Machine Learning Methods for Super-Resolution in Sparse Sensor Arrays

by

Mumin Jin

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2021

Author .....
Department of Electrical Engineering and Computer Science
May 20, 2021

Certified by .....
Atulya Yellepeddi
Company Supervisor, Analog Devices, Inc.
Thesis Supervisor

Certified by .....
Gregory Wornell
Sumitomo Professor of Engineering
Thesis Supervisor

Accepted by .....
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Machine Learning Methods for Super-Resolution in Sparse Sensor Arrays

by

Mumin Jin

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2021, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Due to their robustness to weather and environmental conditions, radars are an important sensor in automotive and industrial applications. However, their utility in more advanced applications is increasingly limited by their resolution, which increases linearly with the number of radar elements in a uniform linear array (ULA). In this work, neural networks are trained to approximate the signals from a large aperture array from the signals obtained from smaller aperture sub-arrays. The training set consists of simulated radar responses to ideal point reflectors in noiseless vacuum, and the network is trained to minimize the squared error between the network output and the normalized log-magnitude of the large aperture array signal in its Fourier Transform domain. In general, given signals from two sets of 12-element sub-arrays, the neural network can reproduce results more than 10 times closer to the signals of the 1024-element array than signals from either of the input sub-array in terms of squared error in the Transform domain. The outputs of the neural network have over 91.97% probability of detection, P_D, with 1.02% probability of false alarm, P_{FA}, compared to 67.53% P_D and 5.57% P_{FA} with data from either sub-array. The results show the possibility for extracting more information by exploiting structure in real-world data with inexpensive, small sensors, and have major implications for the use of radar sensors in the automotive and industrial applications.

Thesis Supervisor: Atulya Yellepeddi
Title: Company Supervisor, Analog Devices, Inc.

Thesis Supervisor: Gregory Wornell
Title: Sumitomo Professor of Engineering

Acknowledgments

First of all, I would like to express my gratitude toward Atulya, my company supervisor at ADI, for introducing me to this project and teaching me the basics about radar. He has been a constant source of guidance and encouragement to me throughout this project. Atulya's timely and detailed feedback on the drafts of this thesis was also extremely helpful in the revision process. Even though the work on this thesis had to be done remotely due to the COVID-19 pandemic, Atulya connected me with many resources and people at ADI so that I had an experience as close to being in person as possible. He has given me many valuable insights from an industry perspective that I would not have gotten from a purely academic environment. His advice from our many Zoom meetings has both deepened my interest in machine learning methods for signal processing. I very much look forward to our paths crossing again in the future.

I am deeply indebted to my MIT thesis supervisor, Professor Gregory Wornell, for pushing me to think more deeply and creatively about the problems I encountered. I would also like to thank Professor Wornell for teaching many courses that have inspired my interests in the field during my undergraduate years.

I could not have done the work in this thesis without the generous support from my ADI colleagues Tao, Alan, and Nicholas, who took time out of their day to discuss ideas with me and show me the ropes. Nicolas LeDortz and Mariana Markova deserve special thanks for being my supervisors during my first 6A assignment at ADI. My lab mate at MIT, Tejas, also provided tremendous support. During our many meet ups both online and in-person, he talked to me about research ideas, shared papers, and imparted wisdom on many topics. I would also like to thank Tricia, the administrative assistant in Professor Wornell's group, for organizing our lab meetings and socializing events. She truly made me feel at home as a newcomer in the group despite everything being remote.

I would also like to acknowledge my many friends and loved ones, Anna, Aline, Megan, Melissa, Nicholas Sass, Alec, Amanda, Dylan, Robert and Tina Sass, Tante Ba, and Tante T.T. for their efforts to distract me from my work with fun and love.

Finally, I need to give my deepest gratitude to my amazing parents, Benlin Mei and Changnian Jin for unconditionally loving me, supporting me, and being my strongest rock and safest harbor through the last twenty-three years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter discusses the motivations behind the work in this thesis on radar signal super-resolution, some related works in radar signal processing, and the formulation of the problem of interest.

1.1 Motivation

Over the past decade, the demand for self-driving cars has been increasing. Consumers have come to expect features such as Advanced Driver Assistance Systems (ADAS) on every new vehicle [4] [5] [6]. Therefore, it is important to make ADAS technologies safe and reliable. The safety and reliability of these technologies depend on the ability of a variety of sensors to produce high-resolution 3-dimensional map of the real world.

