Online learning for dynamic impending collision prediction using FMCW radar

AARTI SINGH, Washington University in St. Louis, USA
NEAL PATWARI, Washington University in St. Louis, USA

Radar collision prediction systems can play a crucial role in safety critical applications, such as autonomous vehicles and smart helmets for contact sports, by predicting impending collision just before it will occur. Collision prediction algorithms use the velocity and range measurements provided by radar to calculate time to collision. However, radar measurements used in such systems contain significant clutter, noise, and inaccuracies which hamper reliability. Existing solutions to reduce clutter are based on static filtering methods. In this paper, we present a deep learning approach using frequency modulated continuous wave (FMCW) radar and inertial sensing that learns the environmental and user-specific conditions that lead to future collisions. We present a process of converting raw radar samples to range-Doppler matrices (RDMs) and then training a deep convolutional neural network that outputs predictions (impending collision vs. none) for any measured RDM. The system is retrained to work in dynamically changing environments and maintain prediction accuracy. We demonstrate the effectiveness of our approach of using the information from radar data to predict impending collisions in real-time via real-world experiments, and show that our method achieves an F1-score of 0.91 and outperforms a traditional approach in accuracy and adaptability.

CCS Concepts: • **Computing methodologies** → *Online learning settings*; • **Hardware** → *Wireless devices*; • **Applied computing** → *Consumer health.*

Additional Key Words and Phrases: Collision prediction, FMCW radar, online learning

ACM Reference Format:
Aarti Singh and Neal Patwari. 2023. Online learning for dynamic impending collision prediction using FMCW radar. *ACM Trans. Internet Things* 1, 1 (August 2023), 27 pages. https://doi.org/10.1145/3616018

1 INTRODUCTION

Collision warning systems (CWS) aid in providing safety measures in a variety of applications. In the field of contact sports, physical collisions between players cause 1.6 to 3.8 million concussions or traumatic brain injuries (TBI) annually in the United States, and American football is a most prominent contributor of these TBIs [16]. Such concussions are life-altering and adversely affect players throughout their lives [21]. Traditional protective gear such as helmets can reduce the severity of such concussions, however, they are limited in their utility, as even with the helmets concussions occur. Preventative measures are an important part of protection. Predicting the collisions that are about to occur before they happen could dramatically improve outcomes [45]. Rather than after-the fact detection of a collision, we argue it is critical to predict collision and warn the player more than their reaction time before the collision. One solution is to equip the player with smart wearable device that predicts the impending collisions. A smart helmet could automatically

Authors' addresses: Aarti Singh, Washington University in St. Louis, 1 Brookings Dr., St. Louis, Missouri, USA, aartisingh@wustl.edu; Neal Patwari, Washington University in St. Louis, 1 Brookings Dr., St. Louis, Missouri, USA, npatwari@wustl.edu.

2577-6207/2023/8-ART $15.00
https://doi.org/10.1145/3616018

and actively adapt its damping to reduce harm upon impact [7]. Alternatively, it could supply a loud auditory warning to the player that allows them to react. New research suggests that warnings prior to impact allow people to 'prepare themselves, effectively mitigating the consequence of the impact' [44]. Our radar-based collision prediction system has the goal of enabling such active responses from players and their helmets.

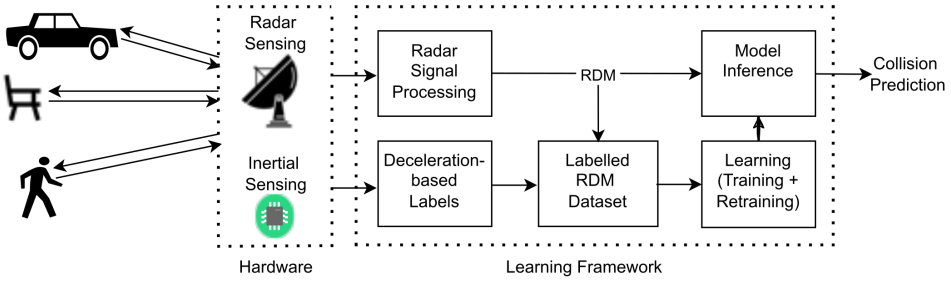


Fig. 1. Proposed collision prediction system overview: The hardware block present on a mobile node is capable of sensing the environment with the help of a FMCW radar-based sensing and notes its inertial measurements. The learning framework uses measurements to generate radar-Doppler matrices (RDMs) that are fed to a trained model for inferring impending collisions. Simultaneously, the model is periodically re-trained with the latest labelled RDMs to improve results in a changing environment.

Collision prediction systems also find utilization in the field of automation, where frequent collisions encountered by the mobile nodes such as unmanned aerial vehicles (UAVs) and robots cause nuisance and damage. Collision warning systems for vehicles [30], [4], [78] have developed solutions that can be applied to combat the collision problem faced in automation. For example, centralized algorithms such as [17], [12] can provide location information to mobile nodes in a network to locate surrounding nodes and predict impending collisions. As an alternative, distributed and cooperative solutions to obtain locations are also well studied [47]. Similarly, mobile nodes can rely on network-provided information to locate themselves and predict any collisions with other nodes [2]. These localization methods are suitable for nodes that are part of a network. However, every object encountered by the mobile node may not be associated with the same network as the mobile node. Therefore, in order for a mobile node to predict impending collisions with any kind of objects in the environment, it should be able to perform standalone sensing of all objects in its environment without requiring a network of sensors.

It should be noted that collision prediction is fundamentally different than localization. Collision prediction must take into account not only the present positions of objects in the environment, or positions at any single future time, but all of the positions between now and a future time in order to know whether a collision will occur at any point in that time period. As a benefit compared to localization, however, collision prediction uses only the relative kinematics (such as relative range, relative velocity), rather than absolute coordinate information of the objects. We address the complexity of estimating positions across a time period; and take advantage of the relative nature of collision prediction in the methods developed in this paper.

We consider a mobile node that is envisioned to be equipped with sensing and processing capabilities that allow it to make standalone measurements from the environment and self-sufficient decisions about impending collisions. The sensors are required to be low cost, compact, lightweight, and capable of processing information to predict collisions with a low false positive rate. There

are several sensor options available for CWS, each with their advantages and limitations [70]. In the case of sensing within a network of nodes, paired communication with other tagged devices is achieved using ultra-wide band (UWB) [61] or radio frequency identification (RFID) sensors. However, for sensing of the entire environment for possible collision with both tagged and untagged objects, standalone sensing is required. LiDARs (light detection and ranging) are expensive sensors that provide rich information about scenes and targets, but can be susceptible to severe weather. Ultrasonic sensors are affordable and compact and are suitable for very short-range detection. Vision-based solutions do not perform well for long distances, in poor light, and in complex, real-world conditions. Radars have been extensively studied in literature, especially for automotive applications [22] and unlike radio frequency tags, in which each device must receive reflections from other tagged devices for pairwise ranging, radars are capable of sensing all objects with electromagnetic properties different from air that are present in the field of view, without requiring additional sensors on the surrounding objects. Moreover, FMCW radars are inexpensive, compact sensors that can measure the relative kinematics (range and range-rate) of the objects in the field-of-view in real-time. Building on this standalone sensing capability of FMCW radars, a performance study of FMCW radars is presented in [11] where the authors analyse the use of an 'impact parameter' in predicting collisions. It must be emphasized that in practice, the accuracy of range and velocity measurements from radars suffers due to system noise and unwanted scattering and reflection, including 'self-motion', which collectively are referred to as 'clutter', which are a function of the material properties of the other objects in the environment that we are not concerned about the robot colliding with, including itself, or anything that would not cause problems if hit. Commonly used techniques to remove clutter are empirical mode decomposition [49], principal component analysis, and independent component analysis. However, working with non-stationary objects, the effect of such interference can be dynamic. Traditional collision prediction methods require these static filtering and pre-processing techniques which may not be effective in the environment in which it is deployed. Traditional methods cannot adapt to the type and style of play of an individual sports player. Learning-based solutions, on the other hand, can offer flexible and data-dependent prediction capabilities in order to efficiently utilize FMCW radars for dynamically changing environments. In the smart helmet application, they can also learn player-specific mobility patterns that do and do not lead to collisions.

Furthermore, the learning-based solutions need to be robust against degrading performance when there is shift in the input data characteristics, therefore it is necessary that a learning-based model's parameters be continuously adjusted for new or dynamic environments. The machine learning paradigm of online learning is a continuous learning process to learn more efficiently with the data arriving incrementally, to perform the same learning task over time, by the model that is already deployed. When a new labeled data sample arrives, online learning allows the existing model to quickly update its parameters to produce the best model so far. However, incrementally updating with the newest data leads to catastrophic forgetting [48], where learning only a small amount of new information can overwrite established knowledge and cause complete loss of ability to operate on previously learned tasks. Incremental batch learning solves the problem of online catastrophic forgetting by utilizing a series of batches of new labeled samples. After a batch has been received, the model loops over the batch until it is adequately learned, and then the model can be tested on information in that batch and previous batches. However, due to limited resources and requirement for fast on-device learning, only the relevant samples that carry new information must be used for learning incrementally. The field of active learning [59] provides methods for selecting those most relevant labelled samples thereby reducing the size of information that is necessary for decision making. Taking advantage of the on-going research in the field of online incremental

learning, we present a functional and practical real-time algorithm that adapt its parameters with the dynamically changing environments.

In this paper, we extend the nascent idea of utilizing FMCW radars for collision prediction in noisy and cluttered environments, when motion is dynamic and changing. Typical CWS methods [10] implement parametric algorithms where collision risk is measured by calculating node specific parameters, such as its range and velocity. However, these parameters are prone to inaccuracies when measured via a radar in cluttered environments. In order to tackle this, we utilize the research done in the field of computer vision and machine learning. As presented in Fig. 1, we propose a learning-based solution to demonstrate the effectiveness of using only the range and radial velocity (range-rate) information obtained from RDMs for predicting impending collision with the help of a convolution neural network (CNN). The proposed deep learning framework extracts adaptable features from the (continuously changing) environment, thereby eliminating the need for static filtering. Moreover, we apply online learning strategies, as well as automated labelling using accelerometer measurements, in order to make the the CNN classifier adaptable and learn from the recent history of the dynamically changing environment.

Challenges. In order to solve the collision prediction problem using FMCW radar and deep learning in a cluttered and dynamic environment, we face the following challenges:

- The presence of clutter in the radar measurements requires extensive filtering to get accurate measurement of the relative range and relative speed for collision prediction.
- Working with static features for machine learning solutions deteriorates the performance of the such solutions operating in new types of environments or dynamically changing environments.
- The dataset obtained for building a collision prediction system suffers from class imbalance due to the fact that real collisions happen only rarely compared to non-collisions.

Contributions. Overall, we make the following contributions.

