IMPROVED EXPLAINABILITY THROUGH UNCERTAINTY ESTIMATION IN AUTOMATIC TARGET RECOGNITION OF SAR IMAGES

by

Nicholas Daniel Blomerus

Submitted in partial fulfilment of the requirements for the degree
Master of Engineering (Electronic Engineering)

in the

Department of Electrical, Electronic and Computer Engineering
Faculty of Engineering, Built Environment and Information Technology

UNIVERSITY OF PRETORIA

September 2021

SUMMARY

IMPROVED EXPLAINABILITY THROUGH UNCERTAINTY ESTIMATION IN AUTOMATIC TARGET RECOGNITION OF SAR IMAGES

by

Nicholas Daniel Blomerus

Supervisor:	Prof. J.P. de Villiers
Co-supervisor(s):	W.A.J. Nel and J.E. Cilliers
Department:	Electrical, Electronic and Computer Engineering
University:	University of Pretoria
Degree:	Master of Engineering (Electronic Engineering)
Keywords:	Automatic Target Recognition (ATR), Synthetic Aperture Radar (SAR), eXplainable Artificial Intelligence (XAI), Interpretability, Bayesian Neural Network (BNN), Uncertainty

In recent years, there has been significant developments in artificial intelligence (AI), with machine learning (ML) implementations achieving impressive performance in numerous fields. The defence capability of countries can greatly benefit from the use of ML systems for Joint Intelligence, Surveillance, and Reconnaissance (JISR). Currently, there are deficiencies in the time required to analyse large Synthetic Aperture Radar (SAR) scenes in order to gather sufficient intelligence to make tactical decisions. ML systems can assist through Automatic Target Recognition (ATR) using SAR measurements to identify potential targets. However, the advancements in ML systems have resulted in non-transparent models that lack interpretability by the human users of the system and, therefore, disqualifying the use of these algorithms in applications that affect human lives and costly property.

Current Deep Machine Learning (DML) implementations applied to ATR are still non-transparent and suffer from over-confident predictions. This study addresses these limitations of DML by investigating the performance of a Bayesian Convolutional Neural Network (BCNN) when applied with the task of ATR using SAR images. In addition, the BCNN is used to perform target detection using data

provided by the Council for Scientific and Industrial Research (CSIR). To improve interpretability, a method is proposed that utilises the epistemic uncertainty of the BCNN detector to visualise high- or low-confidence regions in each of the SAR scenes.

The results of this research showed that the performance of the BCNN in the task of ATR using SAR images is comparable to current DML methods from literature. The BCNN achieves a classification accuracy of 93.1 % which is marginally lower than the performance of a similar Convolutional Neural Network of 96.8 %. The BCNN outperformed the CNN when the networks were given out-of-distribution data. The CNN outputs showed over-confident predictions while the BCNN was able to indicate its lack of confidence by using the epistemic uncertainty in combination with the predictive variance in its output.

Using the dataset from the CSIR, uncertainty heat maps were generated that illustrated regions of high- and low-confidence. The regions with the highest uncertainty were located near large collections of trees and areas near shadows. The high-uncertainty incorrect predictions were fed back into the BCNN, and results showed a reduction in overall uncertainty and detection performance.

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Network
ATR	Automatic Target Recognition
BN	Bayesian Network
BNN	Bayesian Neural Network
BCNN	Bayesian Convolutional Neural Network
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CSIR	Council for Scientific and Industrial Research
DAG	Directed Acylic Graph
DNN	Deep Neural Network
ELBO	Evidence Lower Bound
EOC	Extended Operating Condition
FOA	Focus-of-Attention
GPU	Graphics Processing Unit
Grad-CAM	Gradient-Weighted Class Activation Mapping
HLC	High-Level Classifier
IoU	Intersection Over Union
JISR	Joint Intelligence, Surveillance, and Reconnaissance
KL	Kullback-Leibler
LIME	Local Interpretable Model-Agnostic Explanations
LLC	Low-Level Classifier
MAP	Maximum A Posterior
MC	Monte Carlo
MCMC	Markov Chain Monte Carlo
ML	Machine Learning
MLE	Maximum Likelihood Estimator
MSTAR	Moving and Stationary Target Acquisition and Recognition
RCS	Radar Cross Section
PCA	Principle Component Analysis
RMSProp	Root Mean Squared Propagation
ROI	Regions of Interest

SAR	Synthetic Aperture Radar
SOC	Standard Operating Condition
SVMs	Support Vector Machine
VHF	Very High Frequency
XAI	eXplainable Artificial Intelligence
YOLO	You Only Look Once

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

1.1 PROBLEM STATEMENT

Militaries were one of the earliest adopters of SAR technologies and they continue to use it for Joint Intelligence, Surveillance, and Reconnaissance (JISR) activities [1]. SAR analysts use information from the radar sensors to gain intelligence about a region of interest, including the classification of specific targets and activities. In addition, JISR activities are often time-sensitive and have a high risk associated with them since they involve human lives and property. The problem addressed by this work is the Automatic Target Recognition (ATR) of targets embedded in SAR data using an eXplainable Artificial Intelligence (XAI) framework.

1.1.1 Context of the Problem

The earliest record of radar concepts was shown by Heinrich Hertz in 1886 with the discovery of wave reflections [2]. This fundamental radar concept would later be used by the British to detect aircraft at long distances. Key advantages of SAR are the ability to generate images based on radar reflections in all weather conditions, at any time of the day, and the capability of passing through the foliage at frequencies in the Very High Frequency (VHF) band [3]. Owing to these advantages, SAR has been widely adopted in both military and civilian applications. SAR is a radar method that employs active remote sensors to construct 2-D radar images that achieve a high range and azimuth resolution [4, 5, 6]. SAR exploits the relative motion between the sensor and target to form a large aperture and generate a higher azimuth resolution.

The military has made great strides in developing SAR techniques and currently uses them for JISR operations. SAR analysts are the team that manually extracts intelligence pertaining to the recognition of targets and important events. JISR operations are highly time-dependent [1] - it is a process that requires a high level of efficiency and coordination amongst all teams involved. Bottlenecks in

SAR-based JISR operations are apparent, as SAR analysts have to manually sort through hundreds of kilometres of SAR data. With limitations in sensor hardware, additional challenges arise with sub-meter resolution images that result in targets only being represented by a few pixels. Consequently, this leads to a rise in research to automate the process through ATR of targets within SAR data.

Advancements in machine learning (ML) algorithms and increased availability in hardware have made the challenge of ATR more practical. As a result, numerous ML techniques have been applied to the problem of ATR of SAR images [7, 8, 9, 10], with some Deep Neural Network (DNN) implementations achieving remarkable classification accuracy [11]. JISR activities can greatly benefit from the use of ML algorithms for ATR of SAR as it significantly reduces the time to analyse and classify potential targets of interest. In addition to being time-sensitive, JISR tasks may be dangerous since the decisions made have a direct effect on human lives. However, current state-of-the-art deep learning (DL) algorithms have become "black-box" models that make decisions without any methods of explanation. This has caused end-users to question the reasoning of these algorithms, especially in applications where lives are at risk. The emerging field of XAI aims to remedy this problem by producing ML algorithms that are explainable, interpretable, and facilitate trust in the models' output.

1.1.2 Research Gap

Current state-of-the-art ATR of SAR methods primarily focus on classification accuracy and not explainability and transparency. JISR activities require a high degree of confidence from the ML algorithms to be trusted by the user. Therefore, improvements need to be made to adjust the machine learning algorithms applied to ATR to make them adhere to XAI principles, and for them to explain their decisions as well as facilitate trust in the decision-making process.

A significant challenge that current state-of-the-art ML algorithms face is over-confident predictions [12]. For mass adoption of ML algorithms in the context of ATR of SAR to occur, additional measures are needed to convey realistic confidence in decisions. Furthermore, there is a lack of research into the quantification of uncertainty in the prediction for ATR methods using DL. A measure of uncertainty would decrease the number of over-confident predictions made and, therefore, increase the users' trust in the ATR system.

1.2 RESEARCH OBJECTIVE AND QUESTIONS

Traditional DNNs have proven to achieve state-of-the-art performance in numerous classification activities [13]. However, they are still not suitable for deployment in JISR applications and need to be

supplemented by the addition of XAI techniques. To combat the over-confident predictions made by DNNs, the study investigates the application of a BCNN on the ATR using SAR images. The BCNN has the advantage of uncertainty estimation that eliminates over-confident predictions and allows the user to assess the network's confidence in its output decision[14]. To address the problem of ATR of SAR images using an XAI model, the following research questions are posed:

1. Does the use of a BCNN and the interpretation of uncertainty add transparency and explainability? The end-user should have insight into why the network made a particular classification. This promotes trust and would help in the adoption of such technologies.
2. What is the trade-off in classification performance between traditional DNNs and XAI models? Typically, DNNs achieved the best classification performance at the cost of non-transparency. It is important to investigate the trade-off of higher classification performance for increased explainability and transparency.
3. How reliable are uncertainty estimations from BCNNs? Uncertainty estimations are crucial for the adoption of BNNs as they are the entire premise of Bayesian learning in neural networks. Thus, it is imperative that an investigation be conducted to ensure that the uncertainty is reliable.
4. Can the BCNN be used to perform target detection of various SAR scenes? Target detection would enable a partial ATR system. Therefore, it is critical to investigate the target detection capabilities.
5. Can the uncertainty estimates be used to improve the BCNN's target detection performance? An advantage of the BCNN is its ability to quantify uncertainty, and these uncertainty estimates may be leveraged to penalise high-confident incorrect predictions.

1.3 APPROACH

The research procedure is split into four distinct subsections: data preparation, BCNN and uncertainty estimations, target detection, and integration.

1.3.1 Data Preparation

The dataset used for this research was provided by the Council for Scientific and Industrial Research (CSIR) and consists of SAR measurements taken of the same scene. Each scene may contain numerous buildings, vehicles, and natural terrains and, therefore, the data preparation begins with the creation of the dataset. First, the targets in each scene are identified using data logs captured for each run. These samples are used to generate the ground-truth data. Once each target has been correctly identified, the target is cropped to a specific image size and the pixel values are captured and stored as an image

in its corresponding class. In addition to the dataset provided by the CSIR, another more frequently used dataset is also used. The Moving and Stationary Target Acquisition and Recognition (MSTAR) dataset consists of SAR imagery of ten targets at different elevations captured at X-band frequencies. This dataset is suitable since it has a high resolution of 1m. This dataset has been used in [4], [15], [16], [17], and [18] and is employed to make direct comparisons with ATR of SAR implementations from recent publications. Lastly, both datasets are split into training, validation, and test sets. As the focus is on explainability, minimal pre-processing methods are applied as it would affect classification performance.

1.3.2 BCNNs and Uncertainty Estimations

To address the over-confident prediction made by traditional DNNs, a BCNN is used with the capability of exploiting uncertainty estimations to reduce over-confident outputs. The following method is used:

1. Investigate various BNN methods.
2. Explore the feature space of the training set using data analysis techniques.
3. Investigate the output of the convolutional layers and write up results for publication.
4. Evaluate the BCNN using state-of-the-art DNN architectures and perform a comparison between the BCNN and standard DNN using performance metrics such as classification accuracy and F-score [19].
5. Investigate uncertainty estimates of the BCNN employing the epistemic uncertainties by using SAR images that the BCNN was and was not trained on.
6. Explore methods for transforming the uncertainty of the BCNN from epistemic into interpretable representations.

1.3.3 Target Detection

In this work, an investigation is conducted to determine if the BCNN can be utilised for the detection of targets in a SAR scene. Metrics are used to evaluate the target detection performance of the BCNN detector. In addition, another investigation is performed to determine if the uncertainty estimates can improve the detection performance. The following method is used:

1. Investigate the target detection capabilities of the BCNN using the MSTAR dataset as targets and samples of the environment as clutter.

2. Perform a comparison of the target detection between a similar CNN and the BCNN using the same training set.
3. Experiment with the feedback of highly confident incorrect samples during the BCNN training process. JISR tasks often require a high degree of confidence, and the feedback of high confident incorrect samples might reduce the probability of these predictions.

1.3.4 Integration

Lastly, the components of the BCNN and the detector is integrated using the following steps:

1. Integrate the target detection with the uncertainty estimates and explore the correlation between detected targets and the uncertainty estimates.
2. Investigate the influence of the uncertainty heat maps on the explainability of the BCNN.

1.4 RESEARCH GOALS

The primary objective of this study is the ATR of targets embedded in SAR images using an XAI framework to improve the trust between ML algorithms and their users. The use of a BCNN addresses over-confident decisions made by DNNs through quantifying uncertainties. Thereafter, the models' uncertainty is presented in a format that allows the user to identify regions where the model is unsure and that may require human assistance. This provides users with an additional layer of information regarding the models' confidence and, therefore, fosters increased trust.

1.5 RESEARCH CONTRIBUTION

A comprehensive comparison is presented between the current state-of-the-art ATR of SAR methods from existing publications and a BCNN with a similar architecture. This investigation provides insight into the possible trade-offs between the standard CNN and the Bayesian alternative.

Experiments are conducted to perform target detection using the BCNN. Uncertainty estimates are utilised to provide additional information to the user conveying the confidence of the model over the SAR scene. Lastly, an investigation is conducted using high-uncertainty incorrect detection to further improve the detection and uncertainty estimates of the model.

1.6 RESEARCH OUTPUTS

A conference paper has been accepted for publication at the 24th International Fusion Conference on 3 November 2021. The conference paper is titled: "Improved Explainability through Uncertainty Estimation in Automatic Target Recognition of SAR Images" [20].

1.7 OVERVIEW OF STUDY

This thesis contains six chapters, with every chapter detailing each stage of the work performed. A summary of each chapter is provided in this section.

Chapter 2 presents a comprehensive literature study. A taxonomy of the most current ML algorithms is shown as well as a detailed comparison of the current state-of-the-art ATR in SAR methods. In addition, the chapter discusses an overview of the different uncertainty estimation methods in neural networks. Lastly, it provides an overview of the emerging field of XAI with post-hoc methods for explainability and interpretability.

Chapter 3 provides the theoretical foundation of the BNN, which includes basics on neural networks and CNNs. The chapter illustrates the derivations for the probabilistic modelling that is used for backpropagation training and uncertainty estimation.

Chapter 4 details the BCNN implementation as well as an overview of the implementation to perform the target detection. This chapter concludes by establishing the performance metrics for both classification and target detection.

Chapter 5 contains the results of the classification and the target detection. Detailed comparisons are made between the BCNN and CNNs using similar architectures.

Chapter 6 details the conclusions from the study and how it addresses the problem statement. Lastly, it mentions future works and recommendations.

CHAPTER 2 LITERATURE STUDY

2.1 CHAPTER OVERVIEW

This chapter provides an overview of the literature study for this dissertation. In Section 2.2, a taxonomy is presented on the current state of ATR with SAR images as well as a detailed comparison of various ATR implementations from recent publications. Section 2.3 summarises a taxonomy on the uncertainty estimation methods with an emphasis on Bayesian techniques. Lastly, Section 2.4 provides a brief introduction to XAI.

2.2 ATR WITH SAR

ATR manipulates data from various sensors directed at a particular scene and predicts the targets that have been detected and recognised from the provided data. When ATR originated in the early 1980s, there were two competing fields used to design ATR systems - signal processing and computer vision. With advancements in available hardware and the emergence of artificial intelligence (AI), computer vision has become the dominating field of interest. The problem of ATR has now become how to teach a machine to perform tasks that humans perform automatically [21].

To reduce the vast amount of information initially provided by the SAR images, the ATR system is divided into three distinct stages as shown in Figure 2.1.

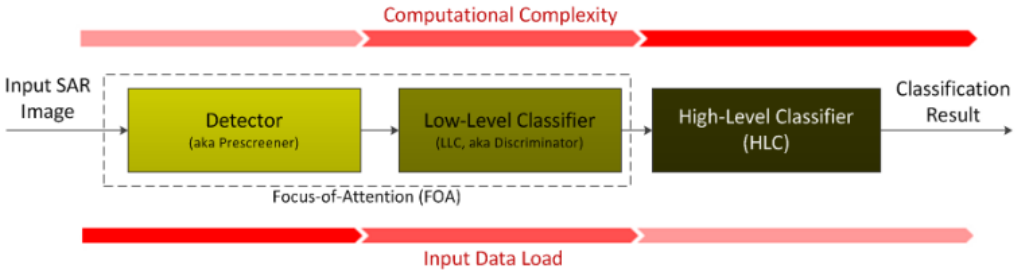


Figure 2.1. Block diagram for an end-to-end ATR of SAR system. Taken from [22], ©IEEE 2016.

The stages are detection (pre-screening), low-level classification (target discrimination), and high-level classification. From Figure 2.1, it is noted that the computational complexity increases after each subsequent stage and the input data decreases after each stage. The combination of the detector and discriminator stages forms the Focus-of-Attention (FOA). The FOA is the main interface between the input data and candidate targets. A key objective of the FOA is the efficient manipulation of input data from the input of the detector to the input of the high-level classifier.

The detection stages are tasked with allowing candidate targets through to the next stage and the elimination of clutter and speckle noise [4]. In addition, the detector performs enhancements to the target region and suppresses the effects of background recognition. In this stage, pre-processing methods are required to decrease the input data load, and this is achieved through various techniques such as image segmentation [23].

The two main objectives of the discrimination stage are the estimation of the position of candidate targets and feature discrimination. In feature discrimination, features are extracted from the targets of interest and the effective information contained in the input SAR image is combined. This is then converted into a feature vector that is fed into the subsequent classifier. Some ATR systems have an additional classifier in the discrimination stage that is only trained on target features, or target features and clutter. The final stage of the ATR system is the high-level classifier (HLC). This stage receives candidate targets and performs the classification of targets into their respective classes. There are a plethora of classification techniques for both the LLC and HLC stages. The main categories are feature-based, semi-model-based, and model-based techniques. Feature-based techniques are dependent on a specific feature to describe each target. The feature is either obtained through feature extractions in the discrimination stage, or they are obtained from image target templates of the target. Feature-based methods operate on the principle that the features of each target from the various classes are situated in distinct separable regions within the feature-space of the training data [22]. Model-based techniques differ from feature-based in terms of the order in which operations are performed. Image recognition starts with the feature extraction from the SAR input data. Afterwards, a model of the targets is used for comparison between the extracted features. The key difference between feature-based and model-based techniques is that feature-based techniques attempt to extract numerous characteristics of the target, which are used to train the classifier, while the model-based methods perform feature hypothesis tests using the pre-defined models of the targets. An advantage of model-based techniques is that it mitigates the difficulties in feature extraction by leveraging prior knowledge into the pre-defined models. The

semi-model methods differ from both feature-based and model-based methods. It does not rely on feature vectors for classifier training and differs from model-based methods as it does not use the same procedure of feature hypothesis verification [24].

Feature-based methods are less complex than model-based methods. This is in part due to the lack of intelligence and reasoning of feature-based methods when compared to model-based methods which have an inherent knowledge built into the system. This enables the model-based methods to be more robust under different operating conditions. Thus, model-based systems are more accurate than feature-based systems. The main disadvantage of model-based systems is the level of complexity required to design them as well as the increase in computation time. There is a balance between accuracy and complexity, and a trade-off must be made between them.

2.2.1 Categorisation of ATR Methods for SAR Images

This section contains a survey of the different methods for learning techniques. Various ATR systems of SAR data are selected from literature and are categorised based on their methods. Owing to the vast amount of published work on the topic of ATR of SAR, only the systems that utilise feature-based techniques were selected, as they are the main focus of this study.

The feature-based techniques reviewed in this study fall under the class of supervised learning. Features are extracted from the input SAR images and converted into feature vectors. The classifier then allocates the feature vector to its respective class. Numerous learning techniques facilitate the learning of the discriminate features, and an overview of these techniques is shown in Figure 2.2. For this research problem, the learning techniques contained in the dotted region are the main focus.

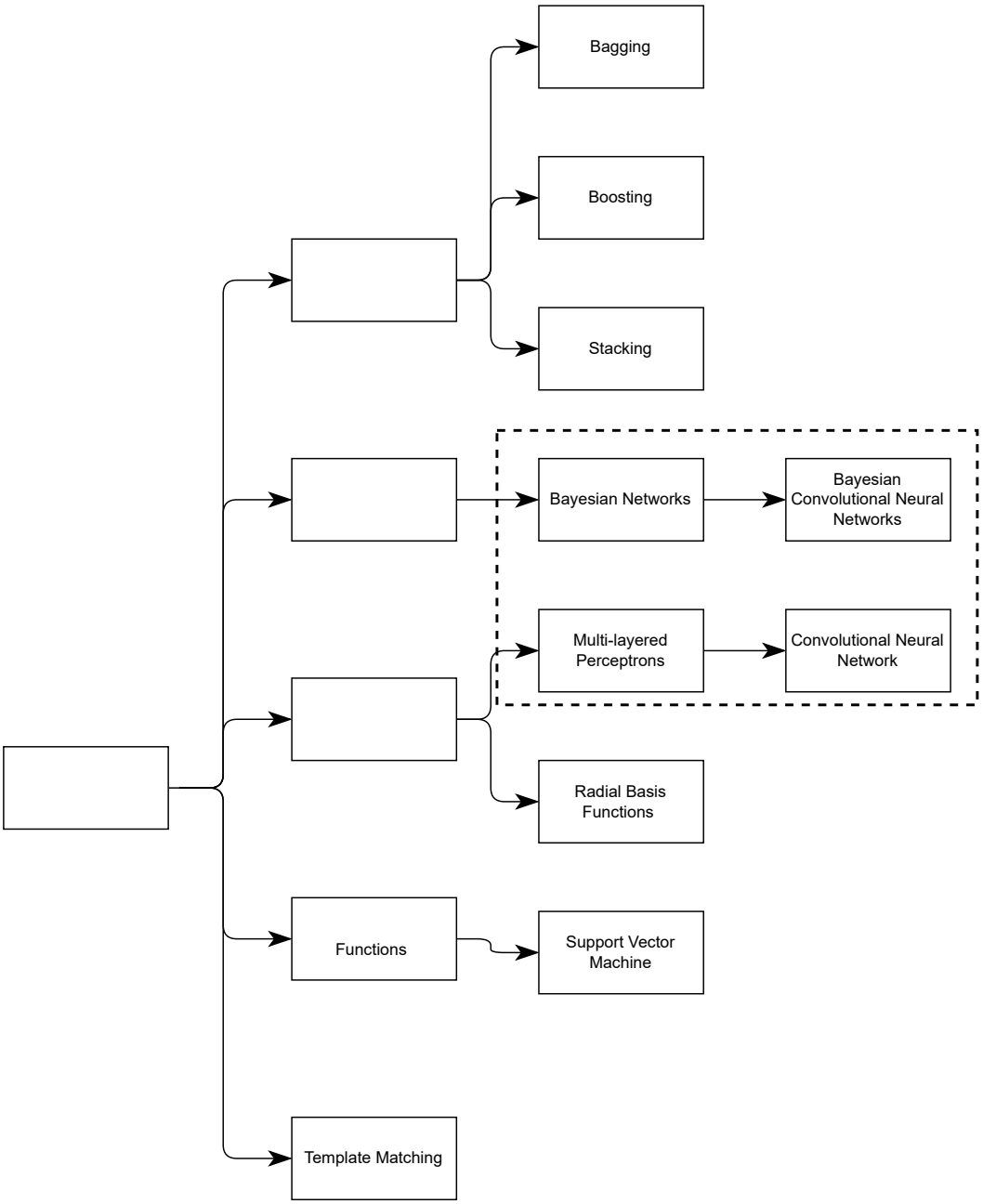


Figure 2.2. Taxonomy of the numerous feature-based learning techniques. This paper focuses on the learning techniques contained in the dotted rectangle.

A background of each of the main sub-categories is provided, with carefully selected published works that use each method. Additional information is provided detailing the benchmark data to allow for a better understanding of the performance of each implementation.

2.2.1.1 Benchmark Dataset

For comparison purposes, the performance methods that use the MSTAR dataset were selected. The MSTAR dataset is a common dataset used throughout the literature and is an appropriate benchmark dataset from which the performance of each method can be evaluated.

The MSTAR dataset consists of two collections taken from the Sandia National Laboratory SAR sensor, with both magnitude and phase information. The first collections of SAR images of the T-72 (T-72 tank), BMP2 (infantry fighting vehicle), and BTR-70 (armoured personnel carrier) were each collected at 15° and 17 ° depression angles. The second collection contains SAR images of the 2S1, BDRM-2, BTR-60, D7, T62, ZIL-131, and ZSU-23/4 taken at 15°, 17°, and 30° depression angles. Each sample was 128x128 pixels in size and consisted of the magnitude data. Samples from the MSTAR dataset are shown in Figure 2.3 where (a) and (b) are optical images of BMP2, BTR70, T72, BTR-60,2S1, BRDM2, D7, T62, ZIL-131, and ZSU23/4, and (c) and (d) are the corresponding SAR images for each target.

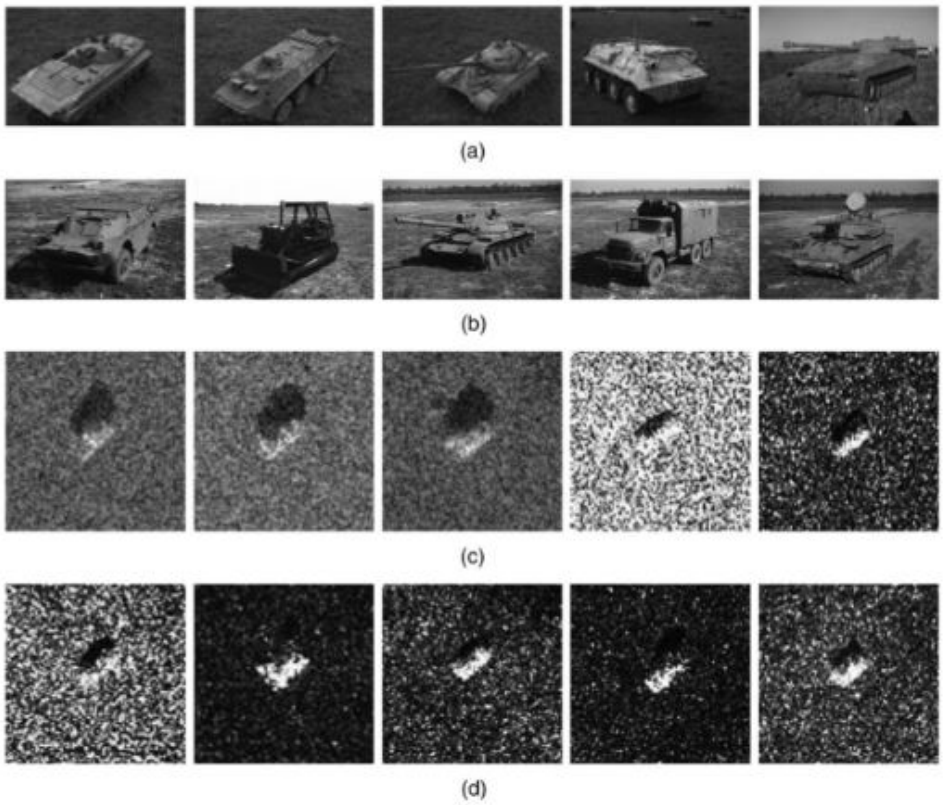


Figure 2.3. MSTAR training samples. The MSTAR dataset contains a total of ten targets of various vehicle types.

The examples below were evaluated using the target configurations, namely, Standard Operation Conditions (SOC) and Extended Operation Conditions (EOC). SOC consists of ten classes with a minimal variation in depression angle of 15° and 17°. EOC includes the same ten classes, with the addition of variation in depression angles (maximum variation of 45°), noise variation (range from -10 to 10 dB), occlusion variation (occlusion levels up to 50% of the target), and resolution variation (resolution range from 0.3m to 0.7m). Given that SOC only has a slight variation in depression angle, the performance metrics achieved were higher when compared to the EOC evaluations. Assume that each example was evaluated using the MSTAR dataset unless stated otherwise.

2.2.1.2 Template-Matching

This is the simplest method that requires the lowest computational complexity. Recognition is achieved through the matching of SAR data with stored templates of a range of variations of each target. The appropriate target is selected using metrics of similarity between the offline templates and input data, such as correlation and matching filters. An example of template-matching is presented in [6], where the attributed scattering centres (ASC) are matched to binary target regions. Image segmentation is performed using basic thresholding to obtain the binary target regions. Then, the binary region is correlated to the stored template region. Lastly, weighted scores are combined from the correlations to determine the appropriate class. Average classification accuracy of 98.34 % was achieved using the MSTAR dataset under SOC. In addition, the system maintained state-of-the-art classification rates under configurations with large depression angles until 30° and partial occlusion of 20 %.