Some common sensors that are deployed include Light Detection and Ranging (LIDAR) sensors, RGB cameras, and Radio Detection and Ranging (radar) sensors. RGB cameras have the advantage in that the data captured by RGB cameras are easily understood and interpreted by humans who have been trained to process visual data since birth. Extensive research has been done on image and video processing, so many algorithms can perform image or video processing tasks, including denoising,

classification, and super-resolution, as well as or better than humans [4]. However, cameras lack sufficient capability in depth sensing and measuring velocities of objects, nor can they operate in darkness or under severe weather conditions like storm or fog. On the other hand, LIDARs can produce a 3d map of the world even without ambient light [5], but LIDARs are also not very weather resistant and lack the ability to measure velocities. By contrast, Frequency Modulated Continuous Wave (FMCW) radar sensors are attractive because they can operate reliably in fog, dust, rain, or snow. Radar sensors' performance also does not depend on ambient light or the presence of plastic enclosures [6].

On the other hand, radar sensors have lower lateral resolution than LIDARs and cameras because the angular resolution of a radar sensor array depends linearly on the aperture, which is the total space spanned by the elements. The aperture is physically limited by the width of the car, and most modern cars are under 2 meters wide. Moreover, in order to avoid spatial aliasing, the radar elements need to be placed with $\lambda/2$ spacing in between, where λ is the wavelength of the received signal [7]. The cost of densely populating a large aperture with radar elements is very high, and the difficulty of calibrating those elements also grows with both the number of elements and the aperture they span. Figure 1-1 shows the relative strengths and weaknesses of these different types of sensors [1]. To more effectively utilize radar sensors in ADAS, we need a way to achieve higher resolution while considering the realizability of the radar arrays.

1.2 Related Works

From a signal processing perspective, the problem of super-resolution can be thought of as a reconstruction problem: we want to reconstruct some high-dimensional signal $\mathbf{x}$ after making low-dimensional observation $\mathbf{y}$ through some measurement process Φ,

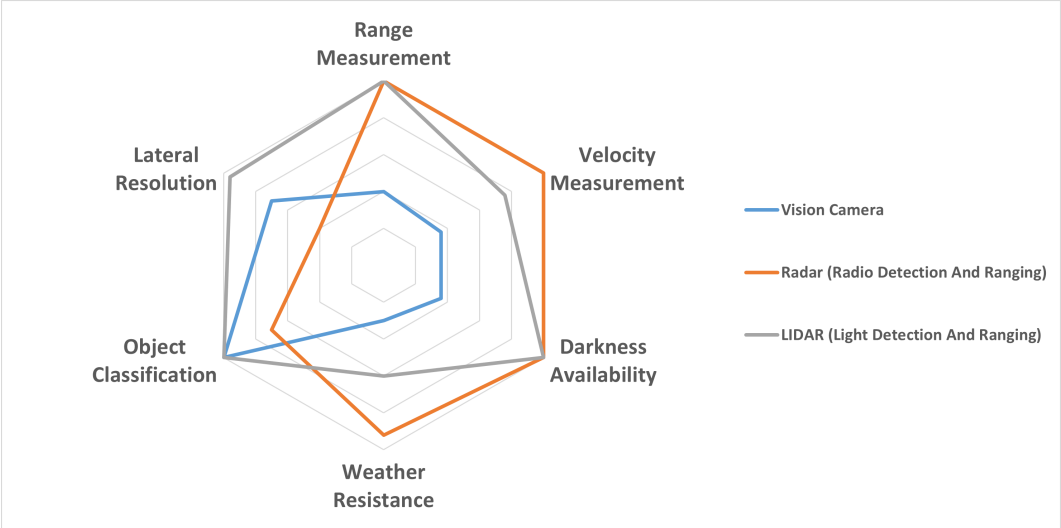


Figure 1-1: Relative strengths and weaknesses of cameras, radar sensors, and LIDAR sensors [1].

i.e. $\mathbf{y} = \Phi(\mathbf{x})$. However, in our problem of interest, $\mathbf{x}$, the ground truth of the world, has infinite dimension, and $\mathbf{y}$, the measurements from radar sensors, has low, finite dimension. As a result Φ does not have a well defined left-inverse that would give a unique solution for $\mathbf{x}$.

There has been a lot of literature on estimating Angles of Arrival (AoA) with higher resolution than allowed by the aperture of the array by taking advantage of the statistics of the data [8][9]. MUltiple SIgnal Classification (MUSIC) algorithm estimates the autocovariance of the signals from the radar elements and performs eigenvalue decomposition of the auto-covariance matrix to estimate noise space and finds signals that are most orthogonal to the noise space to estimate the AoAs. Other algorithms, including linear prediction models, make use of the estimate of the auto-covariance matrix in various ways.