- We develop a novel data-driven, learning-based collision prediction method that uses only radar and inertial data to detect an impending collision.
- We remove the need of static filtering by using the unfiltered, raw radar data for collision prediction, relying on deep learning to extract useful features for predicting collisions.
- We alleviate the class imbalance problem by assigning weights to each class as per their respective frequency in the dataset to achieve high classification accuracy for all the classes.
- We provide a retraining framework by using automated labelling and uncertainty sampling for improving prediction performance in changing environments and for dynamic motion.
- We experimentally collect and publish a large dataset to mimic real-world collision scenarios. We use the dataset to validate that our proposed method performs well in comparison with other non-learning collision warning methods and traditional machine learning methods.

In short, this paper presents and evaluates a novel approach to radar-based collision prediction which adapts, automatically, to the particular environment and characteristics of motion. We expect that it will expand the set of applications in which radar-based CWS can be successful.

2 METHODOLOGY

In this section we formally define the problem statement, the notation for the measurements, the loss function utilized in the learning framework, the learning model used to minimize the loss function, and the retraining procedure in detail.

2.1 Problem Statement

We consider a mobile node which has an attached FMCW radar, an inertial sensor to measure node's acceleration, and a processor for computing.

The mobile node is moving in an environment with several obstacles. Let us consider the scenario where the radar-enabled node is moving towards an obstacle. Let $d(t)$ be the range and $v(t)$ is the relative velocity between the mobile node and the obstacle at time t. During one measurement time period between t_1 and $t_2 > t_1$, duration $T = t_2 - t_1$, the node collects samples from the radar and inertial sensor. At time t_2 it converts them to a range-Doppler matrix, X_{t_2}.

The goal of our system is to raise an alarm at t_2 if at any time $t \in [t_2, t_2 + \delta t]$, that $d(t) < \epsilon$, where $\delta t > 0$ is the time duration into the immediate future for which we must detect a future collision. Also, $\epsilon > 0$ is a collision proximity range threshold. In other words, if the measurement indicates that the node will collide with the object within the next δt period of time, the alarm should be raised. In order to achieve this, we create system that can learn a function f that maps from the range-Doppler matrix to a binary decision about whether there will be an impending collision.

2.2 Learning Framework

For the collision prediction problem, the inputs we use are RDM matrices, named X, that get mapped via the function f to $\hat{y} \in \{0, 1\}$, where class 0 is encoded as 0 for representing 'no impending collision' and class 1 is encoded as 1 representing 'impending collision'. This makes collision prediction a binary classification problem. Supervised learning-based approaches require a training set of 'measurement-label' pairs $[(X_1, y_1), ...(X_n, y_n)]$ where n is the total number of pairs in one training set, X_j is matrix j and y_j is its collision label, such that $j \in \{1, ..., n\}$.

Loss Function

In order to learn the optimal mapping f between RDM and $\hat{y}$, a classifier for two classes needs to minimize a loss function L for the entire training set of size n, given by

$$L = -\frac{1}{n} \sum_{j=1}^{n} [y_j \log(p_j) + (1 - y_j) \log(1 - p_j)], \tag{1}$$

where p_j is the probability of the j^{th} data point belonging to a class as predicted by the mapping. Taking an average over the entire dataset of size n, we get cross entropy loss L, a standard loss function for classification problems.

CNN-based Classifiers

CNNs have become widely popular for image-based machine learning tasks. Compared to the standard MLP architectures, a CNN architecture uses far fewer parameters and can have much deeper architectures which can equip us to solve more complex problems. There are several CNN architectures available with varying degrees of computational complexity and performance.

For our learning-based solution, we test two CNN architectures: (a) ResNet architecture, since it has shown the top-5 error rate on ImageNet dataset [27] and (b) MobileNet-V3 architecture, since it has been optimized for low-resource devices and low latency applications [28].

2.3 Radar Signal Processing

The FMCW radar transmits sequences of a linear frequency modulated (LFM) signal, also called a 'chirp' signal having N samples per chirp, which increases its frequency linearly with time with a bandwidth of B Hz and chirp duration (also called pulse repetition interval) of T_c. The slope s of

the chirp represents the rate of change of frequency, thus $s = B/T_c$. Once the radar receives the reflected signals that bounce back from the target, they are mixed with the transmitted signals to obtain a 'beat signal'. The beat signal is characterized by the beat frequency, f_b, which is equal to $2ds/c$, where d is the range of the object from the radar and c is the speed of light. This frequency is used to infer the range of the target from the radar sensor. A fast Fourier transform (FFT), called the 'range-FFT', is performed on the beat signal to convert it into the frequency domain, thereby obtaining the beat frequencies representing one or multiple objects at various ranges, with a range resolution of $c/2B$ and up to the maximum range of $f_s c/2s$, where f_s is the sampling rate, such that $f_s = N/T_c$ [51]. A second FFT, called the 'velocity-FFT', is then performed across a certain M number of chirps that make one RDM, to estimate the relative radial velocity. It is left to the discretion of the designer to define how many of the chirps are going to be processed together into an RDM (due to processor limitations and/or resolution requirements) [20]. Our proposed method implements this 2-D FFT stage and generate a $N \times M$ matrix, which is referred to as the radar-Doppler matrix (RDM), that indicates the amplitude of scattering from each possible relative range and velocity across the possible range of measured range and velocities.

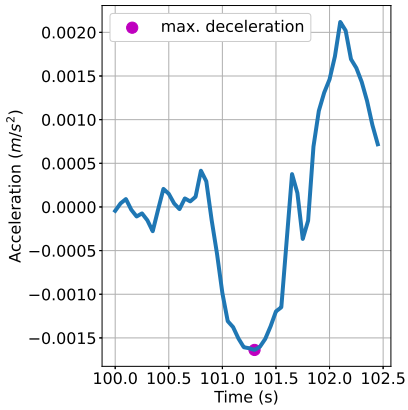
It should be noted that reflection and scattering of the radar signals by objects are a function of the intrinsic properties of the material of the objects, such as dielectric permittivity, magnetic permeability, and electrical conductivity [63]. These properties have been shown to be effective in object distinction and detection [73]. Additionally, the extrinsic characteristic such as the absolute size of the objects and the size of the objects relative to the wavelength of the radar signals play a crucial role in the amplitudes of scattering received in the radar reflections. Therefore, different types of objects produce different amplitudes of scattering within RDMs.

2.4 Label Creation

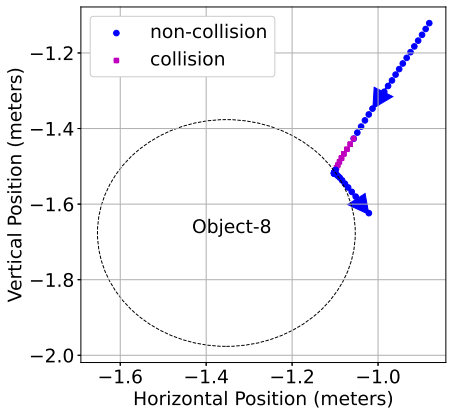
Our proposed system automatically generates (with a sub-second delay) labels for the collected radar data using inertial sensing. This labelled training data can be used to automatically retrain the model in order to improve results for the particular environment and user characteristics that are observed during operations.

As a collision between the moving node and an obstacle occurs, the moving node experiences a sudden change in velocity. This change in velocity can be clearly observed as a change in the measured acceleration, for example, as a sharp drop or negative peak in mobile node's IMU measurements, as seen in Fig. 2a. During normal operation we obtain these 'moments of collision' by the finding peaks in a node's measured acceleration data.

A collision prediction system should alert the nodes about impending collisions before they happen such that the colliding nodes have time to act preemptively to avoid the collision. Considering an alert can be raised instantaneously by a collision prediction system without any mechanical or processing delay, a node receiving this alert, however, will take time equal to its reaction time for any responsive action. To have at least one alert sent by the system to the node before any impending collision, the immediate future δt required to check for impending collisions must be at least equal to the reaction time of the node. Thus, if a collision occurs at δt into the future, it can be avoided or ameliorated by issuing one alert at a time that is δt before the collision. Depending on the reaction time of the node and the number as well as the frequency of alerts needed for the node (robots or human) in any specific application (vehicular or sports-related), the threshold for δt can be adjusted. Based on this approach, for labelling purposes, all the measurements (the RDMs in our case), that happened within δt from each moment of collision as measured by IMU are labelled as 'Impending Collision' (shown as magenta colored square markers on trajectory as shown in Fig. 2b). The rest of the RDMs are labelled as 'No Impending Collision' (shown as blue colored circle markers as shown in Fig. 2b). The labelled images are then used to train our CNN-based model



(a) Selecting moment of collision using the maximum deceleration in moving node's IMU data



(b) Using moment of collision to label steps on the path as impending collision

Fig. 2. Demonstration of the label creation scheme using moving node's deceleration measurement from its IMU

at the subsequent training time to build the next model that provides inference about impending collisions when encountering new, unseen test RDM samples.

It should be noted that in real-world scenarios, false labelling is possible, where, for example due to sensor noise or outdated thresholds, RDMs can be labelled incorrectly through the current labelling scheme. This leads to 'label-shift'. Deep-learning models tend to be robust against infrequent and randomly mislabeled data, treating the labels as noise or outliers. The ability of a model to treat a small number of incorrect labels as noise depends on the quality and quantity of the noisy data, and the overall complexity of the problem being solved. Therefore, precautions are suggested in the choice of IMU sensors and the threshold of the deceleration for the application. Other traditional techniques can also be adapted to handle the label noise prior to training such as label smoothing, example weighting, and custom loss functions [32].

2.5 Class Imbalance Problem

During system operation, the radar is continuously sensing the environment. Even in the environments that are densely populated with obstacles, the events during which a collision is imminent is considerably lower in frequency than the frequency of events in which the node is not in danger of an imminent collision. With the labelling scheme for the generated RDMs described above, the number of RDMs labelled as 'impending collision' are order of magnitudes lower than the number of RDMs labelled as 'no impending collision', which leads to an imbalanced dataset.

Imbalanced datasets, such as the state of the art Give-Me-Some-Credit dataset [33] and statlog heart disease dataset[67], are common in anomaly detection, medical diagnosis, fraud detection, and are challenging to tackle in machine learning domain, as the standard learning algorithms may struggle to perform well on minority classes due to the bias introduced by the majority class. There are many methods to tackle this challenge: re-sampling methods which include over-sampling of the less frequent class using random, SMOTE (Synthetic Minority Over-sampling Technique), and ADASYN (Adaptive Synthetic Sampling) techniques, or under-sampling of the more frequent class using random and Tomek links [13], [26], [66]. However, important considerations needed to be

weighed before applying these. For example, in over-sampling, increasing the size of the training set without gaining any additional information might not add any benefit, and can introduce noise. On the other hand, under-sampling might lead to discarding of the relevant information. Other methods include the use of tree-based algorithms or ensemble methods like bagging and boosting, which might lead to over-fitting [31] and hence poor generalization to new data. There is no one-size-fits-all answer to which method is better for addressing imbalanced datasets, as the effectiveness of different methods or combination of different methods depends on various trade-offs and factors such as computational cost, interpretability, and evaluation metrics. For our system, we refrain from using the over-sampling techniques as they might increase the size of the dataset without adding any useful information, thereby increasing the computational load on the resources required for training the model in general. Furthermore, under-sampling methods are not explored as a possible solution for our system's imbalanced dataset problem as under-sampling may remove the data points that otherwise would provide the highest uncertainty during inference, a required criteria for our retraining framework, such that we can make our system to be robust against data-drift.