2.2.1.3 Linear Discriminate Functions

Support vector machines (SVMs) are linear models that can be used for either classification or regression. SVMs' main objective is to determine the optimal hyper-plane that separates two classes. Through the maximisation of the distance between the hyper-plane and the input data, the upper bound of the expected error is reduced. The optimal hyper-plane divider is determined by the minimisation of the squared norm function of the distance between the different classes [9]. For most datasets, the input space is not separable, meaning there are no hyper-planes that separate the classes. Thus, SVMs perform a higher dimensional mapping of the input such that there exists a hyper-plane that separates the newly mapped data. This is implemented through the kernel function. The kernel function describes an inner product in feature space which can be higher dimensional. The kernel function represents the mapping of the input to an often higher dimensional feature space [25].

An example of an SVM implementation is presented in [16]. The training of the SVM used a Gaussian

kernel, with the kernel size selected based on the mean Euclidean distance between the input data. From the results obtained, the SVM achieved a best classification accuracy of 90.99 % while only using three classes of the MSTAR dataset under SOC. Moreover, the variability of the aspect angle had a large effect on the classification accuracy and a trade-off was made between the sector size and the training accuracy.

2.2.1.4 Neural Networks

An artificial neural network consists of a large number of connected artificial neurons. The neurons are separated into three categories - input layer, hidden layer, and output layer. A neural network uses the linear combination of fixed non-linear basis functions [25]. The linear combination consists of the summations of the contributions from each neuron as an input propagates through the model. This is known as forward propagation. The contribution from one neuron is a multiplication of the weight between the neuron to a particular neuron and its output. These weight values are initialised to a set of normally distributed values. To train the neural network, forward propagation is performed to determine the error at the output layer, and then an optimisation algorithm is applied to minimise a specific error function. After each iteration, the weight values are adjusted such that the error function is minimised. This is repeated until the model meets the required error threshold. Backpropagation is a common training algorithm for training feedforward neural networks. In backpropagation, the error is backpropagated through each layer of the model to minimise the error function. The most common backpropagation technique is gradient descent, which makes incremental adjustments to the weight values after every iteration. Gradient descent achieves high accuracy but can be slow when the input data is large. In addition, non-convex weight spaces may struggle to find a global minimum. Current neural networks use a combination of adaptive learning rate and momentum [8] to negate the pitfalls of plateaus and saddle points in the weight space and allow for faster training times.

One of the most well-known neural networks is the convolutional neural network (CNN). The first CNN was developed by Kunihiko Fukushima [26]. Through the implementation of the convolution operation, CNNs are able to extract geometric features from a given input and are known for being self-organised, since these networks are capable of performing feature extraction when given unprocessed data. The development of the CNN was a massive advancement in the field of image classification and led to the models achieving record-breaking image classification performance. This improvement was a result of the structure of the network being able to accept a large high-dimensional input and reduce the dimensionality through the implementation of multiple convolutional and pooling layers.

This architecture leads to what is known as deep learning models. They are defined to have multiple hidden layers whereas a standard neural network consists of a single hidden layer [27]. In addition, improvements in Graphics Processing Unit (GPU) and Central Processing Unit (CPU) speeds allowed for further advancements in deep learning models with increased efficiency and shorter training times compared to standard deep neural networks.

A representative example of a CNN is given in [17]. The ATR system uses image segmentation methods based on morphological operations in order to reduce the background recognition. The CNN is implemented using a large-margin softmax batch normalisation structure which increases separability in the SAR data after pre-processing. Owing to the structure of the network, an increase in the convergence rate is recorded as well as less proneness to overfitting. The method was capable of achieving a classification accuracy of 96.44 % on the MSTAR dataset under SOC while being robust in EOC, such as large depression angles and configuration variants.

2.2.1.5 Stochastic Techniques

Statistical learning algorithms use probabilistic models that produce the probability that a given input belongs to a class. The most common method is the Bayesian network (BN), a class of probabilistic graphical models. Graphical models provide a natural way to represent and visualise the relationship between variables. The main difficulty in statistical models is the large number of random variables needed to model a problem. Often, it is not possible to capture every data sample, leading to insufficient data needed to fully describe the conditional dependence between the random variables. Even with the information given, it is not feasible to calculate the entire conditional probabilities for each occurrence. Bayesian networks represent the relationship between a set of random variables and its conditional probability distributions through a Directed Acylic Graph (DAG). An example of a DAG is shown in Figure2.4. The random variables are represented by nodes denoted as A, B, C, and D. The directed arrows represent the relationships between the nodes.

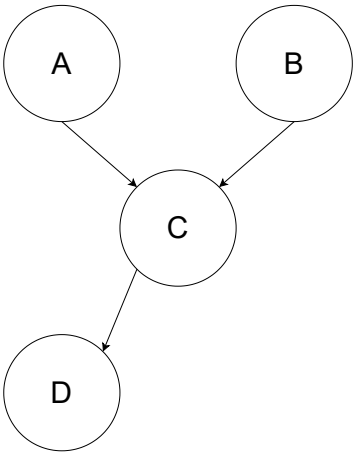


Figure 2.4. Bayesian network structure example.

Nodes that have a one-to-one correspondence are conditionally dependent on each other and nodes that have no connection are conditionally independent of each other.

Bayesian inference is used in order to learn the class probabilities when given data of the BN. The conditional relationships are calculated for each node given its parent node. The joint distributions are determined by the multiplication of the local conditional distributions. This is calculated as follows:

$$P(\mathbf{X} \mid \boldsymbol{\theta}) = \prod_{i=1}^{n} P(x_i \mid pa_i, \boldsymbol{\theta}_i), \tag{2.1}$$

where $\mathbf{X}$ represents the different classes, x_i denotes both the variable and corresponding node, pa_i denotes the parent node of x_i, and $\boldsymbol{\theta}$ is the vector parameters. For the application of a classifier, $\boldsymbol{\theta}$ is the training dataset. Determining the structure of a BN often requires specialist knowledge of the problem. An additional method of determining the structure of the BN is using statistical tests, such as the Chi-squared test, by learning the conditional independence interaction of the nodes in the dataset [10].

An example of a BN is given in [28] for ATR. The feature-based ATR system used Bayesian inference to train the classifier. The BN makes use of a two-level probabilistic model. The two target states used were the target class and azimuth angle, which were quantified from 0° to 360° at a step angle of 48°. The target states were selected in order to obtain good features of the targets at as many angles as possible. The probabilistic inference was used after the conditional probabilities were calculated. Decision-making rules are used to classify each target using known information about the input's

azimuth angle. The BN achieved a best accuracy of 88% using five ground-based targets on a dataset from the Massachusetts Institute of Technology (MIT). The ground targets were all vehicles ranging from a van, truck, and bulldozer.

2.2.1.6 Ensemble Learning

Ensemble learning utilises a finite number of different learning methods. Ensembles use the combination of the outputs from multiple learning algorithms to improve the prediction capabilities. The main ensemble methods are bagging, stacking, and boosting. Bagging methods construct numerous similar models from different sub-samples of the same training dataset and average the predictions in order to improve the classification performance. Stacking methods utilise numerous different models trained on the same training dataset. A separate model is trained on the prediction of the different models in order to better learn to aggregate the predictions. Boosting refers to methods that utilise weighted averages to strengthen weak classifiers. An example of this is the AdaBoost algorithm discussed in Figure 2.5.

An example of a method utilising ensemble methods is given in [15], where a novel pose rectification and image normalisation process is introduced which reduces the variations of the input samples before the feature extraction process. To extract highly discriminative features from ground targets, wavelet decomposition techniques are used. This technique allows for a rich feature set to be extracted that consists of horizontal and vertical edges. Dimensionality reduction is performed to retain the most discriminative features. Decision tree classifiers are utilised to discriminate the features. A statistical analysis of the input data is used to train each base discriminate tree classifier. Additionally, ensemble learning techniques are employed. The ensemble method used is AdaBoost. The principle of boosting is shown in Figure 2.5. In this instance, each model is represented by a decision tree. Fundamentally, the AdaBoost algorithm aims to construct a strong classifier from the linear combination of weak classifiers. Training starts with the strong learner initialising sample weights to an initial number of models. The weights are optimised after each iteration and additional models are added until the stop criteria are met.

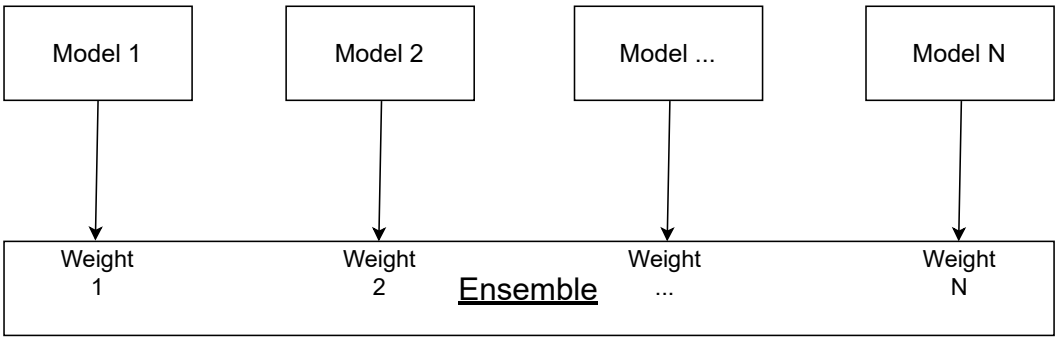


Figure 2.5. Illustration of AdaBoost algorithm.

The method in [15] achieved an average classification accuracy of 97.5% using a feature dimension of 75 on the MSTAR dataset under SOC.

An example of ensemble methods is given in [18]. The ensemble technique consists of a fusion of a CNN and SVM for ATR using the MSTAR dataset. The structure of the CNN consists of three convolution layers each followed by pooling layers and the output layer is fed into the SVM. The SVMs implement the structural risk minimisation principle. For the classification of all ten classes, ten separate SVMs were trained with each following the one-versus-all principle. The one versus all principle divides a multi-class classification into a single binary classification task per class. The main motivation for the combination of a CNN and SVM was to exploit the feature extraction properties of the CNN and reduce the total number of inputs, thus, also improving the generalisation capability. This reduces the total computation time when compared to deep learning models. A best classification accuracy of 98.56% was achieved using the MSTAR dataset under SOC.

2.2.1.7 Comparison of State-of-the-Art ATR SAR Approaches

A comparison of feature-based techniques that have been directly applied to ATR using SAR images is shown in Table 2.1 .

Table 2.1. Comparison of Various Feature-Based ATR of SAR Techniques

	Refs	Classifier Method	Dataset	Features	Classification Accuracy
Template-Matching	[6]	Template-Matching using weighted scores	MSTAR	Binary target regions	98.34 %
Linear Discriminate Functions	[15]	SVM using Gaussian Kernel	MSTAR while only using three targets	Image fed into pose estimator The pose of the image is used as the feature vector	90.99 %
Neural Network	[17]	CNN	MSTAR	Image segmentation is performed using morphological operations	96.44%
	[4]	DCNN	MSTAR	Image segmentation is performed and super resolution image obtained using Generative Adverserial Network	99.31%

Table 2.1. (Continued) Comparison of Various Feature-Based ATR of SAR Techniques

	Refs	Classifier Method	Dataset	Features	Classification Accuracy
Statistical Techniques	[28]	BN	MIT Lincoln Laboratory Target	Total of twelve features used, such as standard deviation, fractical dimension, and weighted-rank fill ratio	88 %
Ensemble Learning	[18]	Fusion of CNN and SVM	MSTAR	Images are converted in dB and data augmentation is performed	98.56%
	[15]	Ensemble of decision trees using AdaBoost	MSTAR	Combination of texture and edges	97.5 %

2.3 UNCERTAINTY ESTIMATIONS IN NEURAL NETWORKS

This section provides an overview of the various methods for quantifying uncertainty in neural networks as well as the different ways in which uncertainty is introduced and propagated from the data acquisition to the prediction stages. The recent advancements in DNNs have allowed for the application of these models in numerous research domains. Earlier examples of this can be observed in several research fields, in particular, speech recognition [29], natural language processing [30], and computer vision [31]. Their state-of-the-art performance has led to applications in more challenging real-world tasks such as medical image analysis [32], surveillance [33], and autonomous driving [34]. However, for sensitive tasks where human lives are affected, their deployment is severely limited. This is owing to the following key factors:

- The “black box” nature of DNNs that are not interpretable and lack sufficient transparency when making predictions [35].
- Lack of dependable confidence measures in the outputs of DNNs. A common challenge for ML models is over-confident predictions [36].
- DNNs are unable to differentiate between samples that are in and out of the training dataset [37].

The commonality between all three factors is that they are linked through the uncertainty either in the training data or uncertainty in the model itself. Therefore, to address each of these limitations, it is crucial that the uncertainty be estimated and interpreted in a meaningful way to benefit users of these models. The advantage of having the uncertainty estimates is that high uncertainty samples can be validated by human specialists to ensure that the model is behaving as expected or that the high uncertainty samples are completely ignored to avoid incorrect predictions. This is essential for high-risk applications such as ATR using SAR data where a high degree of confidence is required.

2.3.1 Sources of Uncertainty

The pipeline from the data acquisition to the uncertainty estimation stage can be described in three steps - data collection, model building, and prediction uncertainty model. Each of these steps introduces uncertainty that affects the output of the model. The following section provides a brief description of each step as well as how it impacts the final uncertainty estimate.

2.3.1.1 Data Collection

Data collection is the operation of gathering and measuring samples and their corresponding target class that is a representation of a specific task or domain. The first key factor responsible for the uncertainty measurement from the data collection process is the inconsistency in environments. The majority of data captured from environments are subjected to constant change, whether it is the new objects, temperature, or clutter. Thus, when testing on measured data, there are changes from the original training dataset which can affect the model’s classification performance. The next factor of uncertainty is the data itself which causes uncertainty in the model’s predictions. Most common is the noise introduced by the data capturing sensor, image resolution, and an insufficient number of samples to capture an accurate approximation of the distribution of the dataset.

2.3.1.2 Model Building

The development of the DNN entails the design of the network and the training method. The designer has numerous free variables when designing the network architecture including the number of layers,

the type of layers, activation functions, and filter sizes. The training process also contains a wide range of parameters to choose from, such as the regularisation, optimisation method, learning rate adjustment, and data augmentation. The uncertainty estimations and performance are influenced by the architecture of the network. This is demonstrated in [38], where the confidence and accuracy of a five-layer network are directly compared to a deeper 110-layer network. It was determined that the average confidence for the five-layer network corresponded to the accuracy, although the average confidence for the deeper network was significantly larger than its accuracy. The higher number of parameters in the 110-layer network leads to overfitting, which is the cause of overconfidence.

The second source of uncertainty occurs during the training procedure. The selection of the training parameters such as learning rate, regularisation, and batch size all affect the optimisation of the network. Owing to the learning method, it seldom occurs that two different networks achieve the same model parameters after training and often find different local minimums. In addition, a lack of variety in the training data can result in the model failing to learn the specific task which leads to poor performance.

2.3.1.3 Prediction Uncertainty Model

The predictive uncertainty is the uncertainty propagated through the model for a given input on the output. The prediction uncertainty is the combination of both the data uncertainty and model uncertainty [39]. The epistemic uncertainty is defined as the uncertainty generated from the model and the aleatoric uncertainty is defined as the uncertainty generated from the data. From the previous sources of uncertainty, the aleatoric uncertainty stems from the data collection step with the introduction of noise, lack of image resolution, and insufficient training samples. The epistemic uncertainty is the combination of uncertainty generated from data collection to the final prediction.

2.3.2 Uncertainty Estimation Methods

This section provides a brief overview of the different uncertainty estimation methods from recent publications [40, 41]. A summary of the different methods is shown in Figure 2.6 with additional information provided on Bayesian techniques.

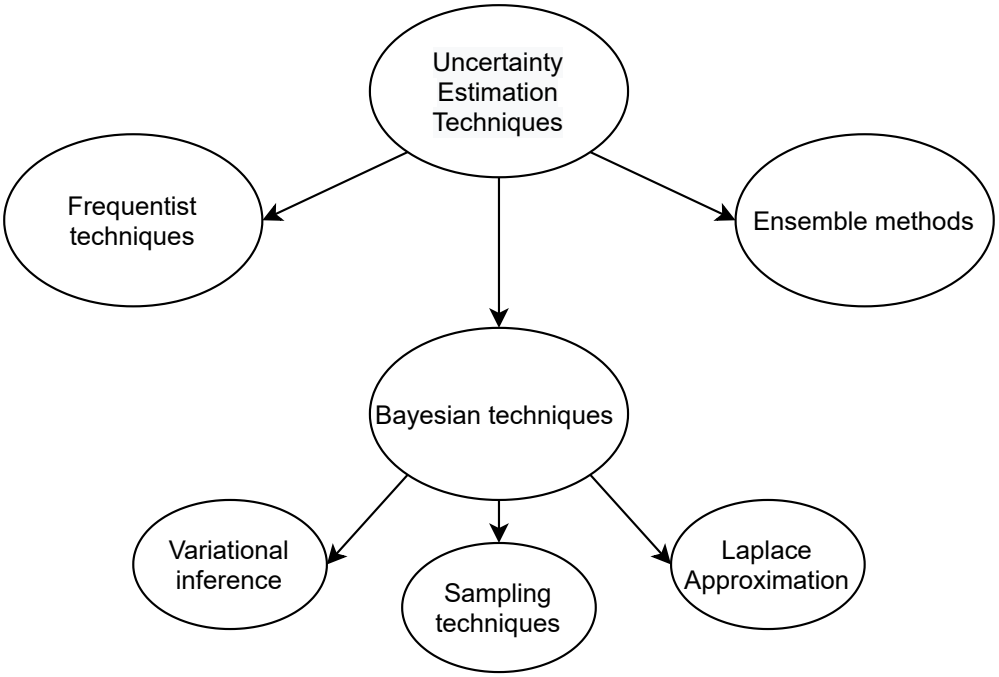


Figure 2.6. Overview of the three various uncertainty estimation techniques, with Bayesian techniques further expanded. Adapted from [40], ©IEEE 2021.

2.3.2.1 Frequentist models

The key attribute that makes a model frequentist is that during each forward pass for any given input, the output is the same for each repetition. This is due to the fixed discrete parameters of the network, whereas for Bayesian models the parameters are sampled during each forward pass. From the literature, the frequentist techniques can be divided into models that are specifically trained to be able to estimate the model's uncertainty [42] and techniques that use supplementary elements to estimate the uncertainty in the model's predictions [43].

The most common method of training the network to estimate uncertainty follows a similar approach to Bayesian techniques. These methods approximate the distribution of the network parameters in contrast to discrete network parameters. Training is performed using loss functions that minimise the difference between the true distribution and the predictive distribution. A popular training method utilises Dirichlet prior networks and can be viewed in [44]. The classification parameters of these networks are generated using the Dirichlet distribution. The classification probabilities are then determined from the classification parameters. The Dirichlet prior network uses a multi-task form that minimises the Kullback-Leibler (KL) divergence between the network and a sharp Dirichlet

distribution with in-distribution data as well as between the network and a flat Dirichlet distribution with out-of-distribution data. An example of this approach is shown in [44]. The proposed method showed an improved separation between the uncertainty for in- and out-of-distribution data. In addition, the method also showed improved performance for detecting miss-classifications.

Supplementary uncertainty estimation methods are external to the predictive model and, therefore, have no effect on the model's predictions. Various approaches [43] train a secondary network that estimates the uncertainty of the first network. Similar to the methods that train the network to estimate the uncertainty, these external approaches have shown the ability to detect out-of-distribution samples.

2.3.2.2 Bayesian Techniques

Bayesian techniques differ from traditional neural network models in the training methods. Frequentist models are trained using the principle of maximum likelihood, while Bayesian methods infer the posterior distribution over the model's parameters. In the case for neural networks, the parameters are given by the weights $\boldsymbol{\theta} = (w_1, ..., w_k)$, the training samples $\boldsymbol{x}$, and class labels $\boldsymbol{y}$. The posterior distribution of the weights, given the data, is provided by using the Bayes theorem:

$$p(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{y}) = \frac{p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\boldsymbol{y} \mid \boldsymbol{x})}. \tag{2.2}$$

The predictive model for a full Bayesian approach is described in [25] for a given input x^* to predict output y^*, and is given by:

$$p(y^* \mid x^*, \boldsymbol{x}, \boldsymbol{y}) = \int p(y^* \mid x^*, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{y}) d\boldsymbol{\theta}. \tag{2.3}$$

Unfortunately, owing to a large number of parameters, no closed-form exists for the integral in (2.3), since there is a lack of conjugate priors for such complex neural networks. Consequently, this has led to increased research into approximate Bayesian inference techniques. A brief overview of the three distinct methods for approximate Bayesian inference is discussed below.

2.3.2.3 Variational Inference

Variational inference approximates the posterior probability distribution on the weights of the network. Variational learning determines the parameters θ of a distribution on the weights. The configuration of the parameters are found such that $q(\theta)$ is close to the true posterior $p(\theta|x, y)$. More recent methods use stochastic variational inference [36, 45] where the optimisation uses mini-batches of samples during the training, similar to more modern training methods.

2.3.2.4 Sampling Methods

Sampling methods are a non-parametric technique for Bayesian inference. Sampling methods sample from a probability distribution, with the advantage of being able to identify unknown distributions using its samples, [46]. Thus, sampling methods are not restricted to a single distribution type, and, as a result, are popular in fields such as importance sampling, particle filtering, and Markov Chain Monte Carlo (MCMC) sampling.

2.3.2.5 Laplace Approximation

The Laplace approximation produces a multivariate Gaussian approximation of the posterior distribution over the network's parameters centred on the Maximum Likelihood Estimator (MLE) [47]. The posterior distribution is determined from the second-order Taylor series expansion from the log of the posterior over the weights. An application of Laplace approximation for deep neural networks is presented in [48].

2.3.2.6 Ensemble Methods

Ensemble methods use multiple models to make a prediction and are based on the principle of group decision-making [49]. Ensemble methods have shown state-of-the-art accuracy and offer a straightforward method for estimating the uncertainty in the prediction by determining the variance in the aggregated prediction.

An important factor to consider when using ensemble methods is that each model should behave differently for a sample with high uncertainty. This allows the ensemble to be able to detect miss-classifications as the average uncertainty of the models will be high, even when one or two models have relatively low uncertainty. The convergence of each model to different local optima is known as multi-mode evaluation [50]. Ultimately, multi-mode evaluation aims to develop models that have a diverse range of strengths and weaknesses such that the combination of each model leads to an improvement in the classification performance and uncertainty estimation.

2.4 EXPLAINABLE ARTIFICIAL INTELLIGENCE

As more ML models are being deployed each day, with increasing levels of complexity, it is apparent that future technological developments can greatly benefit from its use. The human race has come to a pivotal point where the decisions of these models directly affect the lives of humans. Therefore, the demand has increased for the explanation of the decisions made by these ML algorithms [51]. With the increase in AI systems, the introduction of non-transparent models such as DNNs occurred.

These models have been shown to produce impressive results by using efficient training methods and contain a vast number of parameters [52]. With the increase in deployment of these "black-box" models making important decisions in various fields, end-users' need for improved transparency also increased. The decisions by these models are often made without reason and do not come with any logical explanations. This can be dangerous in situations where human lives may be affected. XAI aims to address the problem of the "black-box" model in the following ways: producing models that are explainable while maintaining learning performance, facilitating trust between the user and the algorithms, allowing humans to understand the decision-making process, and interpreting the model's output [35].

There is a distinct difference between the phrases interpretability and explainability [53]. From the literature, there are numerous definitions for both interpretability and explainability [54, 55], however, the definition used in this study is from [56]. Here, interpretability is defined as the extent to which a human can understand the reason for the output. By this definition, a model that is more interpretable allows for the casual relationship between the input and output to be easily identified. An example of this would be an image classification task where certain distinct characteristics in the input have been identified that result in a specific output prediction. Explainability differs from interpretability since it is related to the inner working of the ML model and not the relationship between the input and output of the model. A model that is more explainable provides the user with an enhanced understanding of the inner processes performed for both training and decision-making.

The XAI framework contains methods to increase the interpretability of DNNs. Such methods can be applied to previously trained networks and are often referred to as post-hoc methods [35]. A method for visual explanations from deep networks is presented in [57]. This technique is called Gradient-weighted Class Activation Mapping (Grad-CAM), and it utilises the class-specific gradient data entering into the last convolutional layer of a CNN to construct a rough map of the critical information in the image. Grad-CAM computes the gradient with respect to the feature maps of the last convolutional layer. Once computed, they are fed back after a global average pool operation is performed to obtain the weights. These weight values extract the critical information of the feature maps for each class. To generate the heat maps, the weights of the features are combined and followed by a ReLu activation since only features with a positive influence on the class interest are desired. This is an important XAI tool since it allows for understanding in situations where unexpected predictions are made, thus, ensuring that the classifier is operating as expected and derives its predictions on information from the

desired target. In recent years, the challenge of explainability in ATR has gained much attention with a significant increase in research addressing this challenge. In [58], an example of the application of an XAI method is presented for the task of image classification using the MSTAR dataset. The CNN model consists of three convolutional layers, two max-pooling layers, and one dense layer followed by a softmax layer. Data augmentation is performed by performing various rotations and transitions. The CNN achieved a classification accuracy of 98.78% under SOC. Local Interpretable Model-Agnostic Explanations (LIME) are used to provide model explanations through the visualisation of predictions. LIME allows for the visualisation of the boundary of key characteristics of the target that contributed to the prediction of the CNN.

The quantification of uncertainty in BCNNs provides additional trust to the user through the estimation and visualisation of uncertainty in a model's prediction. The BCNN has been applied in numerous fields such as medicine, finance, and surveillance [59, 60, 61]. In [61], a novel model called the Bayes-SAR Net is proposed. The BCNN achieved comparable classification accuracy when compared to a CNN. Uncertainty estimates were formed from the mean and covariance of the estimated posterior distribution. The model was trained on polarimetric SAR data and it was concluded that the BCNN was more robust to adversarial noise. In [62], a taxonomy for uncertainty representation and evaluation for modelling and decision-making in information fusion is presented. It is further extended in [63]. It contains a discussion on the different types of epistemic and aleatoric uncertainties and where they enter a sensing or fusion system. This taxonomy is applied to investigate the effects of aleatoric and epistemic uncertainties of the BCNN.

2.5 CHAPTER SUMMARY

A comprehensive overview of the literature pertaining to ATR methods for SAR images was presented in Section 2.2. In Section 3.3, a brief taxonomy of the various uncertainty estimation methods was discussed with a focus on Bayesian techniques. Lastly, the emerging field of XAI was detailed with examples of methods to increase interpretability in DNNs.

A comparison of the different ATR methods using SAR images was summarised in Table 2.1. It was found that the methods that used CNNs, on average, achieved the highest classification accuracy. The top performing CNN implementation [4] slightly outperformed the top performing ensemble implementation [18] using both SOC and EOC on the MSTAR dataset. Both implementations contained a CNN and, from the literature, it is apparent that CNNs achieved the best classification performance.

Despite being the best performing ATR method in terms of classification accuracy, CNNs are considered to be "black-box" models and are less explainable than BNs and SVMs. Furthermore, the problem of over-confident predictions is still prevalent in CNNs and needs to be considered for ATR systems. To address the over-confident predictions and attempt to improve interpretability, different uncertainty estimations were investigated in neural networks.

From the literature, it is clear that the number of research pertaining to Bayesian techniques has surged. This is a result of breakthroughs with the use of mini-batch optimisation techniques in Bayesian inference. Numerous works have shown that Bayesian inference is now obtainable with large datasets. These advancements have now been applied to Bayesian Neural Networks (BNNs) and have resulted in models that perform similar to standard neural networks but are better a quantifying uncertainty to prevent over-confident predictions.

Post-hoc methods were identified that improved the interpretability of DNNs. The Grad-CAM technique provided visual explanations of the regions in the input image that contributed the most to the output prediction. This XAI tool provided insight into the causal relationship between the input image and the output of the model. The same concept was used for this study but with the utilisation of uncertainty estimations.

CHAPTER 3 BACKGROUND THEORY

3.1 CHAPTER OVERVIEW

This chapter provides the theoretical background required for the BNN and the method to estimate the uncertainty of the model. Firstly, the fundamental concept of the BNN is presented with the theoretical foundation of the technique of variational inference. The Bayes by backpropagation training method [36] is described in more detail. Lastly, the theory for uncertainty estimation is presented for both the model and data uncertainty.