Another approach is to impose sparsity constraints, taking advantage of the fact that radar data is usually sparse. The sparse signal reconstruction is formulated as

optimizing the following

$$\hat{\mathbf{x}} = \arg\min_{x} D(\mathbf{y}, \Phi(\mathbf{x})) + R(\mathbf{x}) \tag{1.1}$$

where $D(\mathbf{y}, \Phi(\mathbf{x}))$ is a function that measures the difference between $\mathbf{y}$ and $\Phi(\mathbf{x})$, and $R(\mathbf{x})$ is a regularization function that imposes sparsity on $\mathbf{x}$ [8]. For example, if $\mathbf{x}$ only has k non-zero elements, then it works well to choose $D(\mathbf{y}, \Phi(\mathbf{x})) = ||\mathbf{y} - \Phi(\mathbf{x})||_2^2$ and $R(\mathbf{x}) = ||\mathbf{x}||_1$[10]. If the signal has a different kind of sparsity, as illustrated in Figure 1-2 where the signal is non-zero in only a few bands, the Multi-coset imaging algorithm described in [2] shows promising results of reconstructing the signal with randomly place radar elements.

While sparsity is a common assumption made for reconstruction on radar signals, it is not necessarily the most appropriate one in practice. In general, reconstruction given the kind of spatial sparsity as proposed in [2] requires almost randomly placed elements as shown in Figure 1-3. Arrays like the one shown Figure 1-3 are hard to synthesize and calibrate because it is hard to synchronize the outputs across a large distance. Rather, in practice, smaller 3x4 Multiple-Input-Multiple-Output (MIMO) arrays, which function equivalent to virtual 12-element arrays as illustrated in Figure 1-4, are easier to synthesize. Therefore, we assume that we only have access to data from two sets of MIMO arrays placed roughly two meters apart on the front bumper of a car.

While machine learning based algorithms are very popular because they have demonstrated high performance on image and video processing tasks, the research on machine learning algorithms for radar signals is still at its nascent stages due to the lack of standard libraries with large amounts of labeled data [4][11][12]. Most of the research done on processing radar data with machine learning focus on using range-velocity information for object detection and classification, and the research on

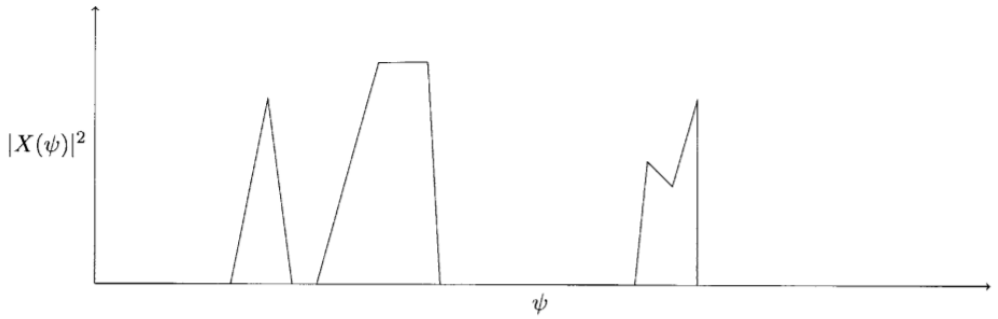


Figure 1-2: Band sparsity.

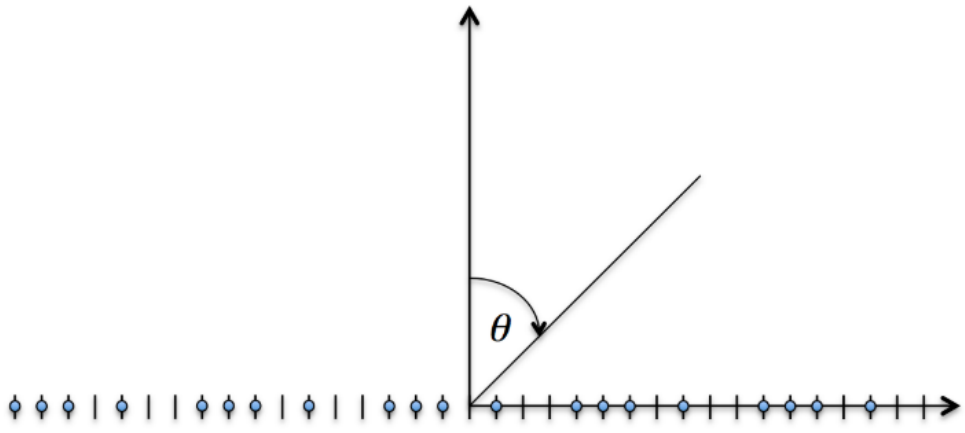


Figure 1-3: Radar element placement proposed in [2].

angular resolution enhancement mostly work by imposing sparsity constraints instead of using machine learning [8] [9]. To the best of our knowledge, there is no prior work to use machine learning to approximate radar signals from a wide aperture array given radar signals taken from small aperture arrays. **Therefore, in this work, we investigate super-resolution by using a neural network to approximate the simulated response of a large aperture radar array to ideal point reflectors given inputs only from practically realizable sub-arrays**.