Instead, we use the method proposed in [37], in which each class is assigned class weights that is inversely proportional to their respective frequencies, such that

$$w_j = \frac{|C_1| + |C_0|}{2|C_j|}, \tag{2}$$

where $|C_j|$ is the number of samples in class $j \in \{0, 1\}$. Assigning a small weight to the cost function for the more frequent class during training results in a smaller error value, and thus, small update to the model coefficients for the more frequent class. A large weight applied to the cost function for the less frequent class will result in a larger error calculation, and in turn, large update to the model coefficients for the less frequent class. This way, we can shift the imbalance of the model so that it could reduce the errors of the less frequent class. Choosing this method provides highest interpretability compared to other methods, while it also maintains original distribution of instances and preserves the inherent patterns and relationships in the data.

An additional problem encountered for imbalanced datasets is that using the accuracy of the model to assess its performance becomes a less reliable metric because an acceptable accuracy is possible to obtain even if the model has excellent classification performance for more frequent class and poor performance on the other, less frequent class [38]. Thus, instead of using accuracy to pick the best model, we use the $F1$ score as a metric that describes the performance of the classifier on both the classes. $F1$ score is the harmonic mean of the positive predictive value (P, also called precision) and true positive rate (R, also called recall or probability of detection(P_D)) given by

$$F1 = \frac{2PR}{P + R}, \tag{3}$$

where P is the fraction of correctly classified positive instances ('impending collision' in our case) among the overall instances that are classified as positive, while R is the fraction of correctly classified positive instances among the overall instances that are truly positive. A high $F1$ score indicates low misclassifications in both of the classes. Therefore, during training epochs, we select the model which has the highest $F1$ score.

2.6 Online Learning Framework

Real-world implementation of any collision system experiences a continuous influx of new data in real-time. A static learning-based solution leads to decreased inference performance, especially if the new data is from different distributions, as a result of, for example, changing user behaviour or environmental conditions compared to the previous data on which the model was trained.

This phenomena is called data-drift, which includes changes in the distribution of the data (co-variate shift), changes in the distribution of the labels (label shift), and changes in the distribution of the relation between data and labels (concept drift) [29]. Domain adaptation, transfer learning, fine-tuning, and continual learning enable models to adapt to the changing data distributions and maintain good performance, however, these methods need large amount of the dynamically changing data. On the other hand, meta-learning provide solutions that have advantage of working with only a few or single instances of new data [18]. However, if the data distribution of the target task significantly deviates from the data distribution of the training tasks, the performance of the meta-learning methods model may degrade and hence, are not seen as a better alternative [72], [76]. Moreover, since our system in question experiences abundant data that has undergone distribution-shift, there is value in utilizing all the relevant data in our application. In order to maintain model's inference performance over time on new, unseen data that has experienced data-drift, one of the techniques is to periodically retrain the model to tackle data-drift in real-world applications. Model retraining is defined as re-running the process that generated the previously selected model, however, on a new training dataset that has experienced data drift. This can be computationally expensive, thus, we explore the branch of active learning and suggest the following retraining strategy:

- Continuously collect system's IMU-labelled RDMs from recent history to form a retraining dataset.
- Based on the system requirement for the retraining latency, select a fraction of the collected dataset using the uncertainty-sampling technique [40].
- Periodically retrain the default or current model using this newly sampled subset of data.
- Repeat the above steps for the newest recent history of dataset collected.

It should be noted in real-world scenarios of using the automated labelling scheme can experience noise or 'label-shift', which further leads to 'co-variate shift' [29]. Therefore, it is necessary to periodically retrain the model that is implemented in dynamic environments. We explore how to implement such a scheme for resource-constrained devices in Section 4.4.

3 EXPERIMENTS

3.1 Objects and Environment

In order to obtain a large quantity of data in which our collision prediction and warning device experiences actual collisions with obstructions, without having to subject a person to carry the device and experience the same collisions, in this paper we implement the following setup. We use a robotic motion platform as our moving node for our experiments that is comprised of an iRobot Roomba, which is controlled by attached Raspberry Pi-3 module as shown in Fig. 3b. The node moves in a laboratory environment as shown in Fig. 3a which has other stationary obstacles. The OptiTrack motion capture system [50] tracks and records the ground truth position co-ordinates of all the objects that are tagged with reflective markers. An 3 m $\times$3 m area is barricaded with PVC pipes in order to confine the moving node and maximize its number of collisions with obstacles within the observed area. Within this obstacle-rich area, randomly placed obstacles are present that are filled with a variety of materials. In order to collect a dataset with a variety of radar reflections for our collision prediction method, we choose three different kinds of materials (gravel, soil, and water), as shown in Fig. 3c, 3d, 3e, and 3f, to simulate the items that are commonly encountered by a moving objects (vehicles or humans) in the real-world. Cylindrical shaped objects are 0.5m in height and 0.3m in diameter, where as rectangular shaped objects are $0.5 \times 0.2 \times 0.3$m in width.

3.2 Sensing Hardware

We use AncorTek's SDR-KIT 2400AD2 [68], a low-power and compact software-defined radar for FMCW-based sensing, sitting at the top deck of the prototype. The center frequency of transmitted signal is adjustable within the 24-26 GHz frequency band. AncorTek provides drivers only for Windows, and thus, we use a Windows laptop to receive and store the complex in-phase and quadrature (I/Q) components of the received FMCW radar signal. For the radar's settings, we choose a bandwidth of 2 GHz at a center frequency 25 GHz during all our experiments. Each chirp is of 1 ms duration with $N = 128$ samples per chirp. With these settings, the range resolution and the maximum range that can be measured are 0.075 m and 4.8 m, respectively. Also, the maximum radial velocity that can be measured is 3 m/s. The radial velocity resolution is dependent on the number of chirps processed together at once for one RDM [20]. Lastly, the moving node also has a BNO055 IMU sensor [64] that collects the acceleration data during our experiments at the rate of 60 Hz. It is noted that this low-cost IMU sensor is prone to noise and thus we use the highly accurate OptiTrack position ground-truth coordinates for estimating the mobile node's higher order kinematics, such as acceleration, which is taken as the second derivative of position with respect to time, and is used for the labelling purposes. All the data files are provided [60]. It should be noted that the size of the radar sensor used for our experiments is $79 \times 56 \times 13$ mm, which is comparable to the size of a commercially available collision warning radar systems. However, in applications such as smart wearables and smart devices, the size of sensing hardware is encouraged to be miniaturized, for example, as accomplished through Google Soli's 12×12 mm radar chip [41].

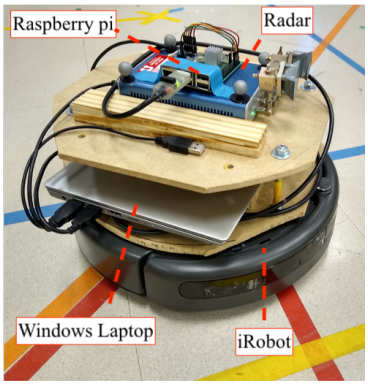
3.3 Motion and Collisions

We conduct a series of experiments with one mobile node moving in a cluttered environment with eight obstacles that are stationary. The maximum velocity that can be reached by the moving node is 0.5 m/s. The moving node is made to travel in two kinds of trajectories, straight and curved. For the straight trajectory, both the wheels of the moving node are programmed to maintain one constant speed (0.5 m/s), and thus, the node moves in a straight line motion. For the curved trajectory, one wheel of the moving node rotates slower than the other, creating a curvature in the trajectory, and mimicking a curved line motion. Several collisions happen between the mobile node and the obstacles during both styles of these trajectories as given in Table 1. When a collision happens, the moving node comes to a complete stop, takes a turn by a random angle, and then starts moving again at the same speed and in the same style of programmed trajectory as before the collision. Lastly, we do not consider the PVC pipes as 'obstacles' since they are used only to keep the robots inside the experimental area, and thus, all the collisions and the corresponding RDMs between the moving node and the PVC pipes are not included in the final dataset.

3.4 Data Characteristics

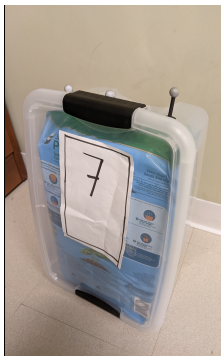
We run 10 experiments, each one either with curved and straight trajectories and each experiment is of 10 minutes duration. The duration of measurements, and thereby size of the dataset affects the convergence time and generalization performance of the model; however, the amount of dataset recorded, and stored or transferred is dependent on available resources and requirements of the system. The maximum duration possible to continuously record the IQ samples through AncorTek GUI is 10 minutes. Each experiment of 10 minutes thus creates one dataset that are shown in Table 1. We process $M = 200$ chirps together to make one RDM, thus for our experiments, one RDM comprises of 200 ms duration of complex-valued radar samples. Moving the RDM window every 50 ms, one 10-minute experiment generates 1.2×10^4 RDMs. The values for M and sliding window size are adjustable as suited by the system's requirements, such as desired velocity resolution

(a) The lab setup with obstacles, moving node, and PVC pipes as boundary.



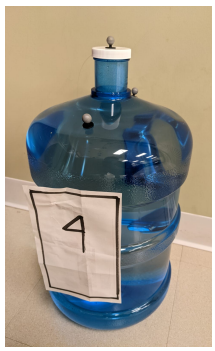
(b) The moving node with a FMCW radar and a Windows laptop to run the radar GUI, attached with Raspberry-Pi to control the trajectories and collect measurements from the IMU sensor.

(c) The obstacle filled with pebbles



(d) The obstacle filled with soil

(e) The obstacle filled with water, type-I



(f) The obstacle filled with water, type-II

Fig. 3. Experiment setup: (a) images from the laboratory environment showing (c), (d), (e), (f) the various types of obstacles of different shapes and material, and (b) the moving node having the sensing hardware.