3.2 BAYESIAN NEURAL NETWORK

A BNN is the combination of probabilistic modelling and Artificial Neural Networks (ANNs). BNNs are created to assess the uncertainty in the training and the problem of over-confident predictions made by NNs. Fundamentally, a BNN is a probabilistic model supplemented by an ANN for its universal function approximation attributes [64].

Blundell *et al.* proposed a method for variational Bayesian learning by introducing uncertainty in the weights of an ANN [36]. An efficient novel algorithm is introduced for regularisation augmented by Bayesian inference on the weights, allowing for a straightforward learning algorithm like back-propagation. It is shown that by introducing uncertainty in the weights, the model gains improved capability by expressing increased uncertainty in areas with little to no data. This results in a model that is more robust to over-fitting while offering uncertainty estimates through its parameters in the form of probability distributions. In this network, all of the weights are expressed by probability distributions, in contrast to regular ANN that uses single-valued weights. A graphical comparison of the two networks is shown in Figure 3.1.

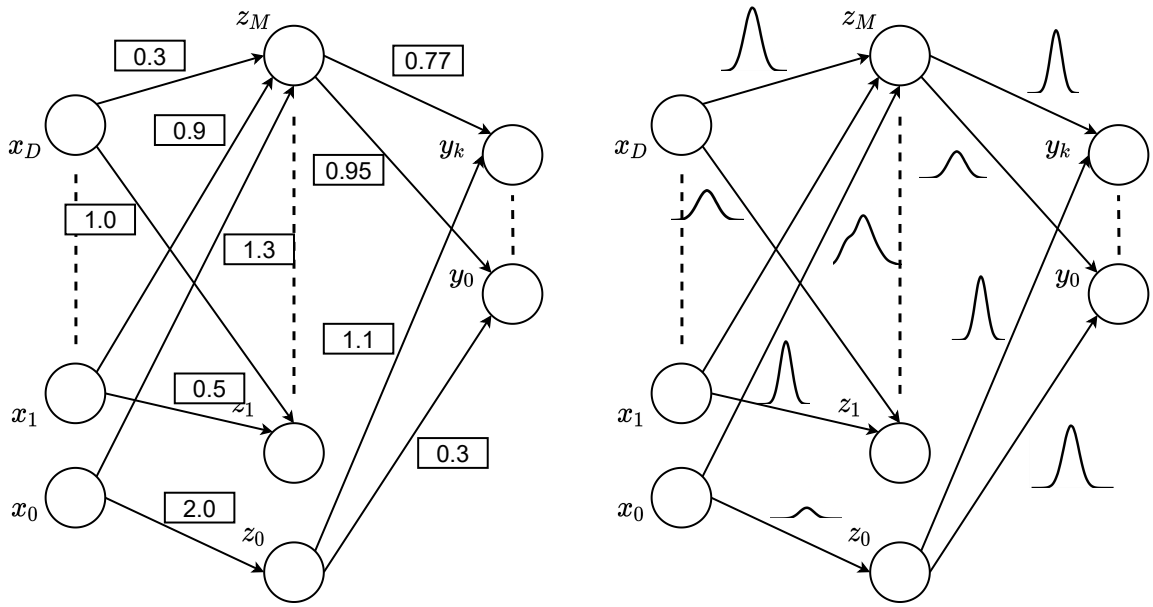


Figure 3.1. Side by side comparison of normal ANN and BNN. The weights of the BNN are represented by probability distributions, while the ANN uses discrete-single values. Adapted from [36], ©IEEE 2015.

The authors introduced the Bayes by backpropagation training algorithm and it is discussed in detail in the following sub-sections. First, the theoretical foundation of the variational inference method is described.

3.2.1 Variational Inference

Variational inference approximates a posterior distribution of latent variables z given observed variables. A family of distributions over latent variables is selected with its own set of variational parameters v [65]. This amounts to an optimisation problem where the objective is to learn the settings of the parameters that closely approximate $q(z)$ to the desired posterior distribution $p(z|x)$. After training, the fitted q can be used instead of the posterior to make predictions [66].

3.2.1.1 Kullback-Leibler Divergence

The Kullback-Leibler (KL) divergence is a good indicator of the similarity between two distributions. Let an undetermined probability distribution be denoted as $q(z)$ and assume that its approximate distribution is $p(z)$. Thus, the KL divergence is used to quantify the closeness of two distributions which is used to determine the free parameter settings that approximate q to the desired posterior. The

KL divergence is defined in [25] as:

$$KL(q||p) = \int_z q(z) \log \frac{q(z)}{p(z|x)}, \tag{3.1}$$

$$= \mathbb{E}\left[\log \frac{q(z)}{p(z|x)}\right]. \tag{3.2}$$

At first glance, it is desirable to want to minimise the KL divergence of q and the posterior. Unfortunately, the KL divergence cannot be minimised exactly, but a function that is proportional to it up to a constant can be minimised.

3.2.1.2 Evidence Lower Bound

The Evidence Lower Bound (ELBO) is derived by first applying Jensen's inequality [67] to the log marginal probability of the observations. The ELBO is described in [68] as:

$$L(q) = \mathbb{E}_q[\log\ p(x,z)] - \mathbb{E}_q[\log\ p(z)], \tag{3.3}$$

where L is the ELBO and the second term is the entropy of z. It is apparent that L is the lower bound for the log probability of the observations.

An expression of the ELBO in terms of the KL divergence for variational inference is given by:

$$L(q) = \log\ p(x) - KL[q||p], \tag{3.4}$$

where the ELBO is independent of q. Notice that maximising the ELBO is equivalent to determining the q that minimises the KL divergence to the posterior distribution. Since the KL divergence is always greater or equal to zero, the ELBO reduces to $L \leq \log\ p(x)$ which is a lower bound of evidence [65]. The evidence is $\log p(x)$. The difference between the ELBO and the log probability is the KL divergence. The lower bound is equal to the log probability if the approximated q is exactly equivalent to the true posterior p.

3.2.2 Bayesian by Backpropagation

Bayes by backpropagation determines the posterior distribution of the weights given the training data $P(\mathbf{w}|\mathcal{D})$. Bayesian inference on the weights of an ANN is intractable due to the vast number of parameters. Variational inference is applied to learn parameters θ of a distribution on the weights $q(\mathbf{w}|\theta)$. The optimal parameters θ^* is described in [36], the expression is in terms of the Kullback-Leibler divergence as:

$$\theta^* = \arg\min_\theta KL[q_\theta(\mathbf{w}|\mathcal{D})||p(\mathbf{w}|\mathcal{D})]. \tag{3.5}$$

As shown previously, the KL divergence is intractable and the negative ELBO is used instead. This optimisation problem is known as variational free energy [69], and the cost function is given by:

$$\mathcal{F}(\mathcal{D},\theta) = KL[q_\theta(\mathbf{w}|\mathcal{D})||p(\mathbf{w})] - \log\ P(\mathcal{D}|\mathbf{w}). \tag{3.6}$$

The cost function is minimised using gradient descent and variational inference searching for q_θ that is closest to the true posterior. The exact cost is shown in [36] and can be represented as:

$$\mathcal{F}(\mathcal{D},\theta) = \sum_{i=1}^{n} \log\ q - \theta(\mathbf{w}^{(i)}|\mathcal{D}) - \log\ p(\mathbf{w}^{(i)}) - \log\ p(\mathcal{D}|\mathbf{w}^{(i)}), \tag{3.7}$$

where $\mathbf{w}^{(i)}$ represents the ith Monte Carlo sample sampled from the variational posterior $q(\mathbf{w}^{(i)}|\theta)$.

3.3 UNCERTAINTY ESTIMATION

In order to determine the predicted class, the predictive distribution for the output y^* and the test input x^* are used. The optimised variational parameter is denoted by $\hat{\theta}$ which is determined by minimising the cost function in (3.7). The optimised variational predictive distribution is described in [70], and is given by:

$$q_{\hat{\theta}}(y^* \mid x^*) = \int_{\mathbf{w}} p(y^* \mid x^*, \mathbf{w})\, q_{\hat{\theta}}(\mathbf{w}) d\mathbf{w}. \tag{3.8}$$

However, the integral is intractable, as discussed in 2.3.2.2, the estimator of the predictive distribution is used:

$$\hat{q}_{\hat{\theta}}(y^* \mid x^*) = \frac{1}{T}\sum_{t=1}^{T} p(y^* \mid x^*, \hat{\mathbf{w}}_t), \tag{3.9}$$

where w_t is drawn from the variational distribution $q_{\hat{\theta}}$, and T is the number of samples. The variance of the variational predictive distribution is also known as the predictive variance. The variance is also referred to as uncertainty. The uncertainty is separated into aleatoric and epistemic uncertainty with the aleatoric representing the randomness in the data while the epistemic uncertainty is generated from the variability in the weights. The combined uncertainty is given by:

$$\begin{aligned}
\mathrm{Var}_{q_{\hat{\theta}}(y^*|x^*)}(y^*) &= E_{q_{\hat{\theta}}(y^*|x^*)}\left\{y^{*\otimes 2}\right\} - E_{q_{\hat{\theta}}(y^*|x^*)}(y^*)^{\otimes 2}, \\
&= \underbrace{\int_{\mathbf{w}} \left[\mathrm{diag}\left\{E_{p(y^*|x^*,\mathbf{w})}(y^*)\right\} - E_{p(y^*|x^*,\mathbf{w})}(y^*)^{\otimes 2}\right] q_{\hat{\theta}}(\mathbf{w}) d\mathbf{w}}_{\text{aleatoric}}, \\
&+ \underbrace{\int_{\mathbf{w}} \left\{E_{p(y^*|x^*,\mathbf{w})}(y^*) - E_{q_{\hat{\theta}}(y^*|x^*)}(y^*)\right\}^{\otimes 2} q_{\hat{\theta}}(\mathbf{w}) d\mathbf{w}}_{\text{epistemic}},
\end{aligned} \tag{3.10}$$

where $\otimes$ denotes the outer product, and where

$$v^{\otimes 2} = vv^T,$$
$$E_{q_{\hat{\theta}}(y^*|x^*)}\{g(y^*)\} = \int g(y^*)\, q_{\hat{\theta}}(y^* \mid x^*)\, dy^*, \tag{3.11}$$
$$E_{p(y^*|x^*,\mathbf{w})}\{g(y^*)\} = \int g(y^*)\, p(y^* \mid x^*, \mathbf{w})\, dy^*.$$

The aleatoric uncertainty represents the intrinsic variance in the data, while the epistemic uncertainty is a result of the variance of the weights $\mathbf{w}$, given the data [70].

The method to explicitly compute the uncertainty as two separate types were developed in [39]. However, this method has two key constraints. Firstly, this method estimates the variance of linear predictors, which is not the case for classifiers, instead, it should model the predictive probabilities. Secondly, the aleatoric uncertainty does not factor in the correlations from the diagonal matrix modelling. The challenges are addressed in [70] and the predictive variance is reduced to

$$\mathrm{Var}_{q_{\hat{\theta}}(y^*|x^*)}(y^*) = \underbrace{\frac{1}{T}\sum_{t=1}^{T} \mathrm{diag}(\hat{p}_t) - \hat{p}_t^{\otimes 2}}_{\text{aleatoric}} + \underbrace{\frac{1}{T}\sum_{t=1}^{T} (\hat{p}_t - \bar{p})^{\otimes 2}}_{\text{epistemic}}, \tag{3.12}$$

where $\bar{p} = \frac{1}{T}\sum_{t=1}^{T} \hat{p}_t$ and $\hat{p}_t = \mathrm{Softmax}(f_{w_t}(x^*))$, where the softmax is described in [71], and is applied to the output of the model.

3.4 CHAPTER SUMMARY

The theoretical foundation for the BNN as well as the method to estimate uncertainty is established in this chapter. The method of Bayes by backpropagation is used in order to perform the Bayesian inference. From the uncertainty in the weights, the BNN is capable of estimating the uncertainty in its output. This provides a measure of the network's confidence which can be used to determine low-confidence predictions.

CHAPTER 4 METHODS

4.1 CHAPTER OVERVIEW

This chapter provides an in-depth description of the implementation of the BCNN. The BCNN is used since it incorporates the uncertainty estimations of the BNN while retaining the improvement in image classification performance from the CNN. In addition, this chapter includes details of the activation function, network architecture, selection of the optimiser, the dataset for classification, and the algorithm for training the network. The method to determine the uncertainty in the model and data is the same as the background theory in Section 3.3.

Section 4.3 describes the method to perform target detection and generate samples. For this study, target detection refers to the ability of the network to locate targets and not target recognition or classification. Using the ground-truth samples, the chapter describes a method to feed back high confidence incorrect detection by using the BCNN.

4.2 BCNN IMPLEMENTATION

This section looks at the implementation of the BCNN, whereas Section 3.2 detailed the theoretical foundation for the BNN. The implementation for the BCNN is based on the same method of variational inference, namely, Bayes by backpropagation. The BCNN implementation used in this study is from [14] as it provides an efficient method to perform variational inference using a CNN. This is achieved through the use of two convolution operations to determine the mean and variance of the weights. This implementation introduces a probability distribution over the weights in the convolutional layers, similar to the weights in the fully-connected layers. Figure 4.1 illustrates the difference in traditional CNNs and BCNNs. In (a), the kernel and the output are represented by single-values, while in (b), the kernels and outputs are represented by probability distributions.

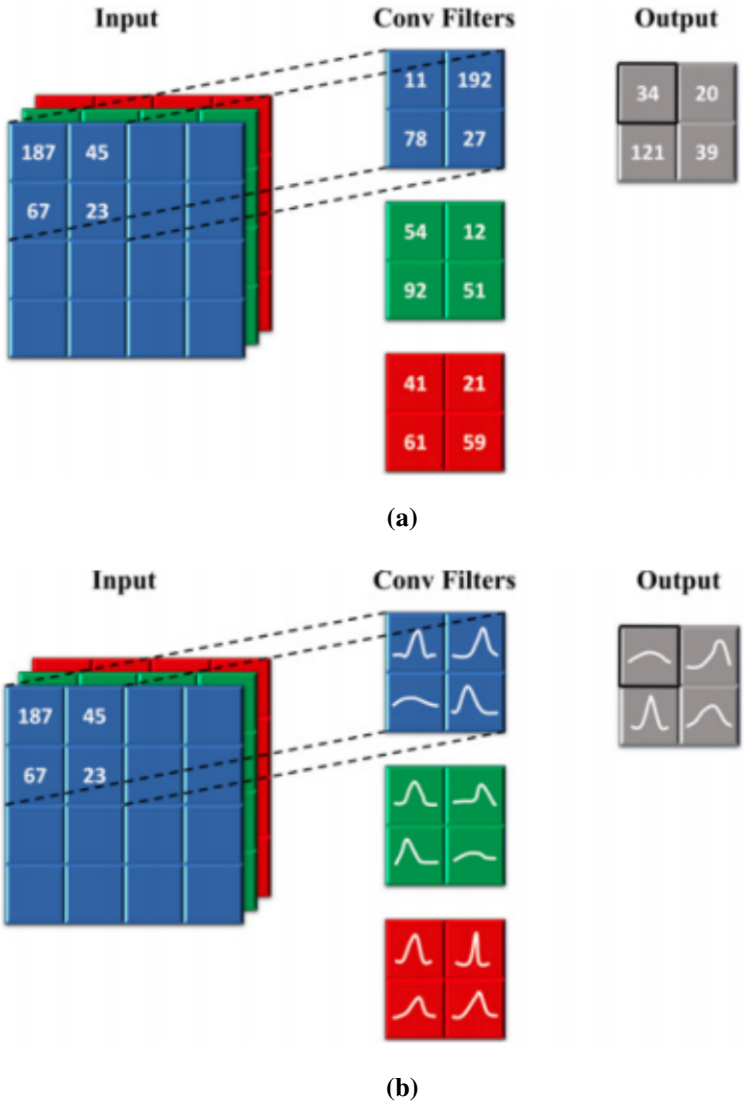


Figure 4.1. Side by side comparison of CNN and BCNN, each with input values, kernels, and output. In the BCNN, the output of each convolutional filter is a probability distribution whereas the output from a CNN is a discrete-single value. Taken from [72], ©IEEE 2021.

4.2.1 Local Reparameterisation Trick

Stochastic gradient variational Bayes is a popular mini-batch optimisation technique that uses a reparameterisation technique in order to generate samples from conditional probability distributions. This technique provides an alternative approach to generate samples from the posterior distribution $q_\theta(\mathbf{w}|\mathcal{D})$ from (3.5). The principle is based on the parameterisation of random parameters such as the weights of a neural network. First, the weight is expressed in a discrete form as $\mathbf{w} = f_\theta(\varepsilon, \mathcal{D})$

where $f(.)$ is a differentiable function and $\varepsilon \sim p(\varepsilon)$ is a random variable [68]. This parameterisation results in an unbiased differentiable estimator from which the expected log-likelihood can be derived. An improved parameterisation which is more computationally efficient is proposed in [73]. This is achieved by not sampling ε straightaway, but, instead, only the intermediate variable $f(\varepsilon)$ that affects the log-likelihood. This transforms the global uncertainty in the weight into the local uncertainty which is independent throughout the data and more convenient to sample. As a result, the reparameterisation is known as the local reparameterisation trick. For the case of the weight following a Gaussian distribution $\mathbf{w} = \mathcal{N}(\mu, \sigma^2)$, the reparameterisation is given by:

$$\mathbf{w} = \mu + \sigma\varepsilon, \tag{4.1}$$

where $\varepsilon = \mathcal{N}(0,1)$ is the auxiliary noise. Since the variance cannot be negative, a softplus transform that is similar to the implementation is often used [36]. The softplus function is discussed in further detail in Subsection 4.2.2.

The reparameterisation for BCNNs is defined in [14]. The local reparameterisation trick for the convolutional layers is given by the output of convolutional layer b as:

$$b_j = A_i * \mu_i + \varepsilon_j \odot \sqrt{A_i^2 * \left(\alpha_i \odot \mu_i^2\right)}, \tag{4.2}$$

where i and j are subscripts for the input of the 2D image, $\varepsilon_j \sim \mathcal{N}(0,1)$, A_i is the receptive field also referred to as the filter size of a layer, * represents the convolutional operation, and $\odot$ the component-wise multiplication. It is important to note that the local reparameterisation b is a function of both the mean μ_i and variance $\alpha_i\mu_i^2$. This determines the parameters that should be updated for the variational posterior distribution $q_\theta(\mathbf{w} \mid \mathcal{D})$. The process to update $q_\theta(\mathbf{w} \mid \mathcal{D})$ requires two convolutional operations [14]. First, the output b is used in a similar fashion to the deterministic neural network to perform the Maximum A Posterior (MAP) of the variational posterior and increase accuracy. This point-estimate is treated at the mean of the $q_\theta(\mathbf{w} \mid \mathcal{D})$ distribution. The second convolutional operation is performed to learn the variance. During each convolutional operation, only a single parameter is updated. This process can be interpreted as the first convolutional operation performing a MAP of the $q_\theta(\mathbf{w} \mid \mathcal{D})$, while the second determines the variance of the $\mathbf{w}$ from the MAP.

4.2.2 Activation Function

As previously described in Subsection 4.2.1, two convolution operations are performed to determine the mean and variance. In order to ensure that the variance is non-negative, the activation function for the convolutional layers is selected to never output a value equal to or less than zero. The softplus activation function is used as it never equals zero for $x \to -\infty$. The equation for the softplus is given

by:

$$\text{Softplus}(x) = \frac{1}{\beta} \cdot \log(1 + \exp(\beta \cdot x)), \tag{4.3}$$

where β is set to 1 for all BCNN models trained. The softplus activation function and ReLu activation are visualised in Figure 4.2. The softplus activation is different to the ReLu near zero, where the softplus is smooth.

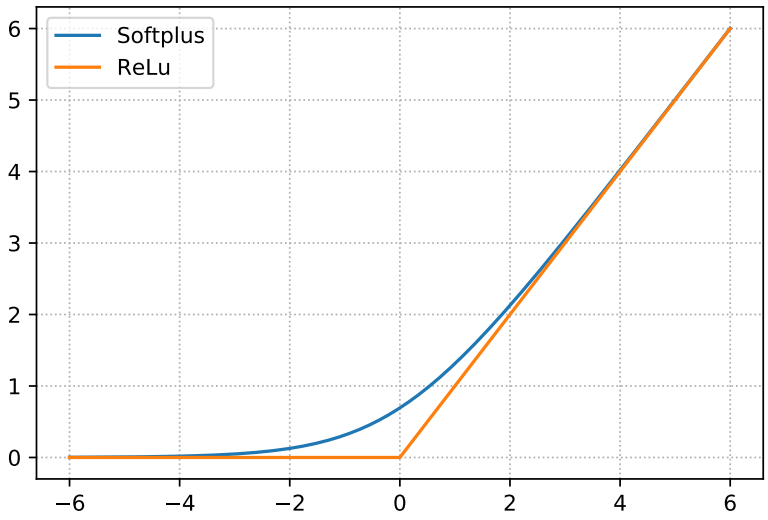


Figure 4.2. Visualisation of activation functions. Their softplus function has a distinct curve at zero. Allowing the functions to always be greater or less than zero.

4.2.3 Architecture

Both the CNN and BCNN trained in this study used the same six-layer architecture. This architecture contains three convolutional layers with three fully-connected layers. A key attribute of the structure is the max pooling layers which were introduced to reduce the overall size of the model [74]. This structure was selected owing to high classification performance and a relatively low number of layers compared to more modern architectures such as VGG16 [31]. Figure 4.3 illustrates the dimensions at each layer of the network. A brief description of the properties of the architecture follows.

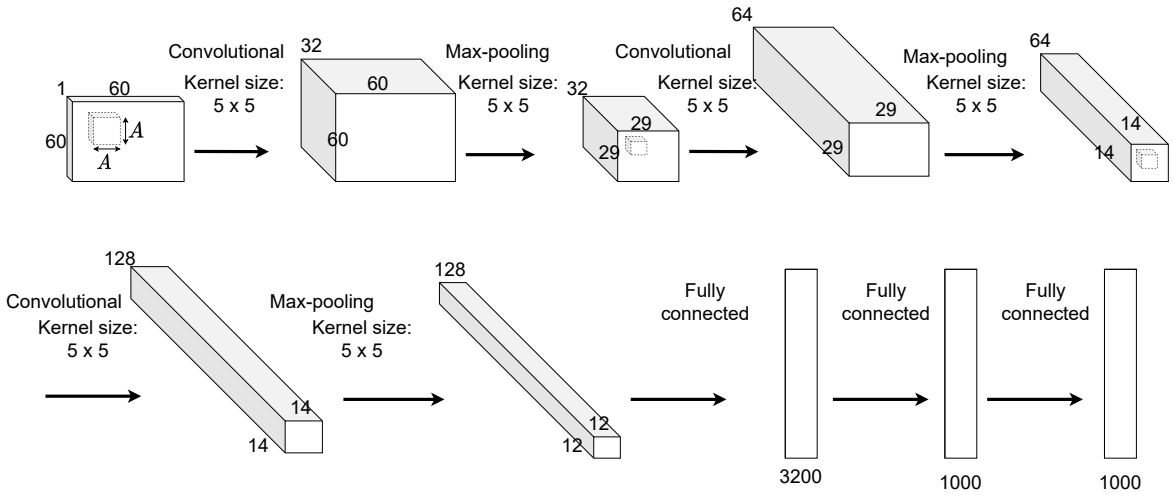


Figure 4.3. Illustration of input and output dimensions for three convolutional and three fully-connected architecture.

4.2.3.1 Layer Type

In this study, only three types of layers are used - convolutional, max-pooling, and fully connected. The convolutional layers are responsible for the feature extraction of the data. The max-pooling layer is used to reduce the dimensions of the feature maps and thereby reduce the total number of weight parameters that need to be trained. The fully-connected layers contain the traditional weights and biases that are connected and used to perform the classification.

4.2.4 Receptive Field

The receptive field is the size of the area in that input image that generates the feature maps in that convolutional layer [75]. From Figure 4.3, the receptive field is denoted by A or the kernel size.

4.2.4.1 Width

The width is the number of filters in the convolutional layer. In Figure 4.3, the width of the network increases after each convolutional layer.

4.2.4.2 Stride

The stride is the number of pixel shifts between the filter operations on the input image. An example of this is when the stride is set to one. This corresponds to the filter moving one pixel between convolution operations.

4.2.4.3 Padding

The purpose of padding is to ensure the feature maps produced by the filters have the same size as the input. This is important since the output shape is reduced after each convolutional layer. For the network to properly learn the features of the dataset, each filter must interact with as many pixels from each input as possible. This ensures that each convolutional layer can extract sufficient features from the data.

4.2.4.4 Architecture Parameters

Table 4.1 shows the architecture used for both the CNN and BCNN with the various network parameters. The value next to the layer type is the kernel size for each layer.

Table 4.1. Six-Layer BCNN Architecture.

Layer Type	Width	Stride	Padding	Input Shape	Activation Function
Convolution (5×5)	32	1	2	$1 \times 1 \times 60 \times 60$	Softplus
Max-pooling (2×2)		2	0	$1 \times 32 \times 60 \times 60$	
Convolution (5×5)	64	1	2	$1 \times 32 \times 29 \times 29$	Softplus
Max-pooling (2×2)		2	0	$1 \times 64 \times 29 \times 29$	
Convolution (5×5)	128	1	1	$1 \times 64 \times 14 \times 14$	Softplus
Max-pooling (2×2)		2	0	$1 \times 128 \times 12 \times 12$	
Fully-connected	1000			3200	Softplus
Fully-connected	1000			1000	Softplus
Fully-connected	10			1000	Softplus or Softmax

4.2.5 Model Initialisation

The BCNN network has two Gaussian distributions that require initialisation. As mentioned in 4.2.1, the variance can never be zero, therefore, the variance is represented as $\sigma = \text{softplus}(\varepsilon)$ where ε is randomly selected.

The prior distribution is initialised with a zero mean and variance of 0.1, while the posterior distribution is initialised with a mean of zero and a ε randomly sampled from -5 to 0.1, similar to [36]. From [36], the parameters were found experimentally by using the validation error to find the parameters that resulted in the lowest validation error.

4.2.6 Optimiser

The Adam optimiser is used to perform the updated steps of the variational parameters. Adam is a combination of Root Mean Squared Propagation (RMSprop) and stochastic gradient descent with momentum. Adam benefits from the advantages of adaptive gradient algorithms and RMSprop. It stores the learning rate which improves performance with sparse gradients and the learning rates are adjusted using the mean of the magnitudes of the gradient of weights, making it more resilient to saddle points. This optimiser has proven to converge faster than both RMSprop and stochastic methods [76].

4.2.7 Bayesian Deep Learning

The training method is described in Algorithm 1. During each forward pass, the activation b is sampled to calculate the KL divergence. The reparameterisation trick is used to sample from each convolutional layer. Training ends after a fixed number of epochs, and the number of epochs is determined during the hyper-parameter optimisation process along with over hyper-parameters. T is the number of times the activation b is sampled per iteration - the number is fixed to twenty in order to reduce the total training time.

Algorithm 1 Bayes by Backpropogation Learning

Input: Dataset $\mathcal{D} = (x_i, y_i)$, learning rate, batch size

Initialisation : Priors and posteriors of weights

for $epoch = 0$ to $numEpochs$ **do**

for batch in $numBatches$ **do**

for $i = 0$ to T **do**

Sample weights

Calculate KL divergence

end for

Calculate ELBO using (3.7)

Determine gradient of the variational parameters θ

Update the mean μ and variance ρ of the weights

Calculate the training and validation accuracy

Calculate the training and validation loss

end for

end for

4.2.8 Hyper-Parameter Optimisation

Hyper-parameters are the parameters that control the training of the network and include the learning rate, number of epochs, batch size, and momentum. Hyper-parameter optimisation can be a time-consuming task if performed manually or when using a grid-search method. To improve the efficiency of optimising the networks, a Bayesian model-based optimisation method is employed. Bayesian methods of optimisation record previous test results that are used to create a statistical model of the hyper-parameter mappings. This maps the probability of an evaluation for a specific cost function. The hyper-parameter optimisation method used is from [77]. The optimisation is performed for both the traditional CNN and BCNN.

