving average encoder with a stop-gradient.

5.3.3 Canonical Correlation

The CCA family originates from the canonical correlation framework [Hotelling 1992]. The goal of CCA is to infer the relationship between two variables by analysing their cross-covariance matrices, therefore maximising the information content of the embedding. Compared to DML and SD family, CCA does not require large batches or memory bank, momentum encoder or stop gradient operation [Bardes 2022].

From the CCA framework arises SSL approaches such as SWAV [Caron 2020], BarlowTwins [Zbontar 2021] or the most recent one VicReg [Bardes 2022]. While BarlowTwins drives the normalised cross-correlation matrix of two embeddings from two views of the same image toward the identity, VicReg aims to balance three objectives (see Figure 5.4). VicReg proposes to minimise the distance between two embedding of the same view while maintaining the variance of the embedding above a threshold and pushing the covariance between embedding variables of a batch to

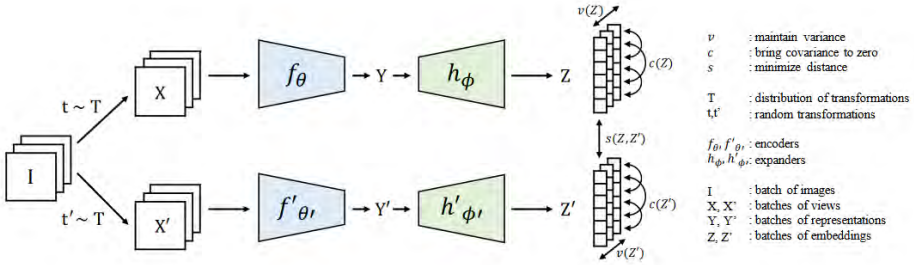


Figure 5.4: VicReg overview. VicReg proposes to minimise the distance between two embedding of the same view while maintaining the variance of the embedding above a threshold and pushing the covariance between embedding variables of a batch to zero. Source: [Bardes 2022]

zero. Maintaining the variance above a threshold prevents collapse, minimising the distance ensures views are encoded similarly, and covariance encourages different dimensions of the representation to capture different features.

5.3.4 Masked Image Modelling

Early SSL techniques for computer vision entailed applying degradation to images and learning to reconstruct the original images. It included generative models such as denoising auto-encoders [Vincent 2008], shuffling image patches [Noroozi 2016] or image in-painting [Pathak 2016]. However, these methods did not reach competitive results with supervised models.

Masked language modelling (MLM) shows impressive results for pre-training language models [Devlin 2019, Brown 2020], however as explained in [Bao 2022], it is not trivial to transpose such a method to vision because there is much more possible output for images than for text. This transposition is known as masked image modelling. In [Dosovitskiy 2021], authors first attempted to pre-train their ViT by masking patches and then teaching the model to predict pixel values directly. However, this pre-training strategy is less effective than supervised training. Instead, authors of BEiT [Bao 2022] remove the pixel-wise reconstruction loss and apply the BERT pre-training strategy to the image. They first encode image patches using a variational auto-encoder and then pre-train their encoder to predict the discrete token values from masked tokens. This approach leads to significantly better performance on downstream tasks than other self-supervised and supervised approaches.

The effectiveness of BEiT paved the way for new MIM pre-training approaches. In [Xie 2022] and [He 2022] (see Figure 5.5), authors propose similar approaches, which consist of randomly masking a high proportion of the input image and learning to reconstruct missing patches using MSE loss. Both approaches exploit the

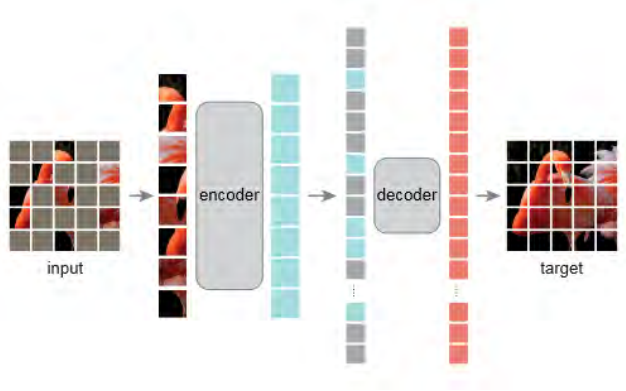


Figure 5.5: MAE overview [He 2022]

Transformer architecture instead of the variational auto-encoder from BEiT and re-introduce the pixel-wise reconstruction loss. Despite simple approaches, masked auto-encoders have achieved competitive performance on various vision tasks (classification, detection, segmentation). Similarly, Woo *et al.* [Woo 2023] propose a Fully Convolutional Masked Auto-Encoder (FCMAE) instead of Transformer to pre-train their model. They propose to mask random patches in the input image and to use sparse convolution to encode the remaining patches. Using a simple decoder, they also show competitive results against fully supervised and self-supervised approaches.

5.4 Pre-training models for object localisation

Collecting annotations for object detection or semantic segmentation is expensive. SSL techniques appear as a good solution to reduce the number of labelled images required to train object detectors or image segmentation models. Most learning frameworks presented in Section 5.3 perform well and are competitive with supervised approaches on downstream tasks as shown in [Ericsson 2021]. However, SSL frameworks are mostly tuned for image classification and lack good localisation representation. Indeed, the data used for the pre-training contains a single object centred in the image. Zhao *et al.* [Zhao 2021] note that due to the transformation applied to the image (random cropping, colour jittering), SSL frameworks are relatively robust to occlusion invariance and tend to learn to use all part of the image to make their predictions.

Recent works [He 2022, Li 2021] suggest that ViT contains superior localisation information in their learned representations and transfers better to object localisation downstream tasks than CNNs. Nevertheless, these algorithms include a localisation objective in their loss function as the patches include the localisation information. Apart from ViT, a solution is to rely on carefully chosen unsupervised

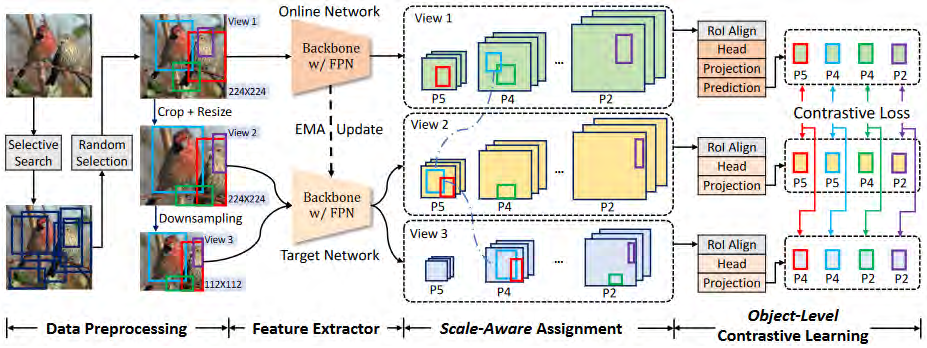


Figure 5.6: SoCo overview. SoCo randomly selects proposals from the selective search algorithm as object priors and constructs three views of an image where the scales and locations of the same objects are different. RoIs are extracted using the RoIAlign operator. Views are encoded using a CNN. The model is trained using the BYOL framework. Source: [Wei 2021]

object priors to learn localised features. Methods propose to modify SSL frameworks with such prior to enhance localisation in their features [Wei 2021, Wang 2021a, Dai 2021, Yang 2021, Bar 2022, Tong 2022, Zhao 2021, Carion 2020b]. It is worth noting that improving localisation features comes at the price of a lower accuracy when transferring features for classification. One example of unsupervised object priors is to modify the training loss to enforce the relationship between extracted features from locations within a single image [Wang 2021a, Yun 2022]. In [Yun 2022], Yun *et al.* encourage adjacent patches within an image to produce similar features by computing a contrastive loss between adjacent patches. Instead of modifying the loss function, some works explicitly add a prior on object location [Wei 2021, Yang 2021]. [Wei 2021] and [Yang 2021] both leverage RoIAlign [He 2017] to pre-train the backbone and the detection head of a CNN in a self-supervised manner to improve localisation on downstream tasks. Instance Localisation [Yang 2021] pastes a randomly chosen patch cut from the foreground of one image onto two other images and extracts features corresponding to only the pasted foreground patch. They use a contrastive loss to force the model to produce similar features regardless of the background and the location of the foreground patch in the image. SoCo [Wei 2021] randomly selects proposals from the selective search algorithm as object priors and constructs three views of an image where the scales and locations of the same objects are different. They train their model using the BYOL [Grill 2020] framework. Figure 5.6 gives an overview of SoCo. Finally, although ViT naturally encodes the position of objects, methods for pre-training DETR [Carion 2020a] family detectors were proposed in [Dai 2021, Bar 2022]. Most of the methods mentioned above enhance the localisation performance of self-supervised learners on object detection and semantic segmentation downstream tasks compared

to fully supervised methods or common SSL frameworks.

5.5 Limits of image-based pre-training strategies for radar

So, how do we transfer pre-training strategies from computer vision to radar? In this section we raise some limitations of applying the framework we presented in Section 5.3 to radar data directly.

The data augmentation problem First, SSL frameworks belonging to the self-distillation, deep metric learning, and canonical correlation analysis families rely on several image transformations to encourage similarities between two views of the same image. However, as explained in Section 2.5.3, most of the existing image transformations used in SSL frameworks cannot be applied to the radar data. Radar data differs from camera images in several ways. For example, it has complex input, energy loss with range (see the radar equation 1.1), and a non-uniform resolution in the angular domain. Except for horizontal and vertical flipping, we cannot transform too much radar data without altering it. Therefore, directly applying methods such as SimCLR [Chen 2020a] or BYOL [Grill 2020] is not possible. One alternative to encourage the similarities between similar objects in radar is to use successive frames. Assuming objects are not moving too much between two successive frames, we can consider two close frames as positive and far frames as negative and then train a model using a Triplet loss or a contrastive loss as in [Pathak 2017, Zhang 2019].

The localisation problem Second, raw radar data contains multiple objects at different distances, velocities or angles. As discussed in Section 5.4, SSL frameworks are designed on datasets with single objects centred in the image. Hence, for efficient pre-training for object localisation models in computer vision, adding a prior about object location in radar is essential. Fortunately, obtaining that prior in radar is easily possible using CFAR-like object detectors. CFAR is an alternative to the selective search algorithm used in [Bar 2022] and [Wei 2021] for pseudo-labelling the data. Experiments in Section 5.6.2 provide encouraging results using such an approach.

Masked image modelling Masked image modelling pre-training [He 2022, Xie 2022, Woo 2023, Bao 2022] shows impressive transfer performance on image classification and downstream tasks. However, MIM techniques could not be suited for radar data. In [He 2022], it has been shown that the higher the masking ratio, the better the performance on downstream tasks. A standard masking ratio

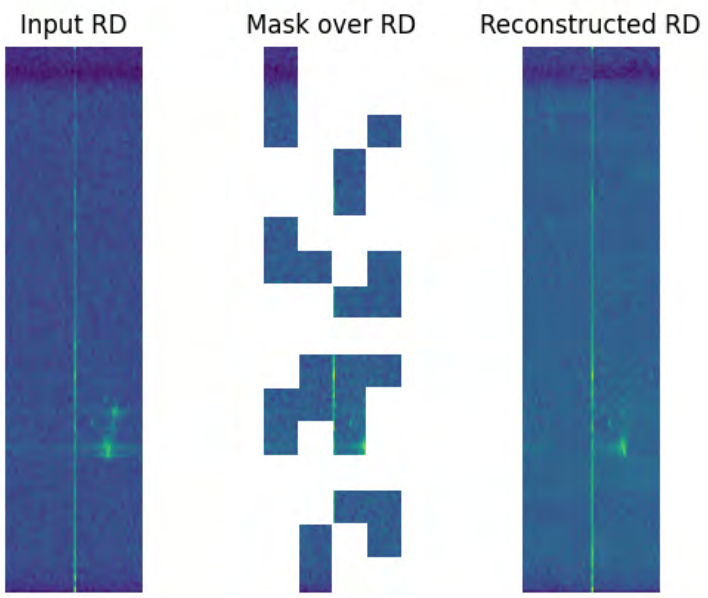


Figure 5.7: Reconstructed RD spectrum using the FCMAE MIM framework. We use a 200-epochs training schedule, a patch size of 16×16 and we set the masking ratio to 0.6. The model learns to reconstruct the noise and the floor reflections (the line in the middle), but cannot reconstruct true targets.

is about 60%. Unlike images, radar data mainly contains noise and tiny targets. Masking 60% of a spectrum means a small probability of masking targets. Thus, the network is more about learning noise rather than the probability distribution of targets, as shown by experiments with the FCMAE framework [Woo 2023] on the CARRADA dataset in Figure 5.7. Also, in the MIM framework, the patches have a 32×32 size, which is larger than the size of an object in radar (8×8 on average), resulting in a low signal-to-noise ratio inside the patch. However, our experiments did not show better reconstruction by decreasing the patch size. This suggests that MIM frameworks in their current form (randomly masking square patches in an image) are unsuitable for radar.

The amount of data Finally, one key ingredient of SSL is the amount of data. Generally, the more data available to pre-train the model, the more accurate the model on downstream tasks. Most SSL frameworks are pre-trained on large unlabelled image datasets (up to one billion) images. However, such large datasets still need to be created for radar and might hamper the benefits of pre-training strategies compared to a fully supervised approach. As a result, in this chapter, we

aim to analyse the advantages of the pre-training radar object detection models to reduce the amount of labelled data during the training instead of achieving higher performance than fully supervised learning.

5.6 Radar Instance Contrastive Learning (RICL)

This section presents a **preliminary work** of a pre-training strategy for radar object detection on the RD spectrum. We address the localisation problem by proposing a variant of the SoCo [Wei 2021] method for radar. We focus on the RD spectrum to consider future generations of radar, as explained in Section 4.6.

In self-supervised learning, the pretext task is fundamental to learning representations without labels. In computer vision, the main pretext tasks are encouraging two views of the same image to be the same [He 2020, Chen 2020a, Bardes 2022] or randomly masking patches of an image and learning to reconstruct them [He 2022, Tong 2022, Xie 2022]. However, as explained in Section 5.5, due to the small number of data transformations available in radar and the noise in the data, it is not possible to use these pretext tasks. Also, methods based on image transformations generally under-performs for localisation tasks.

Although the well-known pretext tasks cannot be applied to radar, a simple way exists to create pseudo-labels for RD spectra. By exploiting CFAR object detectors, we can easily create a binary segmentation mask which contains the position of *possible* objects. Moreover, supposing an object is not moving too much between two successive frames, it is possible to encourage the similarity between the same *possible* objects at different time steps. Then, we propose to leverage both particularities to pre-train a model for object detection. Our approach is a modified version of the SoCo [Wei 2021] framework (see Figure 5.6) but differs in different ways. First, we use the output of a CA-CFAR [Rohling 1983] detector to propose objects. Second, we do not create different views of the same images and proposals. Instead, we use two successive RD maps and apply the contrastive loss between the same object at different time steps. We detail the method in Section 5.6.1. The choice of a BYOL-based framework is motivated because it does not require to sample negatives to apply the contrastive loss. There are multiple objects in spectra, and we do not know the class of those objects. As objects belonging to the same class can be present in the same spectrum, we want to avoid repelling the representations of those objects.

5.6.1 Methodology

We propose an approach for pre-training the backbone and the detection head of an object detection model without labels, named RICL for Radar Instance Contrastive

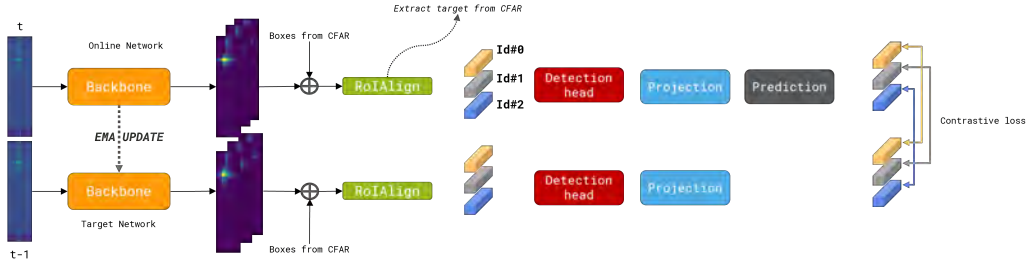


Figure 5.8: RICL framework overview. We use two networks (an online and target network) to encode features from each view in parallel. We use RoIAlign to extract object-level features from CFAR output (each object has an specific id in the figure). The contrastive loss is applied object-wise, no negative samples are required.

Learning. Figure 5.8 displays an overview of RICL. This section details the pre-training strategy.

Overview Given two successive RD maps, we encode features from each view using two identical CNNs (an online and a target network). From the CFAR detection, we then use RoIAlign [He 2017] to extract object-level features. Following the RoIAlign operation, the object-level features are passed to a detection head (a MLP) before being mapped onto another space using a projector (online and student networks) and a predictor (online network only). As for SoCo, we use the BYOL [Grill 2020] framework for learning the representations.

Object proposals generation and matching We use a CA-CFAR [Rohling 1983] detector to generate proposals with *possible* objects. Because CA-CFAR outputs a binary segmentation mask and aims to detect objects as boxes, we transform segmentation masks into bounding boxes. Figure 5.9 summarises the process. For each segmentation mask, we first apply the DBSCAN algorithm to cluster objects and get all the instances in the image. Then, for each object, we define a bounding box $b = \{x_1, x_2, y_1, y_2\}$, where (x_1, y_1) and (y_1, y_2) are the bottom left and the top right coordinates of the object respectively.