Exp.#	Motion type	Number of Collisions	Number of Valid RDMs
1	Straight (S1)	35	5781
2	Straight (S2)	41	6693
3	Straight (S3)	36	5810
4	Straight (S4)	30	4050
5	Straight (S5)	31	4103
6	Curved (C1)	36	6536
7	Curved (C2)	45	7213
8	Curved (C3)	53	6723
9	Curved (C4)	40	6692
10	Curved (C5)	31	6790

Table 1. Details of the experiments conducted

and available computational capabilities to process and store RDMs, as is further investigated in Section 4.2 and Section 4.4. We drop all the collisions with PVC pipes and their corresponding

RDMs, as described above, thus obtaining 6×10^3 RDMs per experiment on average. On average, 32 collisions are encountered during one experiment. With the labeling scheme described in Section 2.4 and taking $\delta t = 400$ ms, we obtain on average 340 RDMs per experiment that can be labelled as 'impending collision', while rest of the RDMs are labelled as 'no impending collision'. Since only 3% of the RDMs can be labelled as 'impending collision', this dataset is highly imbalanced. As explained in Section 2.5, we deal with this issue with the help of Equation 2 by assigning weights to each class, as given in Table 2.

class j	Number of samples (n_{C_j})	class weight w_j
collision	340	8.5015
non-Collision	5441	0.5312

Table 2. Weights for Imbalanced classes in Experiment S1

3.5 Radar Sensing for Traditional CWS

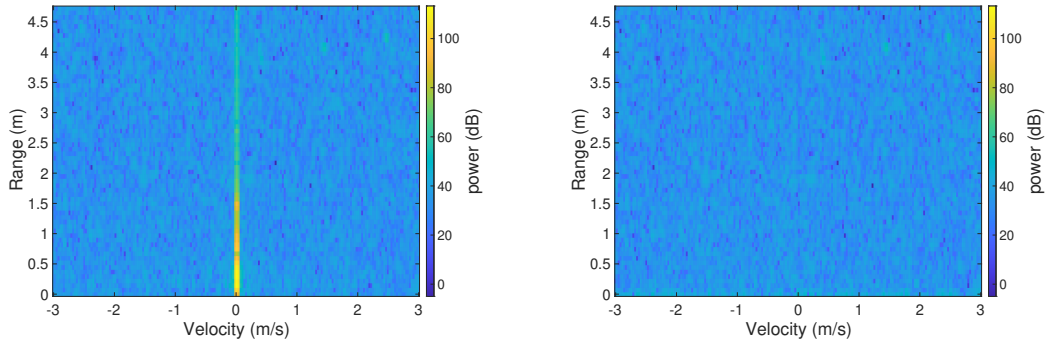
Environments where collisions are likely to happen are characterized by compact spaces with numerous and a variety of objects. Radar systems operating in such environments suffer from additional reflection and scattering from objects which do not cause collisions. For example, scattering from the ground contributes to reflections received at the radar, but in many applications, the node is not going to collide with the ground. Another source of unwanted reflections is the physical object to which the radar is attached, which, for example, in the smart helmet case includes the person themselves.

Overall, we refer 'clutter' as the received radar reflection that are due to any objects within the environment with which the moving node can not possibly collide. Typically these reflections make the largest contribution to the reflections received by the radar and hence, the visibility of the relevant targets in the radar images with which collisions risk must be addressed, get suppressed. Therefore, traditional radar-based CWS systems need to filter out unwanted reflections due to clutter before obtaining range and velocity information.

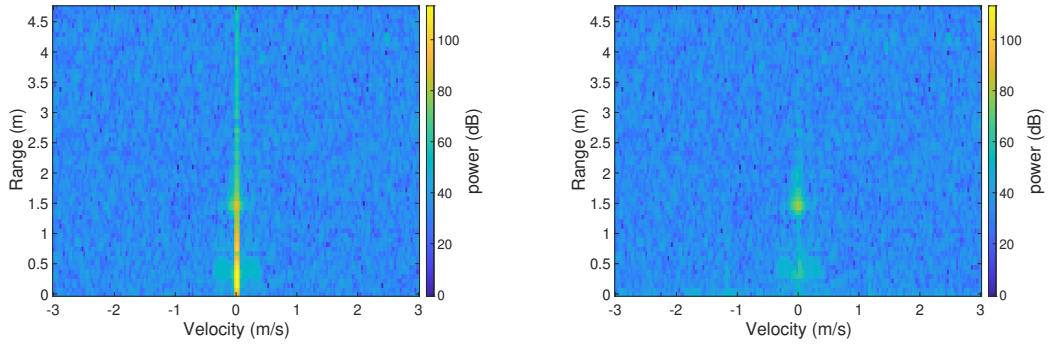
Classical principal component analysis (PCA) has been extensively used in image and video processing applications as a statistical tool to seek the best low-dimensional approximation of the high-dimensional data. We apply an advanced version of PCA as a model-based filtering method in order to remove reflections that are due to the clutter and extract the reflections that are due to the object from the RDMs.

The data can be constituted as a superimposition of two components: 1) L, which is the low-rank matrix and 2) S, which is a matrix that can be sparse or not. This decomposition can be obtained by robust principal component analysis (RPCA) solved via principal component pursuit (PCP) [5]. Due to the correlation between RDMs, the background and clutter are modeled by a low-rank subspace that can gradually change over time, while the objects that are in relative motion with respect to the radar constitute the correlated sparse outliers represented in the sparse matrix.

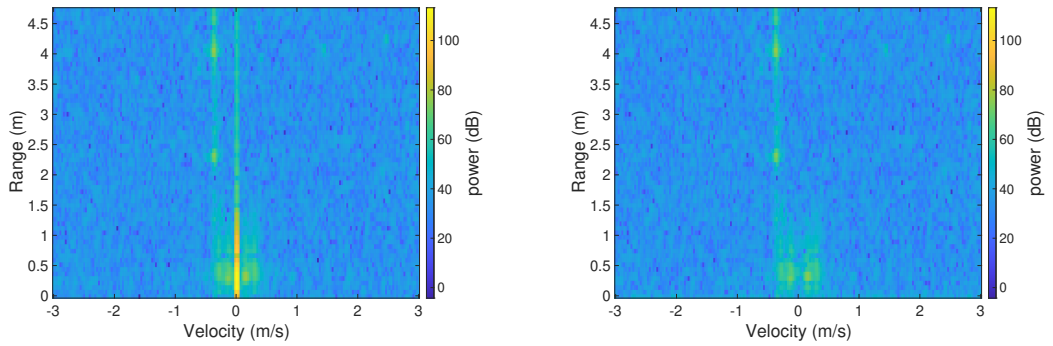
We demonstrate with the help of Fig. 4, the effect of applying RPCA-PCP to RDM images for the three RDMs for various scenarios: (a) radar facing no object in its field of view, (b) radar facing one stationary object in its field of view, and (c) radar facing two moving objects in its field of view. In the left column of Fig. 4, the bright vertical reflection at zero velocity is due to clutter. This clutter reflection is removed with the help of RPCA-PCP reconstruction of the signal, thereafter showing only the target in the RDM, as shown in the right column of Fig. 4. The post-PCA, filtered RDMs are then processed by the range-FFT and velocity-FFT as explained in Section 2.3 to obtain range and radial velocity of the target. It should be mentioned that the amplitude that represent the target



(a) RDM showing no object in view



(b) RDM showing one stationary object



(c) RDM showing two moving objects

Fig. 4. Visual demonstration of the effect of PCA-based clutter removal step for the baseline method: Left column shows the RDMs with clutter reflections present and right column shows the RDMs with clutter reflections removed via the PCA for three different scenarios: (a) RDM with no object in the view; (b) RDM with one stationary object in the view; (c) RDM with two moving objects in the view

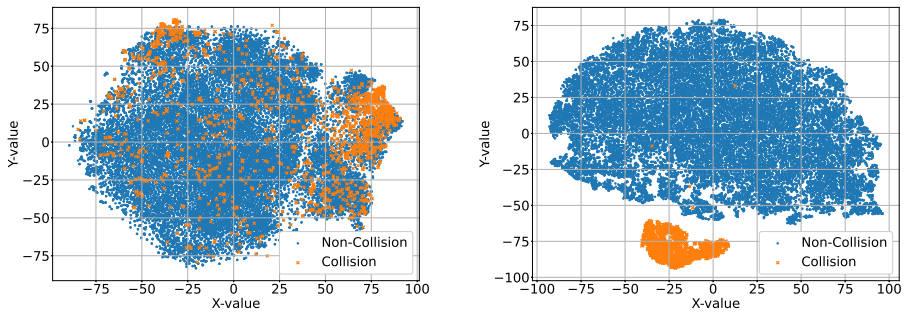
gets reduced after the PCA-based clutter removal step. It should also be noted that this clutter removal step is a requirement only for the baseline method. Our proposed CNN-based solution does not require this additional step of clutter removal since it uses the unfiltered RDMs as input

for the training as well as for the testing stages. Lastly, since all the obstacles get represented by the sparse S matrix, the PCA method successfully filters out the clutter subspace and retains the subspace that is due to obstacle(s), such that reconstructed RDMs can show all and any number of objects within RDMs, as is shown in Fig. 4c.

Baseline Collision Prediction Method. In parametric models for traditional CWS, the time-to-collision (TTC) method is one of the most common methods [25]. For our experiments, we implement the TTC-based baseline method which predicts an impending collision if the measured TTC obtained for every RDM is less than a threshold. We use Equation 4 to calculate one TTC value every RDM, which is given by

$$TTC = \frac{d_r}{r_r}, \tag{4}$$

where d_r is the range and v_r is the relative range rate between the moving node and the nearby obstacle with which a collision is about to happen and hence, for which an alert has to be raised. An impending collision is predicted every RDM for which the respective TTC is less than a threshold, that is δt, as explained in Section 2.4. Using this method for our experiments, we can generate a receiver operating curve (ROC) by computing the probability of false alarm (P_{FA}) and the probability of detection (P_D) for various different threshold values. Note that, if case of two objects, the collision with the closer objects is considered more imminent, therefore, the range for the closer object is selected for the TTC calculations. It should also be noted that the objects are in relative motion with radar, therefore, the calculated range and velocities are relative.



(a) Feature space of the RDMs from an untrained ResNet-18 model.

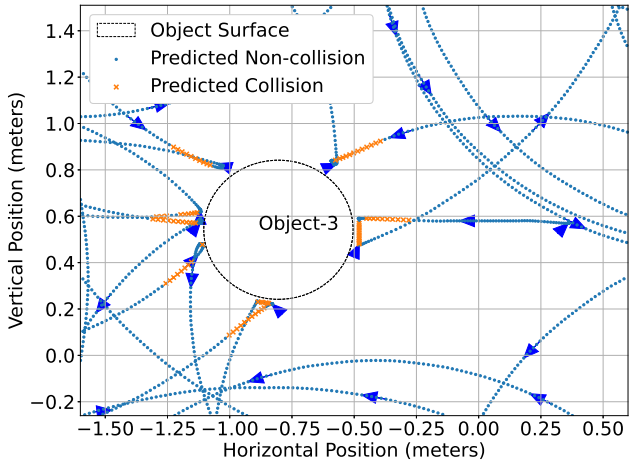
(b) Feature space of the RDMs after training the ResNet-18 model.

Fig. 5. Demonstration of successful training of the ResNet-18 model.