4.3 DETECTION OF TARGETS AND CLUTTER

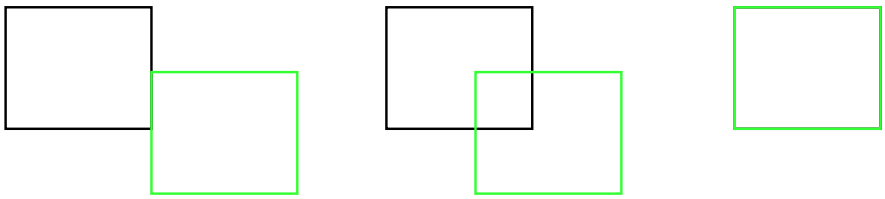
In traditional ATR systems, a key stage is the Focus-of-Attention, since it is responsible for evaluating the entire scene for Regions of Interest (ROIs). This stage can significantly reduce the computation time required to identify every target in a scene by only passing ROIs to the classification algorithms [78] and, thus, minimising the computationally taxing process of passing each sample through the classification algorithm which may consist of a DNN. However, for this research, there was no limitation on the computational time and the main focus was on improving the explainability of ATR systems. As a result, the entire scene was used in order for the uncertainty in the detections to be evaluated. By leveraging the uncertainty estimates over the entire scene, a 2D representation of the uncertainty in the detections was constructed. This 2D image was then investigated to determine if the visualisation of the network's confidence over the scene improved the interpretability for the user. Lastly, an investigation was conducted by feeding back high-confidence incorrect detections in order to improve detection performance.

The following sections describe the methods for ground-truth data generation, target detection, uncertainty heat map generation, and the feed back of high-confident incorrect samples.

4.3.1 Ground-Truth Generation

The method used to generate ground-truth samples is shown in Figure 4.6. Firstly, the known targets are identified and bounding boxes are placed around them. A rudimentary sliding window is used to isolate individual regions in the scene. The sliding-window method has two parameters - the crop size and step size. The crop size is the fixed window size and the step size is the distance the window travels after each iteration. To determine if a sample is a target or clutter, the Intersection over Union (IoU) is used. IoU is a metric that measures the overlap between bounding boxes [79]. A set of examples are

shown in Figure 4.4. An IoU of 0 corresponds to no overlap, while an IoU of 1 corresponds to two bounding boxes overlapping exactly with each other.



$IoU = 0$ $IoU = 0.142$ $IoU = 1$

Figure 4.4. Example of IoU. This shows the direct correlation between the amount of overlap and the IoU value.

The expression for the IoU is given by:

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}, \tag{4.4}$$

where the $Area\ of\ overlap$ is the area of overlap between the ground-truth bounding box and sliding-window bounding box, and the $Area\ of\ Union$ is the combined area that both the ground-truth and sliding-window bounding boxes share. An illustration is shown in Figure 4.5.

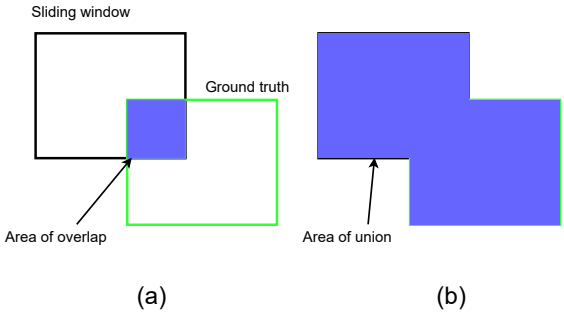


Figure 4.5. Illustration of the area of overlap (a) and area of union between two bounding boxes (b).

After each iteration, the IoU is calculated for the sliding window and each bounding box in the scene. If the IoU is greater than 0.85, then that sample is labelled as a target, since there is a high likelihood that the majority of the target is contained within that window. Samples with an IoU of less than 0.85 are labelled as clutter samples. These samples make up the data for the target and clutter classes to train the BCNN detector.

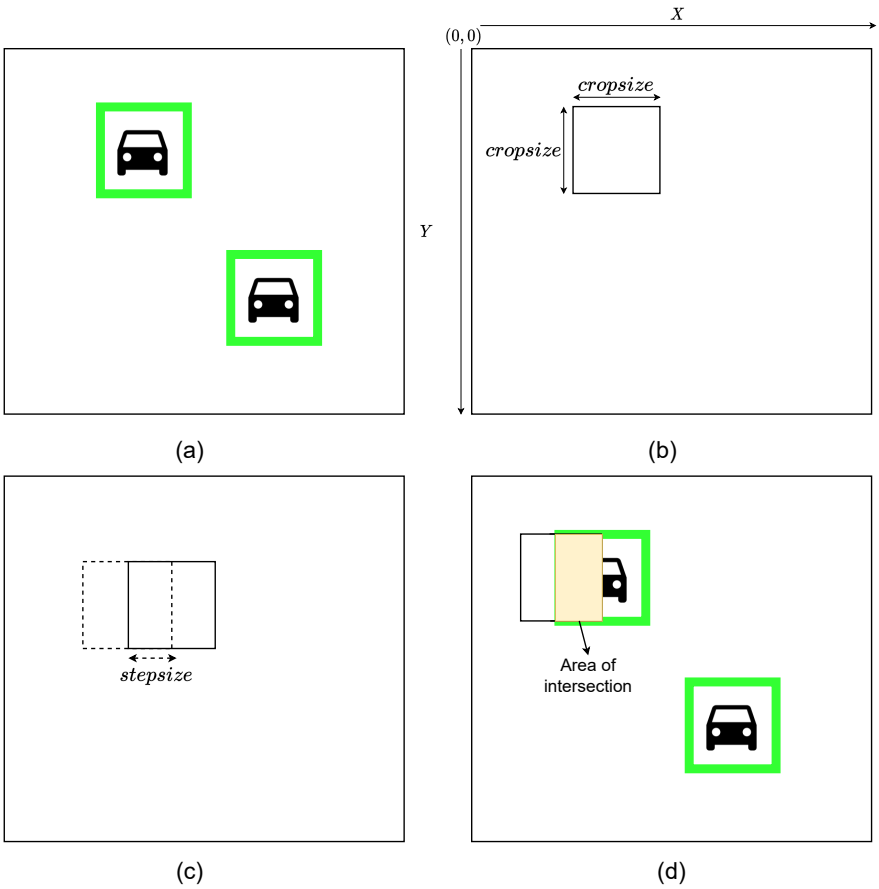


Figure 4.6. Illustration of the method to generate ground-truth data samples. (a) Ground-truth samples with bounding boxes drawn around them. (b) Sliding window, illustrating the crop size parameter. (c) Sliding moving over the SAR scene. (d) Sliding window with the area of intersection used to calculate IoU between ground-truth bounding box.

As seen in Figure 4.6, first, the bounding boxes are drawn around the target (a). Then, the sliding window is started (b) and moved after each iteration by the step size (c). The IoU is calculated at each iteration using the area of intersection (d).

4.3.2 Target Detection Implementation

The dataset used to generate samples for the BCNN detector was supplied by the CSIR. It is similar to the MSTAR dataset since it contains two scenes captured at various elevations and different orientations. To train the detector, the data from the MSTAR was used as the targets, and samples were manually selected from regions known not to contain targets in the CSIR dataset. The BCNN was trained to detect two classes - targets and clutter. Detection is performed using a similar sliding window as

in Subsection 4.3.1 to distinguish between targets and clutter. An overview of the target detection method is shown in Figure 4.7. At each iteration, the window was passed through the network and a prediction was made. The ground-truth data in Subsection 4.3.1 was used to determine the correct and incorrect detections. Once a window was classified as a correct target, a green box was drawn around that sample to indicate that it was correctly detected as a target. Windows correctly detected as clutter was left blank to emphasise the detected targets. Each incorrect detection is contained in a red bounding box drawn around the window, an example of this is shown in Figure 5.23. To evaluate the performance of the detector, the precision was calculated for each scene using the correct and incorrect detections.

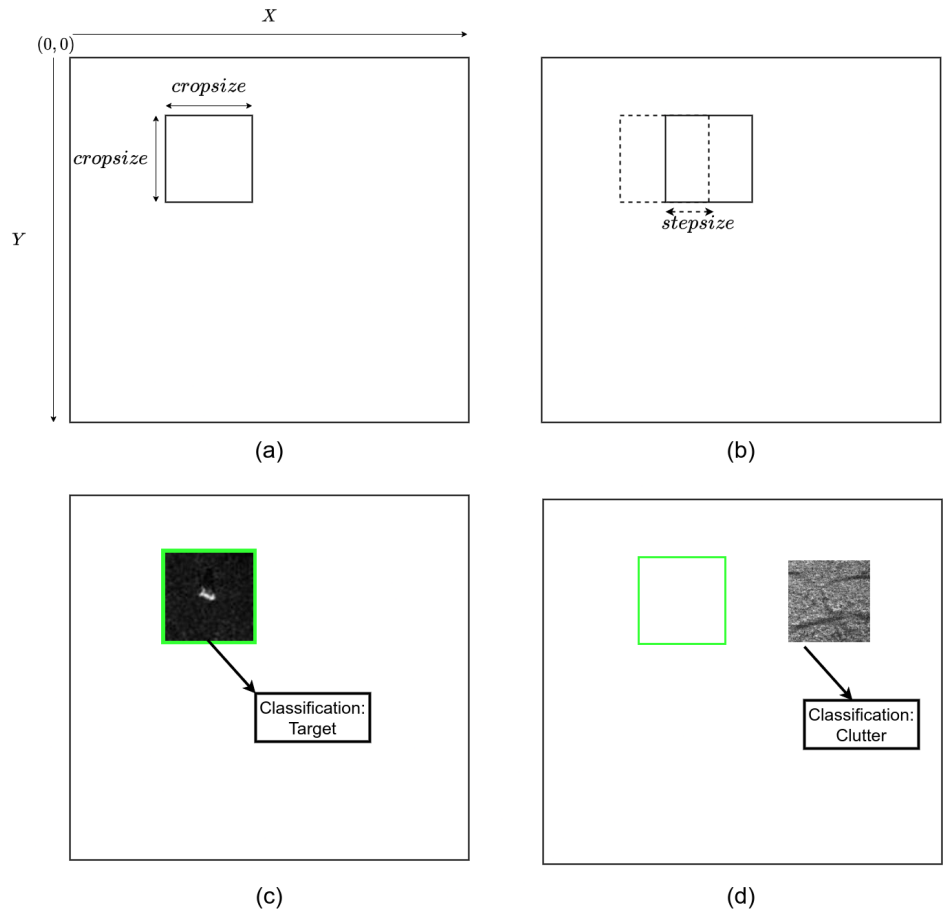


Figure 4.7. Sliding-window target detection using a BCNN. (a) Shows the starting coordinates and axis that the sliding travels. (b) Sliding moves by the step size parameter. (c) Output from the classier of the sliding window. (d) Correct classification of clutter with no bounding box around the sliding window.

In Figure 4.7, (a) illustrates the test scene with dimensions X and Y, and the size of the sliding window is given by the crop size. Box (b) shows the sliding windows moving after one iteration. The distance travelled after each iteration is controlled by the step size. In (c), the sliding window is passed through the classifier and a correct target is detected in that window. Box (d) shows when the classifier correctly predicts a clutter sample and no bounding box is drawn.

4.3.3 Uncertainty Heat Map Generation

To improve explainability, the uncertainty over the scene was visualised. This highlighted the regions of high or low confidence from the network, similar to the Grad-CAM method that visualises regions that contributed the most to the prediction. The method to generate the uncertainty heat maps is shown in Figure 4.8. The same initial process was followed for the target detection. A sliding window was applied to the scene and the epistemic uncertainty of the window was calculated (Figure 4.8 (a) - (c)). The uncertainty heat map was then normalised (Figure 4.8 (d)). For the uncertainty map to be superimposed onto the test scene, an interpolation function was used to transform the dimension from (44 x 34) to (1360 x 1074)(Figure 4.8 (e)). The heat map was then transformed to correspond to a maximum brightness of 255, similar to the SAR scene (Figure 4.8 (f)). Lastly, the uncertainty heat map was superimposed onto the scene to show regions of high-uncertainty. Low-uncertainty regions should have a pixel value of zero and not contribute when superimposed.

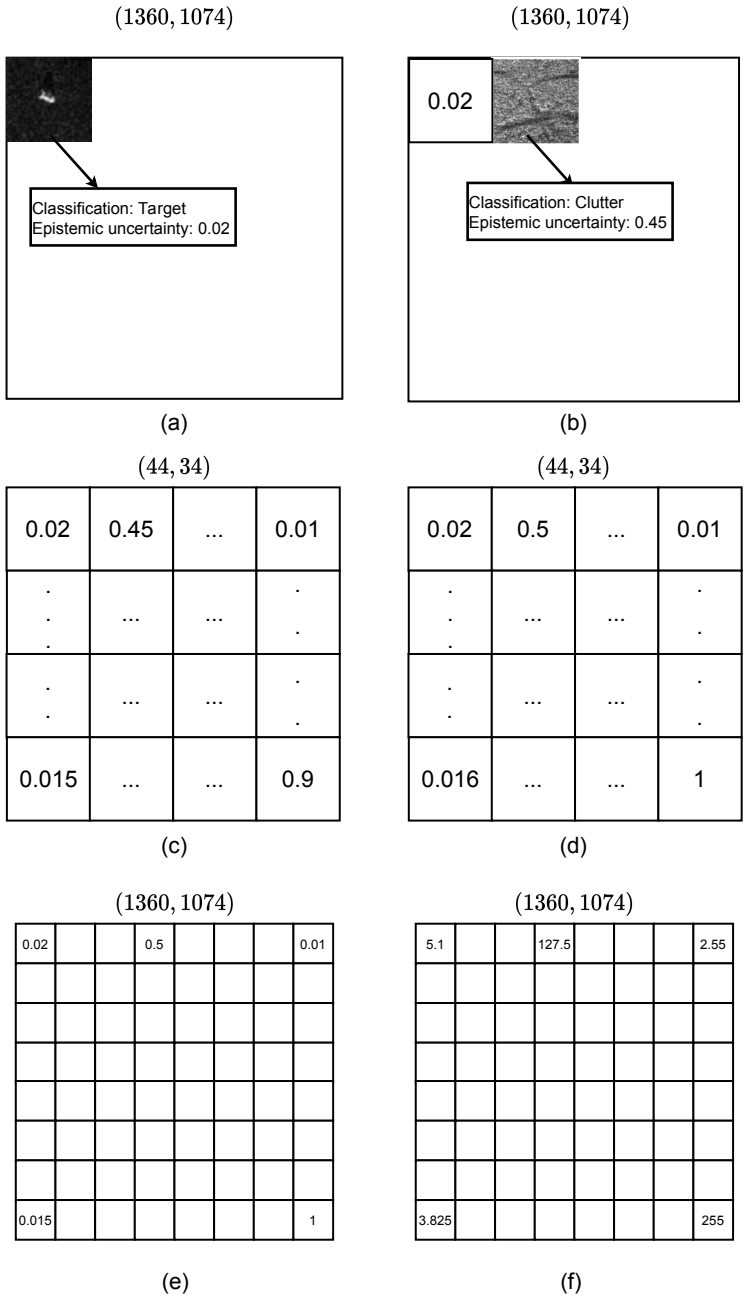


Figure 4.8. Illustration of uncertainty heat map generation process. (a) Each window is passed through the classier and the classification and epistemic uncertainty is captured. (b) Shows the uncertainty value of the previous window, as well as the current classifier output and uncertainty value. (c) Shows all uncertainty values for the SAR scene, which is the base of the uncertainty heat map. (d) The uncertainty heat map is normalised. (e) The heat map goes through an interpolation function. (f) The interpolated heat map is then scaled to a maximum value of 255.

4.3.4 Uncertainty Feedback

To investigate a method to improve uncertainty estimates from the BCNN, a method of feeding back confident incorrect samples was proposed. The main motivation of this method is to penalise incorrect predictions with high certainty. The aim of this investigation is to determine if there is a reduction in incorrect predictions with high certainty after retraining the BCNN. The method is described in Algorithm 2. This implementation borrows components from Subsections 4.3.1 First the sliding window is applied to the SAR scene, and the uncertainty heat map is generated using the method in Subsection 4.3.3. Then for each window in the SAR scene, the IoU is calculated using (4.4). Using the IoU values and prior knowledge of the bounding boxes of the ground truth targets in Subsection 4.3.1, it is now possible to determine correct and incorrect detections. As described previously in Subsection 4.3.1, a window with an IoU value greater than 0.85 over the ground truth targets is classified as a target, and windows with an IoU value less than 0.85 are classified as clutter. An example of correct detection is when the current window has an IoU value greater than the threshold and the classifier has predicted it to be a target or it has an IoU less than 0.85 and is predicted to be clutter. Conversely, an incorrect detection is recorded when the window has an IoU value greater than the threshold and the classifier output is clutter or the window has an IoU less than the threshold and it's predicted to be a target. Once the IoU has been calculated for each window, a check is performed for each bounding box for all of the ground truth targets. If the window is an incorrect detection with high certainty it is stored in a dataset. The BCNN is then retrained on the newly captured data, and an updated uncertainty heat map is generated.

Algorithm 2 Uncertainty Feedback

Input: SAR scene (1360 x 1074), step size = 5

Output: uncertaintyHeatmap (1360 x 1074)

Initialisation : Uncertainty heat map = 0
load BCNN model
for $i = 0$ to *numXWindows* **do**
  for $j = 0$ to *numYWindows* **do**
    Move sliding window
    Calculate prediction and epistemic uncertainty
    Update uncertainty heat map
  end for
end for
Store all windows coordinates
Normalise uncertaintyHeatmap
for window in all windows **do**
  for bb in ground truth bounding boxes **do**
    Calculate IoU between window and bb
  end for
  if any(IoU > 0.85) and epistemicUncertainty < 0.05 and prediction = 1 **then**
    Append window to confident incorrect targets list
  else if all(IoU = 0) and epistemicUncertainty < 0.05 prediction = 0 **then**
    Append window to confident incorrect clutter list
  end if
end for
Create dataset using incorrect high confident samples
retrain BCNN using much lower learning on a new dataset
Perform target detection
Generate uncertainty heat-map

4.4 PERFORMANCE METRICS

4.4.1 Classification Performance

The most prevalent performance metric for classification tasks is accuracy, which measures the capability of the network to discriminate between various classes. By comparison, accuracy has numerous deficiencies such as distinctiveness and informativeness [19].

In most classification problems, the performance evaluation is completed in two stages. The first stage is the training stage, and the performance metric is used to improve classifier performance. In the second stage, the performance metric is used to evaluate the performance of the network on test data. The best performing classifier can be determined from the confusion matrix as shown in Table 4.2. True positives are a prediction when the model predicts the correct positive class. In a similar way, a true negative is a prediction when the model predicts the correct negative class. In contrast, false positives and false negatives occur when the model incorrectly predicts the positive and negative classes. To illustrate the outcomes, an example is described where a positive class is the classification of an enemy target and a negative class is the classification of a friendly target. All four possible outcomes are described as follows: (tp) the enemy is classified and appropriate action is taken, (tn) a friendly target is classified and the appropriate action is taken, (fp) the friendly target is incorrectly classified, therefore, it is decided to respond with aggression, (fn) enemy target is incorrectly classified as friendly and the enemy target is able to attack.

Table 4.2. Confusion Matrix for Binary Classification

	Actual Positive Class	**Actual Negative Class**
Predicted Positive Class	True positive (tp)	False negative (fn)
Predicted Negative Class	False positive (fp)	True negative (tn)

In order to determine the best performing classifier in the training process, additional performance metrics are used and summarised in Table 4.3.

Table 4.3. Additional Threshold Performance Metrics for Classifier Evaluation (adapted from [19]).

Performance Metric	Formula	Evaluation Focus
Error Rate	$\frac{fp+fn}{tp+fp+tn+fn}$	Measures the miss-classification error, which is the ratio of incorrect predictions over the total number of runs performed.
Precision (p)	$\frac{tp}{tp+fp}$	This measures the positive patterns that are correctly predicted from the total predicted patterns in the positive class.
Recall (r)	$\frac{tp}{tp+fn}$	This measures the fraction of positive patterns that are correctly classified.
F-Measure	$\frac{2*p*r}{p+r}$	This metric describes the harmonic mean between the recall and precision values.

To properly evaluate the performance of the BCNN and CNN, Monte Carlo (MC) runs were conducted for each experiment. For each experiment, ten Monte Carlo runs were conducted to better estimate the model's performance [80]. Owing to the lengthy training times of the models, the MC runs must be limited.

4.4.2 Detection Performance

The detection performance uses the same precision metric as the classifier performance, with the addition of an IoU threshold. Firstly, true positives are identified when the prediction is the same as the ground-truth and the IoU between the window is greater than the IoU threshold. The same applies to true negatives.

4.5 CHAPTER SUMMARY

This chapter provided a detailed description of the implementation of the BCNN as well as details regarding the training and evaluation methods. An overview of the MSTAR dataset was provided since it will be used to benchmark the BCNN against other state-of-the-art classification algorithms.

A sliding-window implementation was described to perform target detection as well as generate ground-truth samples using the SAR scene. The chapter also presented the method to generate uncertainty heat maps to improve the explainability. The chapter further detailed an approach to investigate the effects

of feeding back false confident samples by using the combination of target detection and uncertainty heat map generation.

CHAPTER 5 RESULTS

5.1 CHAPTER OVERVIEW

This chapter details all investigations performed to evaluate and compare the performance of the BCNN in both classification and detection tasks. The chapter consists of several distinct sections. Firstly, the posterior predictive distribution of the BCNN is evaluated using in- and out-of-distribution data (Section 5.2). The classification performance between the BCNN and CNN is presented in Section 5.3 using the MSTAR dataset. In addition, the results are compared with recent classification methods from published works.

To improve the target detection capabilities of the BCNN, a brief investigation was conducted into the feature space of the training data. Data analysis was performed, investigating the effects of varying the overlap between training samples. The overlap was caused by the step of the sliding window during each iteration. If the step size is less than the window size, then there will always be an overlap between samples.

The initial results for the target detection are presented in Section 5.5 where the ground-truth images are generated and used to evaluate the detection performance of the model. Results of the visualisations of the uncertainty of the BCNN are depicted in Section 5.6 which illustrates the conversion of the epistemic uncertainty into a visual image of high- and low-confidence regions in the SAR scene. Lastly, an investigation was performed to evaluate the effects of feeding back confident incorrectly classified samples and retraining the model to reduce the number of high-uncertainty zones. The discussion of the results is provided at the end of each section.

5.2 PREDICTIVE UNCERTAINTY IN BCNNS

In this section, the uncertainty estimates from the BCNN are evaluated for in-distribution and out-of-distribution samples. In this study, in-distribution data refers to samples that are similar to the training dataset distribution, while out-of-distribution data samples do not follow the distribution of the training dataset. In addition, the softmax probabilities are shown for both a CNN and BCNN. The BCNN had added error bars for the predictive variance. Along with the softmax outputs, the epistemic uncertainty was calculated. For the epistemic uncertainty, the softplus activation function was used and was normalised similar to the softmax function. From the hyper-parameter optimisation in Subsection 4.2.8, the BCNN and CNN were trained using the parameters in Addendum A. A total of ten MC runs were conducted, and the random variable assessed is the classification accuracy. The model with the highest classification accuracy was selected. The discussion of the results is provided in Subsection 5.2.3.

5.2.1 In-Distribution Data

Three examples were presented to the BCNN and CNN to evaluate the predictive uncertainty. The comparisons of the predictions between the BCNN and CNN are shown in Figures 5.1, 5.3, and 5.5. The corresponding epistemic uncertainty estimates are shown in Figures 5.2, 5.4, and 5.6.

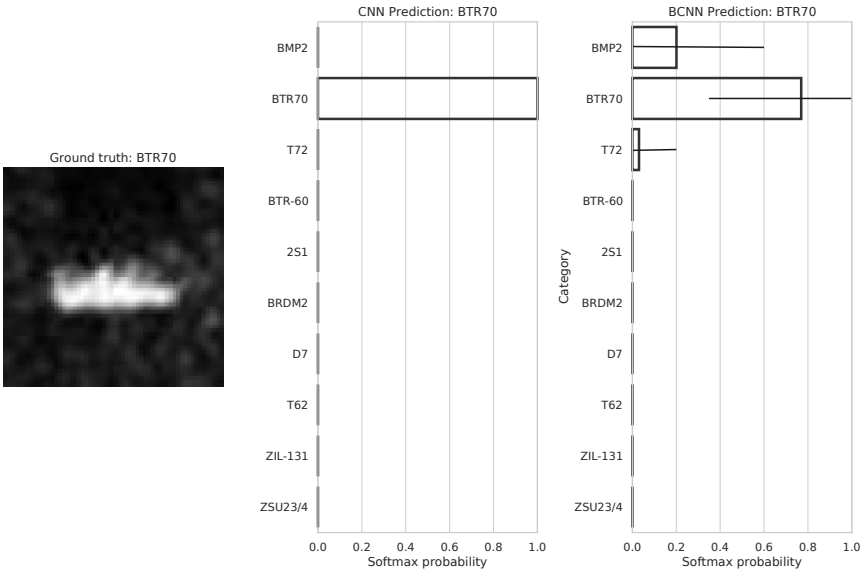


Figure 5.1. Comparison of CNN and BCNN predictions for in-distribution sample one. Both the CNN and BCNN predicted the correct class, with the BCNN having a 77 % probability and the CNN having a 100 % probability.

In Figure 5.1 both networks made correct predictions, with the CNN predicting with absolute certainty and the BCNN giving a 77 % probability to the predicted class. From Figure 5.2, the classes: BMP2, BTR70 and T72 recorded the highest epistemic uncertainties, with the correct class the BTR70 having the highest uncertainty of all the classes.

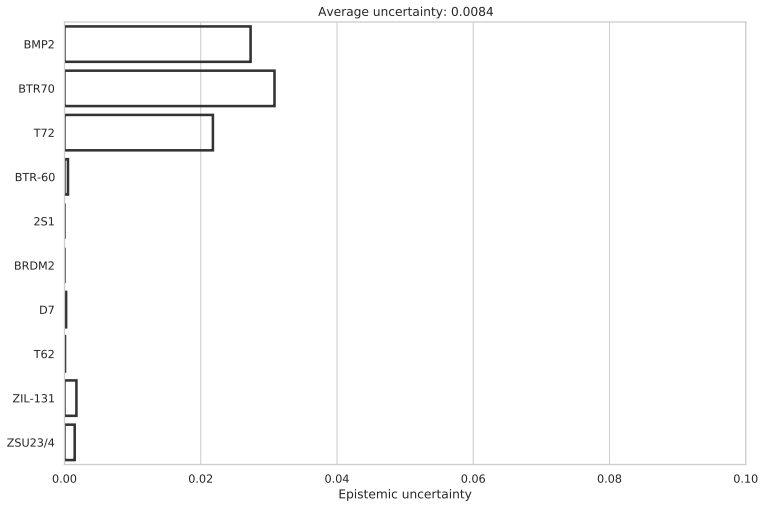


Figure 5.2. Epistemic uncertainty of sample in Figure 5.1. The first three classes have the highest uncertainty compared to the other classes.

From Figure 5.3 is it apparent that the predictive variance was higher than the predictive variance observed in Figure 5.1. This indicates that the BCNN was less certain for this prediction, with the increase in average epistemic uncertainty supporting this deduction.

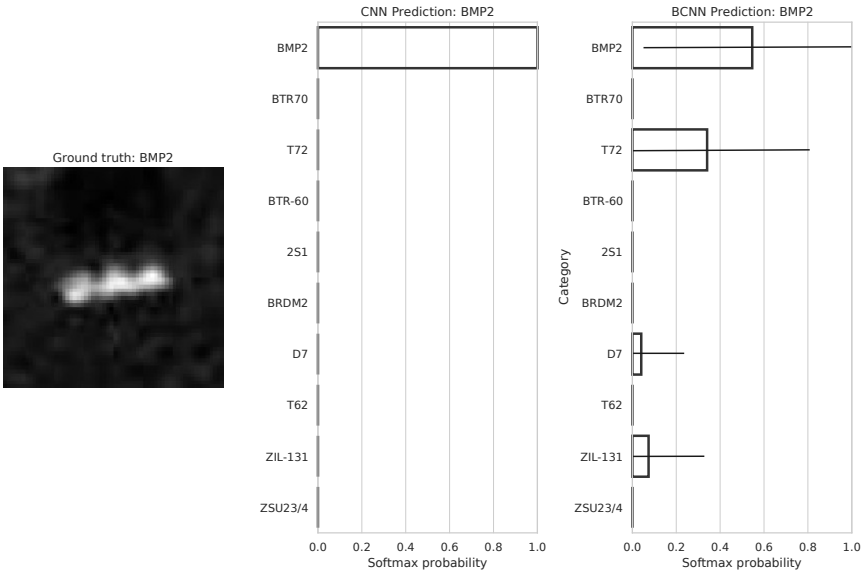


Figure 5.3. Comparison of CNN and BCNN predictions for in-distribution of sample two. The CNN made had a prediction with a softmax probability of 100 % while the BCNN was less certain with a probability of 53 %. The predicted class of the BCNN had a large variance in the predictive variance.