We adopt a simple rule to match objects in two successive frames together. Supposing an object is moving with a mean radial relative speed v_r and the time between two successive frames is t_f, it travels a distance of $d = v_r * t_f$. Supposing the velocity of the object is constant between two frames, two objects are the same if:

$$|d_t - d_{t-1}| \leq d \pm \varepsilon_d \tag{5.1}$$

$$|v_{r_t} - v_{r_{t-1}}| \leq \varepsilon_v \tag{5.2}$$

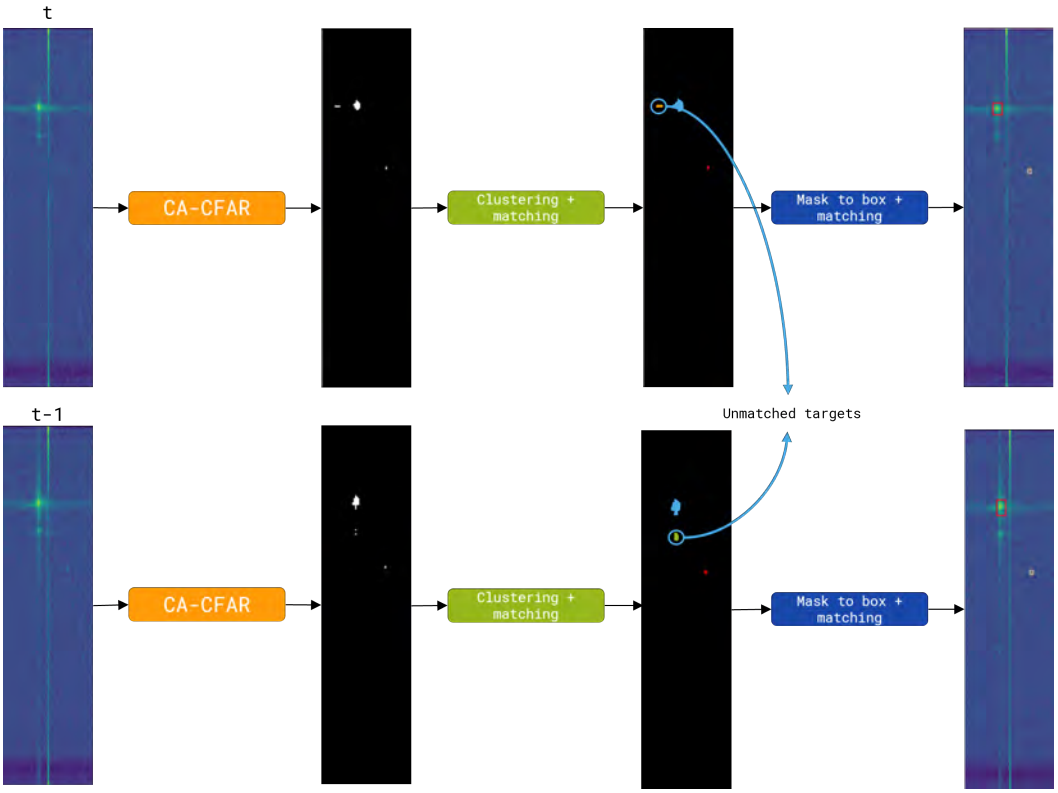


Figure 5.9: RICL object proposals generation and matching. First, CFAR is applied on frames at t and $t-1$. Then we cluster objects to retrieve all the objects in the spectra. Then, we match objects together and return bounding boxes for each valid object.

where d_t, d_{t-1}, v_{r_t}, $v_{r_{t-1}}$ are the range and the velocity of the objects at time t and $t-1$ respectively and, ε_d and ε_v are uncertainty constants for the range and the velocity. Indeed, because we compute the range and the velocity using the centre of the box, this constant is necessary to avoid matching errors.

View construction In our proposed framework, we use two views (instead of three in SoCo), namely R_t and R_{t-1}. The views correspond to two successive range-Doppler maps (separated by 0.1 seconds for the CARRADA dataset).

Object-level contrastive learning We aim to pre-train a model for object detection. Experiments in Chapter 3 show the relevance of the Faster-RCNN [Ren 2017] architecture for radar object detection. Thus, as SoCo, we use the Faster R-CNN framework to pre-train an object detection model for radar. The following explains the modified SoCo framework we use, proposed in [Wei 2021].

First, we compute RD spectrum-level features using a backbone f^S. With the

bounding box representation b, we apply RoIAlign [He 2017] to extract the *foreground* features from the last feature map of the backbone. An R-CNN head f^H is introduced into pre-training to align the pre-training for object detection. For a RD map R and a bounding box b, the object-level feature representation is:

$$h = f^H(\text{RoIAlign}(f^S(R), b)). \tag{5.3}$$

SoCo follows BYOL [Grill 2020] learning framework. Therefore, two neural networks are used to learn: the online and target networks. They share the same architecture, but they have different weights. The weights of the target network f_ξ^S, f_ξ^H are updated using an EMA of the student's weights f_θ^S, f_θ^H with a momentum coefficient τ. τ controls how fast the target's weights are updated. Extending Equation 5.3 to the online and target networks and multiple objects, the object-level feature representation h_i of a set of *possible* objects $\{b_i\}$ in views R_t and R_{t-1} is respectively:

$$h_i = f_\theta^H(\text{RoIAlign}(f_\theta^S(R_t), b_i)), \tag{5.4}$$

$$h_i^{'} = f_\xi^H(\text{RoIAlign}(f_\xi^S(R_{t-1}), b_i)). \tag{5.5}$$

As in BYOL, the online network is appended with a projector g_θ and a predictor $q\theta$ to obtain the latent embedding. The target network is only appended with the projector g_ξ to avoid trivial solutions. g_θ, q_θ and g_ξ are all two-layer MLPs in this chapter. The latent embedding of h_i and $h_i^{'}$ of object-level representations are defined respectively with:

$$v_i = q_\theta(g_\theta(h_i)), \quad v_i^{'} = g_\xi(h_i^{'}) \tag{5.6}$$

The contrastive loss for the i-th *possible* object is defined as:

$$\mathcal{L}_i = -2 \cdot \frac{\langle v_i, v_i^{'} \rangle}{\|v_i\|_2 \cdot \|v_i^{'}\|_2} \tag{5.7}$$

Then, the overall loss function for a pair of RD maps is:

$$\mathcal{L} = \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}_i \tag{5.8}$$

where K is the number of *possible* objects in the RD maps. Finally, as in SoCo and BYOL, the loss is symmetrised by separately feeding R_t to the target network and R_{t-1} to the online network to compute $\tilde{\mathcal{L}}$. At each training iteration, a stochastic optimisation step is performed to minimise $\mathcal{L}^{RICL} = \mathcal{L} + \tilde{\mathcal{L}}$.

5.6.2 Experiments

5.6.2.1 Pre-training settings

Architecture We adopt the Faster R-CNN [Ren 2017] framework to evaluate the pre-training strategy. To match with the SoCo framework, which uses the fifth residual block of a ResNet [He 2016] architecture as the detection head, we use a ResNet-50 backbone. We modify the pooling stride of the ResNet architecture to obtain a feature map with the same size as DAROD (*i.e.* 32×32, see Chapter 3). The RoI align operation is applied on the fourth residual block of the ResNet. We follow SoCo and BYOL using two layers MLPs for the projection and the prediction network. They consist of a linear layer with an output size of 4096 followed by batch normalisation, rectified linear units (ReLU), and a final linear layer with an output dimension of 256. Because we deal with small objects, the size of the RoI pooling is set to four.

Dataset and data augmentation We pre-train Faster R-CNN using the CARRADA [Ouaknine 2021b] dataset, without labels. We apply horizontal and vertical flipping to avoid dimensional collapse. Note that we always apply the flipping for both frames.

Object proposals generation and matching We use the following settings for CA-CFAR. The range and Doppler guard length are set to four and two, respectively. The range and Doppler training length are set to 20 and 10, respectively. We set ε_v to 0.6 and ε_d to 1. We find these values provide the best matching between frames.

Optimisation We use a 100-epoch training schedule to pre-train the model. We use the SGD optimiser with a cosine decay learning rate schedule and a warm-up period of 10 epochs. We set the learning rate to 0.001. The weight decay is set to 1×10^{-3}. The total batch size is set to 16. For the update of the target network, the momentum coefficient τ starts from 0.9 and is increased to one during training. Note that because this chapter presents preliminary results, we did not extensively research the hyper-parameters.

5.6.2.2 Fine-tuning settings

For ease of development, we use the Detectron API[2] to fine-tune the Faster R-CNN framework.

[2] `https://github.com/facebookresearch/Detectron`

% of labels	100%		50%		20%		10%		5%	
	AP	AP@0.5	AP	AP@0.5	AP	AP@0.5	AP	AP@0.5	AP	AP0.5
Supervised	13.62	39.39	10.82	33.86	5.11	17.63	6.26	22.04	3.58	13.17
ImageNet	**16.19**	**42.53**	**16.67**	**40.86**	**8.54**	**25.52**	**11.95**	**35.98**	**7.94**	**25.38**
RICL	13.68	39.34	11.24	34.61	6.99	22.71	8.25	29.81	4.21	15.46

Table 5.1: Comparison of RICL with a supervised approach and ImageNet pre-training for different amounts of data. AP stands for the COCO mAP. AP@0.5 is the AP at IoU threshold 0.5. Best results are in bold, second best are underlined.

Weights initialisation We consider three training schemes: *supervised*, *ImageNet* and *RICL*. The *supervised* model is the default Detectron Faster R-CNN implementation we train from scratch. The *ImageNet* model is the default Detectron Faster R-CNN implementation where the backbone (ResNet-50) is initalised using pre-trained weights from ImageNet. The *RICL* model has the same code base as the *supervised* and *ImageNet*, but the backbone and the detection head are initialised using the weights of the backbone and the detection head from the *online* network. We do not load the weights of the projection and the prediction networks.

Dataset This chapter aims to learn with fewer data. To evaluate the pre-training strategy, we create subsets of the training set of the CARRADA dataset. We train and fine-tune the same model five times with 100%, 50%, 20%, 10% and 5% of the training dataset. We create these splits randomly and use the same subsets for the *supervised*, *ImageNet* and the *RICL* training. We validate and test the model with 100% of the validation and testing sets.

Optimisation We use the same hyper-parameters as in Chapter 3 to train the Faster R-CNN model. We train all the models using an SGD optimiser. The base learning rate is set to 0.02 and increases linearly for the first 1000 iterations [Goyal 2017] and drops twice by a factor of two after 2500 and 5000 steps. We train the models for 20000 iterations and use early stopping to avoid overfitting. Weight decay of 1×10^{-4} is used.

Metrics We report the bounding boxes COCO AP and the AP@0.5. COCO AP corresponds to the mean of all AP@IoU where IoU ranges from 0.5 to 0.95 with 0.05 step.

5.6.2.3 Results

We report in Table 5.1 the preliminary results of the RICL pre-training strategy compared to a supervised approach with random weights initialisation and ImageNet pre-training. Overall, we notice RICL speed-up convergence and outperforms

the supervised training when trained with fewer data but remains less effective than a pre-training on ImageNet. Although performing similarly with 100% and 50% of the training set, RICL boosts the performance of Faster R-CNN when training with 20% and 10% of the data. Indeed, pre-training Faster R-CNN with RICL allows an improvement of five and seven points when 20% and 10% of the dataset are used, respectively.

Experiments using ImageNet pre-trained weights confirm the findings in Chapter 3. No matter the amount of data, ImageNet pre-training outperforms RICL and the training from scratch by far. We notice that the model trained with the ImageNet weights slightly outperforms the model trained from scratch while being trained with only 50% of the training set. Also, this model reaches the same AP@0.5 using 5% and 20% of the training set. However, this should be treated cautiously since we construct the training subsets randomly. For all training strategies, the AP and the AP@0.5 are higher using 10% of the data than using 20%. To avoid this phenomenon, cross-validation should be used.

Finally, though RICL pre-training generally improves performance compared to training from scratch, there is still a gap between a pre-training on ImageNet and our method.

5.7 Conclusion and perspectives

This chapter presented preliminary and encouraging results to pre-train radar object detectors. We modified the SoCo [Wei 2021] framework (which is an extension of the BYOL [Grill 2020]) to the range-Doppler spectrum, and we show that we can leverage the radar knowledge to pseudo-label the data and use the pseudo-labels to learn useful representation. The RICL pre-training strategy generally improves the detection performance when using less data but remains far from a pre-training on a large dataset like ImageNet and a fully supervised approach. We now raise some limitations to our approach and some research tracks to improve it.

The model First, for fast prototyping, we use a ResNet-50 backbone which is not appropriate for radar. This choice has been motivated by the availability of the pre-trained weights on the ImageNet dataset to compare with our approach. Indeed, the results we obtain with this backbone are far from those presented in Chapter 3. Though the aim of this chapter is more to present preliminary results than obtaining a high-performance model, experiments using the DAROD (see Chapter 3) or the RECORD (see Chapter 4) models must be conducted. Indeed, those backbones have been proven to be more adapted to radar data than the ResNet-50 model. Since the ResNet-50 model is larger than DAROD and RECORD, it requires more data to reach good performance. As mentioned throughout this thesis, the size of

radar datasets are not sufficient to train larger models.

Matching objects together We use a greedy method to match the detection from CFAR, sometimes resulting in matching noise with an object or matching different objects when two detected objets are close. It can perturb the pre-training and lead to poor learning of representations. A possible improvement for better pre-training is to track objects using Kalman filtering [Kalman 1960] for a more extended period and use the tracker's output to match objects together. Target tracking allows accurate target matching and an improved dataset for learning representations. Additionally, it enables not to use only two successive frames but to encourage the similarity between the same object at non-consecutive time steps. Indeed, according to the radar equation (see Equation 1.1), the same object at different range and velocity have a variable signature. Encouraging the embedding of RICL to be close to the same object at non-consecutive time steps might help to learn better representations of this object. Finally, in the SoCo paper [Wei 2021], authors show that using more than two views improves the results. For further work, we consider feeding the *target* network with a third frame with the same objects as the two others.

The amount of data We use the CARRADA dataset to learn representation. However, the CARRADA dataset is small (around 4000 frames contain objects), and we might need more data to learn good object-level representation. One key ingredient in SSL is the amount of data. It has been shown for computer vision that the more data, the better the pre-training. We build RICL under the assumption that the radar data is simple (few objects, mostly noise) and that we can learn object-level representation with little data. However, results in Table 5.1 show no differences in the performance between a model trained from scratch and pre-trained with 100% of the dataset. More, using pre-trained weights from the ImageNet dataset outperforms our approach in all scenarios. This suggests that the proposed pre-training is perfectible and can be further improved. Pre-training using larger radar datasets such as RADDet (10000 frames) and RADIal (31000 frames, 8000 with annotations) needs to be conducted. However, using these datasets implies modifying our RICL framework. Indeed, the RADDet dataset does not contain successive frames. As for the RADIal dataset, the data is a MIMO RD spectrum which requires different processing than the non-MIMO RD spectrum. The RADIal dataset is preferred because its size and format are in tune with the future generation of radars.

Conclusion

The search for safer and more robust perception systems led to the massive use of AI models to detect and identify objects in complex urban environments. Presently, most perception systems use cameras and LiDAR sensors to build a representation of the scene. The use of radar sensors remains sporadic and dedicated to tasks requiring speed estimation, and the use of AI for radar processing is limited to point cloud classification. In this thesis, we successfully demonstrated the potential of AI models in enhancing automotive radar perception using raw data. By exploring radar spectra such as range-Doppler, range-angle, and range-angle-Doppler, this work has established their effectiveness in substituting radar point clouds and various aspects of the radar signal processing chain for object detection. In the different chapters of this thesis, we endeavoured to find the best representation of the radar signal to use and the most appropriate formulation to learn to detect and identify objects using raw data.

Can computer vision models be adapted to radar data?

In Chapter 3, we tried to adapt computer vision models to radar spectra, specifically the range-Doppler spectrum, showcasing the relevance of computer vision models in detecting and classifying objects. Notably, this chapter shows that designing light and simple backbones to account for radar data specifics (speed information, smaller input data, varying objects' shape and size) is more effective than using larger ones. This is true for the data we used in this thesis but should be confirmed on high-resolution data and larger datasets. Moreover, we found that using a backbone pre-trained on the ImageNet dataset significantly improved the detection performance. These experiments paved the way for Chapter 5's preliminary work. Finally, this chapter confirmed the superiority of AI models over conventional methods for road-user object detection and classification using RD maps.

Although informative, the comparison with conventional radar object detectors was somewhat unfair because we have compared our model with detection projected onto the RD spectrum. A more appropriate comparison would be to generate the radar point clouds by replacing CFAR with DAROD, computing the direction-of-arrival of the targets and applying the subsequent post-processing steps (tracking, ego-motion compensation). We expect our method to be still superior because DAROD originally outputs fewer false-positives than CFAR.

How do we learn spatio-temporal relationships in radar?

The model introduced in Chapter 4, named RECORD, a combination of convolution and convolutional LSTMs, emerged as a versatile framework capable of learning spatio-temporal relationships across different radar data types and tasks. Notably, this model outperformed existing models based on temporal convolutions, attentions or ViT while maintaining computational efficiency, rendering it a promising solution for embedded systems. Convolutional LSTMs appeared to be more adapted to radar data than temporal convolutions because of their ability to learn long-term spatio-temporal dependencies at multiple scales. One limitation of this work lies in the feasibility of the buffer training mode. On the CARRADA dataset, the model that uses a buffer of N frames outperforms TMVA-Net on both RD and RA views. However, despite having few GMACS compared to TMVA-Net, the runtime of this approach is inappropriate for real-time applications (almost 100ms for a forward pass).

To some extent, this chapter revealed the limitations of multi-view models. While interesting for research purposes, they are long, hard to optimise, and challenging to deploy in real scenarios, as discussed in Chapter 4. As the resolution of radar increases, it requires more memory and computation to store and produce RAD cubes. From Chapter 4, we concluded that single-view models are more appropriate for raw data than multi-view models. They should mainly be used on RD spectra to detect objects and coupled with the direction of arrival estimation method to detect objects in 3D.

Pre-training radar object detectors

Chapter 5 endeavours to address the challenge of learning with limited data, which led to the creation of a pretraining framework inspired by SoCo, named RICL. While yielding promising results, it became evident that pretraining the model with large image datasets (like ImageNet) is more efficient than the proposed method. The results presented in Chapter 5 are preliminary. Thus, this chapter's work must be improved and compared with other image-based SSL frameworks. Among possible improvements, we propose using target tracking algorithms to match the same objects within successive frames and train RICL on a larger radar dataset like RADIal. Also, experiments were conducted using a ResNet-50 backbone, which has been proven not to be the most adequate backbone for radar object detection.

Further investigations using the backbones proposed in this thesis should be conducted. Last, experiments using a backbone pre-trained on ImageNet confirmed that the dataset size used to pre-train the model matters more than the framework. The suggestion to conduct experiments with larger radar datasets, potentially ac-

quired through unannotated radar data collection campaigns, presents an avenue for future research. Owing to the complexity of annotation radar data, self-supervised learning is a promising path to improve AI-driven radar object detection.

Limitations

Single-stage object detectors vs. two-stages object detectors In Chapter 3, we adapt the Faster R-CNN architecture to radar data for object detection on RD maps. However, as explained in Section 2.2.2, two-stage detectors are more accurate than single-stage detectors but slower. For embedded applications, it is preferable to have a fast inference model. Hence, single-stage models should be preferred. We decided to use a two-stage detector to increase the accuracy of our model. Regarding the simple nature of radar data, it is worth asking if two-stage detectors are needed. Single-stage detectors endowed with a relevant backbone have a good chance of being as accurate as two-stage detectors.

Problem formulation Throughout this thesis, we experimented with very different object recognition tasks: object detection, semantic segmentation and point-based detection. Today, there has yet to be a consensus in the radar community about which task to use for radar object detection. In Chapters 3 and 5, we conducted work by representing objects with bounding boxes to account for objects' shape variation and the uncertainty on the localisation. In Chapter 4, we adopted the *object as point* representation from [Wang 2021c] to detect objects in RA maps. We also trained RECORD to segment objects in RA and RD maps. Each task has its advantages and disadvantages.