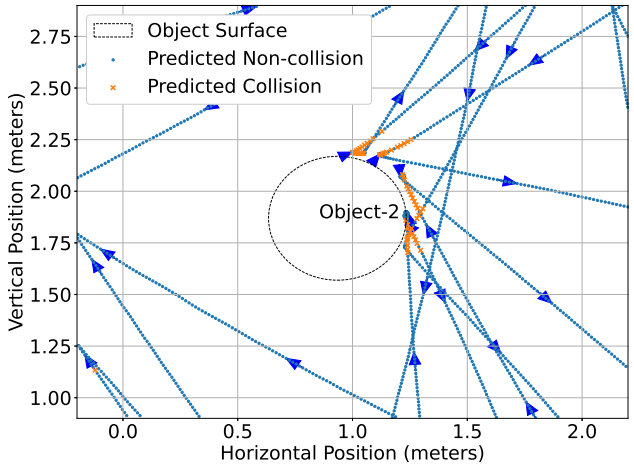
4 ANALYSIS

4.1 Training Performance of Proposed Method

Our goal is to train a classifier which takes RDMs along with their corresponding labels as input for training and predict the label for unseen RDMs. The cross entropy function specified in Section 2.2 is optimized in this process of training the CNN. For training the ResNet-18 CNN as the classifier, Adam [36] is used as the optimizer to update the model's parameters. We also train the MobileNetV3-Small model on the same training dataset and use the RMSProp optimizer [65]. We use D as our training dataset that has all the RDMs from 4 experiments in total: Exp. S1, Exp. S2, Exp. C1, and



(a) Inference results for straight line motion experiment S3 for the moving node near Object-2.



(b) Inference results for curved line motion experiment C3 for the moving node near Object-3.

Fig. 6. Visual representation of our collision prediction model on straight and curved style of trajectories.

Exp. C2. The learning rate for the both the models is set to be 5×10^{-4} and both of the models converge in 30 epochs.

In order to demonstrate that the features space of the RDM dataset is successfully learned by ResNet-18, to be able to distinguish RDMs into two classes based on the dynamically changing distribution of features, we create a visualization of the features space as follows: Each RDM from training dataset D is fed to the 'untrained' ResNet-18, creating a 512-dimensional features space of each RDM in the fully connected layer before the final classification. We then use the t-distributed stochastic neighbor embedding (t-SNE), a popular dimensionality reduction technique, commonly used for data visualization for visualizing high-dimensional features in lower-dimensional space [69]. The parameters for creating t-SNE embeddings using the `sklearn` library are kept as `learning_rate=auto`

and `init=random`. The obtained 2-dimensional t-SNE embedding of the 512-dimensional features space from an untrained ResNet-18 model are shown in Fig. 5a. We see that the untrained ResNet-18 model can not separate the features space of the RDMs into the two desired classes. Now similarly for the 'trained' model, we obtain the 2-dimensional t-SNE embedding of the 512-dimensional features space, as shown in Fig. 5a. We see that the trained ResNet-18 model is now able to linearly separate and cluster the RDMs in the combined dataset into collision and non-collision classes, indicating a successful learning of the features space.

4.2 Inference Performance of Proposed Method

For the inference performance of the trained model on any unseen data, we first provide a visual intuition into the collision prediction system with the help of Fig. 6.

In the two sub-figures, the surroundings of two different obstacles named 'Object-2' in Fig. 6a and 'Object-3' in Fig. 6b are shown, along with the straight and curved trajectories traversed by the moving node in the unseen data from Exp. S3 and Exp. C3, respectively. The blue arrow shows the direction of motion which leads to collision incidents between the stationary obstacles and the moving node. The location of the moving node at the time one RDM gets measured is shown by one dot (•). During the inference stage, an RDM is fed to the trained model and a prediction about the 'impending collision' is made. The steps for which 'no impending collision' is predicted are as represented by blue (•) dots, and the steps for which 'impending collision' is predicted are represented by orange (×) cross. The goal of Fig. 6b is to provide a visual representation of the inference performance made by our trained model, which includes correct and incorrect classifications for both of the classes.

We combine the datasets from Exp. S3 to S5 and Exp. C3 to C5 to use as the test dataset T, that has the RDMs that are not seen by the trained models. For this test dataset T, predictions made by the two trained models (ResNet-18 and MobileNetV3) are collected and compared with the ground truth for collisions according to the IMU's deceleration-based labels. We also investigate the performance of the traditional supervised learning methods such logistic regression (LR), k-nearest neighbors (k-NN), and support vector machine (SVM) using the same datasets D and T for training and testing, respectively. Fig. 7 shows the receiver operating characteristic (ROC) plot for the all the five investigated learning-based classification models as well as the baseline model. The y-axis is probability of detection P_D (or recall R as described in Section 2.5) which measures the fraction of RDMs that are inferred as 'collision' over all the RDMs that were truly labelled as 'collision' during the δt window from moments of collisions, as was done in the labelling scheme of Section 2.4. The x-axis represents the probability of false alarm, P_{FA}, which measures the fraction of RDMs inferred as 'collision' over all the RDMs that were truly labelled as 'non-collision'. The ROC plot shows that ResNet-18 has the highest area under the curve (AUC) with 0.98, outperforming all the other classification methods and thereby, being our investigation's suggested classifier for the collision prediction problem. The reported F1-Score is 0.91. The MobileNetV3 gives the second best AUC (0.88). The traditional supervised learning methods have AUC of 0.78 for SVM, 0.78 for kNN, and 0.74 for LR, while the baseline method of combining PCA-based filtering with the TTC model has an AUC of 0.6.

We further investigate the nature of predictions made by ResNet-18 on the test dataset T, by exploring the prediction accuracy as a function of the time to collision. As seen in Fig. 8, all the false alarms raised by the ResNet-18 classifier are more frequent near the time of the actual collision. This indicates that most of the false alarms (false positives) are close to the time of the threshold of δt. That is to say that the false alarms are less likely to occur when the moving node is further away from objects on the trajectory. Additionally, it is reported that the probability of detection is the nearly consistent when the time of collision is less than the threshold of δt, indicating that the

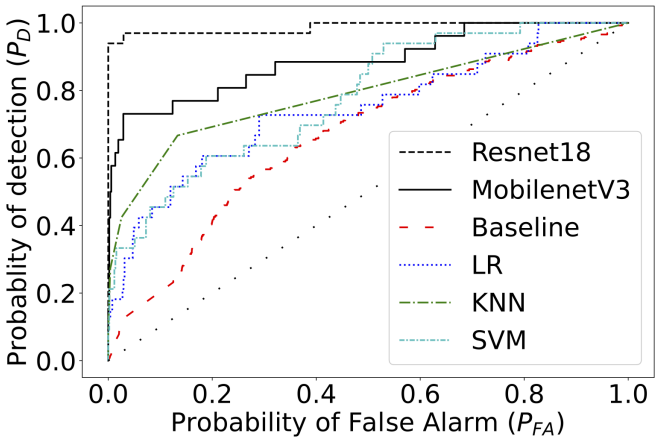


Fig. 7. Performance comparison in the form of probability of detection versus probability of false alarm for various models for the combined dataset D

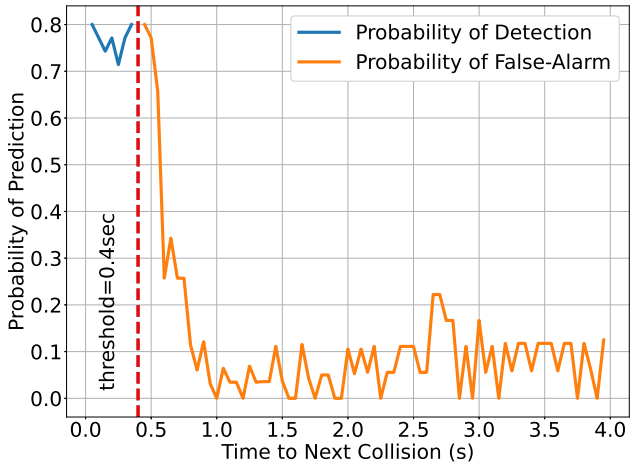


Fig. 8. Probability of detection and false alarm as a function of time to next collision.

classifier is able to detect the impending collisions at the specific threshold of δt as designed for our experiments.

We also investigate the effect of the value of M, that is the number of chirps used to make each RDM on the predictions accuracy of ResNet-18. For this, the radar and IMU measurements from experiments in dataset D and T are processed for varying M values. The frequency of RDM generation is stays the same, that is, the RDM window moves by 50 ms at a time, thereby keeping the dataset size the same for comparison between models processing different RDM window sizes. All the other parameters of measurements s, B, T_c, and N, also stay the same, thus, the direct effect of varying the M is in getting a varying velocity resolution information in RDMs. A larger value of M results in more chirps getting combined to make a RDM, which increases to the processing and

computational cost. For predicting a collision, this implies having a RDM that has chirps of a longer time period before the collision. We report that the AUC score increases with the value of M, as seen in Fig. 9, thereby indicating that having more chirps, which provides more past data and finer velocity information in RDMs, leads to higher learning and decision making performance at the cost of additional computational cost. However, there is an optimal value for M, beyond which the performance degrades or plateaus as the longer duration of RDMs, the older the chirps are in the RDMs. It can be inferred that the accuracy of predictions about future decreases as the age of the data increases because the outdated information can become irrelevant to the immediate future.

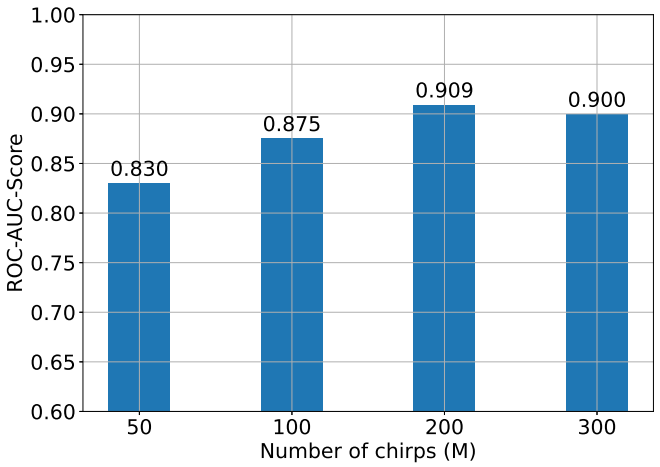


Fig. 9. Effect of various values of M on the inference performance. The ResNet-18 model is trained on RDMs of Dataset D with different window sizes. We see that the chosen value of $M = 200$ for our experiments provided the best AUC performance.

Lastly, we also report the effect of the materials of the obstacles on ResNet-18 model's prediction capabilities. The radar reflections vary with the material of the obstacle but our collision prediction model performed with a P_D of 0.95 or higher, regardless of the material composition of the obstacle as shown in Fig. 10. For the test datset T, we conduct an one-way ANOVA test. With the null hypothesis being that the materials of the object do not affect the probability of detection of our model and the significance level or Type-I error of 0.01, we report a p-value of 0.0254. Thereby, we accept the null hypothesis, indicating that our model's performance does not vary with the material of the encountered obstacles.