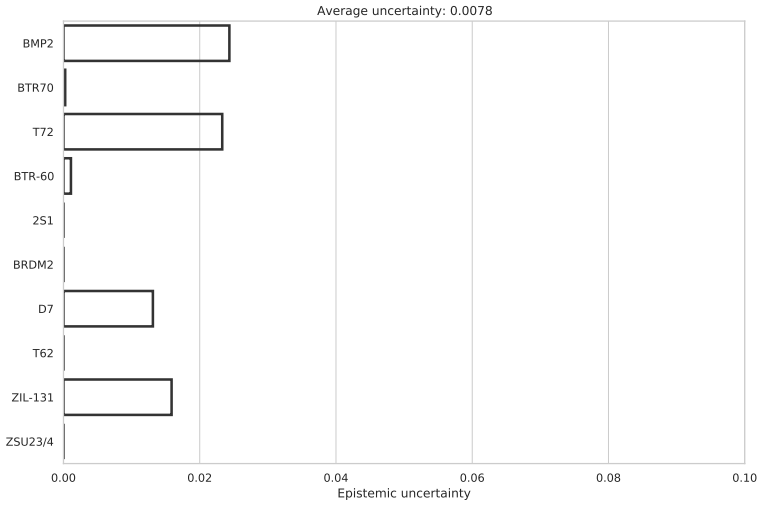


Figure 5.4. Epistemic uncertainty of sample in Figure 5.3. There are numerous classes with almost equal uncertainty. Thus, indicating a spread in the softmax probability in the BCNN's prediction.

It was observed that both models made predictions with absolute certainty as shown in Figure 5.5. In addition, the low average epistemic uncertainty indicates that the BCNN was indeed certain of its

prediction.

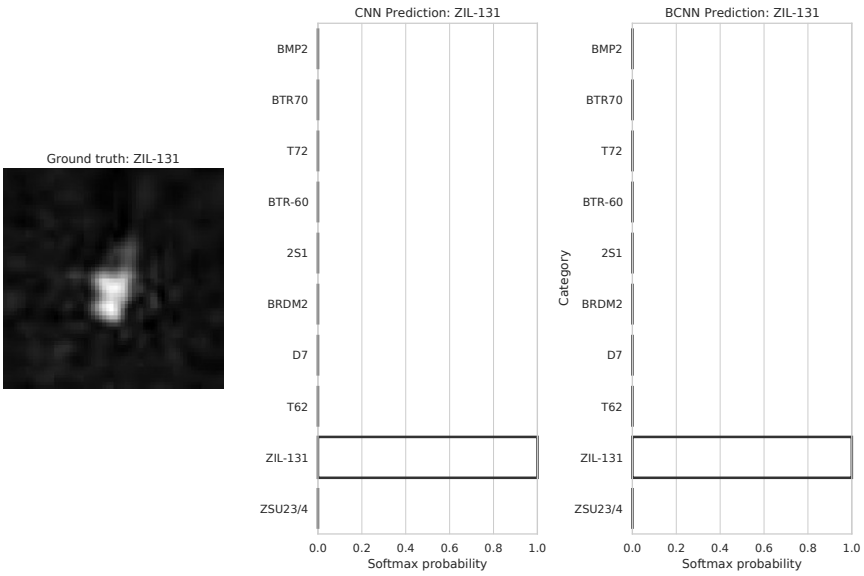


Figure 5.5. Comparison of CNN and BCNN predictions for in-distribution sample three. Both the CNN and BCNN had a predicted softmax probability of 100 %. The recorded predictive variance of the BCNN was approximately zero.

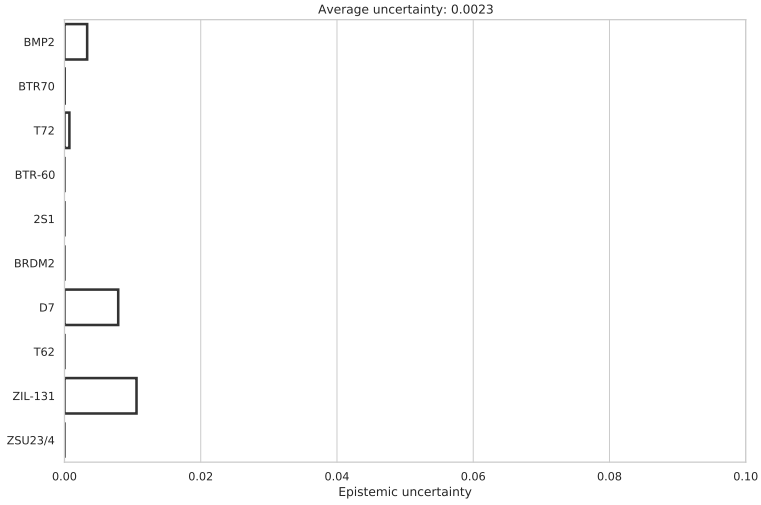


Figure 5.6. Epistemic uncertainty of sample in Figure 5.6. The observed average uncertainty was the lowest for all three in-distribution samples.

5.2.2 Out-of-Distribution Data

For the out-of-distribution data, the first sample was taken from the dataset provided by the CSIR of a region of trees. The second sample was generated from a Gaussian distribution with a mean of 0 and a variance of 0.1. The two examples were presented to the BCNN and CNN to evaluate the predictive uncertainty. The comparisons of the predictions between the BCNN and CNN are shown in Figures 5.7 and 5.9. The corresponding epistemic uncertainty estimates are shown in Figures 5.7 and 5.9.

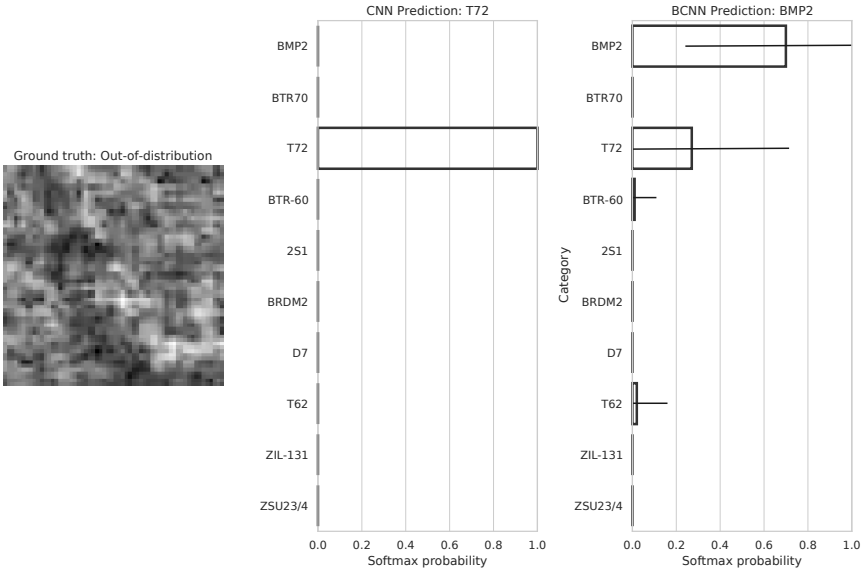


Figure 5.7. Comparison of CNN and BCNN predictions for out-of-distribution sample one. The BCNNs had four numerous classes with probabilities greater than zero, while the CNN only predicted one class.

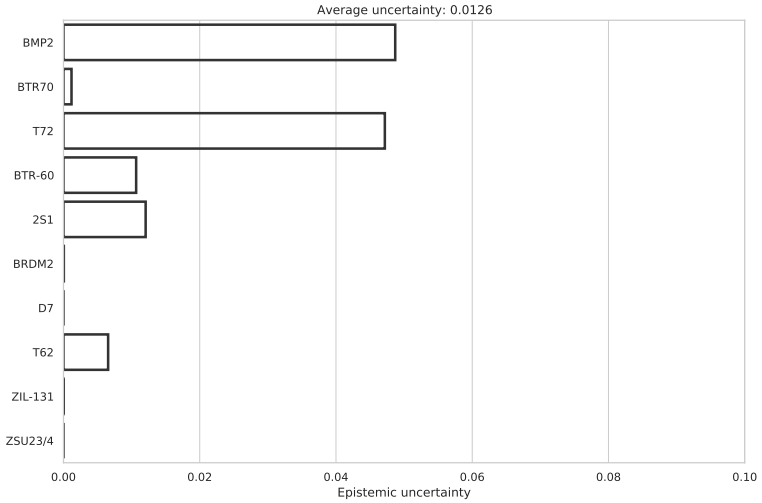


Figure 5.8. Epistemic uncertainty of sample one. This was the highest recorded average uncertainty for both in- and out-of-distribution samples.

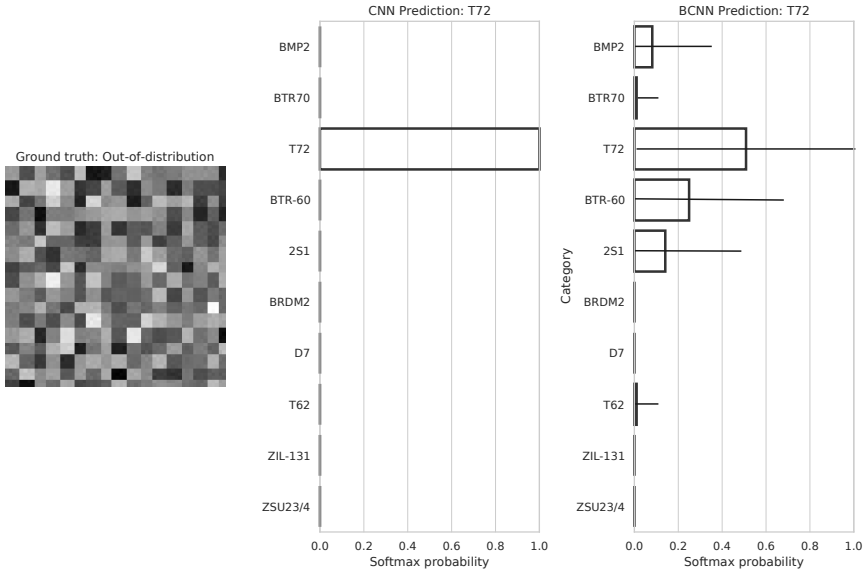


Figure 5.9. Comparison of CNN and BCNN predictions for out-of-distribution sample two. A similar result is observed for the CNN, with a single absolute certain prediction made, while the BCNN had a much more varied prediction. A large predictive variance was observed for multiple classes.

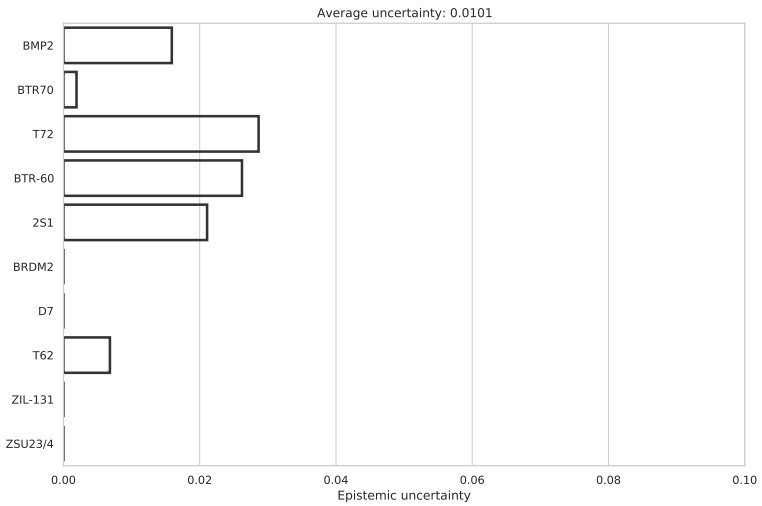


Figure 5.10. Epistemic uncertainty of sample two. The average uncertainty recorded was 0.0101, which was lower than the average uncertainty for sample one. This suggests that sample one was less similar to the training data than sample two.

5.2.3 Discussion of the Predictive Uncertainty of the BCNN

5.2.3.1 In-Distribution Data

From the comparison of softmax outputs for the CNN and BCNN, it is apparent that the CNN makes predictions with absolute confidence by allocating a 1.0 probability to a single class, while the BCNN had a much more varied prediction. It is noted that in Figure 5.5, both the CNN and BCNN were highly confident with the error bar for the BCNN at approximately zero for the in-distribution sample.

The epistemic uncertainty provided additional insight into the confidence of the BCNN with a higher average epistemic uncertainty recorded for predictions with lower confidence values. In the example in Figure 5.5, both the BCNN and CNN were confident in their predictions while the BCNN recorded its lowest average epistemic uncertainty for all in-distribution samples. The epistemic uncertainty provided additional information regarding confidence, especially when highly confident predictions were made.

An important observation is that the epistemic uncertainty for the predicted class was the highest in all of the examples. This can be interpreted as the predicted class having the highest uncertainty, and this corresponds with the error bars in the softmax outputs with the longest bar being over the predicted

class. Despite the predicted class having the highest uncertainty, it does not imply the model is not confident. This is observed in Figure 5.6, where the predicted class has the highest uncertainty but the average epistemic uncertainty for all ten classes is relatively low when compared to average epistemic uncertainty in Figure 5.2 and 5.4.

5.2.3.2 Out-of-Distribution Data

For the out-of-distribution samples in Figures 5.8 and 5.9, it was observed that the CNN still made predictions with absolute certainty, therefore, with maximum confidence, which is clearly unreasonable. Despite being incorrect, this prediction with absolute certainty is very misleading and is not suitable for real-world applications that directly affect humans. On the other hand, the BCNN provided additional information indicating that it was not confident in its prediction. It is noted that the predictive variance of the softmax outputs was higher than the in-distribution variance. An increase in the average epistemic uncertainty was recorded.

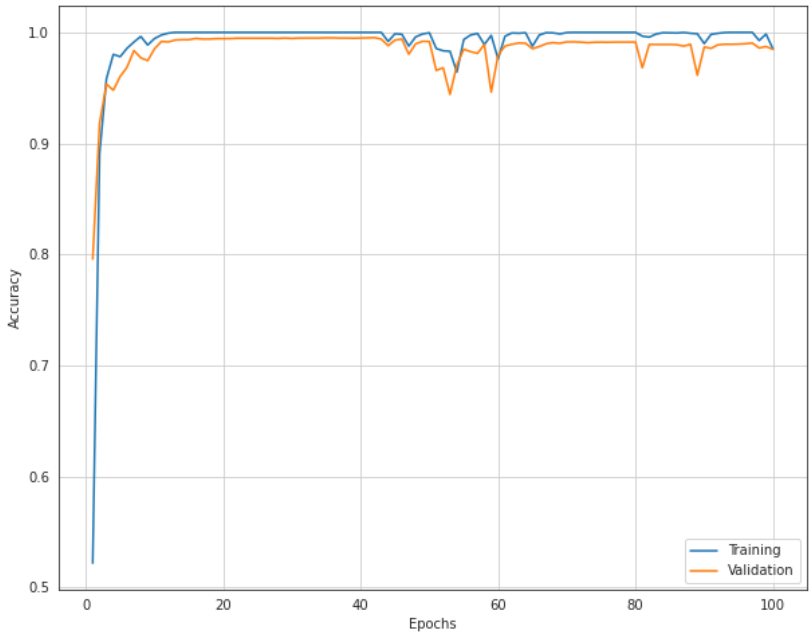
It is apparent that the CNN has no method of managing out-of-distribution data, and the BCNN provides a solution through uncertainty estimation. When using epistemic uncertainty, it is possible to allow the BCNN to withhold its decision when the uncertainty is above a specific threshold. This allows for undecided samples to be evaluated by a human specialist rather than making an incorrect overconfident prediction.

5.3 COMPARISON OF CLASSIFICATION PERFORMANCE BETWEEN BCNN AND CNNS

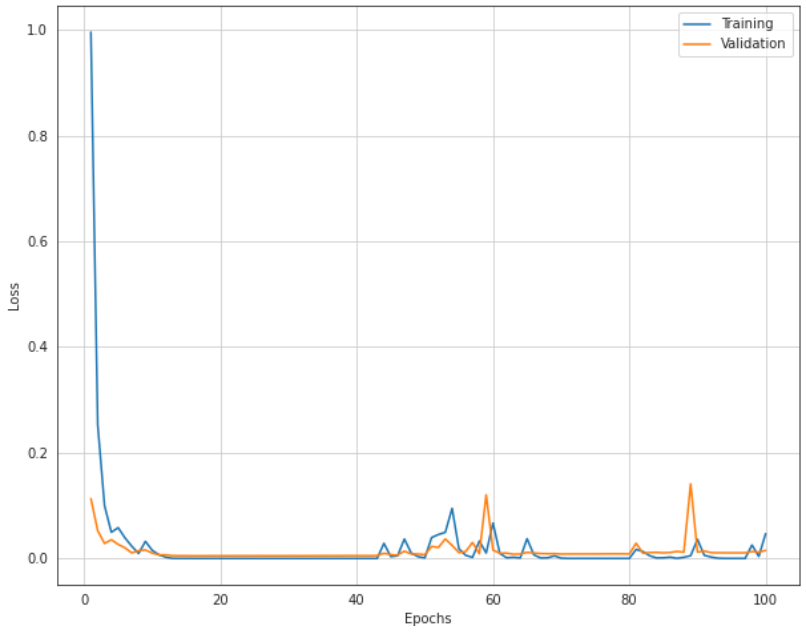
To compare the classification performance between the BCNN and CNN, the hyper-parameters for each network were determined using the Bayesian optimisation technique described in Subsection 4.2.8. The hyper-parameters for both networks are given in Addendum A. The results are discussed in Subsection 5.3.4.

5.3.1 CNN

The CNN was trained using the hyper-parameters in Table A.2 with the six-layer architecture described in Table 4.1. Ten MC runs were performed and the training accuracy and loss curves for a random MC run are shown in Figure 5.11.



(a) The average training accuracy achieved was 1.0, with a similar result for the validation accuracy.



(b) The training and validation curves converge after 20 Epochs, indicating that the model was not over-fit.

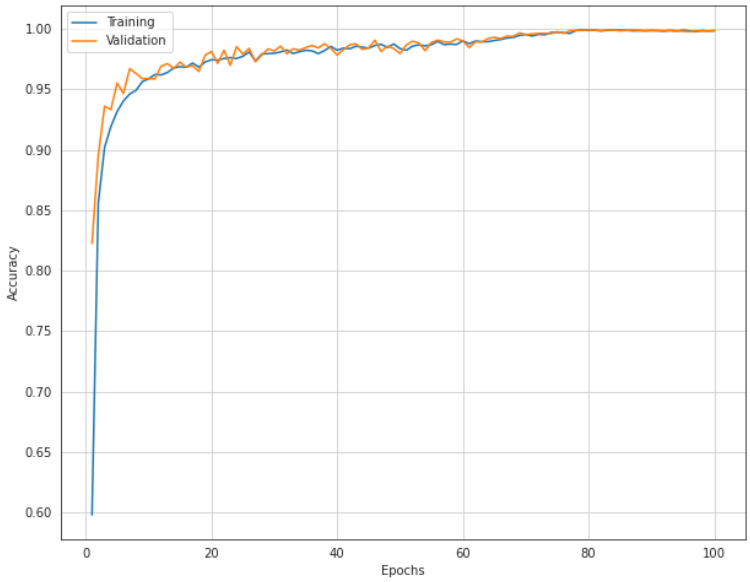
Figure 5.11. (a) Classification accuracy of standard CNN for 100 Epochs with both the training and validation curves. (b) Training and validation loss curves.

The classification performance was evaluated using the test set from the MSTAR data. The confusion matrix was constructed and is shown in Figure 5.12.

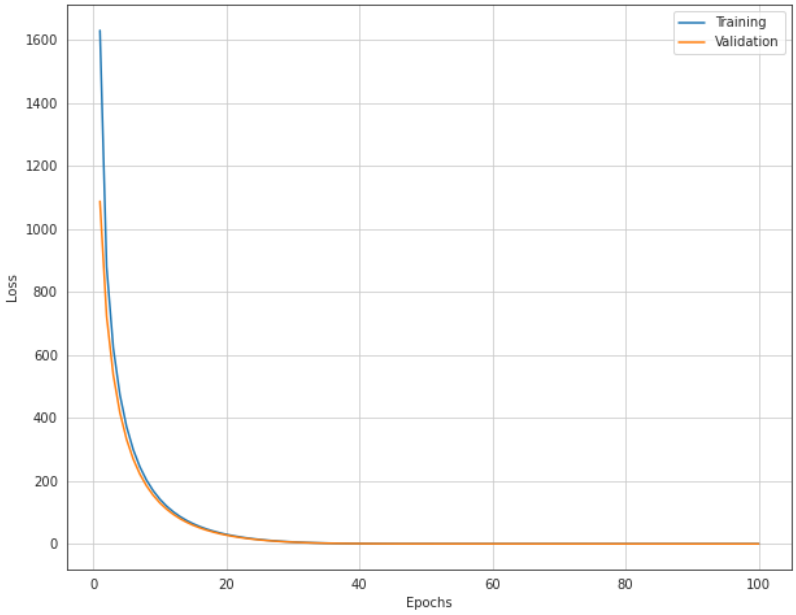
Figure 5.12. Confusion matrix for the CNN for all ten classes of the MSTAR dataset. The columns are normalised using the total number of actual samples, and the rows are normalised using the total number of predicted samples. The bottom right cell is the total accuracy of the BCNN

5.3.2 BCNN

The BCNN was trained using the hyper-parameters in Table A.1 with the six-layer architecture described in Table 4.1. Ten MC runs were performed and the training accuracy and loss curves for a random MC run are shown in Figure 5.13.



(a) The training accuracy increased gradually before plateauing after 80 Epochs.



(b) The training and validation curves converged after 30 Epochs.

Figure 5.13. (a) Classification accuracy of the BCNN for 105 Epochs with both the training and validation curves. (b) Training and validation loss curves.

The classification performance was evaluated using the test set from the MSTAR data. The confusion matrix was constructed and is shown in Figure 5.14.

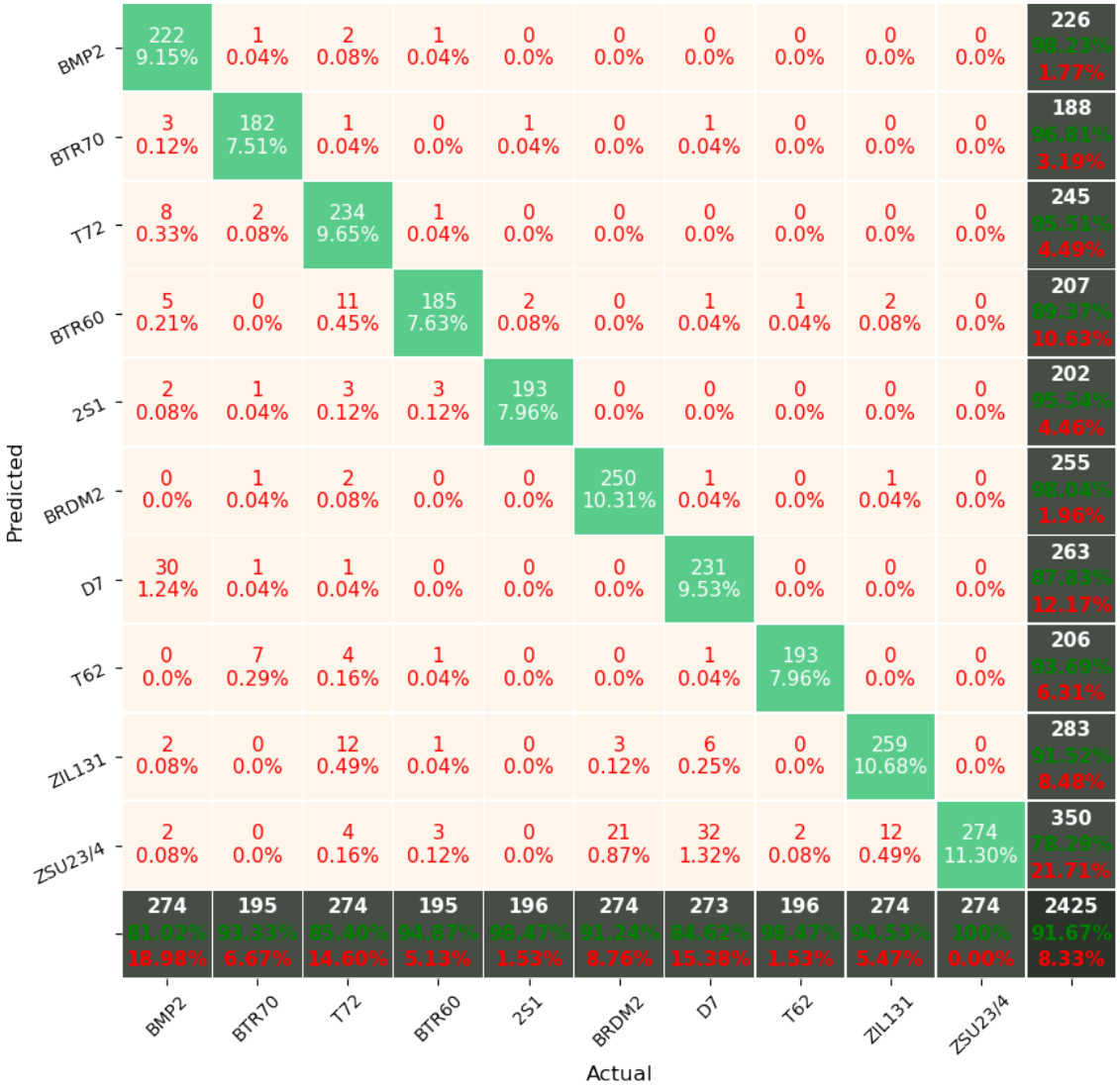


Figure 5.14. Confusion matrix for the BCNN for all ten classes of the MSTAR dataset. The columns are normalised using the total number of actual samples, and the rows are normalised using the total number of predicted samples. The bottom right cell is the total accuracy of the BCNN

5.3.3 Comparison of Classification Performance

The classification accuracy and F-1 scores for the CNN and BCNN are tabulated in 5.1. Additional implementations were selected to distinguish the performance of the BCNN compared to methods from recent publications. The selected methods used either CNNs or an ensemble of different learning methods that included a CNN.

Table 5.1. Comparison of Classification Accuracy between BCNN, CNN, and Implementation from the Literature.

Performance metric	CNN	BCNN	[81]	[18]	[17]
Classification accuracy	0.968	0.931	0.9913	0.986	0.964
F-1 Score	0.937	0.918	-	-	-

5.3.4 Discussion of the Comparison of Classification Performance between BCNN and CNN

From Table 5.1 it was concluded that the CNN performed better than the BCNN by achieving a higher classification and F-1 score while scoring the highest classification accuracy of 96.8%. This was expected as a similar result was achieved in [4]. In Figure 5.11(a) and 5.13(a), there is an observable difference when the classification accuracy reaches a plateau with the CNN converging faster than the BCNN. The difference in batch sizes could be a factor since the hyper-parameter optimisation function proposed a much smaller batch size for the BCNN than the CNN. Another reason for the faster convergence could be the increased number of parameters in the BCNN as the BCNN had twice as many parameters to optimise.

When compared to DNNs from the literature, the BCNN achieved a similar classification performance while retaining the ability to estimate the uncertainty in its prediction. The trade-off with a slight decrease in classification performance is undoubtedly worth the additional information gained from the uncertainty. Furthermore, the ability to eject predictions with low confidence scores should result in improved classification accuracy and close the performance gap between BCNNs and CNNs.

5.4 DATA ANALYSIS FOR TARGET DETECTION DATA

Data analysis is an important step when developing any ML algorithm. Principal Component Analysis (PCA) was performed to investigate the separability between the target and clutter samples. It has been shown that an increase in separability leads to improved predictive performance [82]. This section investigates the effect of varying the overlap parameter for the sliding window when generating the samples for the clutter, since the same sliding-window method is used to extract samples from the individual SAR scenes to generate the dataset for the detector.