On the one hand, representing objects as bounding boxes makes the model more robust to shape and intensity variation and relaxes constraints on object localisation during the training. However, retrieving the object's exact position requires more complex post-processing before being incorporated into the complete radar system. On the other hand, learning to produce segmentation masks is analogous to the output of CFAR detectors, making integrating AI models into the system easier. Applied on range-Doppler maps, semantic segmentation models show a promising path toward real-time application. Lastly, representing objects as points is a well-chosen task for radar object detection. In radar, the default representation of objects is a list of points with position, speed and RCS information. Hence, it would be a natural direction to output a similar representation. Naturally, this requires annotating radar datasets in this way. This thesis did not conclude about the best formulation to detect objects using raw radar data but gives clues about which one to choose in which application.

Comparison with point cloud based approaches This thesis aimed to build deep neural networks for radar object detection using raw data. We demonstrated that strong performances can be achieved using deep learning on radar spectra, particularly on the range-Doppler maps. However, we did not compare our work with AI models on radar point clouds like [Palffy 2022, Saini 2023]. We did not perform this comparison because of the unavailability of radar datasets containing both ADC data (or spectrum) and corresponding annotated point clouds.

Pretraining strategy The results presented in Chapter 5 remain far from those obtained using the weights of a model pre-trained on ImageNet. This chapter aimed to show that a good pre-training strategy allows for a reduction in the number of labelled data without losing performance. Nevertheless, Chapter 5's work is preliminary, and we agree that it can be improved in many ways. First, using a ResNet-50 backbone instead of DAROD, RECORD or other radar-based backbones [Zhang 2021, Rebut 2022, Giroux 2023] might prevent the model from learning relevant features from the data. This choice was motivated by the availability of pre-trained weights on ImageNet for ResNet-50. Second, knowing the key ingredient of self-supervised learning is the amount of data available, we use the smallest radar dataset available. For sure, this hindered the learning of representations. Last, we compared our model with a *supervised* pre-training strategy on the ImageNet dataset. As our pre-training strategy is unsupervised, we believe we could have made a comparison with other image-based pre-training strategies such as SoCo [Wei 2021], BYOL [Grill 2020] or MoCo [He 2020].

Future works

Deploying our models in the real world As we gaze into the short-term perspective, this thesis contributes to deciding on the dimension of future generations of radar hardware accelerators by providing key performance indicators like the number of parameters, GMACS and the runtime required to reach a certain level of performance. Deep learning models are often over parameterised; therefore, pruning and quantisation will be pivotal to optimising the models proposed in this thesis for real-world deployment. Beyond pruning the models, the models proposed in this thesis can serve as base models for NAS, as proposed by Boot *et al.* in [Boot 2023]. In addition to enhancing the efficiency of the models, their integration into the system must be considered. From Chapter 4, we saw that with the increasing resolution of radar sensors, using RAD tensors and, therefore, multi-view models appears difficult for computational reasons. A first step towards integrating deep learning models into radar systems is to use single-view models instead of CFAR detectors to detect objects in the RD view before calculating their direction

of arrival. In this way, we can obtain a list of targets containing the position (distance, azimuth, elevation), the speed and the class of the objects. Besides allowing us to get a representation of the environment, it will enable us to compare raw data-based methods with point cloud-based methods.

Improving RICL As explained above, the pre-training strategy proposed in Chapter 5 offers many avenues for improvement. First, the methodology to match the same objects together must be improved. We consider tracking targets through time to perform the matching. Second, a backbone designed for radar data should be used. Among the available backbones, we will first try to use the backbone proposed in this thesis, but experiments using the FFT-RADNet model [Rebut 2022] or the Radar-ResNet from [Zhang 2021] are contemplated. Last, we will use a larger dataset like RADIal [Rebut 2022] to pre-train the model using RICL.

With the advent of high-resolution imaging radars, another idea for pre-training radar object detectors is to exploit the dense point clouds they produce. Instead of extracting object-level features using CFAR and DBSCAN algorithms, we can project the point clouds on the RD spectrum to extract regions of interest (similar to what Palffy *et al.* do in RTC-Net [Palffy 2020]). Because the point clouds originate from the radar signal processing chain, they are already clustered and tracked. Therefore, using such a hybrid approach (mixing target-level and low-level features) allows more accurate feature extraction than the one we proposed in Chapter 5.

Another research direction is learning the object's location from high-resolution radar point clouds as a pretext task. This is very different from RICL but might be considered for future works. Such a pretext task might ensure the model learning a 3D representation of the environment. Another possibility lies in a sensor fusion-based approach using LiDAR point clouds and radar point clouds.

We think that the success of self-supervised learning for radar lies in increasing the size of the radar datasets. As pre-training radar models do not necessarily require labelled data, we hope this thesis will encourage academic and industrial researchers to release large unlabelled radar datasets with point clouds and raw data available.

Sensor fusion Sensor fusion is emerging as an approach of choice for guaranteeing high-quality perception and improved robustness of ADAS systems. The efficiency of sensor fusion depends upon the robustness and the performance of individual sensor processing. With the models proposed in this thesis and the growing interest in deep learning for radar, the next step towards robust and redundant perception systems is based on sensor fusion.

Building transparent, explainable and reliable models Since the models proposed in this thesis aim to be deployed in critical systems, they must be transparent, explainable and reliable following the recommendation of the European Union in its 2020 report on "Robustness and Explainability of Artificial Intelligence" [Hamon 2020]. This is even truer if we consider replacing the entire radar processing chain with an AI model. While the current radar signal processing chain is fully interpretable (*i.e.* the ability to understand the decision-making process), the AI models we used in this thesis are not. Developing hybrid models is a potential path toward certifiable AI-driven radar systems. Hybrid models consist in developing AI models that mix radar knowledge (*target-level*) and the radar cube (*low-level*). As the point clouds of the generation of radar become dense, it is possible to use the detections from the radar signal processing chain to detect targets and extract regions of interest from the range-Doppler spectrum to classify them following [Palffy 2020]. Using the detection from the radar signal processing chain is the first step for increasing the interpretability of the models because we can explain why the target is detected. The next step in building explainable models is to use explainable AI toolboxes [Fel 2022].

Toward AI-enabled automotive radars From a longer-term perspective, with the emergence of high-resolution 4D radars, we can imagine building an all-in-one AI system able to detect, classify and track objects in a single forward pass. Such a system would substantially reduce the number of processing steps by making the prediction either from the RD spectrum or by learning the signal processing transformation from ADC data as suggested in [Zhao 2023]. To build such a system, one first needs to measure the benefits of AI models over the current radar signal processing chain, its computational costs, the input data to use and its place in the data flow.

To conclude, this thesis represents a fundamental step towards increasing the robustness of driver assistance systems by making better use of radar sensors. This work underscores the ever-evolving synergy between artificial intelligence and radar technology, charting a course toward safer and more advanced automotive systems.

Appendix A

Multiple road-users detection on Range-Doppler maps

A.1 CARRADA dataset

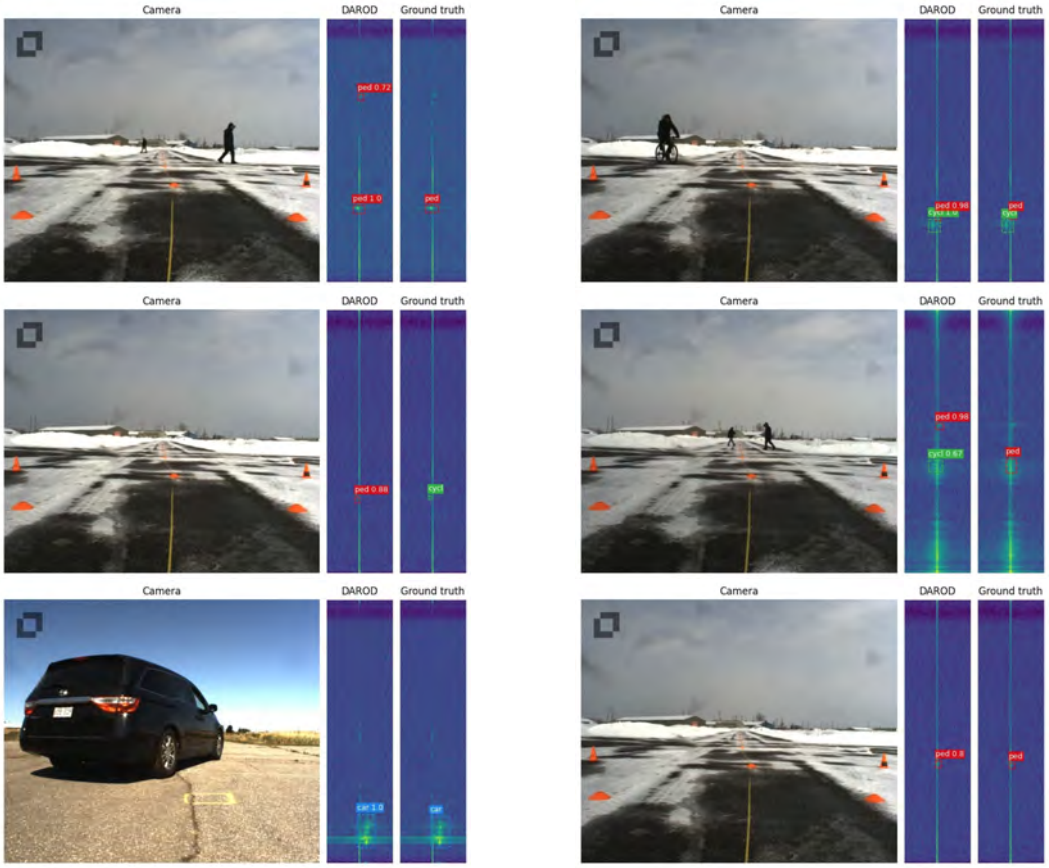


Figure A.1: Qualitative results for **DAROD** on CARRADA dataset.

A.2 RADDet dataset

Figure A.2: Qualitative results for **DAROD** on RADDet dataset.

Appendix B

Online object detection from radar raw data

B.1 UTAE performances improvement.

Table B.1 depicts the performance improvement of UTAE [Fare Garnot 2021] model with and without positional encoding and with a modified number of channels in the encoder and the decoder. We modify the number of channels of UTAE to match our architecture. We found that decreasing the model size and using positional encoding improve the model's performance. We define positional encoding as the time between the first and the k^{th} frames.

# channels		Pos. enc.	AP	AR	Params (M)
Encoder	Decoder				
16, 32, 64, 128	16, 32, 64, 128	Yes	**68.4**	**78.4**	0.79
16, 32, 64, 128	16, 32, 64, 128	No	46.9	64.3	0.79
64, 64, 64, 128	32, 32, 64, 128	Yes	60.8	77.9	1.1

Table B.1: Performances improvement of UTAE model with and without positional encoding and with the default architecture (underlined line). Results are obtained on the test set and on a single seed.

B.2 Single-view object detection

B.2.1 RECORD (online)

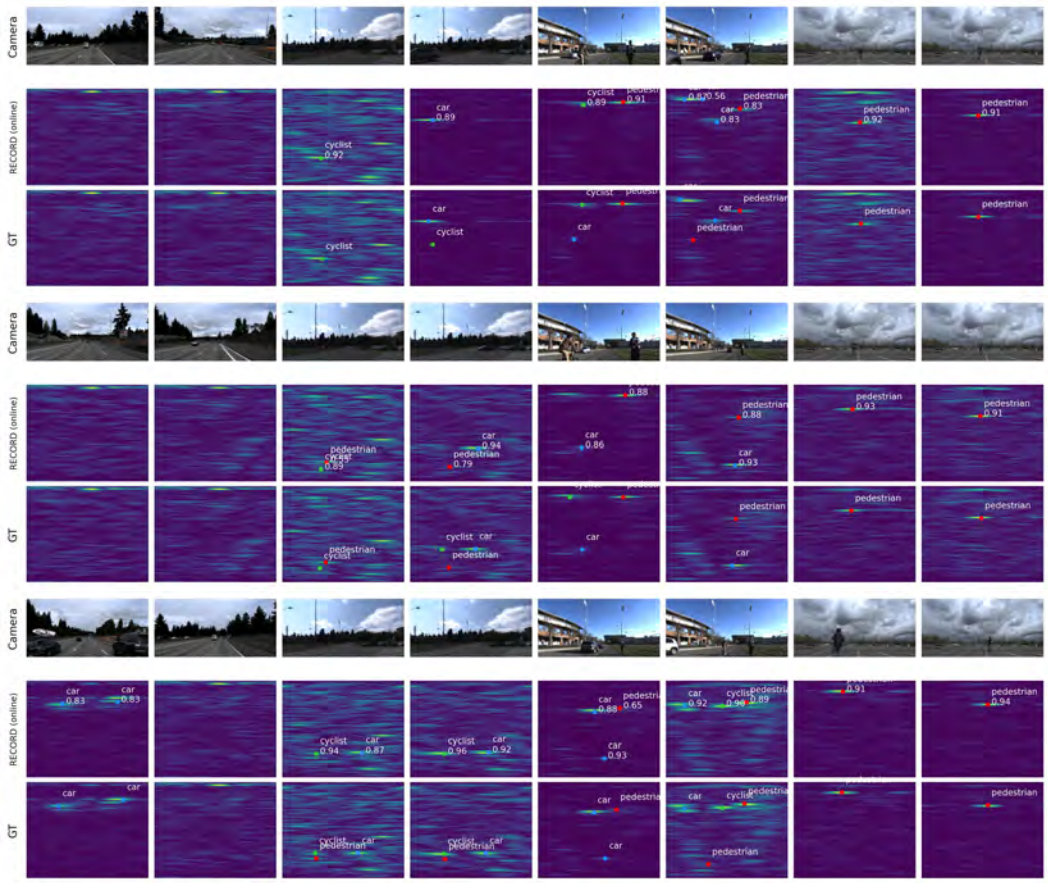


Figure B.1: RECORD (online) qualitative results on the ROD2021 dataset. We sample two example per scenario. From left to right: highway, city street, campus road and parking lot.

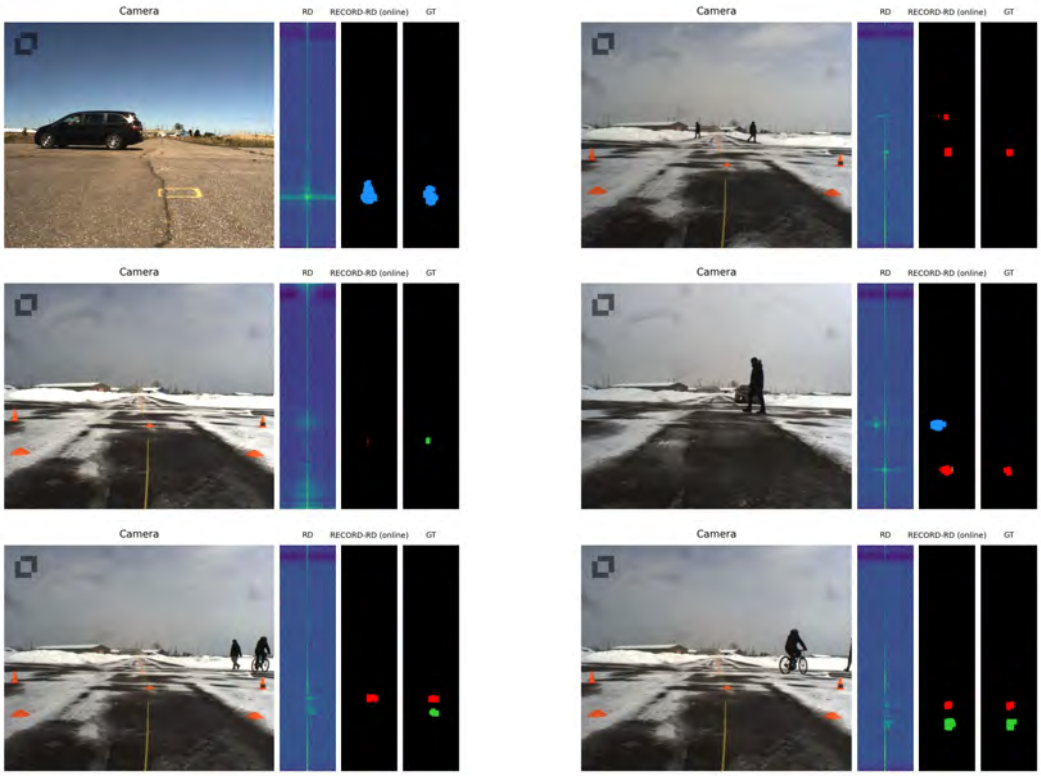


Figure B.2: Qualitative results for RECORD (online) on CARRADA dataset (RD view). From left to right: camera image, RD spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

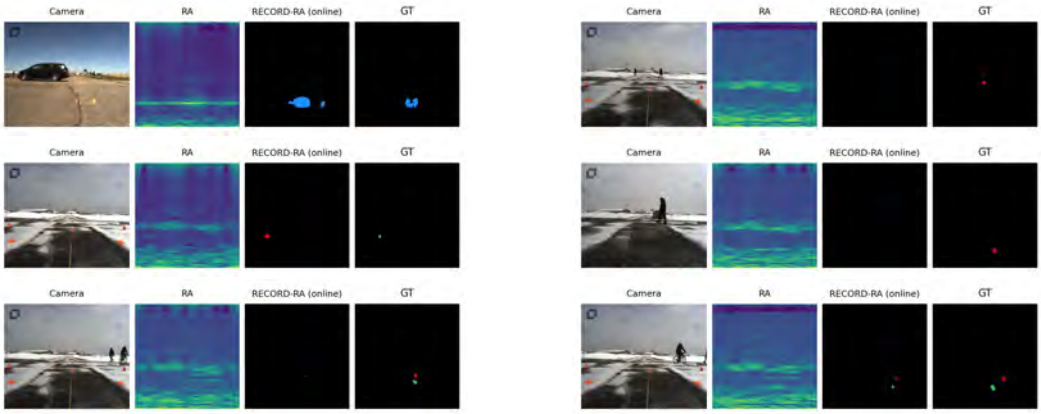


Figure B.3: Qualitative results for RECORD (online) on CARRADA dataset (RA view). From left to right: camera image, RA spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

B.2.2 RECORD (buffer)