4.3 Online Learning Performance

Following the retraining strategy in Section 2.6, the ResNet-18 classifier is investigated in its retraining performance on the previously unseen data. First, a default model is obtained by training a ResNet-18 classifier on one of the straight-trajectory datasets Exp. S1. In order to emulate data drift, a new, unseen dataset from one of the curved-trajectory experiments Exp. C1 is used to obtain the performance of the default model. As presented in Fig. 11, the default model's classification performance on the new, unseen dataset gives an AUC value of 0.69. Next, this curved trajectory dataset is considered as the recent history and 80 : 20 train-test split of this recent history is then used as our training set to retrain the default model. The retrained model generates an AUC of

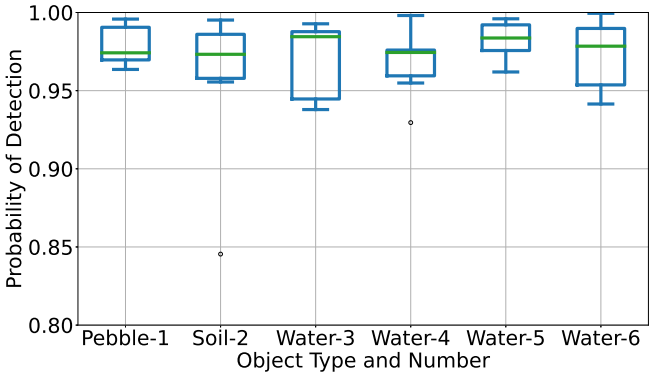


Fig. 10. Impact of obstacle material on F1 score

0.95 on the test set if the full training set is used. It should be noted that large datasets require long training time. Therefore, we apply the uncertainty-sampling technique for selecting uncertainty-based samples from the training set for retraining. We present results for various sample sizes selected for retraining, for example, using top 10% of samples from the full training set that showed highest uncertainty to form as a subset of training set for retraining generates an AUC of 0.83. Similarly, using top 20% and 40% of samples with highest uncertainty generates AUC of 0.884 and 0.885, respectively. We report that the AUCs obtained by retraining on the data subsets of various sizes of the recent history are higher than the AUC from the default model that was not retrained. Furthermore, using uncertainty-sampling enables low latency and data requirements for retraining and provides a better performance than a static, default model.

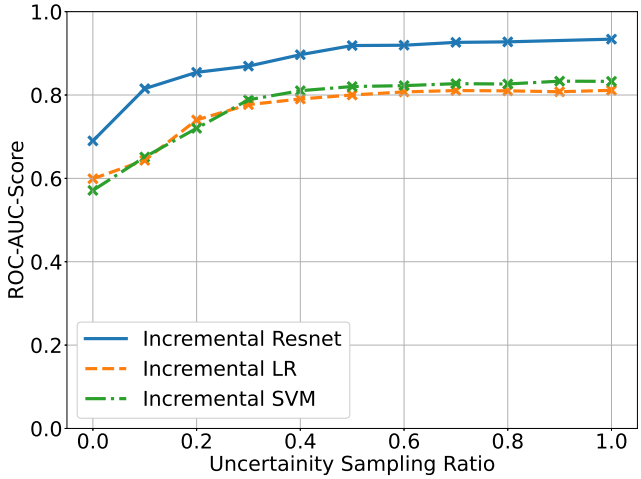


Fig. 11. Retraining performance of ResNet-18 with two additional incrementally learning models for various sizes based on uncertainty sampling.

Online learning with traditional machine learning models. We also compare the performance of our method with the traditional machine learning methods that incrementally update their parameters as new data is collected. Working in mini-batches of the new data, the model's trained parameters from the previous learning are not cleared, but are updated with respect to the new data provided, which is conceptually equivalent to training a model from scratch on a combined dataset. For our binary classification problem statement, we use two incremental learning estimators, a linear support vector machine and logistic regression. Additionally, in order to build a similar architecture to our method for results comparison, we apply the uncertainty sampling principle on these estimators as well. Each of these estimators are given a varying percentage of most uncertain samples of the new data and their model parameters are updated. The performance of our method in comparison with incrementally trained SVM and LR are also presented in Fig. 11. We report that the performance converges at 0.5 of the most uncertain samples from recent history data for incrementally retraining the models. It which shows AUC scores are converge to their optimal after that using only up to half the size of new dataset as training, which will help in reducing the computational load and training time for the models. We report that while traditional CWS methods rely on noise removal techniques to remove unwanted artifacts of motion, our model operates in presence of such artifacts within the RDMs and provides a higher performance metric than the traditional methods. Moreover, since our model is not working with hand crafted features, it selects the appropriate features that continually changing as new data arrives, thereby being robust to data-drift.

4.4 System Design and Limitations

This section discusses how to our system for real-time operation in a resource constrained wearable device.

There are multiple steps in the processing pipeline of the proposed system. First, a base model is trained using prior data collected beforehand. This training is presumably performed in the cloud using a cloud-offloading approach, similar to [54]. Thereafter,

(1) A radar sensor collects a complex-valued signal data vector.
(2) An RDM is generated using a 2D-FFT. An FFT (including a 2D-FFT) is a common operation performed by embedded devices.
(3) The RDM is input to the CNN-based model, performed locally on the embedded processor, to classify the measurement;
(4) Later, when wireless communications are available, the device uploads recently collected and labelled RDMs to the cloud for retraining of the model, and downloading of the new model parameters. To be clear, this communication is not expected to happen in real-time, for example, during a sports game. Instead, the data upload and subsequent model parameter download might occur after the end of a game.

We have used an Intel-Xeon E52666 (CPU) for the processing of radar signals, but this step can be performed on any embedded microprocessor in real-time. For example, in common smart wearables, a 1.6 GHz Quad-A7 Snapdragon is the system-on-chip (SoC), which is reported to generate every second a total of 1200 RDMs that are of similar configuration as our experiments. This corresponds to a latency of 0.8 milliseconds per RDM [41].

To run the deep-learning pipeline on resource-constrained smart devices, several libraries provide run-time optimizations to accelerate the inference performance of models [53]. The overall system design for smart wearables can benefit from exploring the options in terms of available SoCs and their performance [75]. In terms of computational cost, advanced architectures of CNNs, such as

[15], can provide alternatives, however, a comparison to such recent and advanced architectures in terms of accuracy is beyond the scope of this study.

In terms of communication, smart wearables are known to support NFC, WiFi 802.11ac, and Bluetooth LE and can communicate with any a local device with powerful GPUs and CPUs. The number of wearables that can directly connect with the cloud are increasing as the demand grows [52]. Apart from the communication delay of computational off-loading, the overall processing time for one inference is primarily composed of on-device model's inference latency, which is dependent on the model of choice, deep learning libraries, and the hardware.

As an example of the relative inference latencies, we compare the two CNN architectures' inference latencies for one RDM on Intel-Xeon E52666 in Table 3. It can be seen that MobileNetV3 achieves a 16 times reduction in the model latency compared to the widely popular ResNet-50 and 6 times reduction compared to our work's proposed model of ResNet-18. The CNN architectures also differ in their number of trainable parameters. MobileNet architecture has 10 times lower number of parameters to train in comparison to the ResNet architecture. Therefore, we suggest MobileNetV3 for solutions where swifter predictions are required to be made while having better performance than traditional supervised learning methods. Lastly, for implementing our design on smart devices, MobileNet architecture has shown similar inference performance on SoCs such as Snapdragon 430 to 888, thereby making our suggested architecture a suitable option for smart devices and mobile applications [75].

Model	Inference time (sec)	# Parameters ($\times 10^6$)
MobileNetV3-Small	0.0165	2.54
MobileNetV2	0.0608	3.50
ResNet50	0.2545	25.56
ResNet18	0.1032	11.69

Table 3. A comparison of inference latencies of the various CNN architectures in processing one RDM for real-time applications.

The above suggestions are in accordance with the current challenges faced by smart devices, in that the majority of them are resource-constrained in terms of computational and communication capabilities. The system design for a smart wearable such as smart helmet, thus requires innovative schemes to implement our method's pipeline and suggestions to capitalize on available resources.

5 RELATED WORK

5.1 Smart Helmets and Collision Prediction

Two of the most relevant fields that deploy preventative measures similar to our problem statement of predicting and avoiding collisions are of smart helmets and vehicular CWS. Smart helmets have gained substantial interest from the research community in the past decade. The goal of smart helmets can be diverse, most commonly used for collision prevention, by either deactivating the paired vehicle if helmets are not in the vicinity [56] or by deciding that a collision has occurred and call for help after the collision event [46]. They do not actively predict impending collisions before they happen. Therefore, there is scope for incorporating the smart helmets with advanced sensors and algorithms to instead predict the collisions before they happen, which is crucial in altering the wearer of these helmets to prepare themselves upon receiving an alert according to the prediction. Efforts for a camera-based collision prediction algorithm is presented in [8], where future positions are 'extruded' and the decision about a future collision is made. Another camera-based helmet for

avoiding rear end collisions in real-time is presented in [58]. Most of the work in smart helmets utilize cameras or a combination of vibration sensors. To best of our knowledge, solely using radar-based smart helmets for predicting collisions before they happen is novel.

5.2 Vehicular CWS and Radars

Vehicular CWS, on the other hand, are well researched and are adapted as a necessity for the automotive industry. The automotive industry commonly utilizes the 24 GHz and 77 GHz frequency bands in radar systems [22]. Both frequency bands have their advantages and limitations. The choice of sensor for the vehicular CWS is most commonly a combination of camera and radar sensors. Radar performance may be degraded performance due to clutter and noise. In order to minimize the clutter reflections received and noise captured in these radar sensors, filtering techniques such as [6] are actively being researched. However, these filtering methods are not dynamic and need to be updated frequently whenever the environment in which the radars operate changes. Instead, deep learning can be utilized to work with measurements from radars that operating in cluttered and dynamic environments. Multiple published papers that are using deep learning on radar measurements have resulted in successful learning-based solutions in the field of motion and object detection, classification [35], localization and tracking [77], [42], [3], [43], and human health and activity monitoring [14], [55].

5.3 Vehicular CWS and Deep Learning

Although the progress in deep learning have found much application in radar-based sensing, the learning-based algorithms for CWS are still being developed. In [62] and [19], the authors present a non-learning method in the former and a CNN-based classifier in the latter to use a combination of camera and millimeter-wave radar and predict collisions. Instead of using sensing hardware like radars, another option is to provide input information is vehicle-to-vehicle (V2V) or vehicle-to-infrastructure (V2I) communication streams for obtaining range and velocity information for collision predictions. Based on V2I communication, in [39] a collision warning algorithm using a multi-layer perceptron (MLP) neural network called MCWA is presented and reported to have higher performance compared to non-learning methods. A more advanced architecture using VGG-CNN, called RCPM, for a real-time collision detection using trajectory information from videos is presented in [71]. These attempts pave the way for exploring faster machine learning architectures for making swifter predictions for safety critical applications such as collision warning. In our wor, we propose a ResNet-18 based collision prediction method that only uses entirely independent and ego-centric measurements obtained from inexpensive and lightweight FMCW radar and IMU sensors. Thus, our method is an infrastructure-free, self-sufficient solution that operates in cluttered environment and is faster and more accurate than traditional parametric and other learning-based methods.