5.4.1 Investigation of Varying Overlap of Training Samples

Initially, the clutter samples were captured using a step size of five pixels. This resulted in a large number of samples being generated with numerous samples overlapping one another. As illustrated in Figure 4.5, the overlap of samples is the area of intersection between those samples. The 2D dimensional PCA space is shown in Figure 5.15 with samples that were captured with a step size of five pixels.

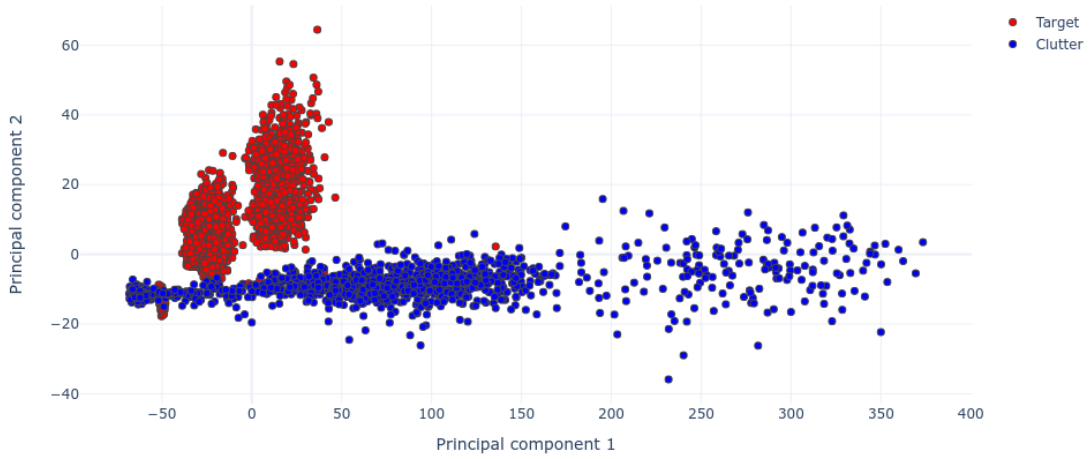


Figure 5.15. 2D PCA plot for samples generated using a step size of five pixels. Four distinct clusters are observed.

The 3D PCA space is shown in Figures 5.16 and 5.17 at two different angles.

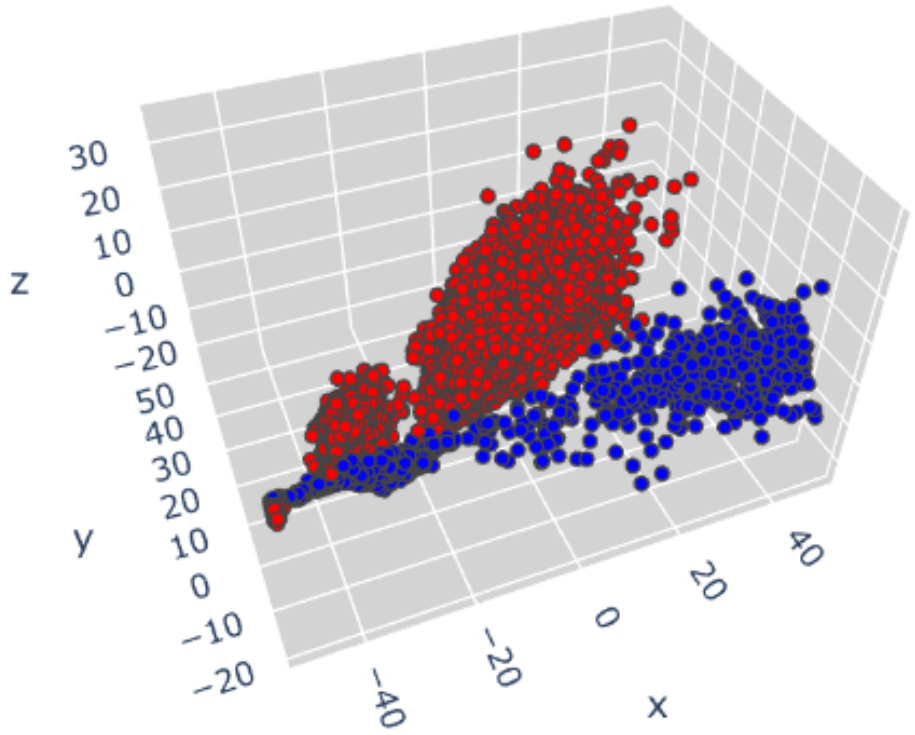


Figure 5.16. 3D PCA plot for samples generated using a step size of five pixels. There appear to be two planes in PCA space where the target and clutter clusters are concentrated.

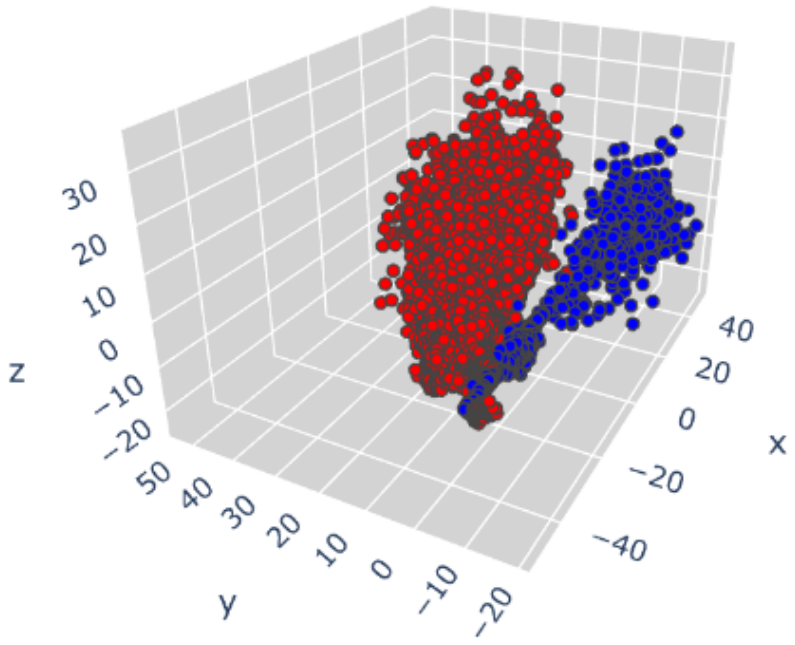


Figure 5.17. 3D PCA plot for samples generated using a step size of five pixels taken from a different orientation. This orientation emphasises the separation between two classes.

The 2D PCA space is shown in Figure 5.18 with samples captured with no overlap.

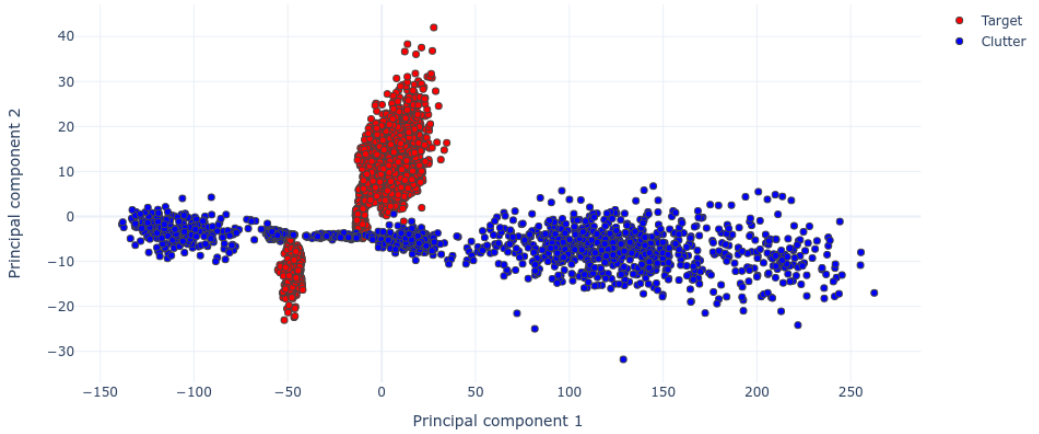


Figure 5.18. 2D PCA plot for samples generated with no overlap. Three main clusters are now observed. One cluster on the X-axis for the clutter sample, and two opposing clusters for the target samples.

The 3D PCA space is shown in Figures 5.19 and 5.20 at two different angles.

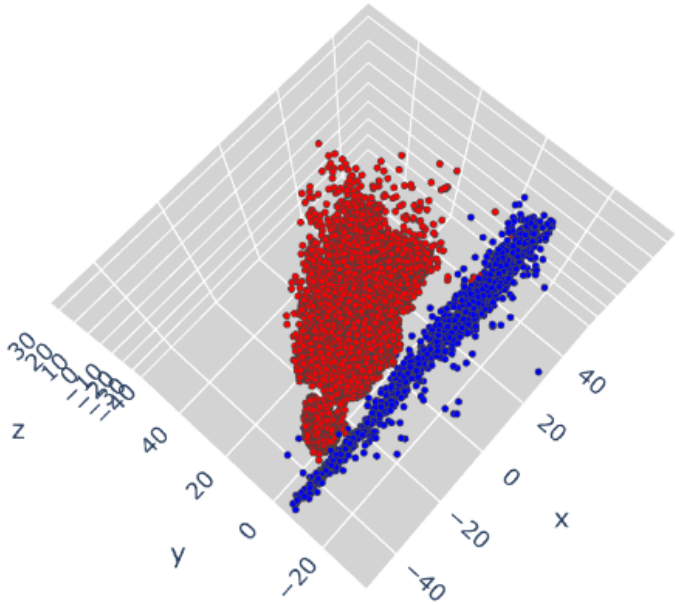


Figure 5.19. 3D PCA plot for samples generated with no overlap. Compared to the 3D PCA plot in Figure 5.16, the clusters appear more condensed for the samples generated with no overlap.

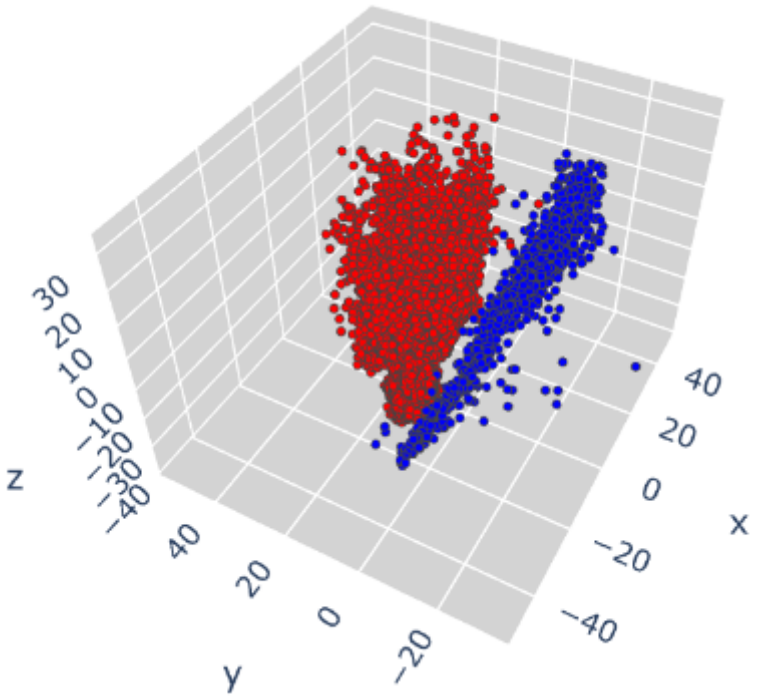


Figure 5.20. 3D PCA plot for samples generated with no overlap. Illustrating the increase in separation between the target and clutter clusters.

5.4.2 Discussion on the Investigation of Varying Overlap of Training Samples

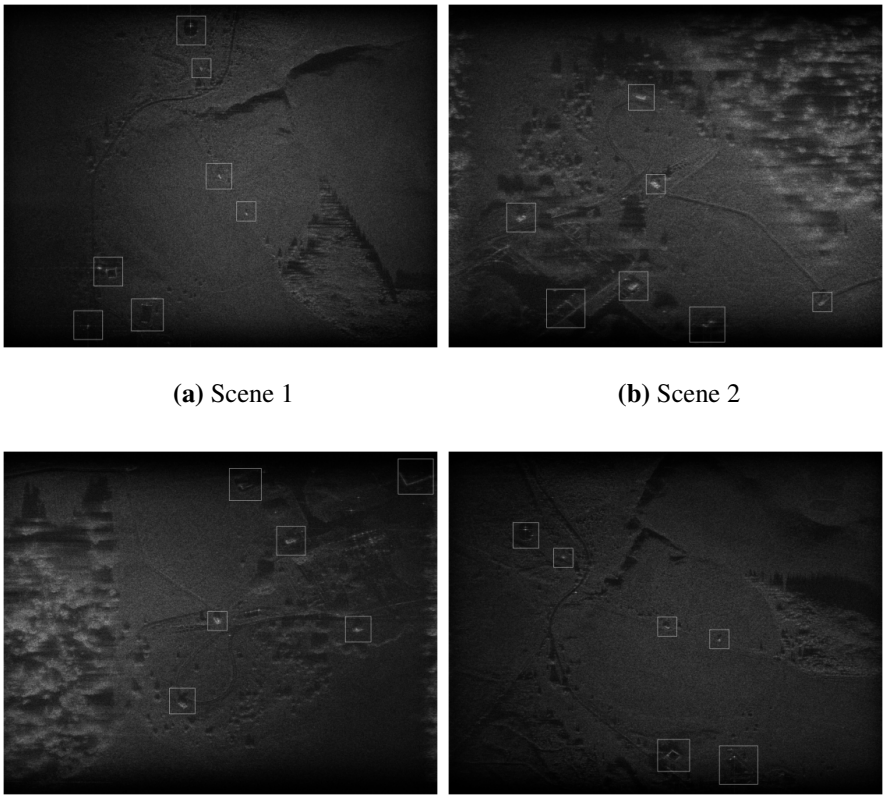
From the PCA plots in Figures 5.15 to 5.17, three distinct clusters can be observed for the target samples, with one single large cluster for the clutter samples. It was noted that there was a region where the two target clusters and clutter clusters intersected. This large entanglement can reduce detection performance if the network is not able to distinguish between the two classes. Thus, the overlap between cluster samples must be removed in order to reduce the correlation between neighbouring samples taken in proximity to one another. The results of the removal of the overlap between neighbouring samples are observed in Figures 5.15 to 5.20. It was observed that the number of target clusters was reduced to only two. In addition, the intersection of the target and clutter clusters was reduced when compared to the PCA plot that was generated from samples with significant overlap. The additional clusters could have been generated due to the increase in correlation of samples that have repeating patches owing to regions of overlap. When the overlap was removed, this led to the samples being more independent from each other.

5.5 TARGET DETECTION

The detection method detailed in Section 4.3.2 was implemented. The results for the detection and uncertainty heat maps were generated using the same four scenes as in Figure 5.21.

5.5.1 Ground Truth

The ground-truth measurements were generated from Figure 5.21. Target samples must have an IoU greater than 0.85 to be labelled as a ground-truth target. Clutter samples are defined as samples with an IoU of less than 0.85. Table 5.2 shows the number of ground-truth samples for both target and clutter contained in each of the scenes for a step size of eight.



(a) Scene 1 **(b)** Scene 2

(c) Scene 3 **(d)** Scene 4

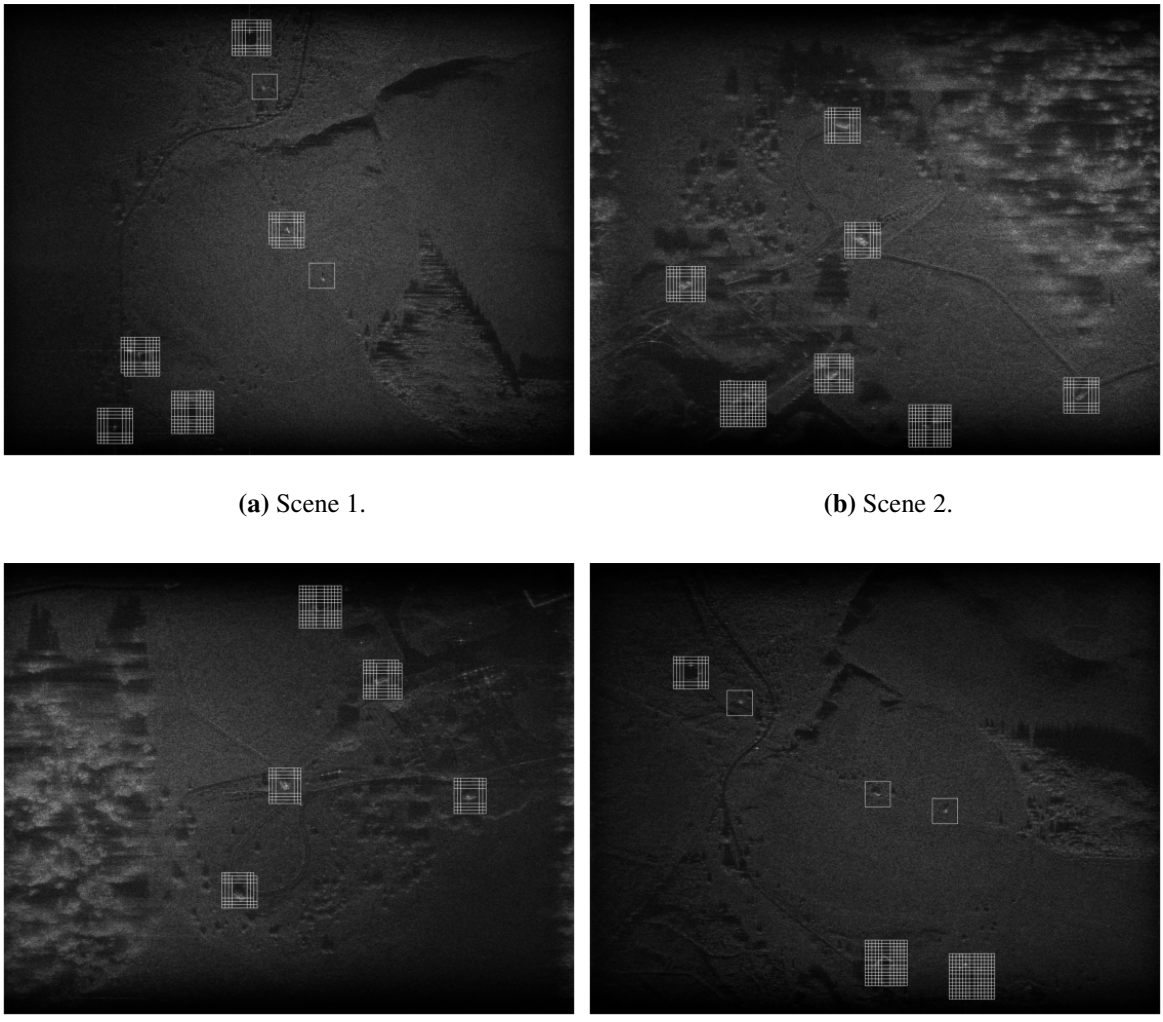
Figure 5.21. SAR scenes provided by the CSIR with corresponding bounding boxes around the targets. (a) Scene 1 was captured over a large open grass area, it also contains a small portion of the tree to the bottom right. (b) Scene 2 contains a large collection of trees that is next to a few man-made structures. (c) Scene 3 is the same area as scene 2 but captured at a different orientation, at this angle more of the forest area is exposed. (d) Scene 4 was captured in a similar region as scene 1 but at a different orientation. This scene highlights more of the tree and hill areas.

Table 5.2. Number of Ground-Truth Samples for Targets and Clutter

Image	Number of ground-truth targets	Number of ground-truth clutter
(a)	113	20588
(b)	226	20475
(c)	99	20602
(d)	106	20595

The total number of ground-truth targets for Table 5.2 are visualised in Figure 5.22. From Table 5.2, the ground-truth data is severely imbalanced with a disproportionate number of ground-truth clutter samples to ground-truth target samples. As stated previously in Subsection 4.3.2, the detector uses the MSTAR dataset for the target data and the ground-truth clutter samples from Table 5.2 as the clutter data. The imbalance of the increased number of ground-truth clutter samples is mitigated by randomly selecting an equal amount of ground-truth clutter samples as target samples in the MSTAR dataset.

Figure 5.22 corresponds with the detector correctly detecting every possible target in each scene. It is noted that there are a number of overlapping bounding boxes over the targets. In general, the current detection algorithm uses an additional network to predict the bounding box coordinates. The prediction of the bounding boxes can be addressed in future works. Furthermore, the main objective of this study is the improvement in explainability, and the current method is sufficient to address the research questions posed.



(a) Scene 1. **(b)** Scene 2.

(c) Scene 3. **(d)** Scene 4.

Figure 5.22. Illustration of the ground-truth targets for each scene. (a) Scene 1 has seven ground-truth targets. (b) Scene 2 also has seven ground-truth targets. Scene 3 has five ground-truth targets, due to the orientation two targets are not visible at this angle. (d) Scene 4 has six ground-truth targets.

From Figure 5.22, it is noted that there are multiple detections for each target. This is caused by the larger target, such as the building, where multiple windows have an IoU greater than the threshold. As a result, there are numerous redundant detections. However, this method is sufficient to further investigate the research problem.

5.5.2 Target Detection Performance

The results for the target detection are shown in Figures 5.23 to 5.26 using the ground-truth data in Subsection 5.5.1. Correct detections are represented by green bounding boxes and incorrect detections are represented by red bounding boxes. It is noted that there are multiple detections for specific targets. This is factored into the results to generate confusion matrices. This would increase the difficulty of detecting every target in the ground-truth data as there are significantly more samples. Since the precision is used to measure the performance of the detector and not the accuracy, the proposed implementation still results in a fair evaluation of the detector's performance.

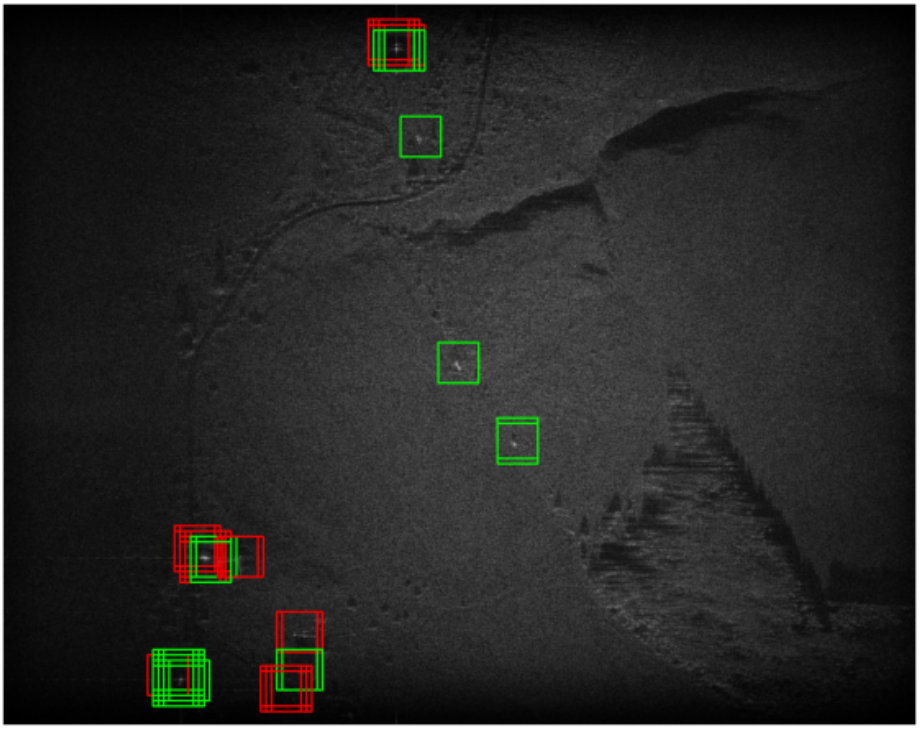


Figure 5.23. Target detection using BCNN for scene 1. There are correct detections for all seven targets, with false detections recorded at the edges of structures.

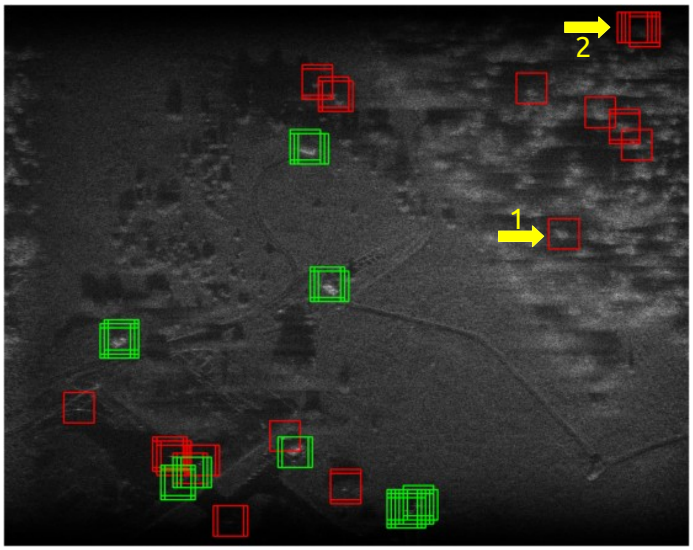


Figure 5.24. Target detection using BCNN for scene 2. The majority of the targets were detected correctly. The tree area did result in multiple false detections.

In Figure 5.24, the numbered arrows show two examples of incorrect predictions of trees as targets because they have similar characteristics as the targets.

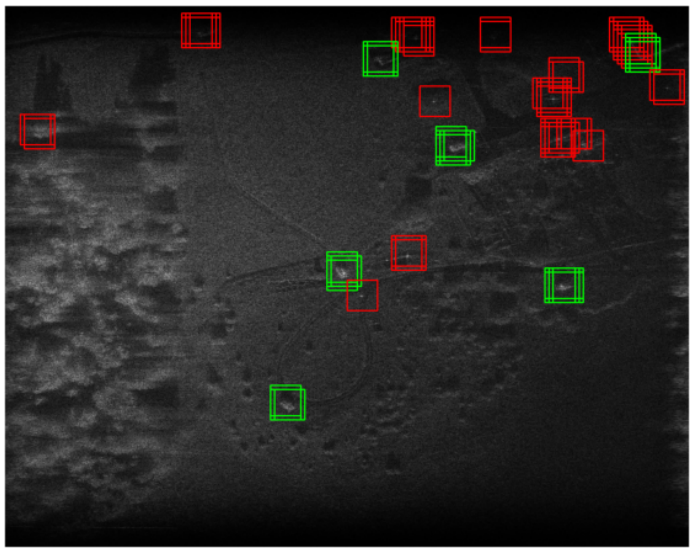


Figure 5.25. Target detection using BCNN for scene 3. Despite all of the ground-truth targets being correctly detected, there are far more incorrect detections. This illustrates the different orientation angles can have on SAR measurements. Previously, the majority of false detections were concentrated on the trees, and now they are focused on the area near the structures.