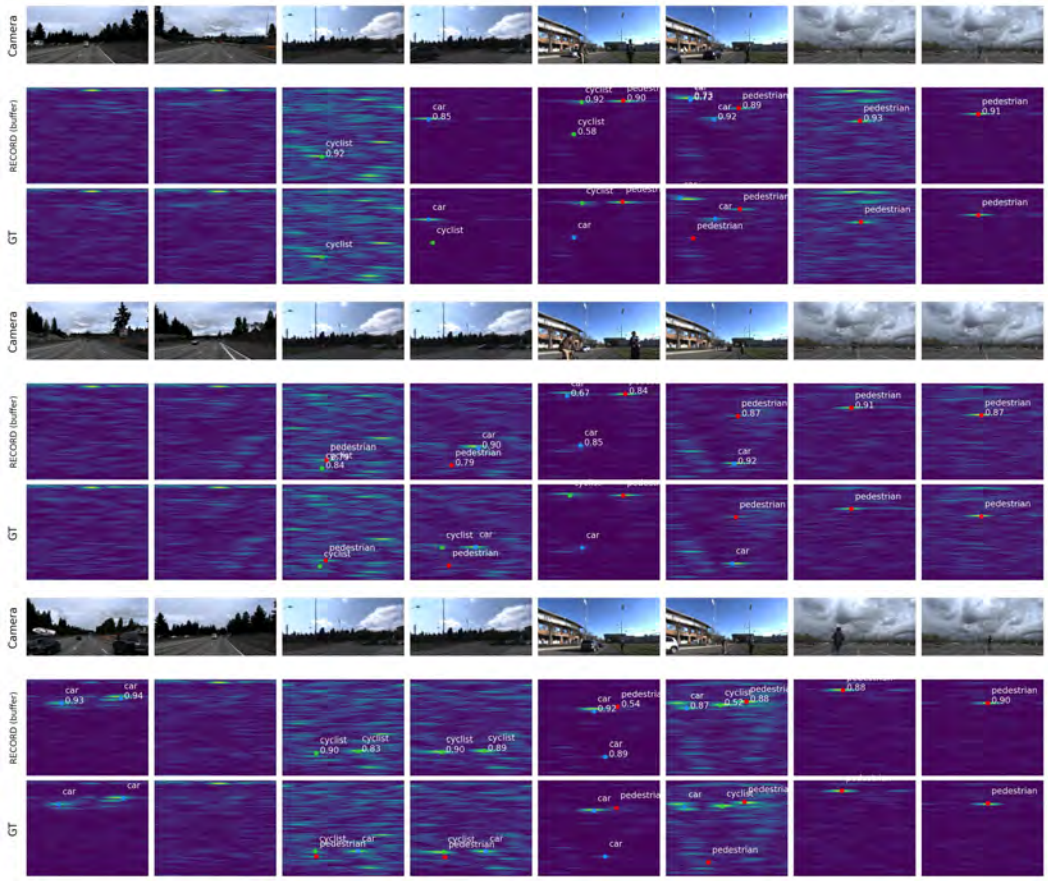


Figure B.4: RECORD (buffer) qualitative results on the ROD2021 dataset. We sample two example per scenario. From left to right: highway, city street, campus road and parking lot.

B.2.3 RECORD (no LSTM, multi)

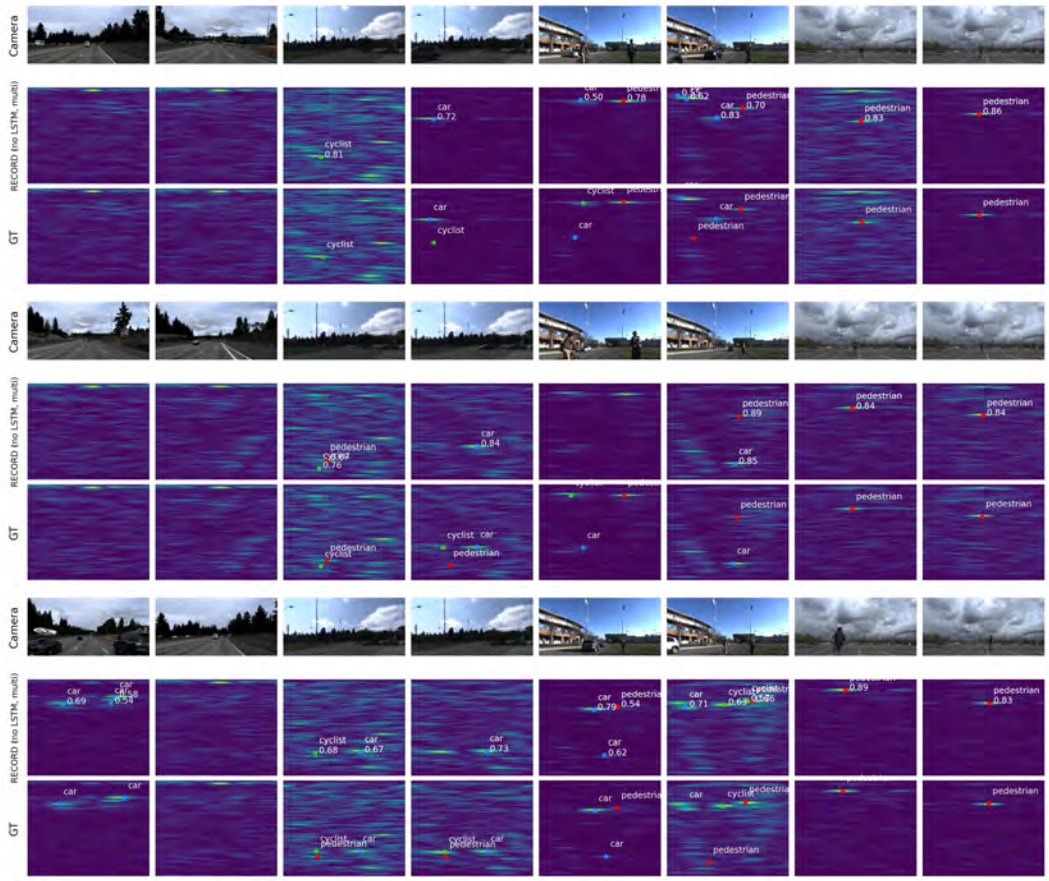


Figure B.5: RECORD (no LSTM, multi) qualitative results on the ROD2021 dataset. We sample two example per scenario. From left to right: highway, city street, campus road and parking lot.

B.2.4 RECORD (no LSTM, single)

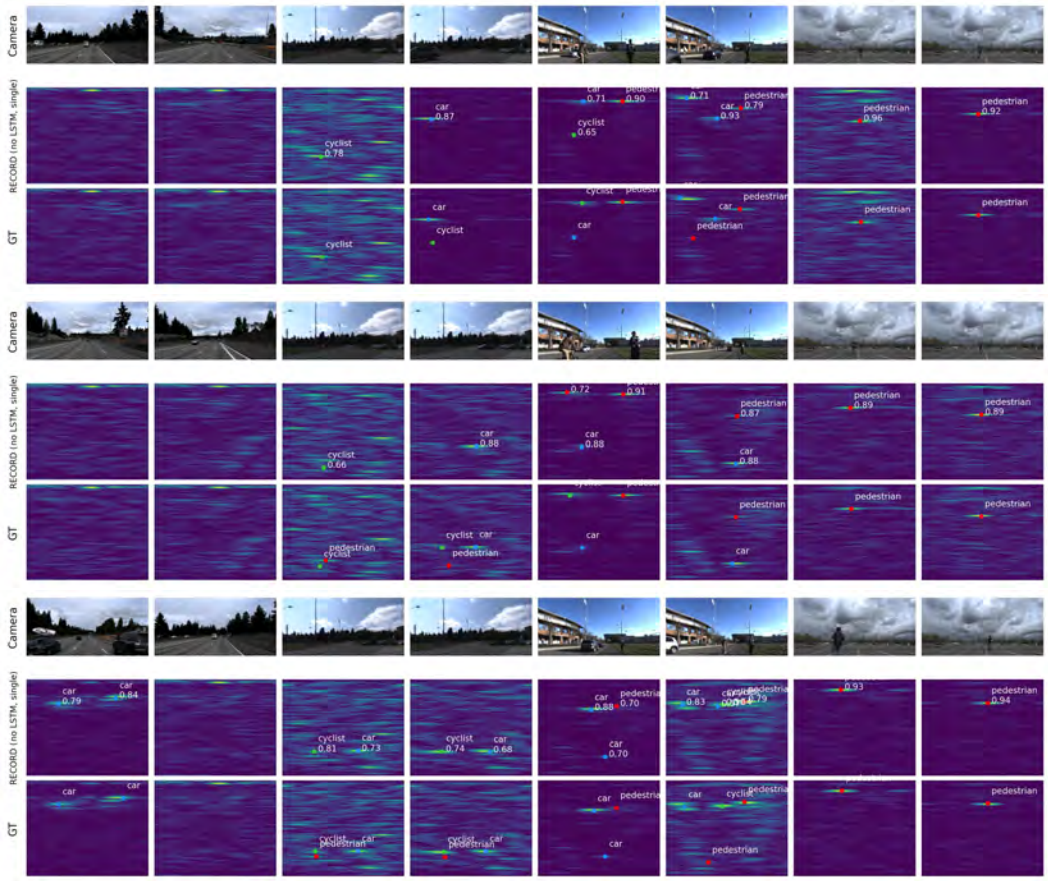


Figure B.6: RECORD (no LSTM, single) qualitative results on the ROD2021 dataset. We sample two example per scenario. From left to right: highway, city street, campus road and parking lot.

B.2.5 DANet

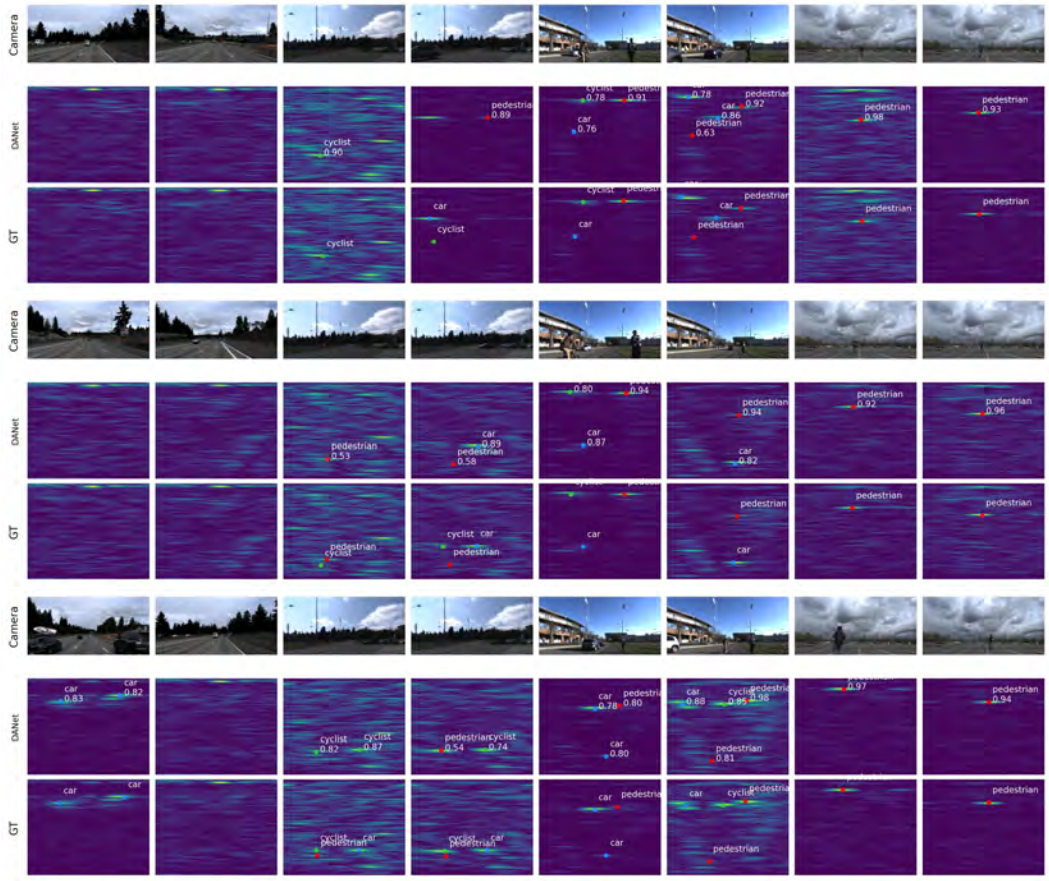


Figure B.7: DANet [Ju 2021] qualitative results on the ROD2021 dataset. We sample two example per scenario. From left to right: highway, city street, campus road and parking lot.

B.2.6 UTAE

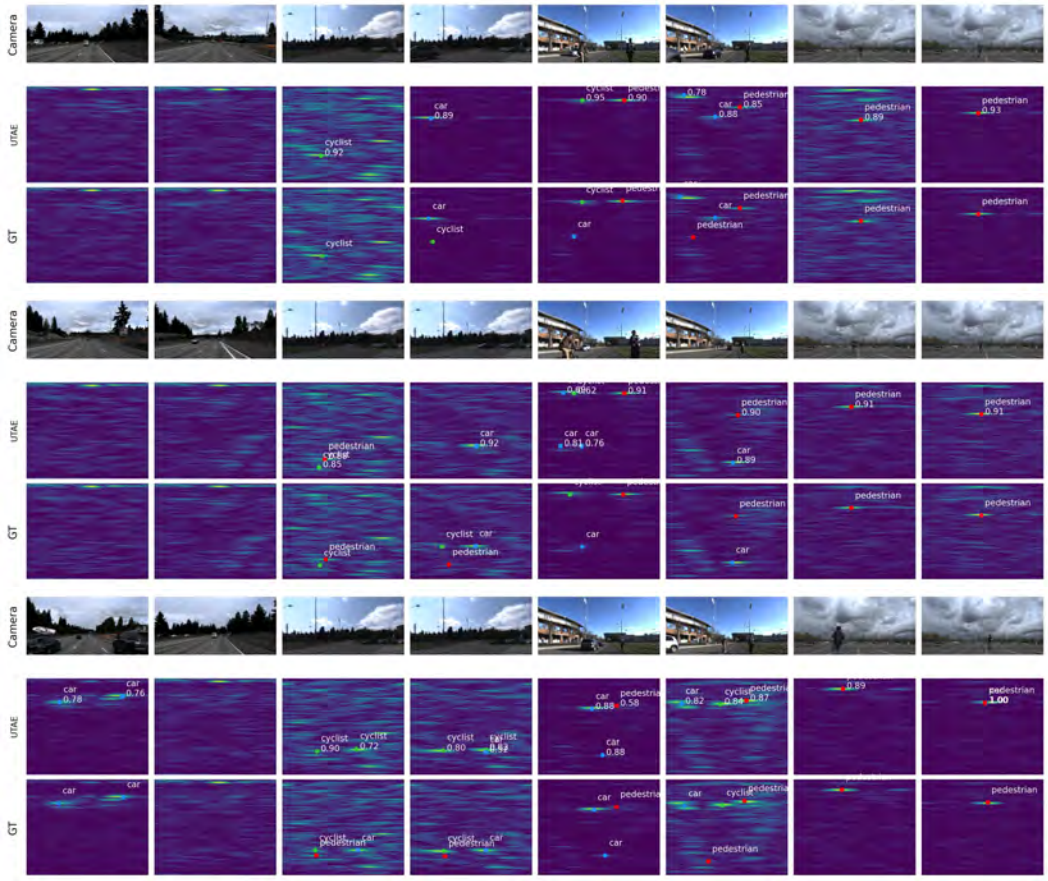


Figure B.8: UTAE [Fare Garnot 2021] qualitative results on the ROD2021 dataset. We sample two example per scenario. From left to right: highway, city street, campus road and parking lot.

B.2.7 T-RODNet

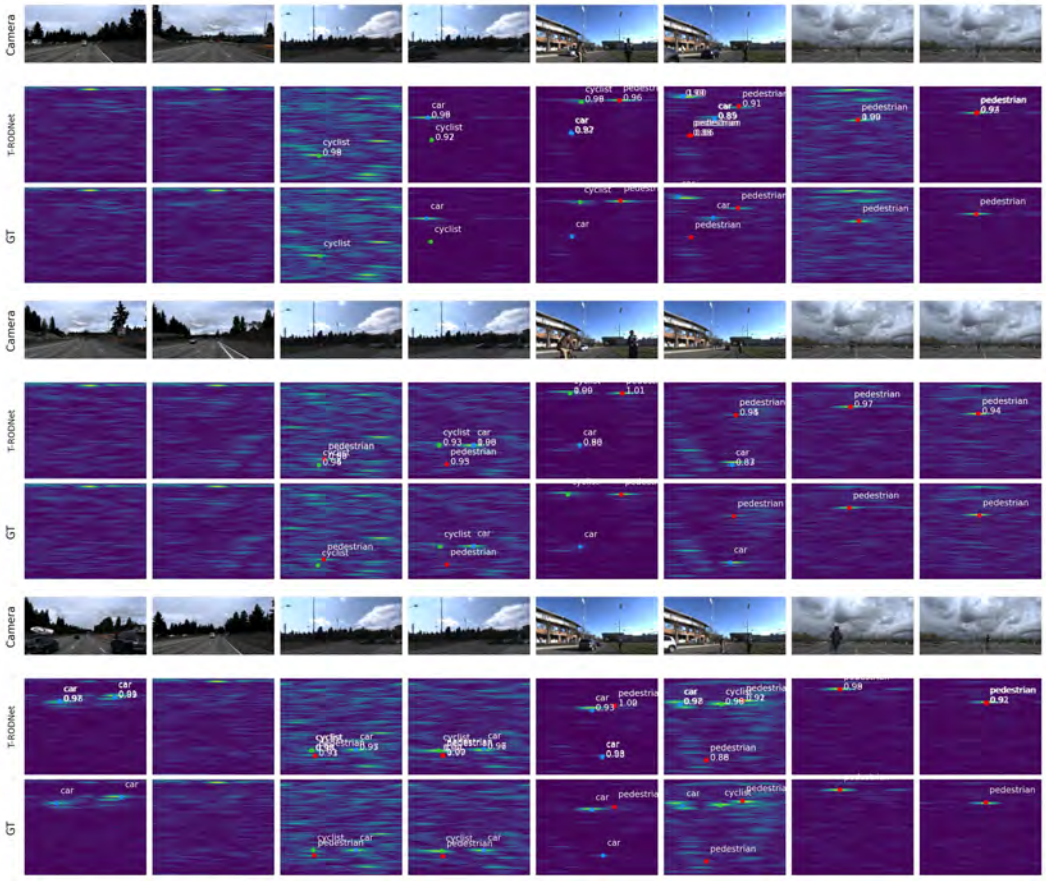


Figure B.9: T-RODNet [Jiang 2023] qualitative results on the ROD2021 dataset. We sample two example per scenario. From left to right: highway, city street, campus road and parking lot.