5.4 Online Learning and Radars

Finally,learning-based collision prediction models should use self-labelled measurements from recent history and continue to learn. Therefore, a related field to explore is that of online learning, which has seen significant research in building algorithms that continuously and efficiently learn new information while retaining previously learned information. Most recently, rehearsal-based approaches for incremental learning have been implemented in which a model maintains a subset of previous examples that are mixed with new samples to update the model [34], [57], [23]. More specifically for radars, in order to operate in changing environmental conditions, cognitive or fully adaptive radars (FAR) can provide a feedback and optimization mechanism for improved system performance through the subsequent measurements [9], [24]. Another approach is of drift

detection with incremental learning, where modification to the learning model occurs only when a change is detected in the environment distribution from which data are drawn, as is presented through a simulation study in [1]. In [74], a SVM-based method of working with hand-crafted features of the environment is presented that extends its underlying model by adjusting its hidden layer and retraining on selected incorrect samples while retaining previously learned information. Incremental learning, thus, is further explored in our work as the framework for our system to continuously learn with the changing environment using samples with highest uncertainties.

6 LIMITATIONS AND FUTURE WORK

6.1 Limitations of the Experiments Conducted

First, we note that the experiments in our work could be extended. For example, our experiments utilize only single-receiver radar measurements. In contrast, the use of multi-antenna radar systems that can beamform to improve the strength of the desired signal by combining signals from multiple antennas, adds the ability to measure angle and reduces the impact of interference. However, it adds to the processing and computational costs and has not been explored in this work. Also, our experiments use 24 GHz radar, and future experiments could explore performance in the 60 and 77 GHz bands which allow higher bandwidth and thus greater resolution. Finally, the speed of our moving prototype node is not representative of the target application of our system, such as sports players. However, radars are capable of capturing different speeds, according to the requirements by the application and radar operation settings, and extended experiments can include different speeds into the measurements.

6.2 Limitations of the Proposed Method

A collision prediction system for a particular application must have classification error rates such that the application needs are met. The proposed system, as presented in this paper, may not meet the needs of all applications. Future research is critical to advance collision prediction methods to meet the real-world usability standards in particular applications.

6.3 Future Work

Recurrent neural networks (RNNs) utilize the sequential nature of the data by using a time-series of multiple data points to make one inference. In our system, the data points are RDMs, and they are indeed sequential in time, being produced one after the previous. Using multiple sequential RDMs as one input to a RNN-based model may have value to explore in order to get improved performance compared to a CNN-based model. As a future direction to explore, other deep-learning architectures such as RNNs and recent modifications in CNNs that have shown to offer a lower complexity and inference latency, can be studied to present a comparative analysis of using raw radar data in predicting collisions. Sensor fusion with other modalities, such as video and lidar, can also be used to provide complementary information that could dramatically improve the classification rates of collision prediction. These methods may be beneficial in improving the increasing a machine learning model's inference accuracy.

7 CONCLUSIONS

In this paper, we present a deep learning-based methodology for predicting impending collisions using radar and inertial sensing. Approaching the problem as a binary classification problem, we study the CNN architectures of ResNet and MobileNet to optimize the cross-entropy loss function using labelled range-Doppler matrices as input and collision prediction labels for impending collisions as the outcome. We report ResNet-18 provided an F1 score of 0.91 and outperformed traditional

supervised learning methods. We also presented a retraining framework for incorporating online learning capabilities to the ResNet-18 CNN architecture in order to make it adaptable to the changes in the input data distribution due to the dynamic environmental conditions such as changes in the style of motion and object materials. Lastly, the system is designed to automate the labelling process by using its inertial sensor to provide real-time acceleration measurements which are used to label the present radar measurements for the continuous, online learning scheme presented in this study.

ACKNOWLEDGMENT

This work is supported by the US National Science Foundation under Grant No. 1622741. We also thank our anonymous reviewers for providing the valuable feedback which helped us in producing the final manuscript.

REFERENCES

[1] Murat Akcakaya, Satyabrata Sen, and Arye Nehorai. 2016. A Novel Data-Driven Learning Method for Radar Target Detection in Nonstationary Environments. *IEEE Signal Processing Letters* 23, 5 (2016), 762–766. https://doi.org/10.1109/LSP.2016.2553042

[2] Daniele Alessandrelli, Andrea Azzarà, Matteo Petracca, Christian Nastasi, and Paolo Pagano. 2012. ScanTraffic: Smart Camera Network for Traffic Information Collection, Vol. 7158. 196–211. https://doi.org/10.1007/978-3-642-28169-3_13

[3] Yasin Almalioglu, Mehmet Turan, Chris Xiaoxuan Lu, Niki Trigoni, and A. Markham. 2021. Milli-RIO: Ego-Motion Estimation With Low-Cost Millimetre-Wave Radar. *IEEE Sensors Journal* 21 (2021), 3314–3323.

[4] Matthias Althoff, Olaf Stursberg, and Martin Buss. 2009. Model-based probabilistic collision detection in autonomous driving. *IEEE Transactions on Intelligent Transportation Systems* 10, 2 (2009), 299–310.

[5] A. Aravkin, S. Becker, V. Cevher, and P. Olsen. 2014. A variational approach to stable principal component pursuit. In *Conference on Uncertainty in Artificial Intelligence (UAI)*.

[6] M. Ash, M. Ritchie, and K. Chetty. 2018. On the Application of Digital Moving Target Indication Techniques to Short-Range FMCW Radar Data. *IEEE Sensors Journal* 18, 10 (2018), 4167–4175. https://doi.org/10.1109/JSEN.2018.2823588

[7] Jonathan P. Aston, Nik Benko, Takara Truong, Alia Zaki, Nathaniel Olsen, Ebsa Eshete, Nathaniel G. Luttmer, Brittany Coats, and Mark A. Minor. 2020. Optimization of a Soft Robotic Bladder Array for Dissipating High Impact Loads: an Initial Study in Designing a Smart Helmet. In *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*. 607–614. https://doi.org/10.1109/RoboSoft48309.2020.9116034

[8] S. Atev, O. Masoud, R. Janardan, and N. Papanikolopoulos. 2005. A collision prediction system for traffic intersections. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 169–174. https://doi.org/10.1109/IROS.2005.1545407

[9] Kristine L Bell, Christopher J Baker, Graeme E Smith, Joel T Johnson, and Muralidhar Rangaswamy. 2014. Fully adaptive radar for target tracking part I: Single target tracking. In *2014 IEEE Radar Conference*. IEEE, 0303–0308.

[10] Francesco Bella and Roberta Russo. 2011. A Collision Warning System for rear-end collision: a driving simulator study. *Procedia - Social and Behavioral Sciences* 20 (2011), 676–686.

[11] Hossein Mehrpour Bernety, Suresh Venkatesh, and David Schurig. 2018. Performance Analysis of a Helmet-Based Radar System for Impact Prediction. *IEEE Access* 6 (2018), 75124–75131. https://doi.org/10.1109/ACCESS.2018.2882768

[12] Bernhard Buchli, Felix Sutton, and Jan Beutel. 2012. GPS-Equipped Wireless Sensor Network Node for High-Accuracy Positioning Applications. 179–195. https://doi.org/10.1007/978-3-642-28169-3_12

[13] N. Chawla, K. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *ArXiv* abs/1106.1813 (2002).

[14] Changhao Chen, A. Markham, Niki Trigoni, Bing Wang, Linhai Xie, and Peijun Zhao. 2020. Heart Rate Sensing with a Robot Mounted mmWave Radar. *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), 2812–2818.

[15] Zhe Chen, Cai Chao, Tianyue Zheng, Jun Luo, Jie Xiong, and Xin Wang. 2021. RF-Based Human Activity Recognition Using Signal Adapted Convolutional Neural Network. (10 2021).

[16] Daniel Daneshvar, Christopher Nowinski, Ann Mckee, and Robert Cantu. 2011. The Epidemiology of Sport-Related Concussion. *Clinics in sports medicine* 30 (01 2011), 1–17, vii. https://doi.org/10.1016/j.csm.2010.08.006

[17] Bram Dil, Stefan Dulman, and Paul Havinga. 2006. Range-Based Localization in Mobile Sensor Networks. *European Conference on Wireless Sensor Networks*, 164–179. https://doi.org/10.1007/11669463_14

[18] Shuya Ding, Zhe Chen, Tianyue Zheng, and Jun Luo. 2020. RF-Net: A Unified Meta-Learning Framework for RF-Enabled One-Shot Human Activity Recognition. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems* (Virtual Event, Japan) *(SenSys '20)*. Association for Computing Machinery, New York, NY, USA, 517–530. https://doi.org/10.1145/3384419.3430735

[19] Lei Fan, Qi Yang, Yang Zeng, Bin Deng, and Hongqiang Wang. 2021. Real-Time Collision Warning and Status Classification Based Camera and Millimeter Wave-Radar Fusion. In *Artificial Intelligence*, Lu Fang, Yiran Chen, Guangtao Zhai, Jane Wang, Ruiping Wang, and Weisheng Dong (Eds.). Springer International Publishing, Cham, 103–114.

[20] E Guerrero-Menéndez. 2018. Frequency-modulated continuous-wave radar in automotive applications. *Guerrero-Menéndez* (2018).

[21] Kimberly G Harmon, Jonathan A Drezner, Matthew Gammons, Kevin M Guskiewicz, Mark Halstead, Stanley A Herring, Jeffrey S Kutcher, Andrea Pana, Margot Putukian, and William O Roberts. 2013. American Medical Society for Sports Medicine position statement: concussion in sport. *British Journal of Sports Medicine* 47, 1 (2013), 15–26. https://doi.org/10.1136/bjsports-2012-091941 arXiv:https://bjsm.bmj.com/content/47/1/15.full.pdf

[22] Jürgen Hasch, Eray Topak, Raik Schnabel, Thomas Zwick, Robert Weigel, and Christian Waldschmidt. 2012. Millimeter-Wave Technology for Automotive Radar Sensors in the 77 GHz Frequency Band. *IEEE Transactions on Microwave Theory and Techniques* 60, 3 (2012), 845–860. https://doi.org/10.1109/TMTT.2011.2178427

[23] Tyler L. Hayes, Nathan D. Cahill, and Christopher Kanan. 2019. Memory Efficient Experience Replay for Streaming Learning. *2019 International Conference on Robotics and Automation (ICRA)* (2019), 9769–9776.

[24] Simon Haykin. 2009. Cognition is the Key to the Next Generation of Radar Systems. In *2009 IEEE 13th Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop*. 463–467. https://doi.org/10.1109/DSP.2009.4785968

[25] John C. Hayward. 1972. Near-miss determination through use of a scale of danger. *Highway Research Record* (1972).

[26] Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. 1322–1328. https://doi.org/10.1109/IJCNN.2008.4633969

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778. https://doi.org/10.1109/CVPR.2016.90

[28] Andrew Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. (04 2017).

[29] C. Huyen. 2022. *Designing Machine Learning Systems: An Iterative Process for Production-ready Applications*. O'Reilly Media, Incorporated. https://books.google.com/books?id=BAy_zgEACAAJ

[30] Jonas Jansson. 2005. *Collision Avoidance Theory: With application to automotive collision mitigation*. Ph. D. Dissertation. Linköping University Electronic Press.