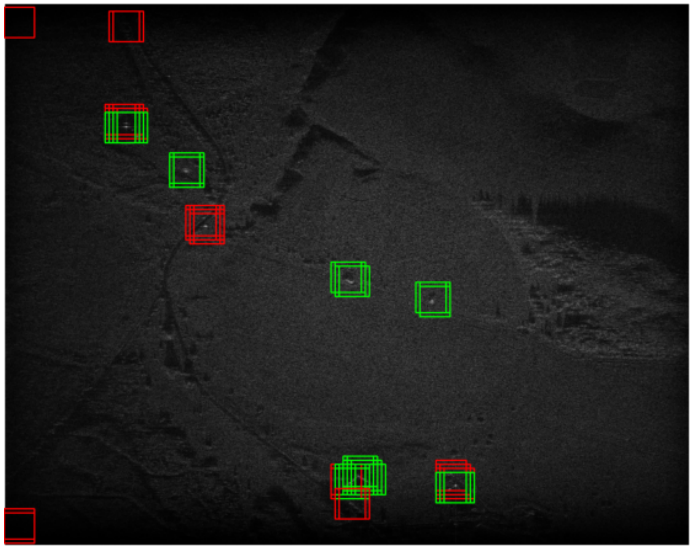
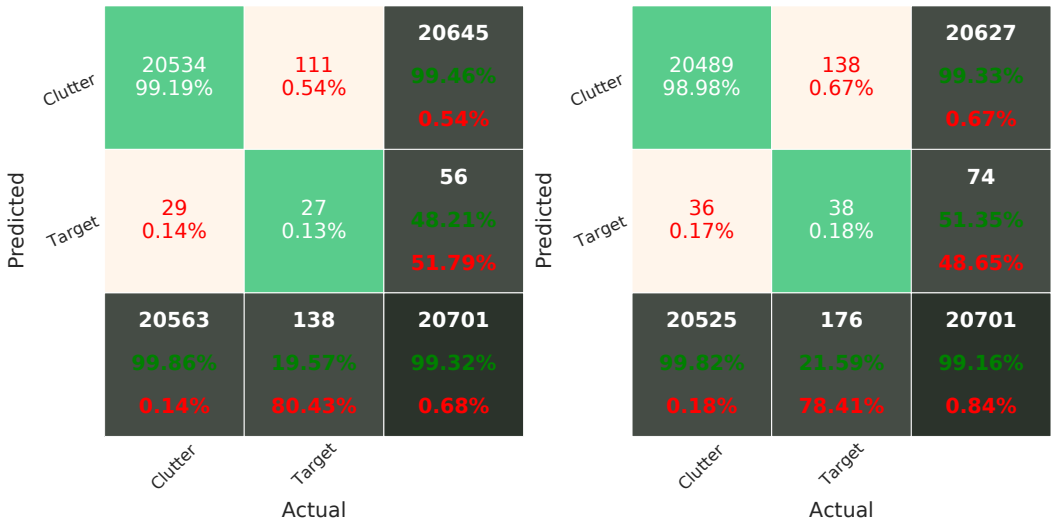


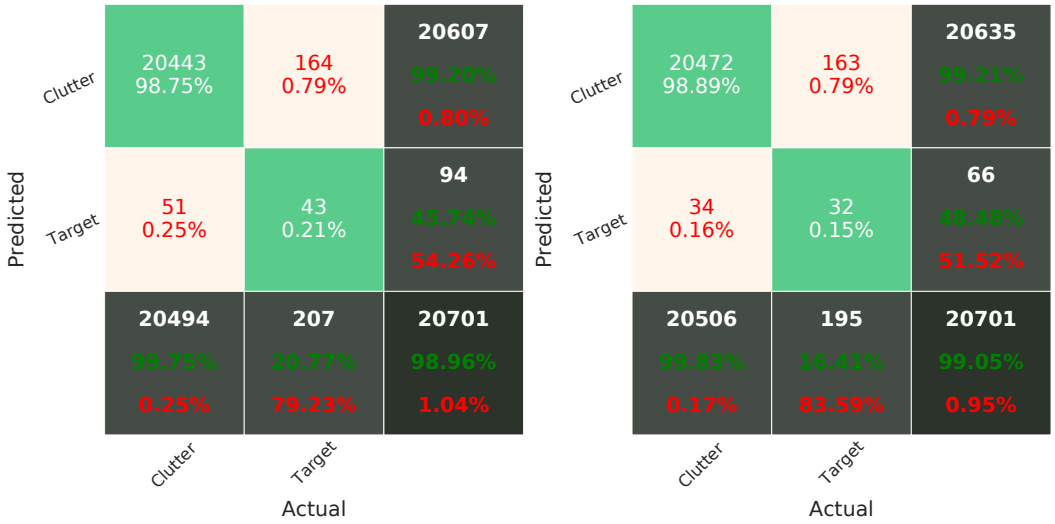
Figure 5.26. Target detection using BCNN for scene 4. From the detector results in scenes 1 and 4, it is apparent that the open grass area had a lower number of false detections than the scenes with a large forest area. The change in orientation did result in more incorrect detections than in scene 1, similar to what was observed in scene 3.

To evaluate the detection performance of the BCNN detector, the method detailed in Subsection 4.4.2 was used. Firstly, the confusion matrices for each scene is shown in Figure 5.27. On the confusion matrix, the percentages of all the samples predicted to belong to each class that are correctly and incorrectly classified are shown in the column on the right of the plot. The percentages of all the examples belonging to each class that are correctly and incorrectly classified are shown in the row at the bottom of the figure. The percentages on the far right columns are normalised using the total number of predicted samples for each class, and the percentages on the bottom row are normalised using the total number of available samples. The precision scores were calculated from the confusion matrices and are shown in Table 5.3.



(a) Scene 1.

(b) Scene 2.



(c) Scene 3.

(d) Scene 4.

Figure 5.27. Confusion matrices for detections in each scene. The columns are normalised using the total number of actual samples, and the rows are normalised using the total number of predicted samples. The bottom right cell is the total accuracy of the BCNN. The columns on the far right represent all the samples predicted to belong to each class, which are correctly and incorrectly classified.

Owing to the imbalance in the number of targets and clutter, there are significantly more clutter detections, as seen in Figure 5.27. As a result, the accuracy of over 99 % is recorded for all scenes. However, this is inaccurate and falsely depicts the performance. The target accuracy shows a much

better representation of the true performance, which is the block above the accuracy on the bottom right.

Table 5.3. Precision of Detection using BCNN

Image	Number of ground-truth targets	Precision
(a)	113	0.472
(b)	135	0.562
(c)	107	0.311
(d)	100	0.508

5.5.3 Discussion of Target Detection Results

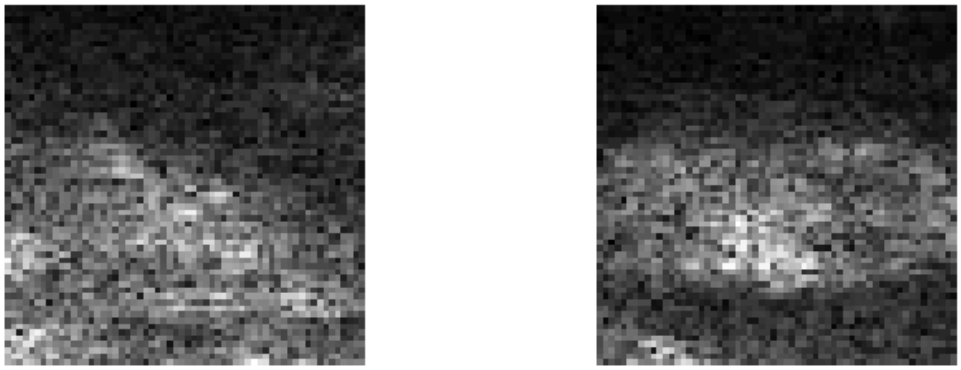
5.5.3.1 Ground Truth

The number of samples generated for the ground-truth scenes is varied by the step size used and the IoU threshold. By increasing the step size, fewer ground-truth samples were created. This affected the detection capabilities of the BCNN since the model was trained on data captured with the target centred in the middle of the image. Another factor was the size of the bounding boxes around the targets. If the bounding boxes are too small, it makes the conditions to meet the IoU threshold more difficult. This can result in missed detections as the network recognises the target but the amount of overlap is insufficient for it to be labelled as a correct detection.

5.5.3.2 Target Detection Performance

The detection performance over the four scenes is varied, with the lowest precision recorded for scene 3 and the best precision recorded for scene 2. An interesting observation is that scenes 2 and 3 were of the same region, but captured at a different orientation. The discrepancy in detection performance could be a result of the reduced amplitude in that region. This is caused by a lower Radar Cross Section (RCS) of the target at this orientation and the decrease in spotlight illumination since the region would have been further from the radar.

The BCNN was able to correctly detect the majority of the ground-truth targets. Despite the target data only being supplied by the MSTAR, the BCNN was still able to detect non-vehicular targets such as a building. However, numerous missed detections were recorded over the large region of trees in Figures 5.24 and 5.25. Upon closer inspection, the miss-detected targets in the trees appeared to resemble targets. Examples of these are shown in Figure 5.28.



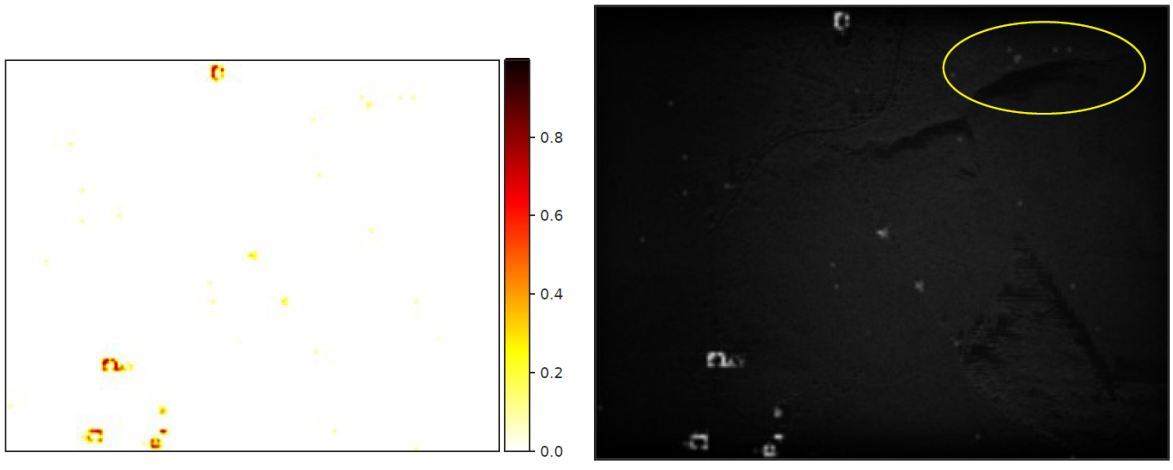
(a) Missed detection 1 from Figure 5.24. **(b)** Missed detection 2 from Figure 5.24.

Figure 5.28. Examples of missed detections for trees that appear to be targets.

Most of them had a bright region in the centre with a form of shadow similar to the MSTAR data. Given an increased amount of training samples, the network should be able to correctly distinguish them.

5.6 UNCERTAINTY HEAT MAPS

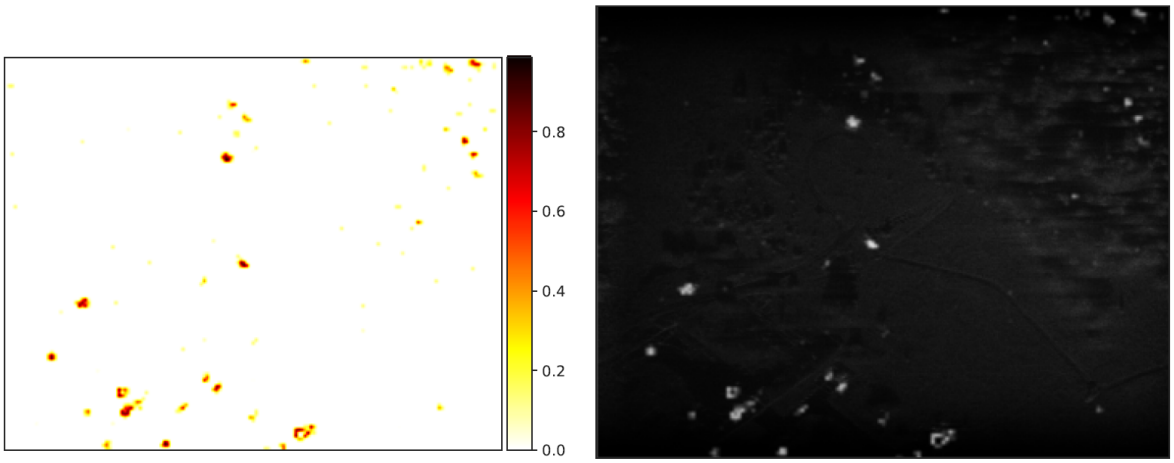
The uncertainty heat maps were generated using the method described in Subsection 4.3.3 and are shown in Figures 5.29(b) to 5.32(b).



(a) **(b)**

Figure 5.29. Normalised uncertainty heat map of scene 1 (a). Uncertainty heat map superimposed onto scene 1 (b), the yellow ellipse contains low-certainty clutter detection.

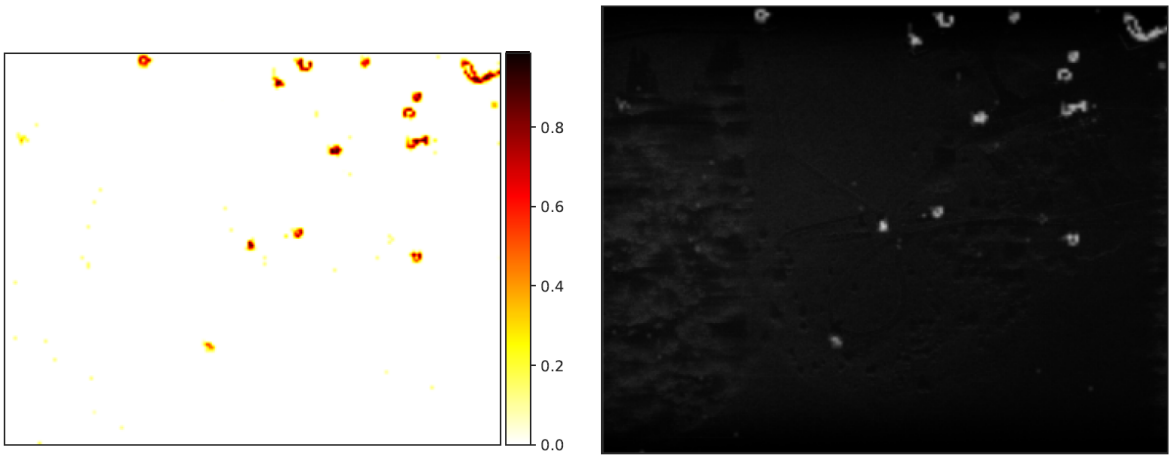
The region contained in the yellow ellipse has examples of low-certainty clutter detections. The highest uncertainty regions are observed over the targets and in the region in the yellow ellipse.



(a) **(b)**

Figure 5.30. Normalised uncertainty heat map of scene 2 (a). Uncertainty heat map superimposed onto scene 2 (b). A group of high uncertainty detections are observed over the forest area.

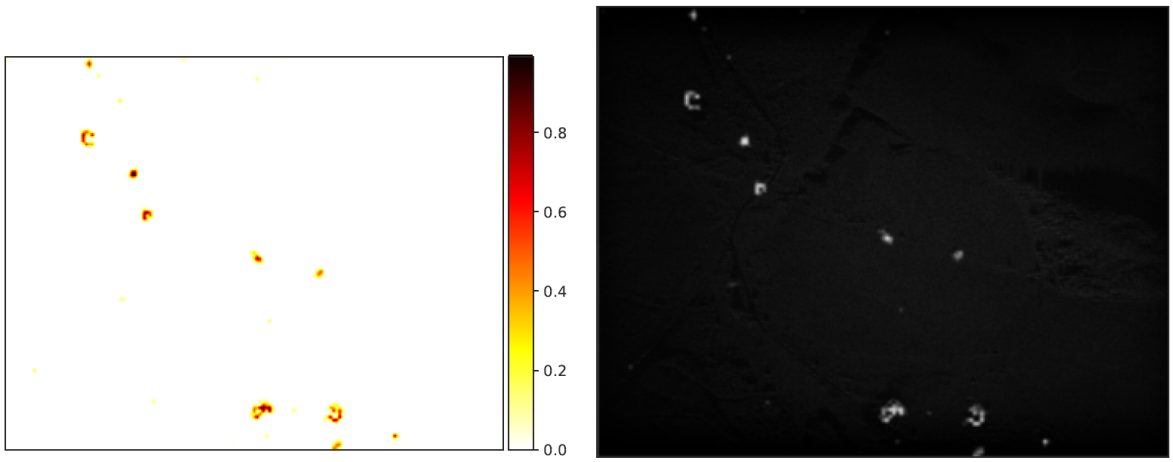
The area of dense trees and the regions over the targets have numerous high-uncertainty regions.



(a) **(b)**

Figure 5.31. Normalised uncertainty heat map of scene 3 (a). Uncertainty heat map superimposed onto scene 3 (b). There appear to be high uncertainty areas at the edges of vehicles and structures. With the outline of the building being particularly prominent, an example of this is observed at the top right.

The highest uncertainty regions are concentrated in the top right over the buildings. It is noted that there is a small cluster of trees with high uncertainty.



(a) **(b)**

Figure 5.32. Normalised uncertainty heat map of scene 4 (a). Uncertainty heat map superimposed onto scene 4 (b). There are minimal areas of high uncertainty over the scene, except at the targets.

It was observed that the uncertainty estimate for scene 4 was low, except over the target. A small region by the hill appeared to have high uncertainty.

5.6.1 Discussion of Uncertainty Heat Map Results

From Figures 5.29(b) to 5.30(b), the most distinct elements are the regions that contain targets. Across all examples, the uncertainty over the targets was the highest. This is attributed to the results in Section 5.2, where it was shown that the predictive uncertainty was always the highest for the predicted class. This is apparent even for miss-detection when the BCNN classifies a tree as a target. However, there were regions with a high uncertainty that were correctly classified as clutter. An example of this can be observed in Figure 5.29(b) near the large shadow by the top right hill in the area contained in the yellow ellipse.

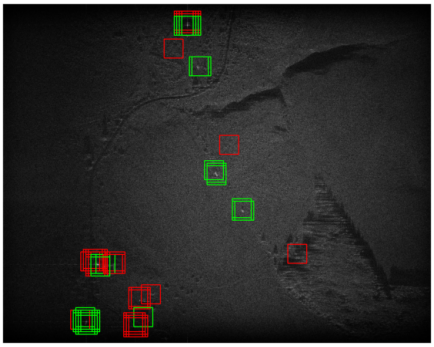
An interesting phenomenon occurred around the targets where the outer edges had increased uncertainty relative to the centre of the targets. This is especially prominent for buildings where a distinct outline can be observed and is less prominent for smaller targets such as vehicles. Examples of the high-uncertainty traces are observed in Figures 5.29(b) and 5.31(b) .

The uncertainty heat maps deliver an additional level of information to the user along with the detections. The uncertainty map provides a visual representation of the confidence of the model over each scene, thus, improving the trust between the user and the network. The user may now perceive which areas in the network have high or low confidence. This allows for human aid to be requested to assist the network when there are high-uncertainty targets. The incorporation of uncertainty estimates should improve the explainability of ATR systems.

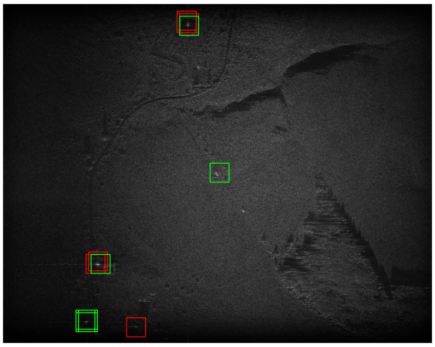
5.7 FEEDBACK ON HIGH-CONFIDENCE INCORRECT SAMPLES

The method to feed back samples as discussed in Subsection 4.3.4 was implemented. The results are shown in Figures 5.33 to 5.36. To investigate the effects of retraining, the BCNN with confident incorrect samples was explored. First, the detection performance and uncertainty heat map were determined for each scene. Once the BCNN was retrained, the detection performance and uncertainty heat map were determined again for comparison. To retrain and not completely alter the current configuration of the weight parameters, the learning rate was reduced by a factor of 25. This ensured that the BCNN was able to adjust to weights appropriately for the new data but make a significant change that would drastically change the detection performance. In addition, multiple runs were

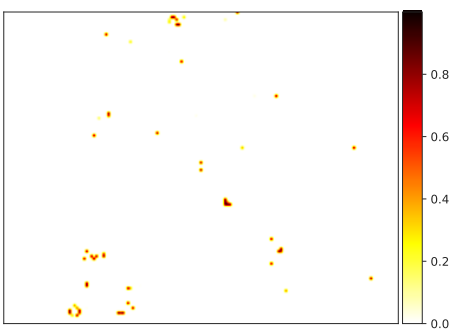
performed to gather confident incorrect samples. A total of five runs were performed for the results obtained.

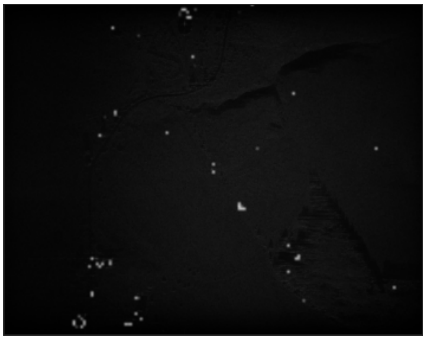


(a) Detection before feedback.

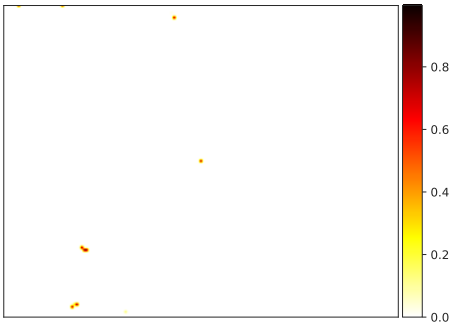


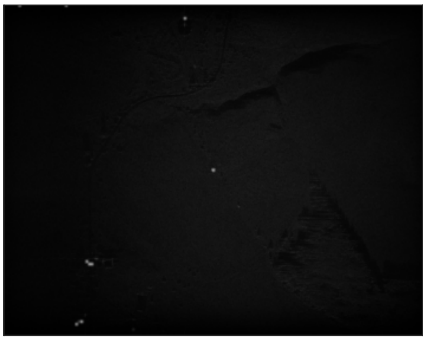
(b) Detection after feedback.





(c) Uncertainty heat map before feedback.



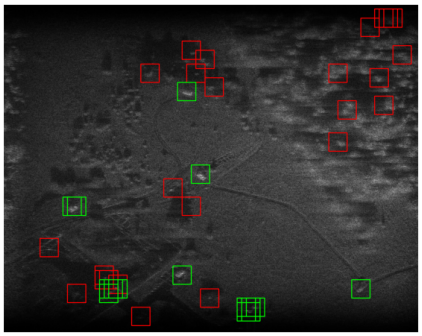


(d) Uncertainty heat map after feedback.

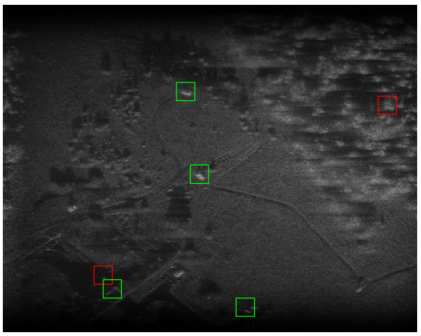
Figure 5.33. Comparison of the effect of retraining BCNN on confident incorrect detections for scene 1. The key difference in (a) and (b) is the reduction in false detections, and in (c) and (d) is the reduction in the number of high uncertainty regions.

From the comparisons, it is clear that there is a reduction in the overall uncertainty of the scene.

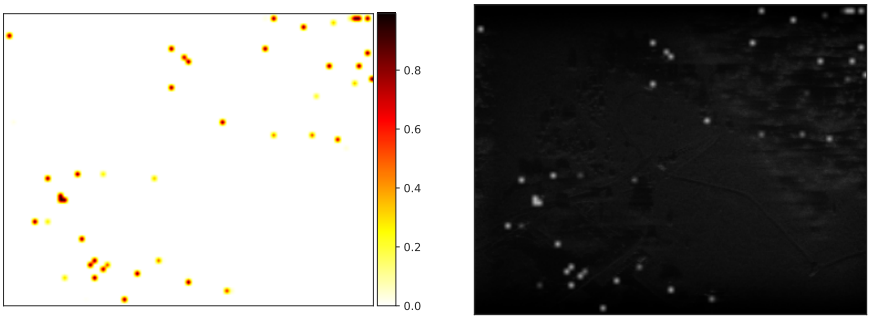
Notable changes are observed near the large area of trees and high-uncertainty regions in the centre of the scene, with numerous high-uncertainty regions disappearing after retraining.



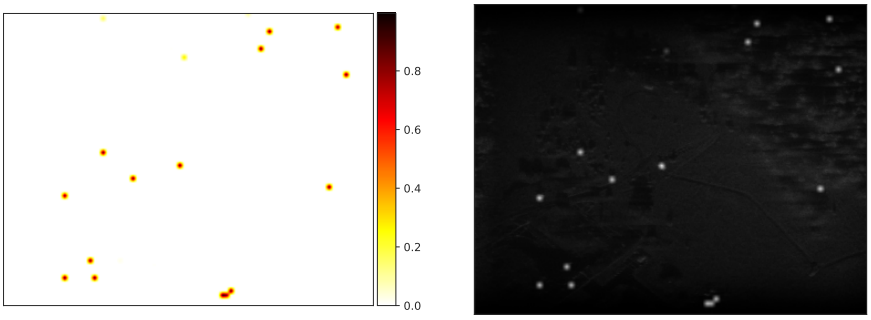
(a) Detection before feedback.



(b) Detection after feedback.



(c) Uncertainty heat map before feedback.

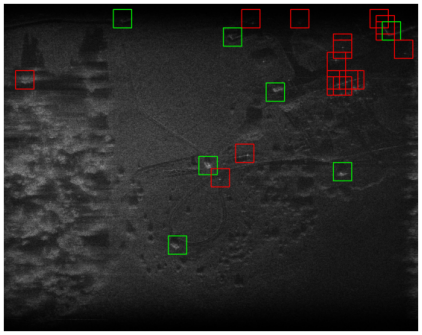


(d) Uncertainty heat map after feedback.

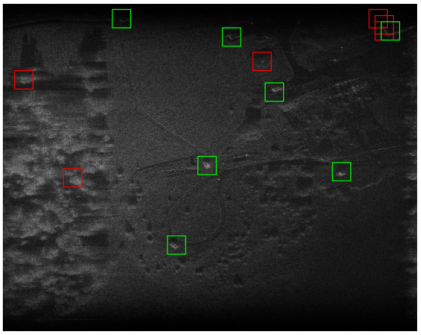
Figure 5.34. Comparison of the effect of retraining BCNN on confident incorrect detections for scene 2. The main takeaway from (a) and (b) is the removal of false detections over the forest region at the top right. (c) and (d) follow a similar trend to Figure 5.33 where there was a reduction in the number of high uncertainty areas. Scene 2 showed the most improvement in terms of precision.

After retraining, there was a reduction in the number of high-uncertainty regions. Noteworthy areas

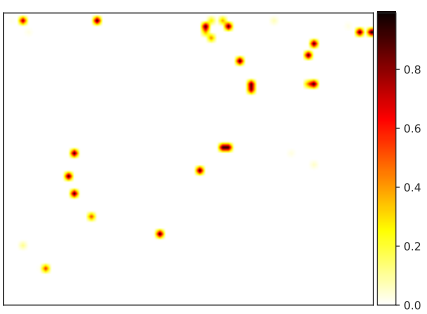
were the large cluster of trees in the top left and near the building towards the bottom left of the image. In addition, there were far fewer missed detections.

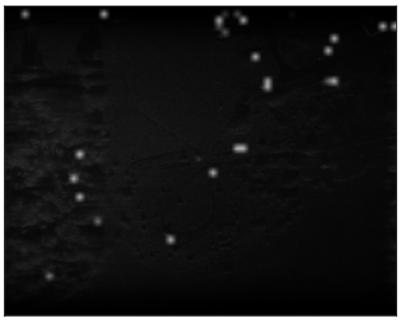


(a) Detection before feedback.

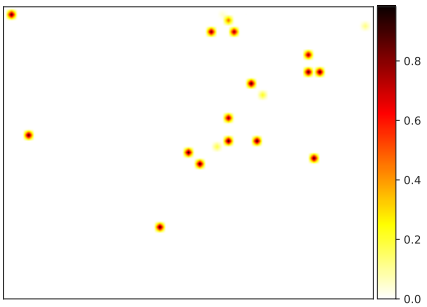


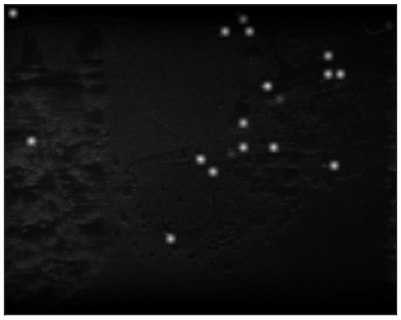
(b) Detection after feedback.





(c) Uncertainty heat map before feedback.