B.3 Multi-view object detection

B.3.1 MV-RECORD (online)

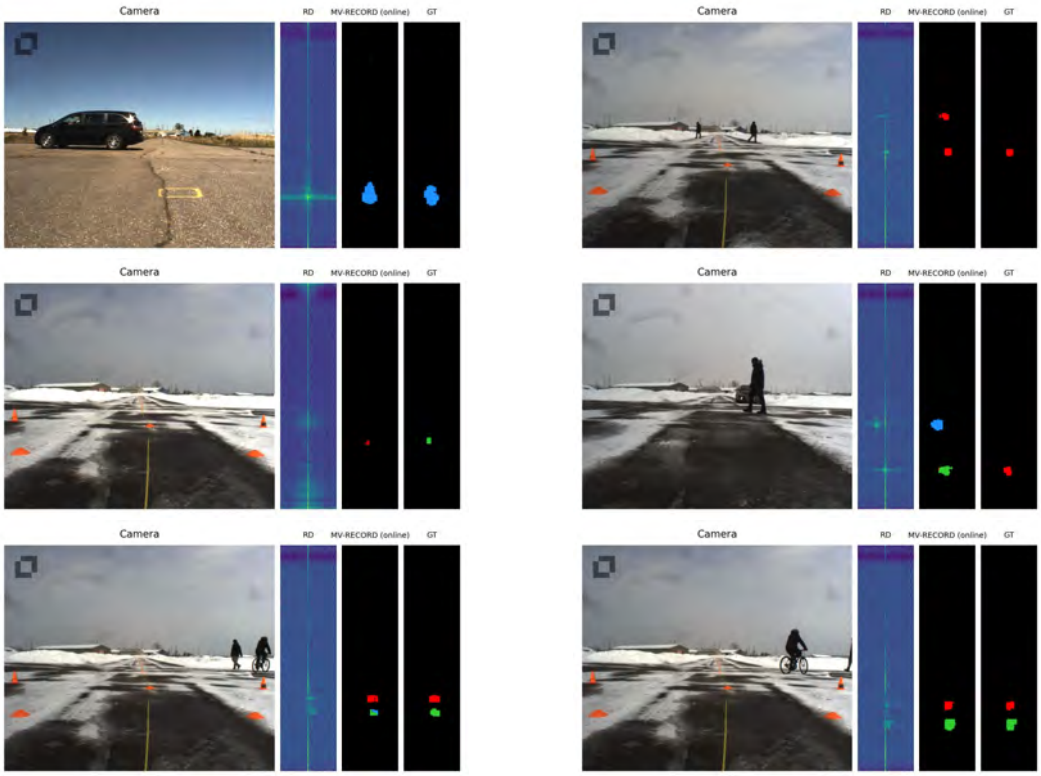


Figure B.10: Qualitative results for **MV-RECORD (online)** on CARRADA dataset (RD view). From left to right: camera image, RD spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

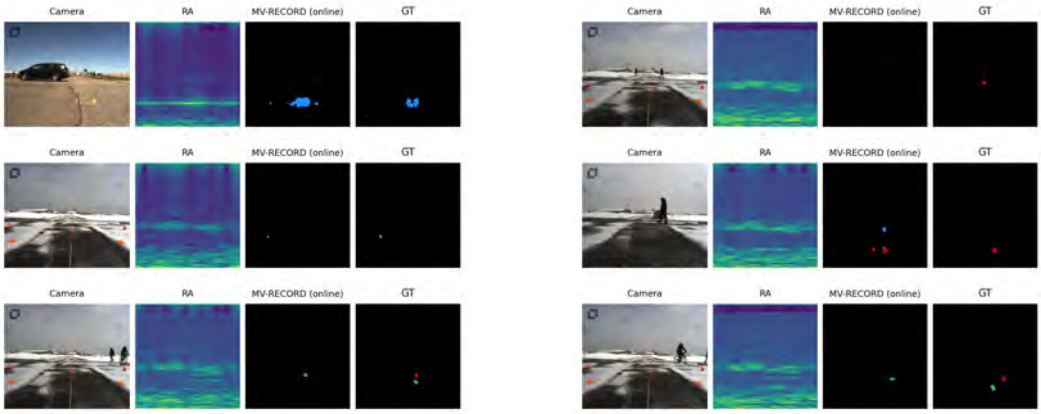


Figure B.11: Qualitative results for **MV-RECORD (online)** on CARRADA dataset (RA view). From left to right: camera image, RA spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

B.3.2 MV-RECORD (buffer)

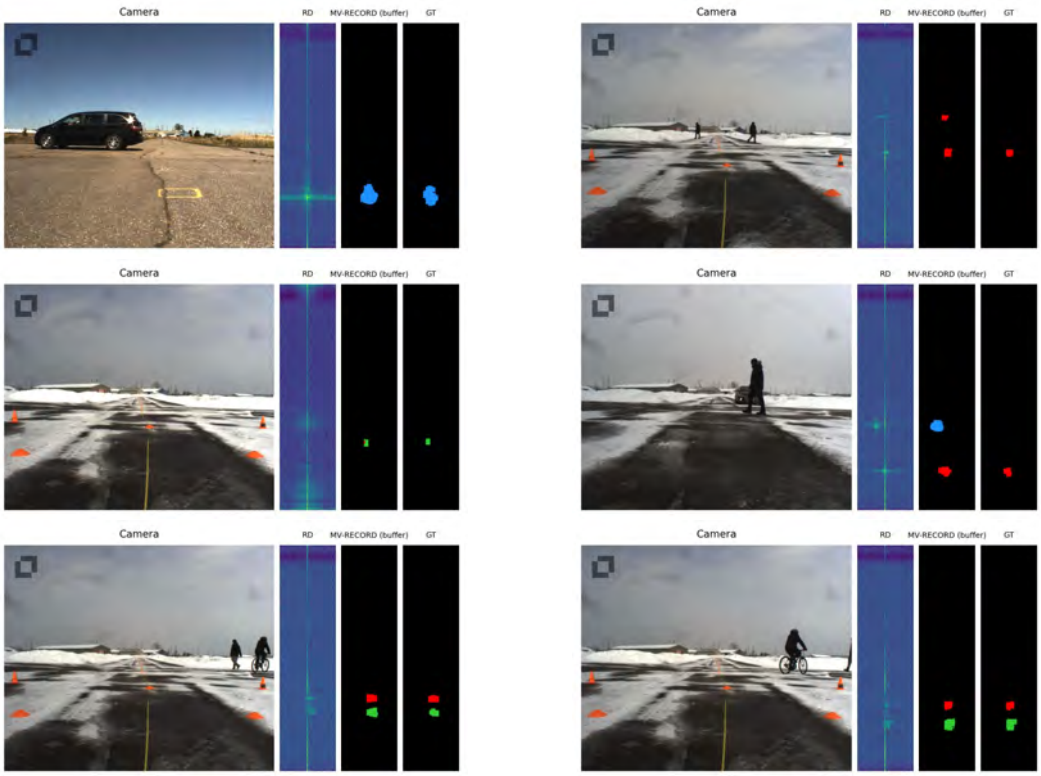


Figure B.12: Qualitative results for **MV-RECORD (buffer)** on CARRADA dataset (RD view). From left to right: camera image, RD spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

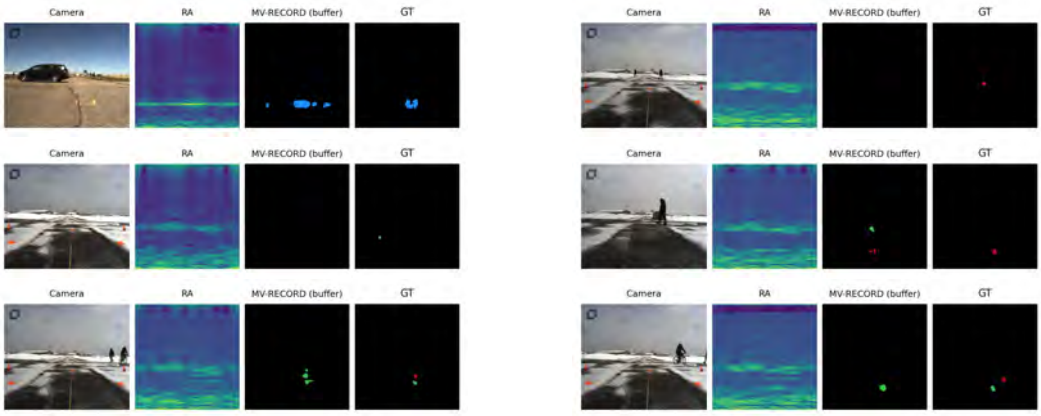


Figure B.13: Qualitative results for **MV-RECORD (buffer)** on CARRADA dataset (RA view). From left to right: camera image, RA spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

B.3.3 TMVA-Net

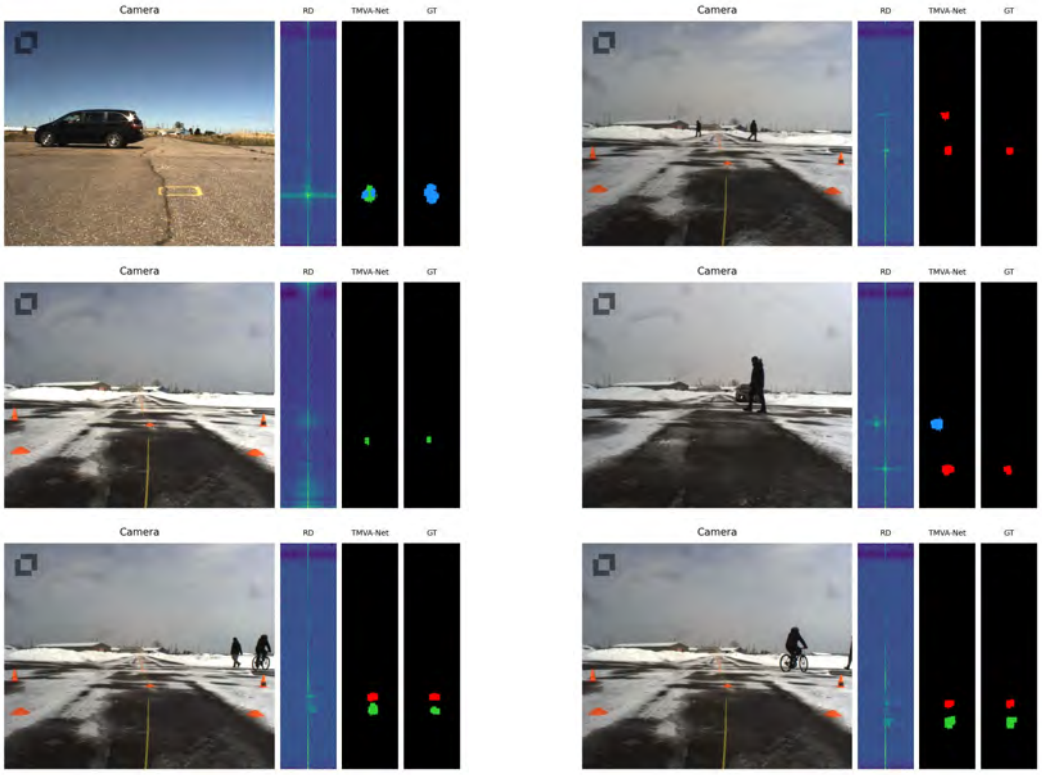


Figure B.14: Qualitative results for **TMVA-Net** on CARRADA dataset (RD view). From left to right: camera image, RD spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

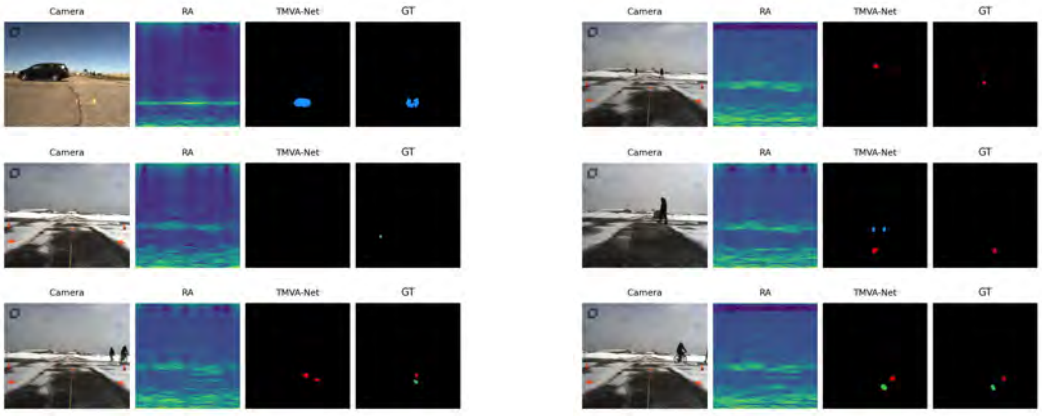


Figure B.15: Qualitative results for **TMVA-Net** on CARRADA dataset (RA view). From left to right: camera image, RA spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

B.3.4 MV-Net

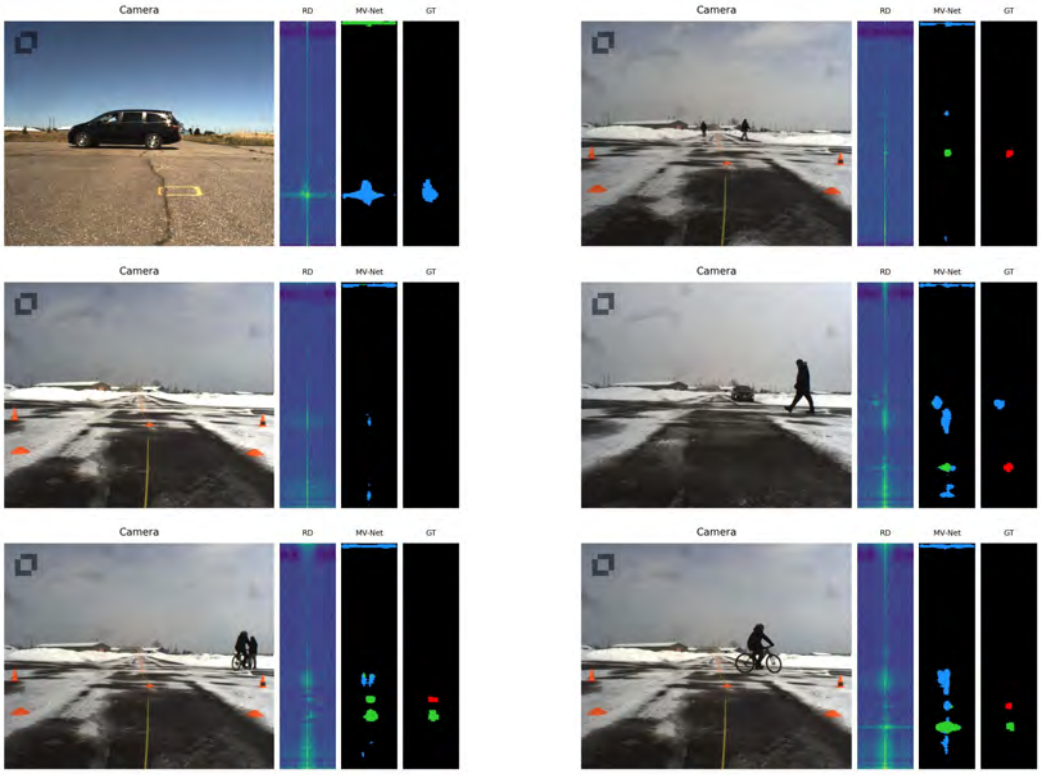


Figure B.16: Qualitative results for **MV-Net** on CARRADA dataset (RD view). From left to right: camera image, RD spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

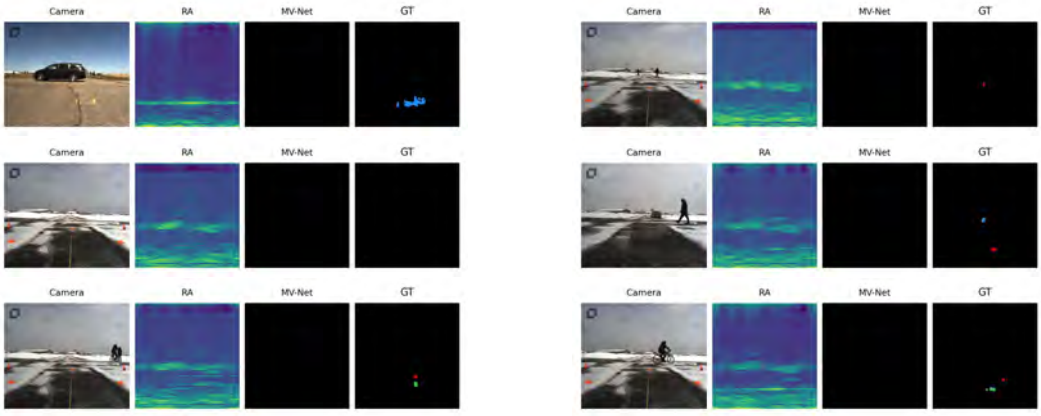


Figure B.17: Qualitative results for **MV-Net** on CARRADA dataset (RA view). From left to right: camera image, RA spectrum, predicted mask and ground truth mask. Legend: pedestrians, bicyclists, cars.

Bibliography

[Akita 2019] Tokihiko Akita and Seiichi Mita. Object Tracking and Classification Using Millimeter-Wave Radar Based on LSTM. In 2019 IEEE Intelligent Transportation Systems Conference (ITSC), pages 1110–1115, 2019. (Cited in pages 3, 33, 62, and 74.)

[Angelov 2018] Aleksandar Angelov, Andrew Robertson, Roderick Murray-Smith and Francesco Fioranelli. Practical classification of different moving targets using automotive radar and deep neural networks. IET Radar, Sonar & Navigation, vol. 12, no. 10, pages 1082–1089, 2018. (Cited in page 62.)

[Arnab 2021] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić and Cordelia Schmid. ViViT: A Video Vision Transformer. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pages 6816–6826, 2021. (Cited in page 114.)

[Balestriero 2023] Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tianet al. A cookbook of self-supervised learning. arXiv preprint arXiv:2304.12210, 2023. (Cited in pages 117 and 119.)

[Ballas 2016] Nicolas Ballas, Li Yao, Chris Pal and Aaron C. Courville. Delving Deeper into Convolutional Networks for Learning Video Representations. In 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings, 2016. (Cited in page 107.)

[Bao 2022] Hangbo Bao, Li Dong, Songhao Piao and Furu Wei. BEiT: BERT Pre-Training of Image Transformers. In International Conference on Learning Representations, 2022. (Cited in pages 121 and 124.)

[Bar 2022] Amir Bar, Xin Wang, Vadim Kantorov, Colorado J Reed, Roei Herzig, Gal Chechik, Anna Rohrbach, Trevor Darrell and Amir Globerson. DETReg: Unsupervised Pretraining with Region Priors for Object Detection. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 14585–14595, 2022. (Cited in pages 123 and 124.)