[31] Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intell. Data Anal.* 6 (2002), 429–449.

[32] Lu Jiang, Di Huang, Mason Liu, and Weilong Yang. 2020. Beyond synthetic noise: Deep learning on controlled noisy labels. In *International conference on machine learning*. PMLR, 4804–4815.

[33] Kaggle. 2011. Snapdragon Neural Processing Engine SDKr. https://www.kaggle.com/c/GiveMeSomeCredit

[34] Ronald Kemker and Christopher Kanan. 2018. FearNet: Brain-Inspired Model for Incremental Learning. *ArXiv* abs/1711.10563 (2018).

[35] Woosuk Kim, Hyun-Woong Cho, Jongseok Kim, Byungkwan Kim, and Seongwook Lee. 2020. YOLO-Based Simultaneous Target Detection and Classification in Automotive FMCW Radar Systems. *Sensors (Basel, Switzerland)* 20 (2020).

[36] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2015).

[37] Sotiris Kotsiantis, D. Kanellopoulos, and P. Pintelas. 2005. Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering* 30 (11 2005), 25–36.

[38] Valliappa Lakshmanan, Sara Robinson, and Michael Munn. 2020. *Machine Learning Design Patterns*. O'Reilly Media.

[39] Donghoun Lee and Hwasoo Yeo. 2016. Real-Time Rear-End Collision-Warning System Using a Multilayer Perceptron Neural Network. *IEEE Transactions on Intelligent Transportation Systems* 17 (2016). https://doi.org/10.1109/TITS.2016.2537878

[40] David Lewis, Jason Catlett, W. Cohen, and Haym Hirsh. 1996. Heterogeneous Uncertainty Sampling for Supervised Learning. (1996).

[41] Jaime Lien, Nicholas Edward Gillian, Mustafa Emre Karagozler, Patrick Amihood, Carsten Schwesig, Erik Olson, Hakim Raja, and Ivan Poupyrev. 2016. Soli: ubiquitous gesture sensing with millimeter wave radar. *ACM Trans. Graph.* 35 (2016), 142:1–142:19.

[42] Chris Xiaoxuan Lu, Stefano Rosa, Peijun Zhao, Bing Wang, Changhao Chen, John A. Stankovic, Niki Trigoni, and A. Markham. 2020. See through smoke: robust indoor mapping with low-cost mmWave radar. *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services* (2020).

[43] Chris Xiaoxuan Lu, Muhamad Risqi U. Saputra, Peijun Zhao, Yasin Almalioglu, Pedro Porto Buarque de Gusmão, Changhao Chen, Ke Sun, Niki Trigoni, and A. Markham. 2020. milliEgo: single-chip mmWave radar aided egomotion estimation via deep sensor fusion. *Proceedings of the 18th Conference on Embedded Networked Sensor Systems* (2020).

[44] Nathaniel G. Luttmer, Takara E Truong, Alicia M. Boynton, David Carrier, and Mark A. Minor. 2020. Treadmill Based Three Tether Parallel Robot for Evaluating Auditory Warnings While Running. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 9135–9142. https://doi.org/10.1109/ICRA40945.2020.9196600

[45] Nathaniel G Luttmer, Takara E Truong, Alicia M Boynton, Andrew S Merryweather, David R Carrier, and Mark A Minor. 2022. Impactful Robots: Evaluating Visual and Audio Warnings to Help Users Brace for Impact in Human Robot Interaction. *arXiv preprint arXiv:2207.13835* (2022).

[46] Michele Magno, Angelo D'Aloia, Tommaso Polonelli, Lorenzo Spadaro, and Luca Benini. 2016. Shelmet: an intelligent self-sustaining multi sensors smart helmet for bikers. In *International Conference on Sensor Systems and Software*. Springer, 55–67.

[47] Evangelos Mazomenos, Dirk Jager, Jeffrey Reeve, and Neil White. 2011. A Two-Way Time of Flight Ranging Scheme for Wireless Sensor Networks, Vol. 6567. 163–178. https://doi.org/10.1007/978-3-642-19186-2_11

[48] Michael McCloskey and Neal J. Cohen. 1989. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. *Psychology of Learning and Motivation* 24 (1989), 109–165.

[49] Isar Mostafanezhad, Ehsan Yavari, Olga Boric-Lubecke, Victor M. Lubecke, and Danilo P. Mandic. 2013. Cancellation of Unwanted Doppler Radar Sensor Motion Using Empirical Mode Decomposition. *IEEE Sensors Journal* 13 (2013), 1897–1904.

[50] Optitrack. 2022. Motion Capture Systems. https://optitrack.com/software/motive/

[51] Sujeet Milind Patole, Murat Torlak, Dan Wang, and Murtaza Ali. 2017. Automotive radars: A review of signal processing techniques. *IEEE Signal Processing Magazine* 34, 2 (2017), 22–35. https://doi.org/10.1109/MSP.2016.2628914

[52] Carlos Pereira and Ana Aguiar. 2014. Towards efficient mobile M2M communications: Survey and open challenges. *Sensors* 14, 10 (2014), 19582–19608.

[53] Qualcomm. 2023. Snapdragon Neural Processing Engine SDKr. https://developer.qualcomm.com/sites/default/files/docs/snpe/overview.html

[54] Kiran Rachuri, Cecilia Mascolo, Mirco Musolesi, and Peter Rentfrow. 2011. SociableSense: Exploring the Trade-offs of Adaptive Sampling and Computation Offloading for Social Sensing. *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*, 73–84. https://doi.org/10.1145/2030613.2030623

[55] Tauhidur Rahman, Alexander Travis Adams, Ruth Vinisha, Mi Zhang, Shwetak N. Patel, Julie A. Kientz, and Tanzeem Choudhury. 2015. DoppleSleep: a contactless unobtrusive sleep sensing system using short-range Doppler radar. *Proc. 2015 ACM Intl. Joint Conference on Pervasive and Ubiquitous Computing* (2015).

[56] Mohd Khairul Afiq Mohd Rasli, Nina Korlina Madzhi, and Juliana Johari. 2013. Smart helmet with sensors for accident prevention. In *2013 International Conference on Electrical, Electronics and System Engineering (ICEESE)*. IEEE, 21–26.

[57] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, G. Sperl, and Christoph H. Lampert. 2017. iCaRL: Incremental Classifier and Representation Learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), 5533–5542.

[58] Sudhir Rao Rupanagudi, Sumukha Bharadwaj, Varsha G. Bhat, S. Eshwari, S. Shreyas, B. S. Aparna, Anirudh Venkatesan, Amrit Shandilya, Vikram Subrahmanya, and Fathima Jabeen. 2015. A novel video processing based smart helmet for rear vehicle intimation & collision avoidance. *2015 International Conference on Computing and Network Communications (CoCoNet)* (2015), 799–805.

[59] Burr Settles. 1995. Active Learning Literature Survey. *Science* 10, 3 (1995), 237–304.

[60] Aarti Singh and Dr. Neal Patwari. 2023. 24GhzFMCWRadarIndoorsMobile. https://doi.org/10.7910/DVN/80XS3Y

[61] Aarti Singh and Neal Patwari. 2021. Collision Prediction using UWB and Inertial Sensing: Experimental Evaluation. In *2021 IEEE International Conference on Autonomous Systems (ICAS)*. 1–5. https://doi.org/10.1109/ICAS49788.2021.9551118

[62] Wenjie Song, Yi Yang, Mengyin Fu, Fan Qiu, and Meiling Wang. 2018. Real-Time Obstacles Detection and Status Classification for Collision Warning in a Vehicle Active Safety System. *IEEE Transactions on Intelligent Transportation Systems* 19 (2018), 758–773.

[63] J. A. Stratton. 1941. *Electromagnetic theory*. McGrow-Hill Book Company Inc.

[64] Harry Thornburg. 2022. Adafruit BNO055 absolute orientation sensor. http://ccrma.stanford.edu/~jos/bayes/bayes.html

[65] Tijmen Tieleman, Geoffrey Hinton, et al. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4, 2 (2012), 26–31.

[66] Ivan Tomek. 1976. Two Modifications of CNN.

[67] UCI. 2023. UCI Repository of Machine Learning Databases. https://archive.ics.uci.edu/ml/datasets/statlog+(heart)

[68] UserManual.wiki. 2016. Ancortek Manual. https://usermanual.wiki/Document/AncortekManual.401224494

[69] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605. http://jmlr.org/papers/v9/vandermaaten08a.html

[70] Jorge Vargas, Suleiman Alsweiss, Onur Toker, Rahul Razdan, and Joshua Santos. 2021. An Overview of Autonomous Vehicles Sensors and Their Vulnerability to Weather Conditions. *Sensors* 21, 16 (2021). https://doi.org/10.3390/s21165397

[71] Xin Wang, Jing Liu, Tie Qiu, Chaoxu Mu, and Pan Zhou. 2020. A Real-Time Collision Prediction Mechanism With Deep Learning for Intelligent Transportation System. *IEEE Transactions on Vehicular Technology* PP (06 2020), 1–1. https://doi.org/10.1109/TVT.2020.3003933

[72] Fupin Yao. 2021. Cross-domain few-shot learning with unlabelled data. *ArXiv* abs/2101.07899 (2021).

[73] Hui-Shyong Yeo, Gergely Flamich, Patrick Schrempf, David Harris-Birtill, and Aaron Quigley. 2016. RadarCat: Radar Categorization for Input & Interaction. 833–841. https://doi.org/10.1145/2984511.2984515

[74] Jing Zhang, Lu Chen, Chao Wang, Li Zhuo, Qi Tian, and Xi Liang. 2017. Road Recognition From Remote Sensing Imagery Using Incremental Learning. *IEEE Transactions on Intelligent Transportation Systems* 18, 11 (2017), 2993–3005. https://doi.org/10.1109/TITS.2017.2665658

[75] Qiyang Zhang, Xiang Li, Xiangying Che, Xiao Ma, Ao Zhou, Mengwei Xu, Shangguang Wang, Yun Ma, and Xuanzhe Liu. 2022. A Comprehensive Benchmark of Deep Learning Libraries on Mobile Devices. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 3298–3307. https://doi.org/10.1145/3485447.3512148

[76] An Zhao, Mingyu Ding, Zhiwu Lu, Tao Xiang, Yulei Niu, Jiechao Guan, Ji rong Wen, and Ping Luo. 2020. Domain-Adaptive Few-Shot Learning. *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2020), 1389–1398.

[77] Peijun Zhao, Chris Xiaoxuan Lu, Jianan Wang, Changhao Chen, Wei Wang, Agathoniki Trigoni, and A. Markham. 2019. mID: Tracking and Identifying People with Millimeter Wave Radar. *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)* (2019), 33–40.

[78] Zhiguo Zhao, Liangjie Zhou, Qiang Zhu, Yugong Luo, and Keqiang Li. 2017. A review of essential technologies for collision avoidance assistance systems. *Advances in Mechanical Engineering* 9 (10 2017), 168781401772524. https://doi.org/10.1177/1687814017725246