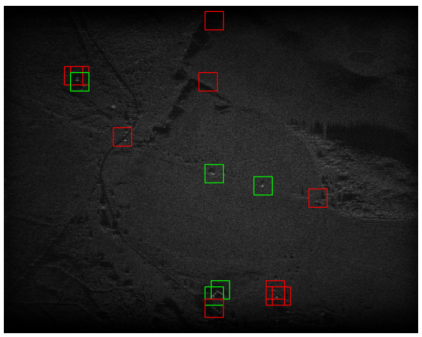




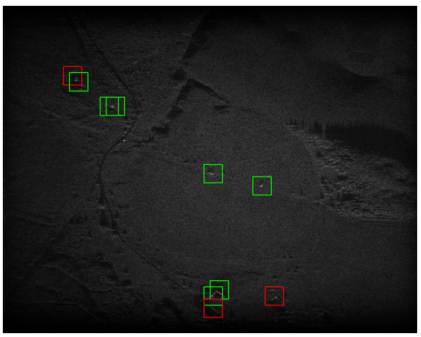
(d) Uncertainty heat map after feedback.

Figure 5.35. Comparison of the effect of retraining the BCNN on confident incorrect detections for scene 3. (a) and (b) continue to follow the trend of a reduction in false detections. Interestingly, the BCNN did not perform better in the tree region and had one additional false detection. Contrary to the previous scenes, in (c) and (d) there was no reduction in the number of high uncertainty regions.

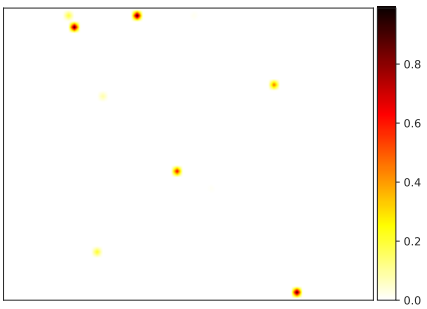
When compared to Figures 5.33 and 5.34, the reduction in uncertainty for scene 3 is lower. However, there was still a decrease in uncertainty of the overall scene.



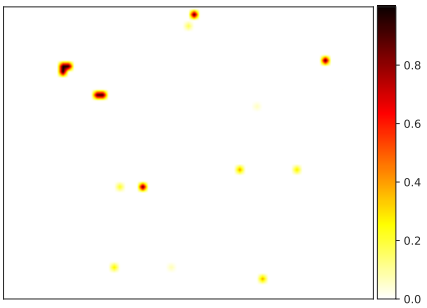
(a) Detection before feedback.

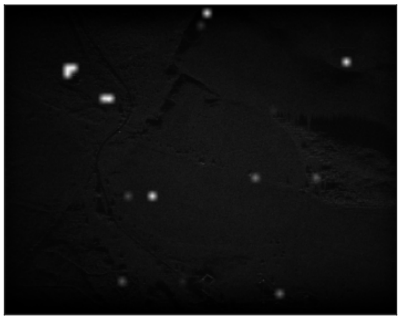


(b) Detection after feedback.



(c) Uncertainty heat map before feedback.





(d) Uncertainty heat map after feedback.

Figure 5.36. Comparison of the effect of retraining BCNN on confident incorrect detection for scene 4. (a) and (b) showed a significant improvement in the reduction of false detections, while being able correctly to detect all ground-truth targets. After retraining, a slight reduction in uncertainty was observed between (c) and (d). The effect of retraining appears to shift high uncertainty regions to over-the-ground-truth targets.

It was observed that there was an increase in the number of high-uncertainty regions in scene 4, while there was an improvement in the number of correct detections.

To compare the precision and uncertainty estimation, ten MC runs were conducted to determine the mean of the precision and uncertainty for pre- and post-feedback. The results are tabulated in Tables 5.4 and 5.5.

Table 5.4. Comparison of Detection Precision for Pre- and Post-Feedback of Confident Incorrect Samples.

Scene	Pre-feedback precision			Post-feedback precision		
	Average	Worst	Best	Average	Worst	Best
1	0.426	0.397	0.478	0.429	0.300	0.538
2	0.407	0.375	0.454	0.853	0.667	1.0
3	0.356	0.291	0.391	0.396	0.300	0.538
4	0.449	0.333	0.583	0.543	0.429	0.7

Table 5.5. Comparison of Average Epistemic Uncertainty for Pre- and Post-Feedback of Confident Incorrect Samples.

Scene	Average epistemic uncertainty		
	Pre-feedback	Post-feedback	% Reduction
1	0.000440	0.000190	58.82
2	0.00166	0.000521	68.61
3	0.0014970	0.001582	-5.68
4	0.000558	0.0005165	7.44

5.7.1 Discussion of the Feedback on High Confidence Incorrect Samples

The effect of feeding back confident incorrect samples resulted in fewer missed detections. This can be observed with the overall increase in precision shown in Table 5.4 for all scenes. However, it also resulted in fewer correct detections but the average precision was improved. This was to be expected as the model was retrained on data that had previously caused missed detections.

The feedback of confident incorrect samples reduced the number of high-uncertainty predictions for each scene. A significant decrease in the high-uncertainty regions was observed from Figures 5.33(d) to 5.36(d). This may be attributed to the decrease in the number of target detections. However, regions with known targets had a similar uncertainty to the uncertainty map before the feedback. This indicated that the network had learnt that those samples were indeed targets and adjusted its predictions for incorrect detections. In Figure 5.33(b), this was observed where the number of missed detections in the upper right corner was significantly reduced after the feedback while the network was still able to detect the majority of the targets correctly.

From Table 5.5, it was shown that the average epistemic uncertainty for each scene was reduced after the network was retrained. This implies that the network was more confident regarding its predictions. It is observed that the feedback of high-uncertainty incorrect samples is beneficial to the network. This method may not be practical during real-time applications where ground truth may be available but it was practical in a controlled environment during the training of the network. This method may assist in training and evaluation to improve performance and uncertainty estimation.

The visualisation of the epistemic uncertainty of the BCNN provided an indication of how confident the network was over the scene. Regions where it is typically difficult to detect targets, such as dense areas of trees, showed high uncertainty, while in flat field regions, a lower uncertainty was observed. The uncertainty heat maps provided improved interpretability by showing the network's confidence given various SAR scenes. These heat maps can be used to aid the user by providing additional information to make decisions and increase the user's trust in the BCNN decisions.

CHAPTER 6 CONCLUSION

The popularity of SAR sensors has significantly increased since it became an essential surveillance tool in World War II with the British famously using SAR to identify aircraft at relatively close ranges. Military forces have made great strides in advancing SAR techniques and currently use them for JISR operations. SAR analysts are human agents who manually extract intelligence pertaining to the recognition of targets and events. JISR operations require timely decisions and could have dangerous consequences. It is a process that requires a high level of efficiency and coordination amongst all teams involved [1]. Bottlenecks in data and information processing are apparent, as SAR analysts have to manually sort through hundreds of kilometres of SAR data. With limitations in sensor hardware, additional challenges arise with low-resolution images that result in targets only being represented by a few pixels. Consequently, this work aims to address the challenge of ATR of SAR images through the application of ML algorithms to aid in the classification process. In addition, the ML model uses eXplainable Artificial Intelligence (XAI) principles of transparency and explainability with model agnostic tools that act as an interpreter between the ML algorithm and the end-user.

One of the most pressing difficulties for ML methods is the non-transparency and over-confident predictions made by DNNs. These models have become more complex to achieve increased performance but have achieved this at the cost of transparency. As the field of XAI is still in its infancy, research to quantify a model's explainability and transparency is not yet well defined. Progress is being made at an increasing rate as regulatory bodies are pushing to embrace XAI principles to ensure that future AI models are made safer since human lives are being affected by their decisions.

The key results of this study are summarised next.

6.1 PREDICTIVE UNCERTAINTY OF BCNN

The predictive uncertainty showed that the BCNN made fewer over-confident predictions than the CNN while providing insight into the confidence of its prediction. When both networks were presented with samples that were in- and out-of-distribution to the MSTAR dataset, the BCNN demonstrated it was a significant improvement over the CNN. The CNN still made over-confident predictions, allocating a 100 % probability to a single incorrect class. However, the BCNN probabilities were much more spread out and with the aid of the epistemic uncertainty, it was apparent that the network was not confident. This difference when managing out-of-distribution data emphasises the necessity for alternative DNN implementation such as the BCNN. As a result, the BCNN is capable of handling out-of-distribution samples and responds accordingly by refusing to classify them. In this situation, operators may be notified to address the uncertain sample. This is in contrast to traditional CNNs that would proceed with an incorrect high-confidence prediction.

6.2 CLASSIFICATION PERFORMANCE BETWEEN CNN AND BCNN

The BCNN was able to correctly classify targets in the MSTAR dataset. Achieving a classification and F-1 score of 96.8 % and 93.7 %, respectively, was comparable to numerous classification implementations from recent publications. When compared to a CNN trained with the same architecture, the BCNN had a lower classification and F-1 score which correlated with other publications [4, 14].

6.3 TARGET DETECTION PERFORMANCE

The BCNN was used to distinguish targets from clutter to perform detections over various SAR scenes. Using the performance metric of precision, an average precision of 46.3 % was achieved. It was found that the detector was able to correctly detect the majority of the targets. The regions of trees resulted in the highest number of miss-detections. This is a result of the similarities between specific trees and targets in the MSTAR dataset since they have a shadow with a brighter region in the centre.

6.4 UNCERTAINTY HEAT MAP

The uncertainty heat maps provided a 2D visualisation of the confidence of the model over each region where the brighter regions indicated a higher uncertainty and the dimmer regions corresponded to a low uncertainty. The regions of high uncertainty appeared in areas that are more difficult to detect targets, such as large areas of trees or areas that heavily cast a shadow. The high uncertainty over the targets was a result of the epistemic uncertainty where the predictive class always had the highest uncertainty. The uncertainty maps improved the explainability of the system by utilising the uncertainty estimations. The BCNN is able to provide an explanation of how confident it was. Thus, the detection

map, supplemented with an uncertainty heat map, allows the user to have more trust in the target detector. Ultimately, this is a step in the direction of improving the current ML method to foster increased trust, interpretability, and transparency.

6.5 FEEDBACK OF HIGH UNCERTAINTY INCORRECT SAMPLES

An additional advantage of the uncertainty estimates is that they may be used to improve the performance of the network and reduce the number of high-uncertainty predictions. It was found that by feeding back high-confident incorrect samples, the precision of the detector was improved and an overall reduction in average epistemic uncertainty was observed. This method may be used as the last step before a model is deployed to make small adjustments to the network to reduce the number of high-confident incorrect detections.

6.6 ANSWERS TO RESEARCH QUESTIONS

1. Does the use of a BCNN and the interpretation of uncertainty add transparency and explainability? The use of the BCNN and uncertainty estimates improved the interpretability of the system through the use of uncertainty heat maps. However, the BCNN did not add transparency since the network does not provide additional insight into the decision-making processes.
2. What is the trade-off in classification performance between traditional DNNs and XAI models? The trade-off in classification accuracy was a 3 % difference between the BCNN and CNN. This trade-off was made for the advantage of the uncertainty estimation capabilities, and it was shown that the BCNN was less prone to over-fitting while the CNN made incorrect predictions with absolute certainty.
3. How reliable are uncertainty estimations from BCNNs? When the uncertainty estimates were evaluated, it was found that there was a distinct difference between samples that were in- and out-of-distribution. The BCNN could reliably differentiate between the in- and out-of-distribution samples using epistemic uncertainty.
4. Can the BCNN be used to perform target detection of various SAR scenes? The BCNN was used in the implementation of a detector, and it was found that the BCNN was able to detect targets in a variety of SAR scenes. The detector achieved an average precision of 46.33 % for the four different scenes. From the results, the detector was able to locate the majority of the known targets.
5. Can the uncertainty estimates be used to improve the BCNN's target detection performance? The proposed method to feed back incorrect high-certainty detections resulted in an overall reduction in the epistemic uncertainty and improvement in precision for all scenes. Thus, it was

concluded that the uncertainty estimates did improve the BCNN's detection performance as well as reduce the overall uncertainty.

6.7 SUGGESTED FUTURE WORK

This dissertation has emphasised the importance of uncertainty estimation in CNNs and how they can reduce over-confident predictions as well as aid in improving the trust of the end-user in the ML algorithm. However, there are still numerous research avenues to pursue, which include the following.

XAI is still in its infancy, and, as a result, there are no widely adopted standard metrics to measure the explainability of a particular ML algorithm. For the mass adoption of these algorithms, standards need to be in place to measure the interpretability so various methods can be assessed to select the most explainable model. Once qualitative standards have been developed, such methods will be used to measure the increase in explainability of incorporating uncertainty heat maps.

This implementation used a rudimentary sliding window to perform the target detection and uncertainty heat map generation, and the approach consisted of feeding the individual windows into the BCNN to estimate the uncertainty. This method is computationally expensive as the classifier computes the output for each individual window. Current detection algorithms such as You Only Look Once (YOLO) have been shown to perform with a high degree of speed and precision [83, 84]. In addition, YOLO factors in the targets with different resolutions and aspects and is adaptable to complex datasets with multiple overlapping classes. The next step would be the incorporation of a BCNN with a YOLO implementation to achieve a state-of-the-art detection algorithm with the benefits of the improvement in explainability through the uncertainty estimations.

Lastly, the focus would extend the BCNN to manage complex data. For this dissertation, only the magnitude was used for training and evaluation. It has been shown that a significant amount of information is contained in the complex data compared to only the magnitude [85]. Implementing a complex BCNN should improve the performance of the network, especially for the application of ATR using SAR measurements.

REFERENCES

[1] P. Berens, "Introduction to synthetic aperture radar (SAR)," *NATO*, pp. 1–10, 2006.

[2] M. Guarnieri, "The early history of radar," *Industrial Electronics Magazine, IEEE*, vol. 4, pp. 36 – 42, Oct 2010.

[3] F. Amato, A. Farina, M. Fiorini, and S. Gallone, "Surveillance unattended foliage penetrating radar for border control and homeland protection," *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, vol. 7, pp. 193–196, 06 2013.

[4] X. Shi, F. Zhou, S. Yang, Z. Zhang, and T. Su, "Automatic target recognition for synthetic aperture radar images based on super-resolution generative adversarial network and deep convolutional neural network," *Remote Sensing*, vol. 11, p. 135, 01 2019.

[5] Y. Chen, Q. Zhang, X.-P. Chen, Y. Luo, and F.-F. Gu, "An imaging algorithm of sparse stepped frequency SAR based on multiple measurement vectors model," *Journal of Electronics and Information Technology*, vol. 36, pp. 2986–2993, 12 2014.

[6] J. Tan, X. Fan, S. Wang, and Y. Ren, "Target recognition of sar images via matching attributed scattering centers with binary target region," *Sensors*, vol. 18, p. 3019, 09 2018.

[7] R. Flórez-López, "Reviewing RELIEF and its extensions: A new approach for estimating attributes considering high-correlated features," in *Proceedings IEEE International Conference on Data Mining*, Maebashi City, Japan, 2002, pp. 605–608.

[8] G. P. Zhang, "Neural networks for classification: A survey," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 30, no. 4, pp. 451–462, 2000.

[9] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer, 2000, pp. 128–180.

[10] Y. Tang and S. N. Srihari, "Efficient and accurate learning of Bayesian networks using Chi-squared independence tests," in *Proceedings - International Conference on Pattern Recognition (ICPR)*, Tsukuba, Japan, 2012, pp. 2723–2726.

[11] C. Coman and R. Thaens, "A deep learning SAR target classification experiment on MSTAR dataset," in *Proceedings International Radar Symposium*, Bonn, Germany, 2018, pp. 1–6.

[12] Y. Pang, S. Cheng, J. Hu, and Y. Liu, "Evaluating the robustness of bayesian neural networks against different types of attacks," *CoRR*, vol. abs/2106.09223, 2021. [Online]. Available: https://arxiv.org/abs/2106.09223

[13] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, and et al., "Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors," in *Conference: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 3296–3297.

[14] K. Shridhar, F. Laumann, and M. Liwicki, "A Comprehensive guide to Bayesian Convolutional Neural Network with Variational Inference," *arXiv*, pp. 1–38, 2019. [Online]. Available: http://arxiv.org/abs/1901.02731

[15] X. Zhao, Y. Jiang, and T. Stathaki, "Automatic Target Recognition Strategy for Synthetic Aperture Radar Images Based on Combined Discrimination Trees," *Computational Intelligence and Neuroscience*, pp. 1–18, 11 2017.

[16] Q. Zhao and J. C. Principe, "Support vector machines for SAR automatic target recognition," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 2, pp. 643–654, 2001.

[17] F. Zhou, L. Wang, X. Bai, and Y. Hui, "SAR ATR of Ground Vehicles Based on LM-BN-CNN," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, pp. 7282–7293, 07 2018.

[18] S. Wagner, "Combination of convolutional feature extraction and support vector machines for radar ATR," in *Proceedings 17th International Conference on Information Fusion*. Salamanca, Spain: International Society of Information Fusion, 2014, pp. 1–6.

[19] M. Hossin and M. Sulaiman, "A Review on Evaluation Metrics for Data Classification Evaluations," *International Journal of Data Mining & Knowledge Management Process*, vol. 5, pp. 1–11, 2015.

[20] N. Blomerus, P. de Villiers, and W. Nel, "Improved explainability through uncertainty estimation in automatic target recognition of sar images," in *Conference: 2021 25th International Fusion Conference (IFC)*, North West, NW, SA, November 2021.

[21] B. J. Schachter, *Automatic Target Recognition*. Washington, WA, USA: SPIE PRESS, 2016, vol. TT118, pp. 1–5.

[22] K. El-Darymli, E. Gill, P. McGuire, D. Power, and C. Moloney, "Automatic target recognition in synthetic aperture radar imagery: A state-of-the-art review," *IEEE Access*, vol. 4, pp. 6014–6058, 10 2016.

[23] Z. Zhang, W. Hu, and X. Du, "A new SAR chip image segmentation method by exploiting spatial relation between target and shadow," in *International Geoscience and Remote Sensing Symposium (IGARSS)*. Vancouver, BC, Canada: IEEE, 2011, pp. 1658–1661.

[24] M. DeVore, J. O'Sullivan, S. Anand, and N. Schmid, "Probabilistic approach to model extraction from training data," in *Algorithms for Synthetic Aperture Radar Imagery*, E. G. Zelnio, Ed., International Society for Optics and Photonics. Orlanda, Florida, United States: SPIE, 08 2001, pp. 358 – 366.

[25] C. M. Bishop, *Machine Learning and Pattern Recognition*. New York, NY, USA: Springer-Verlag, 2006.

[26] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[27] L. Arnold, S. Rebecchi, and S. Chevallier, "An Introduction to Deep Learning," in *European Symposium on Artificial Neural Networks (ESANN)*. Bruges, Belgium: ESANN, 2011.

[28] J. Liu and K.-C. Chang, "Feature-based target recognition with Bayesian inference," in *Conference of the North American Fuzzy Information Processing Society (NAFIPS)*. College Park, MD, USA: IEEE, 1995, pp. 95–100.

[29] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, and et al., "Deep speech: Scaling up end-to-end speech recognition," *ArXiv preprint arXiv:1412.5567*, 2014. [Online]. Available: https://arxiv.org/abs/1412.5567

[30] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, vol. 2, 01 2010, pp. 1045–1048.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ArXiv preprint arXiv:1409.1556*, 2014.

[32] J.-G. Lee, S. Jun, Y. Cho, H. Lee, G. Kim, J. B. Seo, and N. Kim, "Deep learning in medical imaging: General overview," *Korean Journal of Radiology*, vol. 18, p. 570, 07 2017.

[33] J. Li, X. Huang, and J. Gong, "Deep neural network for remote-sensing image interpretation: status and perspectives," *National Science Review*, vol. 6, no. 6, pp. 1082–1086, 05 2019. [Online]. Available: https://doi.org/10.1093/nsr/nwz058

[34] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: Automated testing of deep-neural-network-driven autonomous cars," *ArXiv preprint arXiv:1708.08559*, 2018. [Online]. Available: https://arxiv.org/abs/1708.08559

[35] A. Barredo Arrieta, N. Diaz Rodriguez, J. Del Ser, and A. Bennetot, "Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI," *Information Fusion*, vol. 58, pp. 1–8, 10 2019.

[36] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *32nd International Conference on Machine Learning (ICML)*. Lille, France: JMLR, 2015, p. 1613–1622.

[37] K. Lee, H. Lee, K. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=ryiAv2xAZ

[38] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70. Sydney, NSW, Australia: JMLR, 08 2017, pp. 1321–1330.

[39] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 5580–5590.

[40] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, and et al., "A survey of uncertainty in deep neural networks," *ArXiv preprint arXiv:2107.03342*, vol. abs/2107.03342, 2021.

[41] H. M. D. Kabir, A. Khosravi, M. A. Hosen, and S. Nahavandi, "Neural network-based uncertainty quantification: A survey of methodologies and applications," *IEEE Access*, vol. 6, pp. 36 218–36 234, 06 2018.

[42] J. Nandy, W. Hsu, and M. L. Lee, "Towards maximizing the representation gap between in-domain & out-of-distribution examples," *ArXiv preprint arXiv:2010.10474*, vol. abs/2010.10474, pp. 1–7, 2020.

[43] M. Raghu, K. Blumer, R. Sayres, Z. Obermeyer, B. Kleinberg, and et al., "Direct uncertainty prediction for medical second opinions," in *Proceedings of the 36th International*

Conference on Machine Learning, vol. 97, 06 2019, pp. 5281–5290. [Online]. Available: https://arxiv.org/abs/1807.01771

[44] A. Malinin and M. Gales, "Predictive uncertainty estimation via prior networks," *ArXiv preprint arXiv:1802.10501*, 2018.

[45] A. Graves, "Practical variational inference for neural networks," *Advances in neural information processing systems*, vol. 24, 2011.

[46] A. Lye, A. Cicirello, and E. Patelli, "A review of stochastic sampling methods for bayesian inference problems," in *Proceedings of the 29th European Safety and Reliability Conference (ESREL)*, Hannover, Germany, 01 2019, pp. 1866–1873.

[47] P. Alquier, "Approximate Bayesian Inference," *Entropy*, vol. 22, p. 1272, 11 2020.

[48] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[49] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[50] R. Figueiredo, K. Schröter, A. Weiss-Motz, M. L. Martina, and H. Kreibich, "Multi-model ensembles for assessment of flood losses and associated uncertainty," *Natural Hazards and Earth System Sciences*, vol. 18, no. 5, pp. 1297–1314, 2018.

[51] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. New Jersey, NJ, USA: Prentice Hall, 2003, vol. 82.

[52] R. J. Soldin, "SAR Target Recognition with Deep Learning," in *IEEE Applied Imagery Pattern Recognition Workshop*. Washington, DC, USA: IEEE, 2018, pp. 1–8.

[53] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, "Explainable ai: A review of machine learning interpretability methods," *Entropy*, vol. 23, p. 18, 12 2020.

[54] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. A. Specter, and L. Kagal, "Explaining explanations: An overview of interpretability of machine learning," in *IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*. Turin, Italy: IEEE, 2018, pp. 80–89.

[55] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *Arxiv preprint arXiv:1702.08608*, 2017. [Online]. Available: https://arxiv.org/abs/1702.08608

[56] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial Intelligence*, vol. 267, 06 2017.

[57] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 336–359, 2020.

[58] Mandeep, H. S. Pannu, and A. Malhi, "Deep learning-based explainable target classification for synthetic aperture radar images," in *2020 13th International Conference on Human System Interaction (HSI)*. Tokyo, Japan: IEEE, 06 2020, pp. 34–39.

[59] G. Zhao, F. Liu, J. Oler, M. Meyerand, N. Kalin, and R. Birn, "Bayesian convolutional neural network based mri brain extraction on nonhuman primates," *NeuroImage*, vol. 175, 03 2018.

[60] J. Ticknor, "A bayesian regularized artificial neural network for stock market forecasting," *Expert Systems with Applications*, vol. 40, p. 5501–5506, 10 2013.

[61] D. Dera, G. Rasool, N. C. Bouaynaya, A. Eichen, S. Shanko, J. Cammerata, and S. Arnold, "Bayes-sar net: Robust sar image classification with uncertainty estimation using bayesian convolutional neural network," in *2020 IEEE International Radar Conference (RADAR)*, Washington, DC, USA, 2020, pp. 362–367.

[62] J. P. de Villiers, A.-L. Jousselme, A. de Waal, G. Pavlin, K. Laskey, and et al., "Uncertainty evaluation of data and information fusion within the context of the decision loop," in *Proceedings*

19th International Conference on Information Fusion (FUSION), Heidelberg, Germany, 2016, pp. 766–773.

[63] J. P. de Villiers, A.-L. Jousselme, G. Pavlin, K. Laskey, E. Blasch, and et al., "Uncertainty representation and reasoning evaluation framework for information fusion," *Journal of Advances in Information Fusion*, vol. 13, pp. 198 – 215, 2018.

[64] V. Mullachery, A. Husain, and A. Khera, "A study of Bayesian Neural Networks," *ArXiv preprint arXiv:1801.07710*, 2015. [Online]. Available: https://arxiv.org/abs/2006.12024

[65] S. Odaibo, "Tutorial: Deriving the standard variational autoencoder (vae) loss function," *ArXiv preprint arXiv:1907.08956*, 2019. [Online]. Available: https://arxiv.org/abs/1907.08956

[66] D. Blei, A. Kucukelbir, and J. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, 01 2016.

[67] E. J. McShane, "Jensen's inequality," *Bulletin of the American Mathematical Society*, vol. 43, pp. 521–527, 1937.

[68] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *ArXiv preprint arXiv:1312.6114*, 2013. [Online]. Available: https://arxiv.org/abs/1312.6114

[69] R. M. Neal and G. E. Hinton, "A View of the EM Algorithm that Justifies Incremental, Sparse, and other Variants," in *Learning in Graphical Models*. Department of Computer Science, University of Toronto, 1998.

[70] Y. Kwon, J.-H. Won, B. J. Kim, and M. C. Paik, "Uncertainty quantification using bayesian neural networks in classification: Application to biomedical image segmentation," *Computational Statistics and Data Analysis*, vol. 142, pp. 1–6, 2020.

[71] R. D. Luce, *Individual Choice Behavior: A Theoretical analysis*. New York, NY, USA: Wiley, 1959.

[72] R. Feng, N. Balling, D. Grana, J. S. Dramsch, and T. M. Hansen, "Bayesian convolutional neural networks for seismic facies classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–8, 2021.

[73] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," *Advances in neural information processing systems*, vol. 28, pp. 2575–2583, 2015.

[74] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[75] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 4905–4913.

[76] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, (ICLR)*, vol. abs/1412.6980, San Diego, CA, USA, 2015.

[77] M. Pelikan, D. Goldberg, and E. Cantú-Paz, "Boa: the bayesian optimization algorithm," in *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, p. 525–532.

[78] L. Kaplan, "Improved sar target detection via extended fractal features," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 2, pp. 436–451, 2001.

[79] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, 2019, pp. 658–666.

[80] M. I. Bellil and X. Xu, "Affine and magnitude transforms based data augmentation comparison

for SAR automatic target recognition," in *2018 IEEE 4th International Conference on Computer and Communications (ICCC)*. Chengdu, China: IEEE, 2018, pp. 1649–1653.

[81] S. Chen, H. Wang, F. Xu, and Y.-Q. Jin, "Target classification using the deep convolutional networks for sar images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, pp. 1–12, 04 2016.

[82] A. Schilling, A. Maier, R. Gerum, C. Metzner, and P. Krauss, "Quantifying the separability of data classes in neural networks," *Neural Networks*, vol. 139, pp. 278–293, 2021.

[83] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, 2016, pp. 779–788.

[84] Q. Huang, W. Zhu, Y. Li, B. Zhu, T. Gao, and P. Wang, "Survey of target detection algorithms in sar images," in *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, vol. 5, Chongqing, China, 2021, pp. 1756–1765.

[85] J. Cilliers, "Information theoretic limits on non-cooperative airborne target recognition by means of radar sensors," Ph.D. dissertation, Department of Electronic and Electrical Engineering, University College London, London, UK, 2018.

ADDENDUM A HYPER-PARAMETER OPTIMISATION

The optimised hyper-parameters are listed below for both the CNN and BCNN.

Table A.1. Optimised Hyper-Parameters for BCNN.

Parameter	Value
Learning rate CNN	0.00065124
Network type	3 convolutional layer 3 fully-connected
Validation size	0.1
MC runs	10
Priors	$\mu = 0, \sigma = 0.1$
Learning rate BCNN	0.00035393
Batch size BCNN	8
Number of epochs	105
Patience	11
Activation function	softplus

Table A.2. Optimised Hyper-Parameters for CNN.

Parameter	Value
Learning CNN	0.00065124
Network type	3 convolutional layer 3 fully-connected
Validation size	0.1
MC runs	10
Batch size BCNN	24
Number of epochs	110
Activation function	ReLu