[Bardes 2022] Adrien Bardes, Jean Ponce and Yann LeCun. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. In International Conference on Learning Representations, 2022. (Cited in pages 120, 121, and 126.)

[Barnes 2020] Dan Barnes, Matthew Gadd, Paul Murcutt, Paul Newman and Ingmar Posner. The Oxford Radar RobotCar Dataset: A Radar Extension to the Oxford RobotCar Dataset. In 2020 IEEE International Conference on Robotics and Automation (ICRA), pages 6433–6438, 2020. (Cited in page 58.)

[Bay 2006] Herbert Bay, Tinne Tuytelaars and Luc Van Gool. SURF: Speeded Up Robust Features. In Aleš Leonardis, Horst Bischof and Axel Pinz, editors, Computer Vision – ECCV 2006, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. (Cited in page 61.)

[Bertasius 2018] Gedas Bertasius, Lorenzo Torresani and Jianbo Shi. Object Detection in Video with Spatiotemporal Sampling Networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu and Yair Weiss, editors, Computer Vision – ECCV 2018, pages 342–357, Cham, 2018. Springer International Publishing. (Cited in page 95.)

[Bertozzi 2004] M. Bertozzi, A. Broggi, A. Fascioli, A. Tibaldi, R. Chapuis and F. Chausse. Pedestrian localization and tracking system with Kalman filtering. In IEEE Intelligent Vehicles Symposium, 2004, pages 584–589, 2004. (Cited in page 25.)

[Blake 1988] Stephen Blake. OS-CFAR theory for multiple targets and nonuniform clutter. IEEE transactions on aerospace and electronic systems, vol. 24, no. 6, pages 785–790, 1988. (Cited in pages 4, 6, 25, 28, and 61.)

[Boot 2023] Thomas Boot, Nicolas Cazin, Willem Sanberg and Joaquin Vanschoren. Efficient-DASH: Automated Radar Neural Network Design Across Tasks and Datasets. In 2023 IEEE Intelligent Vehicles Symposium (IV), pages 1–7, 2023. (Cited in page 138.)

[Brodeski 2019] Daniel Brodeski, Igal Bilik and Raja Giryes. Deep Radar Detector. In 2019 IEEE Radar Conference (RadarConf), pages 1–6, 2019. (Cited in pages 64, 65, 70, and 74.)

[Bromley 1993] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger and Roopak Shah. Signature Verification using a "Siamese" Time Delay Neural Network. In J. Cowan, G. Tesauro and J. Alspector, editors, Advances in Neural Information Processing Systems, volume 6. Morgan-Kaufmann, 1993. (Cited in page 117.)

[Brookhuis 2001] Karel A. Brookhuis, Dick de Waard and Wiel H. Janssen. Behavioural impacts of Advanced Driver Assistance Systems–an overview.

European Journal of Transport and Infrastructure Research, vol. 1, no. 3, Jun. 2001. (Cited in page 18.)

[Brown 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever and Dario Amodei. Language Models are Few-Shot Learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. (Cited in pages 116 and 121.)

[Caesar 2019] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. arXiv preprint arXiv:1903.11027, 2019. (Cited in pages 1, 12, 43, and 53.)

[Cai 2019] Xiuzhang Cai and Kamal Sarabandi. A Machine Learning Based 77 GHz Radar Target Classification for Autonomous Vehicles. In 2019 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting, pages 371–372, 2019. (Cited in page 62.)

[Carion 2020a] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox and Jan-Michael Frahm, editors, Computer Vision – ECCV 2020, pages 213–229, Cham, 2020. Springer International Publishing. (Cited in pages 48 and 123.)

[Carion 2020b] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox and Jan-Michael Frahm, editors, Computer Vision – ECCV 2020, pages 213–229, Cham, 2020. Springer International Publishing. (Cited in page 123.)

[Caron 2018] Mathilde Caron, Piotr Bojanowski, Armand Joulin and Matthijs Douze. Deep Clustering for Unsupervised Learning of Visual Features. In

Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu and Yair Weiss, editors, Computer Vision – ECCV 2018, pages 139–156, Cham, 2018. Springer International Publishing. (Cited in page 117.)

[Caron 2020] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski and Armand Joulin. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 9912–9924. Curran Associates, Inc., 2020. (Cited in page 120.)

[Caron 2021] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jegou, Julien Mairal, Piotr Bojanowski and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pages 9630–9640, 2021. (Cited in pages 48, 90, 113, 119, and 120.)

[Charles 2017] R. Qi Charles, Hao Su, Mo Kaichun and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 77–85, 2017. (Cited in page 60.)

[Chen 2006] V.C. Chen, F. Li, S.-S. Ho and H. Wechsler. Micro-Doppler effect in radar: phenomenon, model, and simulation study. IEEE Transactions on Aerospace and Electronic Systems, vol. 42, no. 1, pages 2–21, 2006. (Cited in page 62.)

[Chen 2015] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy and Alan L. Yuille. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. In Yoshua Bengio and Yann LeCun, editors, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. (Cited in page 51.)

[Chen 2017] Liang-Chieh Chen, George Papandreou, Florian Schroff and Hartwig Adam. Rethinking Atrous Convolution for Semantic Image Segmentation. ArXiv, vol. abs/1706.05587, 2017. (Cited in page 52.)

[Chen 2018a] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy and Alan L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, pages 834–848, 2018. (Cited in pages 51 and 66.)

[Chen 2018b] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff and Hartwig Adam. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu and Yair Weiss, editors, Computer Vision – ECCV 2018, pages 833–851, Cham, 2018. Springer International Publishing. (Cited in pages 52 and 95.)

[Chen 2020a] Ting Chen, Simon Kornblith, Mohammad Norouzi and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In Hal Daumé III and Aarti Singh, editors, Proceedings of the 37th International Conference on Machine Learning, volume 119 of Proceedings of Machine Learning Research, pages 1597–1607. PMLR, 13–18 Jul 2020. (Cited in pages 90, 117, 118, 124, and 126.)

[Chen 2020b] Yihong Chen, Yue Cao, Han Hu and Liwei Wang. Memory Enhanced Global-Local Aggregation for Video Object Detection. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 10334–10343, 2020. (Cited in page 95.)

[Chen 2021] Xinlei Chen and Kaiming He. Exploring Simple Siamese Representation Learning. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 15745–15753, 2021. (Cited in page 120.)

[Cho 2014] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. (Cited in page 39.)

[Chollet 2017] François Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1800–1807, 2017. (Cited in page 52.)

[Cordts 2016] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3213–3223, 2016. (Cited in page 49.)

[Dai 2021] Zhigang Dai, Bolun Cai, Yugeng Lin and Junying Chen. UP-DETR: Unsupervised Pre-training for Object Detection with Transformers. In

2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 1601–1610, 2021. (Cited in page 123.)

[Dalbah 2023] Yahia Dalbah, Jean Lahoud and Hisham Cholakkal. RadarFormer: Lightweight and Accurate Real-Time Radar Object Detection Model. In Image Analysis: 23rd Scandinavian Conference, SCIA 2023, Sirkka, Finland, April 18–21, 2023, Proceedings, Part I, pages 341–358. Springer, 2023. (Cited in page 67.)

[Danzer 2019] Andreas Danzer, Thomas Griebel, Martin Bach and Klaus Dietmayer. 2D Car Detection in Radar Data with PointNets. In 2019 IEEE Intelligent Transportation Systems Conference (ITSC), pages 61–66, 2019. (Cited in page 60.)

[Decourt 2022a] Colin Decourt, Rufin VanRullen, Didier Salle and Thomas Oberlin. DAROD: A Deep Automotive Radar Object Detector on Range-Doppler maps. In 2022 IEEE Intelligent Vehicles Symposium (IV), pages 112–118, 2022. (Cited in pages 75 and 96.)

[Decourt 2022b] Colin Decourt, Rufin VanRullen, Didier Salle and Thomas Oberlin. A recurrent CNN for online object detection on raw radar frames. arXiv preprint arXiv:2212.11172, 2022. (Cited in page 95.)

[Devlin 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. (Cited in pages 116 and 121.)

[Ding 2022] Fangqiang Ding, Zhijun Pan, Yimin Deng, Jianning Deng and Chris Xiaoxuan Lu. Self-Supervised Scene Flow Estimation With 4-D Automotive Radar. IEEE Robotics and Automation Letters, pages 1–8, 2022. (Cited in page 60.)

[Ding 2023] Fangqiang Ding, Andras Palffy, Dariu M. Gavrila and Chris Xiaoxuan Lu. Hidden Gems: 4D Radar Scene Flow Learning Using Cross-Modal Supervision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 9340–9349, June 2023. (Cited in page 60.)

[Dong 2020] Xu Dong, Pengluo Wang, Pengyue Zhang and Langechuan Liu. Probabilistic Oriented Object Detection in Automotive Radar. In 2020

IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 458–467, 2020. (Cited in pages 67 and 70.)

[Dosovitskiy 2021] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021. (Cited in pages 113 and 121.)

[Dreher 2020] Maria Dreher, Emeç Erçelik, Timo Bänziger and Alois Knoll. Radar-based 2D Car Detection Using Deep Neural Networks. In 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), pages 1–8, 2020. (Cited in page 60.)

[Dubey 2020] Anand Dubey, Jonas Fuchs, Maximilian Lübke, Robert Weigel and Fabian Lurz. Generative Adversial Network based Extended Target Detection for Automotive MIMO Radar. In 2020 IEEE International Radar Conference (RADAR), pages 220–225, 2020. (Cited in pages 68 and 69.)

[Duke 2021] Brendan Duke, Abdalla Ahmed, Christian Wolf, Parham Aarabi and Graham W. Taylor. SSTVOS: Sparse Spatiotemporal Transformers for Video Object Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 5912–5921, June 2021. (Cited in page 95.)

[Ericsson 2021] Linus Ericsson, Henry Gouk and Timothy M. Hospedales. How Well Do Self-Supervised Models Transfer? In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 5410–5419, 2021. (Cited in page 122.)

[Ester 1996] Martin Ester, Hans-Peter Kriegel, Jörg Sander and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96, page 226–231. AAAI Press, 1996. (Cited in page 25.)

[Ettinger 2021] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhouet al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 9710–9719, 2021. (Cited in pages 1, 12, and 43.)

[Everingham 2012] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012), 2012. (Cited in pages 43 and 49.)

[Fang 2022] Shihong Fang, Haoran Zhu, Devansh Bisla, Anna Choromanska, Satish Ravindran, Dongyin Ren and Ryan Wu. ERASE-Net: Efficient Segmentation Networks for Automotive Radar Signals. arXiv preprint arXiv:2209.12940, 2022. (Cited in pages 64 and 65.)

[Fare Garnot 2021] Vivien Sainte Fare Garnot and Loic Landrieu. Panoptic Segmentation of Satellite Image Time Series with Convolutional Temporal Attention Networks. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pages 4852–4861, 2021. (Cited in pages 7, 95, 96, 104, 114, 143, and 150.)

[Fatseas 2019] Konstantinos Fatseas and Marco J.G. Bekooij. Neural Network Based Multiple Object Tracking for Automotive FMCW Radar. In 2019 International Radar Conference (RADAR), pages 1–5, 2019. (Cited in pages 33, 64, and 68.)

[Fatseas 2022] Konstantinos Fatseas and Marco J.G. Bekooij. Weakly Supervised Semantic Segmentation for Range-Doppler Maps. In 2021 18th European Radar Conference (EuRAD), pages 70–73, 2022. (Cited in page 68.)

[Fel 2022] Thomas Fel, Lucas Hervier, David Vigouroux, Antonin Poche, Justin Plakoo, Remi Cadene, Mathieu Chalvidal, Julien Colin, Thibaut Boissin, Louis Bethune, Agustin Picard, Claire Nicodeme, Laurent Gardes, Gregory Flandin and Thomas Serre. Xplique: A Deep Learning Explainability Toolbox. Workshop on Explainable Artificial Intelligence for Computer Vision (CVPR), 2022. (Cited in page 140.)

[Feng 2019] Zhaofei Feng, Shuo Zhang, Martin Kunert and Werner Wiesbeck. Point Cloud Segmentation with a High-Resolution Automotive Radar. In AmE 2019 - Automotive meets Electronics; 10th GMM-Symposium, pages 1–5, 2019. (Cited in page 60.)

[Fent 2023] Felix Fent, Philipp Bauerschmidt and Markus Lienkamp. RadarGNN: Transformation Invariant Graph Neural Network for Radar-based Perception. arXiv preprint arXiv:2304.06547, 2023. (Cited in page 60.)

[Franceschi 2022] Roberto Franceschi and Dmytro Rachkov. Deep Learning-Based Radar Detector for Complex Automotive Scenarios. In 2022 IEEE Intelligent Vehicles Symposium (IV), pages 303–308, 2022. (Cited in pages 64, 65, and 96.)

[Galvani 2019] Marco Galvani. History and future of driver assistance. IEEE Instrumentation and Measurement Magazine, vol. 22, no. 1, pages 11–16, 2019. (Cited in page 17.)

[Gao 2019a] Teng Gao, Zhichao Lai, Zengyang Mei and Qisong Wu. Hybrid SVM-CNN Classification Technique for Moving Targets in Automotive FMCW Radar System. In 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP), pages 1–6, 2019. (Cited in pages 62 and 63.)

[Gao 2019b] Xiangyu Gao, Guanbin Xing, Sumit Roy and Hui Liu. Experiments with mmWave Automotive Radar Test-bed. In 2019 53rd Asilomar Conference on Signals, Systems, and Computers, pages 1–6, 2019. (Cited in pages 33, 52, 62, and 64.)

[Gao 2021] Xiangyu Gao, Guanbin Xing, Sumit Roy and Hui Liu. RAMP-CNN: A Novel Neural Network for Enhanced Automotive Radar Object Recognition. IEEE Sensors Journal, vol. 21, no. 4, pages 5119–5132, 2021. (Cited in pages 3, 4, 64, 65, 66, 67, 70, 71, and 96.)

[Gavrila 2001] D.M. Gavrila. Sensor-based pedestrian protection. IEEE Intelligent Systems, vol. 16, no. 6, pages 77–81, 2001. (Cited in page 25.)

[Giroux 2023] James Giroux, Martin Bouchard and Robert Laganiere. T-FFTRadNet: Object Detection with Swin Vision Transformers from Raw ADC Radar Signals. arXiv preprint arXiv:2303.16940, 2023. (Cited in pages 3, 64, 69, 113, 114, and 138.)

[Girshick 2014] Ross Girshick, Jeff Donahue, Trevor Darrell and Jitendra Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In 2014 IEEE Conference on Computer Vision and Pattern Recognition, pages 580–587, 2014. (Cited in page 45.)

[Girshick 2015] Ross Girshick. Fast R-CNN. In 2015 IEEE International Conference on Computer Vision (ICCV), pages 1440–1448, 2015. (Cited in pages 45 and 76.)

[Goodfellow 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville and Yoshua Bengio. Generative Adversarial Nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems, volume 27. Curran Associates, Inc., 2014. (Cited in page 69.)

[Goodfellow 2016] Ian Goodfellow, Yoshua Bengio and Aaron Courville. Deep learning. MIT Press, 2016. http://www.deeplearningbook.org. (Cited in pages 36, 38, and 117.)

[Goyal 2017] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. arXiv preprint arXiv:1706.02677, 2017. (Cited in page 131.)

[Goyal 2021] Priya Goyal, Mathilde Caron, Benjamin Lefaudeux, Min Xu, Pengchao Wang, Vivek Pai, Mannat Singh, Vitaliy Liptchinsky, Ishan Misra, Armand Joulinet al. Self-supervised pretraining of visual features in the wild. arXiv preprint arXiv:2103.01988, 2021. (Cited in page 116.)

[Goyal 2022] Priya Goyal, Quentin Duval, Isaac Seessel, Mathilde Caron, Ishan Misra, Levent Sagun, Armand Joulin and Piotr Bojanowski. Vision models are more robust and fair when pretrained on uncurated images without supervision. arXiv preprint arXiv:2202.08360, 2022. (Cited in page 117.)

[Grill 2020] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos and Michal Valko. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 21271–21284. Curran Associates, Inc., 2020. (Cited in pages 90, 117, 119, 123, 124, 127, 129, 132, and 138.)

[Gu 2022] Huanyu Gu. The Importance of Imaging Radar, February 2022. https://www.nxp.com/company/blog/the-importance-of-imaging-radar:BL-THE-IMPORTANCE-OF-IMAGING-RADAR. (Cited in pages 11, 12, and 13.)

[Guo 2022] Zuyuan Guo, Haoran Wang, Wei Yi and Jiahao Zhang. Efficient Radar Deep Temporal Detection in Urban Traffic Scenes. In 2022 IEEE Intelligent Vehicles Symposium (IV), pages 498–503, 2022. (Cited in pages 68 and 69.)

[Hadsell 2006] R. Hadsell, S. Chopra and Y. LeCun. Dimensionality Reduction by Learning an Invariant Mapping. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), volume 2, pages 1735–1742, 2006. (Cited in page 117.)

[Hameed 2022] Syed Waqar Hameed. Peaks Detector Algorithm after CFAR for Multiple Targets Detection. EAI Endorsed Transactions on AI and Robotics, vol. 1, no. 1, 7 2022. (Cited in page 29.)

[Hamon 2020] R Hamon, H Junklewitz and JI Sanchez Martin. Robustness and Explainability of Artificial Intelligence. no. KJ-NA-30040-EN-N (online), 2020. (Cited in page 140.)

[Hasch 2012] Jürgen Hasch, Eray Topak, Raik Schnabel, Thomas Zwick, Robert Weigel and Christian Waldschmidt. Millimeter-Wave Technology for Automotive Radar Sensors in the 77 GHz Frequency Band. IEEE Transactions on Microwave Theory and Techniques, vol. 60, no. 3, pages 845–860, 2012. (Cited in page 20.)

[He 2014] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In David Fleet, Tomas Pajdla, Bernt Schiele and Tinne Tuytelaars, editors, Computer Vision – ECCV 2014, pages 346–361, Cham, 2014. Springer International Publishing. (Cited in page 51.)

[He 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Deep Residual Learning for Image Recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2016. (Cited in pages 6, 9, 41, 42, 51, and 130.)

[He 2017] Kaiming He, Georgia Gkioxari, Piotr Dollár and Ross Girshick. Mask R-CNN. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 2980–2988, 2017. (Cited in pages 9, 45, 46, 74, 88, 115, 123, 127, and 129.)

[He 2020] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie and Ross Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 9726–9735, 2020. (Cited in pages 90, 120, 126, and 138.)

[He 2022] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár and Ross Girshick. Masked Autoencoders Are Scalable Vision Learners. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 15979–15988, 2022. (Cited in pages 117, 121, 122, 124, and 126.)

[Hendrycks* 2020] Dan Hendrycks*, Norman Mu*, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer and Balaji Lakshminarayanan. AugMix: A Simple Method to Improve Robustness and Uncertainty under Data Shift. In International Conference on Learning Representations, 2020. (Cited in page 70.)

[Heuel 2011] Steffen Heuel and Hermann Rohling. Two-stage pedestrian classification in automotive radar systems. In 2011 12th International Radar Symposium (IRS), pages 477–484, 2011. (Cited in page 61.)

[Hochreiter 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. Neural Computation, vol. 9, no. 8, pages 1735–1780, 11 1997. (Cited in page 39.)

[Hotelling 1992] Harold Hotelling. Relations between two sets of variates. In Breakthroughs in statistics: methodology and distribution, pages 162–190. Springer, 1992. (Cited in page 120.)

[Howard 2017] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. CoRR, vol. abs/1704.04861, 2017. (Cited in pages 41 and 52.)

[Howard 2019] Andrew Howard, Mark Sandler, Bo Chen, Weijun Wang, Liang-Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, Yukun Zhu, Ruoming Pang, Hartwig Adam and Quoc Le. Searching for MobileNetV3. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 1314–1324, 2019. (Cited in pages 41 and 42.)

[Hu 2018] Jie Hu, Li Shen and Gang Sun. Squeeze-and-Excitation Networks. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7132–7141, 2018. (Cited in page 42.)

[Huber 1964] Peter J. Huber. Robust Estimation of a Location Parameter. The Annals of Mathematical Statistics, vol. 35, no. 1, pages 73 – 101, 1964. (Cited in page 78.)

[Jiang 2023] Tiezhen Jiang, Long Zhuang, Qi An, Jianhua Wang, Kai Xiao and Anqi Wang. T-RODNet: Transformer for Vehicular Millimeter-Wave Radar Object Detection. IEEE Transactions on Instrumentation and Measurement, vol. 72, pages 1–12, 2023. (Cited in pages 7, 104, 113, and 151.)

[Jones 2018] Michael J. Jones, Alexander Broad and Teng-Yok Lee. Recurrent Multi-frame Single Shot Detector for Video Object Detection. In British Machine Vision Conference (BMVC), September 2018. (Cited in page 97.)

[Ju 2021] Bo Ju, Wei Yang, Jinrang Jia, Xiaoqing Ye, Qu Chen, Xiao Tan, Hao Sun, Yifeng Shi and Errui Ding. DANet: Dimension Apart Network for Radar Object Detection. In Proceedings of the 2021 International Conference on

Multimedia Retrieval, ICMR '21, page 533–539, New York, NY, USA, 2021. Association for Computing Machinery. (Cited in pages 4, 6, 7, 67, 94, 96, 103, 104, 112, 113, and 149.)

[Kalman 1960] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. Journal of Basic Engineering, vol. 82, no. 1, pages 35–45, 03 1960. (Cited in pages 25, 32, and 133.)

[Kaul 2020] Prannay Kaul, Daniele de Martini, Matthew Gadd and Paul Newman. RSS-Net: Weakly-Supervised Multi-Class Semantic Segmentation with FMCW Radar. In 2020 IEEE Intelligent Vehicles Symposium (IV), pages 431–436, 2020. (Cited in pages 64, 67, and 96.)

[Khalid 2019] Habib-ur-Rehman Khalid, Sofie Pollin, Maxim Rykunov, André Bourdoux and Hichem Sahli. Convolutional Long Short-Term Memory Networks for Doppler-Radar Based Target Classification. In 2019 IEEE Radar Conference (RadarConf), pages 1–6, 2019. (Cited in pages 3 and 62.)

[Kiefer 1952] J. Kiefer and J. Wolfowitz. Stochastic Estimation of the Maximum of a Regression Function. The Annals of Mathematical Statistics, vol. 23, no. 3, pages 462 – 466, 1952. (Cited in page 38.)

[Kim 2018] Sangtae Kim, Seunghwan Lee, Seungho Doo and Byonghyo Shim. Moving Target Classification in Automotive Radar Systems Using Convolutional Recurrent Neural Networks. In 2018 26th European Signal Processing Conference (EUSIPCO), pages 1482–1486, 2018. (Cited in page 62.)

[Krähenbühl 2011] Philipp Krähenbühl and Vladlen Koltun. Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems, volume 24. Curran Associates, Inc., 2011. (Cited in page 52.)

[Kraus 2020] Florian Kraus, Nicolas Scheiner, Werner Ritter and Klaus Dietmayer. Using Machine Learning to Detect Ghost Images in Automotive Radar. In 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), pages 1–7, 2020. (Cited in page 60.)

[Krizhevsky 2012] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C.J. Burges, L. Bottou and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems, volume 25. Curran Associates, Inc., 2012. (Cited in pages 40, 41, 45, and 50.)

[Lang 2019] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang and Oscar Beijbom. PointPillars: Fast Encoders for Object Detection From Point Clouds. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 12689–12697, 2019. (Cited in page 60.)

[Lecun 1989] Yann Lecun. Generalization and network design strategies. Elsevier, 1989. (Cited in pages 39 and 40.)

[Lee 2017] Seongwook Lee, Young-Jun Yoon, Jae-Eun Lee and Seong-Cheol Kim. Human–vehicle classification using feature-based SVM in 77-GHz automotive FMCW radar. IET Radar, Sonar & Navigation, vol. 11, no. 10, pages 1589–1596, 2017. (Cited in page 61.)

[Lee 2019] Dajung Lee, Colman Cheung and Dan Pritsker. Radar-based Object Classification Using An Artificial Neural Network. In 2019 IEEE National Aerospace and Electronics Conference (NAECON), pages 305–310, 2019. (Cited in page 62.)

[Li 2018] Xiaobo Li, Haohua Zhao and Liqing Zhang. Recurrent RetinaNet: A Video Object Detection Model based on Focal Loss. In International conference on neural information processing, pages 499–508. Springer, 2018. (Cited in page 95.)

[Li 2021] Zhaowen Li, Zhiyang Chen, Fan Yang, Wei Li, Yousong Zhu, Chaoyang Zhao, Rui Deng, Liwei Wu, Rui Zhao, Ming Tang and Jinqiao Wang. MST: Masked Self-Supervised Transformer for Visual Representation. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang and J. Wortman Vaughan, editors, Advances in Neural Information Processing Systems, volume 34, pages 13165–13176. Curran Associates, Inc., 2021. (Cited in page 122.)

[Li 2022] Peizhao Li, Pu Wang, Karl Berntorp and Hongfu Liu. Exploiting Temporal Relations on Radar Perception for Autonomous Driving. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 17050–17059, 2022. (Cited in pages 6, 64, 94, and 96.)

[Lin 2014] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In ECCV, 2014. (Cited in pages 43, 44, and 49.)

[Lin 2017a] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan and Serge Belongie. Feature Pyramid Networks for Object Detection. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 936–944, 2017. (Cited in pages 45, 47, and 51.)

[Lin 2017b] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He and Piotr Dollár. Focal Loss for Dense Object Detection. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 2999–3007, 2017. (Cited in page 45.)

[Liu 2016] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu and Alexander C. Berg. SSD: Single Shot MultiBox Detector. In Bastian Leibe, Jiri Matas, Nicu Sebe and Max Welling, editors, Computer Vision – ECCV 2016, pages 21–37, Cham, 2016. Springer International Publishing. (Cited in pages 45, 48, and 51.)

[Liu 2021] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pages 9992–10002, 2021. (Cited in page 48.)

[Liu 2022] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell and Saining Xie. A ConvNet for the 2020s. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 11966–11976, 2022. (Cited in page 48.)

[Lloyd 1982] S. Lloyd. Least squares quantization in PCM. IEEE Transactions on Information Theory, vol. 28, no. 2, pages 129–137, 1982. (Cited in page 25.)

[Long 2015] Jonathan Long, Evan Shelhamer and Trevor Darrell. Fully convolutional networks for semantic segmentation. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3431–3440, 2015. (Cited in page 50.)

[Madani 2022] Sohrab Madani, Jayden Guan, Waleed Ahmed, Saurabh Gupta and Haitham Hassanieh. Radatron: Accurate Detection Using Multi-resolution Cascaded MIMO Radar. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella and Tal Hassner, editors, Computer Vision – ECCV 2022, pages 160–178, Cham, 2022. Springer Nature Switzerland. (Cited in pages 56 and 68.)

[Major 2019] Bence Major, Daniel Fontijne, Amin Ansari, Ravi Teja Sukhavasi, Radhika Gowaikar, Michael Hamilton, Sean Lee, Slawomir Grzechnik and Sundar Subramanian. Vehicle Detection With Automotive Radar Using Deep Learning on Range-Azimuth-Doppler Tensors. In 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), pages 924–932, 2019. (Cited in pages 6, 64, 65, 66, 67, 70, 91, 94, and 97.)

[Meyer 2019] Michael Meyer and Georg Kuschk. Automotive Radar Dataset for Deep Learning Based 3D Object Detection. In 2019 16th European Radar Conference (EuRAD), pages 129–132, 2019. (Cited in pages 53 and 54.)

[Meyer 2021] Michael Meyer, Georg Kuschk and Sven Tomforde. Graph Convolutional Networks for 3D Object Detection on Radar Data. In 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), pages 3053–3062, 2021. (Cited in pages 68 and 74.)

[Mitchell 1997] Tom Mitchell. Machine learning. McGraw Hill, 1997. `http://www.cs.cmu.edu/afs/cs.cmu.edu/user/mitchell/ftp/mlbook.html`. (Cited in page 36.)

[Nagpal 2022] Raj Kumar Nagpal and Edo Cohen. Automotive electronics revolution requires faster, smarter interfaces, May 2022. https://www.embedded.com/automotive-electronics-revolution-requires-faster-smarter-interfaces/, Last accessed on 2023-07-23. (Cited in page 17.)

[Naseer 2021] Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan and Ming-Hsuan Yang. Intriguing Properties of Vision Transformers. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang and J. Wortman Vaughan, editors, Advances in Neural Information Processing Systems, volume 34, pages 23296–23308. Curran Associates, Inc., 2021. (Cited in page 113.)

[Newell 2016] Alejandro Newell, Kaiyu Yang and Jia Deng. Stacked Hourglass Networks for Human Pose Estimation. In Bastian Leibe, Jiri Matas, Nicu Sebe and Max Welling, editors, Computer Vision – ECCV 2016, pages 483–499, Cham, 2016. Springer International Publishing. (Cited in page 67.)

[Niederlöhner 2022] Daniel Niederlöhner, Michael Ulrich, Sascha Braun, Daniel Köhler, Florian Faion, Claudius Gläser, André Treptow and Holger Blume. Self-Supervised Velocity Estimation for Automotive Radar Object Detection Networks. In 2022 IEEE Intelligent Vehicles Symposium (IV), pages 352–359, 2022. (Cited in pages 53, 60, 61, and 96.)

[Noroozi 2016] Mehdi Noroozi and Paolo Favaro. Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles. In Bastian Leibe, Jiri Matas, Nicu Sebe and Max Welling, editors, Computer Vision – ECCV 2016, pages 69–84, Cham, 2016. Springer International Publishing. (Cited in page 121.)

[Ouaknine 2021a] Arthur Ouaknine, Alasdair Newson, Patrick Pérez, Florence Tupin and Julien Rebut. Multi-View Radar Semantic Segmentation. In

2021 IEEE/CVF International Conference on Computer Vision (ICCV), pages 15651–15660, 2021. (Cited in pages 3, 4, 6, 7, 64, 65, 66, 67, 70, 74, 91, 94, 96, 100, 108, and 109.)

[Ouaknine 2021b] Arthur Ouaknine, Alasdair Newson, Julien Rebut, Florence Tupin and Patrick Pérez. CARRADA Dataset: Camera and Automotive Radar with Range- Angle- Doppler Annotations. In 2020 25th International Conference on Pattern Recognition (ICPR), pages 5068–5075, 2021. (Cited in pages 3, 5, 7, 9, 33, 52, 54, 55, 74, 78, 84, 85, 95, 108, 115, and 130.)

[Paek 2022] Dong-Hee Paek, SEUNG-HYUN KONG and Kevin Tirta Wijaya. K-Radar: 4D Radar Object Detection for Autonomous Driving in Various Weather Conditions. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho and A. Oh, editors, Advances in Neural Information Processing Systems, volume 35, pages 3819–3829. Curran Associates, Inc., 2022. (Cited in pages 56, 59, 68, and 69.)

[Palffy 2020] Andras Palffy, Jiaao Dong, Julian F. P. Kooij and Dariu M. Gavrila. CNN Based Road User Detection Using the 3D Radar Cube. IEEE Robotics and Automation Letters, vol. 5, no. 2, pages 1263–1270, 2020. (Cited in pages 3, 62, 63, 139, and 140.)

[Palffy 2022] Andras Palffy, Ewoud Pool, Srimannarayana Baratam, Julian F. P. Kooij and Dariu M. Gavrila. Multi-Class Road User Detection With 3+1D Radar in the View-of-Delft Dataset. IEEE Robotics and Automation Letters, vol. 7, no. 2, pages 4961–4968, 2022. (Cited in pages 52, 53, 54, 60, and 138.)

[Patel 2019] Kanil Patel, Kilian Rambach, Tristan Visentin, Daniel Rusev, Michael Pfeiffer and Bin Yang. Deep Learning-based Object Classification on Automotive Radar Spectra. In 2019 IEEE Radar Conference (RadarConf), pages 1–6, 2019. (Cited in pages 62 and 63.)

[Patel 2022] Kanil Patel, William Beluch, Kilian Rambach, Michael Pfeiffer and Bin Yang. Improving Uncertainty of Deep Learning-based Object Classification on Radar Spectra using Label Smoothing. In 2022 IEEE Radar Conference (RadarConf22), pages 1–6, 2022. (Cited in page 62.)

[Pathak 2016] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell and Alexei A. Efros. Context Encoders: Feature Learning by Inpainting. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2536–2544, 2016. (Cited in pages 117 and 121.)

[Pathak 2017] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell and Bharath Hariharan. Learning Features by Watching Objects Move. In 2017

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 6024–6033, 2017. (Cited in pages 117 and 124.)

[Patole 2017] Sujeet Milind Patole, Murat Torlak, Dan Wang and Murtaza Ali. Automotive Radars: A Review of Signal Processing Techniques. IEEE Signal Processing Magazine, vol. 34, no. 2, pages 22–35, 2017. (Cited in pages 18 and 21.)

[Paulraj 1985] A. Paulraj, R. Roy and T. Kailath. Estimation Of Signal Parameters Via Rotational Invariance Techniques- Esprit. In Nineteeth Asilomar Conference on Circuits, Systems and Computers, 1985., pages 83–89, 1985. (Cited in pages 25 and 30.)

[Pfeuffer 2019] Andreas Pfeuffer, Karina Schulz and Klaus Dietmayer. Semantic Segmentation of Video Sequences with Convolutional LSTMs. In 2019 IEEE intelligent vehicles symposium (IV), pages 1441–1447. IEEE, 2019. (Cited in page 95.)

[Prophet 2018a] Robert Prophet, Marcel Hoffmann, Alicja Ossowska, Waqas Malik, Christian Sturm and Martin Vossiek. Image-Based Pedestrian Classification for 79 GHz Automotive Radar. In 2018 15th European Radar Conference (EuRAD), pages 75–78, 2018. (Cited in pages 61 and 62.)

[Prophet 2018b] Robert Prophet, Marcel Hoffmann, Alicja Ossowska, Waqas Malik, Christian Sturm and Martin Vossiek. Pedestrian Classification for 79 GHz Automotive Radar Systems. In 2018 IEEE Intelligent Vehicles Symposium (IV), pages 1265–1270, 2018. (Cited in pages 61 and 63.)

[Pérez 2018] Rodrigo Pérez, Falk Schubert, Ralph Rasshofer and Erwin Biebl. Single-Frame Vulnerable Road Users Classification with a 77 GHz FMCW Radar Sensor and a Convolutional Neural Network. In 2018 19th International Radar Symposium (IRS), pages 1–10, 2018. (Cited in page 62.)

[Qi 2017] Charles Ruizhongtai Qi, Li Yi, Hao Su and Leonidas J Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017. (Cited in page 60.)

[Rao 2018] Sandeep Rao. MIMO Radar, 2018. (Cited in pages 29, 30, and 31.)

[Rebut 2022] Julien Rebut, Arthur Ouaknine, Waqas Malik and Patrick Pérez. Raw High-Definition Radar for Multi-Task Learning. In Proceedings of

the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 17021–17030, June 2022. (Cited in pages 9, 10, 56, 57, 68, 69, 70, 89, 114, 115, 138, and 139.)

[Redmon 2016] Joseph Redmon, Santosh Divvala, Ross Girshick and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 779–788, 2016. (Cited in pages 45, 47, 48, 65, 68, and 81.)

[Redmon 2017] Joseph Redmon and Ali Farhadi. YOLO9000: Better, Faster, Stronger. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 6517–6525, 2017. (Cited in pages 47 and 48.)

[Ren 2017] Shaoqing Ren, Kaiming He, Ross Girshick and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pages 1137–1149, 2017. (Cited in pages 4, 14, 45, 46, 51, 73, 77, 78, 128, and 130.)

[Richards 2005] M.A. Richards. Fundamentals of radar signal processing. McGraw-Hill Education (India) Pvt Limited, 2005. (Cited in pages 19 and 28.)

[Robbins 1951] Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. The Annals of Mathematical Statistics, vol. 22, no. 3, pages 400 – 407, 1951. (Cited in page 38.)

[Rohling 1983] Hermann Rohling. Radar CFAR Thresholding in Clutter and Multiple Target Situations. IEEE Transactions on Aerospace and Electronic Systems, vol. AES-19, no. 4, pages 608–621, 1983. (Cited in pages 28, 126, and 127.)

[Rohling 2010] Hermann Rohling, Steffen Heuel and Henning Ritter. Pedestrian detection procedure integrated into an 24 GHz automotive radar. In 2010 IEEE Radar Conference, pages 1229–1232, 2010. (Cited in page 25.)

[Ronneberger 2015] Olaf Ronneberger, Philipp Fischer and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells and Alejandro F. Frangi, editors, Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, pages 234–241, Cham, 2015. Springer International Publishing. (Cited in pages 50, 51, 64, 69, and 100.)

[Rumelhart 1986] David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams. Learning representations by back-propagating errors. Nature, vol. 323, pages 533–536, 1986. (Cited in page 38.)

[Russakovsky 2015] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), vol. 115, no. 3, pages 211–252, 2015. (Cited in pages 10, 40, 90, and 116.)

[Saini 2023] Loveneet Saini, Axel Acosta and Gor Hakobyan. Graph Neural Networks for Object Type Classification Based on Automotive Radar Point Clouds and Spectra. In ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 1–5, 2023. (Cited in pages 62, 63, and 138.)

[Sandler 2018] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov and Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4510–4520, 2018. (Cited in pages 7, 41, 42, 95, 98, and 99.)

[Scheiner 2018] Nicolas Scheiner, Nils Appenrodt, Jürgen Dickmann and Bernhard Sick. Radar-based Feature Design and Multiclass Classification for Road User Recognition. In 2018 IEEE Intelligent Vehicles Symposium (IV), pages 779–786, 2018. (Cited in pages 2 and 59.)

[Scheiner 2019] Nicolas Scheiner, Nils Appenrodt, Jürgen Dickmann and Bernhard Sick. Radar-based Road User Classification and Novelty Detection with Recurrent Neural Network Ensembles. In 2019 IEEE Intelligent Vehicles Symposium (IV), pages 722–729, 2019. (Cited in page 59.)

[Scheiner 2020] Nicolas Scheiner, Ole Schumann, Florian Kraus, Nils Appenrodt, Jürgen Dickmann and Bernhard Sick. Off-the-shelf sensor vs. experimental radar - How much resolution is necessary in automotive radar classification? In 2020 IEEE 23rd International Conference on Information Fusion (FUSION), pages 1–8, 2020. (Cited in pages 2 and 59.)

[Schmidt 1986] R. Schmidt. Multiple emitter location and signal parameter estimation. IEEE Transactions on Antennas and Propagation, vol. 34, no. 3, pages 276–280, 1986. (Cited in pages 25, 30, and 67.)

[Schroff 2015] Florian Schroff, Dmitry Kalenichenko and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 815–823, 2015. (Cited in page 118.)

[Schumann 2018] Ole Schumann, Markus Hahn, Jürgen Dickmann and Christian Wöhler. Semantic Segmentation on Radar Point Clouds. In 2018 21st International Conference on Information Fusion (FUSION), pages 2179–2186, 2018. (Cited in pages 60 and 96.)

[Schumann 2021] Ole Schumann, Markus Hahn, Nicolas Scheiner, Fabio Weishaupt, Julius F. Tilly, Jürgen Dickmann and Christian Wöhler. RadarScenes: A Real-World Radar Point Cloud Data Set for Automotive Applications. In 2021 IEEE 24th International Conference on Information Fusion (FUSION), pages 1–8, 2021. (Cited in pages 53 and 54.)

[Sermanet 2018] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine and Google Brain. Time-Contrastive Networks: Self-Supervised Learning from Video. In 2018 IEEE International Conference on Robotics and Automation (ICRA), pages 1134–1141, 2018. (Cited in page 118.)

[Sheeny 2020] Marcel Sheeny, Andrew Wallace and Sen Wang. RADIO: Parameterized Generative Radar Data Augmentation for Small Datasets. Applied Sciences, vol. 10, no. 11, 2020. (Cited in page 70.)

[Sheeny 2021] Marcel Sheeny, Emanuele De Pellegrin, Saptarshi Mukherjee, Alireza Ahrabian, Sen Wang and Andrew Wallace. RADIATE: A Radar Dataset for Automotive Perception in Bad Weather. In 2021 IEEE International Conference on Robotics and Automation (ICRA), pages 1–7, 2021. (Cited in pages 58 and 59.)

[Shi 2015] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong and Wang-chun WOO. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 28. Curran Associates, Inc., 2015. (Cited in pages 7, 98, and 107.)

[Simonyan 2015] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Yoshua Bengio and Yann LeCun, editors, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. (Cited in pages 5, 40, 41, 45, 46, 51, and 76.)

[Svenningsson 2021] Peter Svenningsson, Francesco Fioranelli and Alexander Yarovoy. Radar-PointGNN: Graph Based Object Recognition for Unstructured Radar Point-cloud Data. In 2021 IEEE Radar Conference (RadarConf21), pages 1–6, 2021. (Cited in page 53.)

[Szegedy 2015] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich. Going deeper with convolutions. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1–9, 2015. (Cited in page 41.)

[Tatarchenko 2023] Maxim Tatarchenko and Kilian Rambach. Histogram-based Deep Learning for Automotive Radar. arXiv preprint arXiv:2303.02975, 2023. (Cited in pages 2 and 59.)

[Tian 2019] Zhi Tian, Chunhua Shen, Hao Chen and Tong He. FCOS: Fully Convolutional One-Stage Object Detection. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 9626–9635, 2019. (Cited in pages 45 and 48.)

[Tilly 2020] Julius F. Tilly, Stefan Haag, Ole Schumann, Fabio Weishaupt, Bharanidhar Duraisamy, Jürgen Dickmann and Martin Fritzsche. Detection and Tracking on Automotive Radar Data with Deep Learning. In 2020 IEEE 23rd International Conference on Information Fusion (FUSION), pages 1–7, 2020. (Cited in page 60.)

[Tong 2022] Zhan Tong, Yibing Song, Jue Wang and Limin Wang. VideoMAE: Masked Autoencoders are Data-Efficient Learners for Self-Supervised Video Pre-Training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho and A. Oh, editors, Advances in Neural Information Processing Systems, volume 35, pages 10078–10093. Curran Associates, Inc., 2022. (Cited in pages 123 and 126.)

[Tran 2018] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun and Manohar Paluri. A Closer Look at Spatiotemporal Convolutions for Action Recognition. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6450–6459, 2018. (Cited in page 67.)

[Ulrich 2021] Michael Ulrich, Claudius Gläser and Fabian Timm. DeepReflecs: Deep Learning for Automotive Object Classification with Radar Reflections. In 2021 IEEE Radar Conference (RadarConf21), pages 1–6, 2021. (Cited in pages 2, 59, 60, 85, 86, 87, and 89.)

[Ulrich 2022] Michael Ulrich, Sascha Braun, Daniel Köhler, Daniel Niederlöhner, Florian Faion, Claudius Gläser and Holger Blume. Improved Orientation Estimation and Detection with Hybrid Object Detection Networks for Automotive Radar. In 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), pages 111–117, 2022. (Cited in pages 60 and 61.)

[Urtasun 2012] R. Urtasun, P. Lenz and A. Geiger. Are we ready for autonomous driving? The KITTI vision benchmark suite. In 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3354–3361, Los Alamitos, CA, USA, jun 2012. IEEE Computer Society. (Cited in pages 1, 12, and 43.)

[Vaswani 2017a] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017. (Cited in page 37.)

[Vaswani 2017b] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser and Illia Polosukhin. Attention is All You Need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. (Cited in page 67.)

[Ventura 2019] Carles Ventura, Miriam Bellver, Andreu Girbau, Amaia Salvador, Ferran Marques and Xavier Giro-i Nieto. RVOS: End-To-End Recurrent Network for Video Object Segmentation. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 5272–5281, 2019. (Cited in pages 95 and 96.)

[Vincent 2008] Pascal Vincent, Hugo Larochelle, Yoshua Bengio and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In William W. Cohen, Andrew McCallum and Sam T. Roweis, editors, Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008, volume 307 of ACM International Conference Proceeding Series, pages 1096–1103. ACM, 2008. (Cited in pages 117 and 121.)

[Wang 2015] Xiaolong Wang and Abhinav Gupta. Unsupervised Learning of Visual Representations Using Videos. In 2015 IEEE International Conference on Computer Vision (ICCV), pages 2794–2802, 2015. (Cited in page 117.)

[Wang 2021a] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong and Lei Li. Dense Contrastive Learning for Self-Supervised Visual Pre-Training. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 3023–3032, 2021. (Cited in page 123.)

[Wang 2021b] Yizhou Wang, Zhongyu Jiang, Xiangyu Gao, Jenq-Neng Hwang, Guanbin Xing and Hui Liu. RODNet: Radar Object Detection using

Cross-Modal Supervision. In 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), pages 504–513, 2021. (Cited in pages 3, 4, 6, 67, 68, 70, 74, 91, 94, 96, 102, and 103.)

[Wang 2021c] Yizhou Wang, Gaoang Wang, Hung-Min Hsu, Hui Liu and Jenq-Neng Hwang. Rethinking of Radar's Role: A Camera-Radar Dataset and Systematic Annotator via Coordinate Alignment. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pages 2815–2824, June 2021. (Cited in pages 3, 7, 9, 55, 67, 74, 95, 102, 115, and 137.)

[Wei 2021] Fangyun Wei, Yue Gao, Zhirong Wu, Han Hu and Stephen Lin. Aligning Pretraining for Detection via Object-Level Contrastive Learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang and J. Wortman Vaughan, editors, Advances in Neural Information Processing Systems, volume 34, pages 22682–22694. Curran Associates, Inc., 2021. (Cited in pages 9, 116, 123, 124, 126, 128, 132, 133, and 138.)

[Weinberger 2009] Kilian Q. Weinberger and Lawrence K. Saul. Distance Metric Learning for Large Margin Nearest Neighbor Classification. Journal of Machine Learning Research, vol. 10, no. 9, pages 207–244, 2009. (Cited in page 118.)

[Woo 2023] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon and Saining Xie. ConvNeXt V2: Co-Designing and Scaling ConvNets With Masked Autoencoders. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 16133–16142, June 2023. (Cited in pages 122, 124, and 125.)

[Xiao 2018] Fanyi Xiao and Yong Jae Lee. Video Object Detection with an Aligned Spatial-Temporal Memory. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu and Yair Weiss, editors, Computer Vision – ECCV 2018, pages 494–510, Cham, 2018. Springer International Publishing. (Cited in page 95.)

[Xie 2017] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu and Kaiming He. Aggregated Residual Transformations for Deep Neural Networks. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 5987–5995, 2017. (Cited in page 47.)

[Xie 2022] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai and Han Hu. SimMIM: a Simple Framework for Masked Image Modeling. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 9643–9653, 2022. (Cited in pages 121, 124, and 126.)

[Xiong 2022] Weiyi Xiong, Jianan Liu, Yuxuan Xia, Tao Huang, Bing Zhu and Wei Xiang. Contrastive Learning for Automotive mmWave Radar Detection Points Based Instance Segmentation. In 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), pages 1255–1261, 2022. (Cited in page 60.)

[Xu 2021] Baowei Xu, Xinyu Zhang, Li Wang, Xiaomei Hu, Zhiwei Li, Shuyue Pan, Jun Li and Yongqiang Deng. RPFA-Net: a 4D RaDAR Pillar Feature Attention Network for 3D Object Detection. In 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), pages 3061–3066, 2021. (Cited in page 60.)

[Xu 2022] Li Xu, Yueqi Li and Jin Li. Improved Regularization of Convolutional Neural Networks with Point Mask. In Xingming Sun, Xiaorui Zhang, Zhihua Xia and Elisa Bertino, editors, Advances in Artificial Intelligence and Security, pages 16–25, Cham, 2022. Springer International Publishing. (Cited in page 70.)

[Yamada 2005] N. Yamada, Y. Tanaka and K. Nishikawa. Radar cross section for pedestrian in 76GHz band. In 2005 European Microwave Conference, volume 2, pages 4 pp.–1018, 2005. (Cited in page 25.)

[Yang 2021] Ceyuan Yang, Zhirong Wu, Bolei Zhou and Stephen Lin. Instance Localization for Self-supervised Detection Pretraining. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 3986–3995, 2021. (Cited in page 123.)

[Yang 2023] Bo Yang, Ishan Khatri, Michael Happold and Chulong Chen. ADCNet: End-to-end perception with raw radar ADC data. arXiv preprint arXiv:2303.11420, 2023. (Cited in page 69.)

[Yu 2022] Ye Yu, Jialin Yuan, Gaurav Mittal, Li Fuxin and Mei Chen. BATMAN: Bilateral Attention Transformer in Motion-Appearance Neighboring Space for Video Object Segmentation. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella and Tal Hassner, editors, Computer Vision – ECCV 2022, pages 612–629, Cham, 2022. Springer Nature Switzerland. (Cited in page 95.)

[Yun 2019] Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo and Junsuk Choe. CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 6022–6031, 2019. (Cited in page 70.)

[Yun 2020] Sangdoo Yun, Seong Joon Oh, Byeongho Heo, Dongyoon Han and Jinhyung Kim. Videomix: Rethinking data augmentation for video classification. arXiv preprint arXiv:2012.03457, 2020. (Cited in page 71.)

[Yun 2022] Sukmin Yun, Hankook Lee, Jaehyung Kim and Jinwoo Shin. Patch-level Representation Learning for Self-supervised Vision Transformers. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 8344–8353, 2022. (Cited in page 123.)

[Zbontar 2021] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun and Stephane Deny. Barlow Twins: Self-Supervised Learning via Redundancy Reduction. In Marina Meila and Tong Zhang, editors, Proceedings of the 38th International Conference on Machine Learning, volume 139 of Proceedings of Machine Learning Research, pages 12310–12320. PMLR, 18–24 Jul 2021. (Cited in page 120.)

[Zeiler 2014] Matthew D. Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In David Fleet, Tomas Pajdla, Bernt Schiele and Tinne Tuytelaars, editors, Computer Vision – ECCV 2014, pages 818–833, Cham, 2014. Springer International Publishing. (Cited in page 46.)

[Zhang 2018] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin and David Lopez-Paz. mixup: Beyond Empirical Risk Minimization. In International Conference on Learning Representations, 2018. (Cited in page 70.)

[Zhang 2019] Chen Zhang and Joohee Kim. Modeling Long-and Short-Term Temporal Context for Video Object Detection. In 2019 IEEE international conference on image processing (ICIP), pages 71–75. IEEE, 2019. (Cited in pages 87, 88, 95, and 124.)

[Zhang 2021] Ao Zhang, Farzan Erlik Nowruzi and Robert Laganiere. RADDet: Range-Azimuth-Doppler based Radar Object Detection for Dynamic Road Users. In 2021 18th Conference on Robots and Vision (CRV), pages 95–102, 2021. (Cited in pages 3, 5, 6, 9, 55, 64, 65, 66, 67, 74, 78, 79, 80, 88, 89, 96, 115, 138, and 139.)

[Zhao 2021] Nanxuan Zhao, Zhirong Wu, Rynson W.H. Lau and Stephen Lin. Distilling Localization for Self-Supervised Representation Learning. Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 12, pages 10990–10998, May 2021. (Cited in pages 122 and 123.)

[Zhao 2023] Peijun Zhao, Chris Xiaoxuan Lu, Bing Wang, Niki Trigoni and Andrew Markham. CubeLearn: End-to-end Learning for Human Motion Recognition

from Raw mmWave Radar Signals. IEEE Internet of Things Journal, pages 1–1, 2023. (Cited in pages 69 and 140.)

[Zheng 2021] Zangwei Zheng, Xiangyu Yue, Kurt Keutzer and Alberto Sangiovanni Vincentelli. Scene-Aware Learning Network for Radar Object Detection. In Proceedings of the 2021 International Conference on Multimedia Retrieval, ICMR '21, page 573–579, New York, NY, USA, 2021. Association for Computing Machinery. (Cited in pages 67, 70, 71, and 96.)

[Zhou 2018] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4490–4499, 2018. (Cited in page 60.)

[Zhou 2019] Xingyi Zhou, Dequan Wang and Philipp Krähenbühl. Objects as Points. ArXiv, vol. abs/1904.07850, 2019. (Cited in pages 45, 48, and 67.)

[Zhu 2017] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan and Yichen Wei. Flow-Guided Feature Aggregation for Video Object Detection. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 408–417, 2017. (Cited in page 95.)

[Zhu 2018] Menglong Zhu and Mason Liu. Mobile Video Object Detection with Temporally-Aware Feature Maps. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5686–5695, 2018. (Cited in pages 7, 95, 98, 99, and 107.)

Titre : Extraction et identification de cibles multiples pour radar automobile à l'aide d'intelligence artificielle

Mots clés : Radar FMCW, Intelligence artificielle, Détection, Identification, Suivi

Résumé : Ces dernières années, l'apparition de véhicules de plus en plus connectés ont ouvert la voie à des modes de transports plus sûrs et plus autonomes. Ces véhicules s'appuient sur des systèmes avancés d'aide à la conduite (ADAS) et utilisent divers capteurs comme le radar, la caméra, le LiDAR et le V2X pour créer un cocon de sécurité à 360° autour du véhicule. Si l'intelligence artificielle et l'apprentissage profond ont permis la détection et l'identification d'objets en temps réel à l'aide de caméras et de LiDAR, l'utilisation de ces algorithmes sur des données radar est encore limitée. Pourtant, les radars présentent des avantages, notamment celui de fonctionner dans des conditions météorologiques difficiles et d'offrir de bonnes performances en terme de résolution en distance, angulaire et en vitesse, à un coût inférieur à celui du LiDAR. Cependant, les données renvoyées par les radars actuels contiennent peu d'information concernant les cibles détectées et plusieurs étapes de pré-traitement et de post-traitement sont nécessaires pour les obtenir. Ces étapes de traitement dénaturent le signal brut réfléchi par les objets, pouvant affecter les performances des algorithmes d'intelligence artificielle. Ce doctorat vise à développer de nouveaux algorithmes d'apprentissage profond spécifiquement adaptés aux données radar, visant à être incorporés dans des systèmes automobiles. Ces algorithmes auront pour but de détecter et identifier les objets autour d'un véhicule dans des environnements complexes. Outre les algorithmes, cette thèse étudiera quelles types de données radar, et donc quelle quantité de pré-traitement, permettent d'obtenir les meilleures performances. Les algorithmes proposés dans cette thèse devront satisfaire aux contraintes des environnements automobiles: faible puissance, faible complexité et temps de réaction rapide.

Title: Multiple target extraction and identification for automotive radar with A.I.

Key words: FMCW Radar, Artificial intelligence, Detection, Classification, Tracking

Abstract: In recent years, connected vehicles have paved the way for safer and more automated transportation systems. These vehicles rely heavily on Advanced Driving Assistance Systems (ADAS) and use various sensors like radar, camera, LiDAR, and V2X to ensure 360° safety type of cocoon around the vehicle. While artificial intelligence and deep learning have enabled real-time object detection and identification using cameras and LiDAR, the use of such algorithms on radar data is still limited. Radar sensors offer advantages, such as working in challenging weather conditions and providing good performance in distance, angular and speed resolution, at a lower cost than LiDAR. However, radars output relatively low content information regarding the detected targets and several pre and post-processing steps are required to obtain those. Since the processing steps filter the raw signal returned by objects, it can affect the performance of AI algorithms. This PhD aims to develop new deep learning algorithms explicitly tailored for raw radar data to integrate them into automotive systems. These algorithms will detect and identify objects in complex environments. Additionally, this thesis will explore the optimal types of radar data and pre-processing steps for achieving the best performance. The algorithms will have to meet automotive constraints, including low power consumption, simplicity, and fast response times.