1.3 Problem Formulation

We formulate the problem of super-resolution as a Bayesian inference task. If we could populate the full aperture of 2 meters with 1024 radar elements, we could observe data $\mathbf{y_{full}}$ which will give a good estimate for the ground truth scene $\mathbf{x}$, but such a large array is highly impractical for reasons discussed in Section 1.1. Therefore, we assume that we can only observe $\mathbf{y_0}$ and $\mathbf{y_1}$ from practically realizable 12-element sub-arrays as seen in Figure 3-8. Therefore, we want to approximate $\mathbf{y_{full}}$ given $\mathbf{y_0}$ and $\mathbf{y_1}$ by finding some function $f(\cdot)$ to minimize the expected value of some cost function $C(\cdot,\cdot)$ given $f([\mathbf{y_0}, \mathbf{y_1}])$ and $\mathbf{y_{full}}$ as in equation

$$\hat{f} = \arg\min_{f} \mathbb{E}_{\mathbf{x}}[C(f([\mathbf{y_0}, \mathbf{y_1}]), \mathbf{y_{full}})] \tag{1.2}$$

Computation of the actual expectation requires knowledge about the prior distribution for $\mathbf{x}$, but deep neural network has the potential to learn the prior distribution through training on data set $\mathcal{D}$ of samples $(\mathbf{y_0}, \mathbf{y_1}, \mathbf{y_{full}}) \in \mathcal{D}$ and minimizing the training loss in equation 1.3

$$L(f, \mathcal{D}) = \arg\min_{\mathbf{w}} \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{y_0}, \mathbf{y_1}, \mathbf{y_{full}}) \in \mathcal{D}} ||P(\mathbf{y_{full}}) - f_{\mathbf{w}}([P'(\mathbf{y_0}), P'(\mathbf{y_1})])||_2^2 \tag{1.3}$$

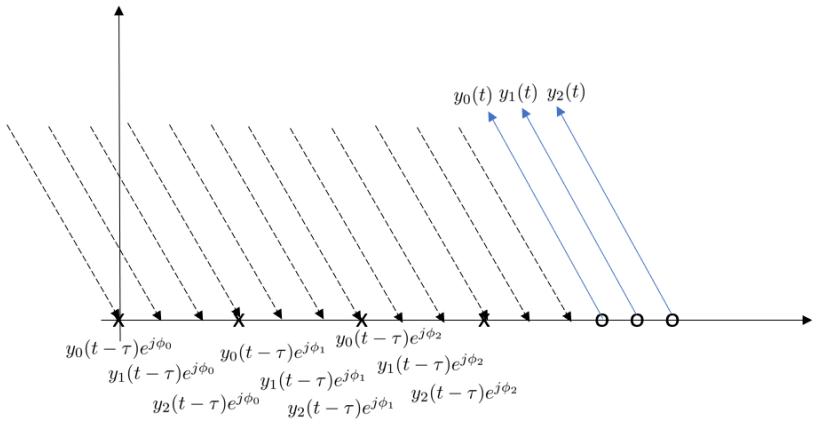


Figure 1-4: MIMO radar and virtual elements.

where $P(\cdot)$ and $P'(\cdot)$ are some processing done on the raw data $\mathbf{y_{full}}, \mathbf{y_0}, \mathbf{y_1}$, which we will describe in as described in section 3.2.2. $f_{\mathbf{w}}(\cdot)$ is the forward pass of the neural network with weights $\mathbf{w}$. $||\cdot||_2$ is the ℓ-2 norm.

Under this formulation, we can generate a data set with a large amount of simulation samples, design a suitable neural network, and train the network with gradient descent.

Figure 1-5: Problem set up

Chapter 2

Background

This chapter introduces how FMCW radar sensors work to estimate ranges and angles of arrival of targets in the world and some important characteristics of the data that make radar signal processing challenging.

2.1 Principles of FMCW Radar Operation

2.1.1 Range Estimation

An FMCW radar element has four components: a transmit (Tx) antenna, a receive (Rx) antenna, a mixer that takes two input signals and produces one output signal, and an ADC as shown in Fig 2-1 [7]. The output signal of the mixer has instantaneous frequency equal to the difference of the instantaneous frequencies of the inputs. The Tx antenna sends out a pulse of linear chirp, that sweeps over a bandwidth B over time T_c, as seen in the blue curve in Figure 2-1, and the Rx antenna receives the chirp echoed by a target, the orange curve in Figure 2-1. The frequency of the output signal from the mixer $y(t)$ is proportional to the time delay between the echoed pulse and the transmitted pulse. Finally, ADC samples $y(t)$ with period T to get final output signal $y[n]$.

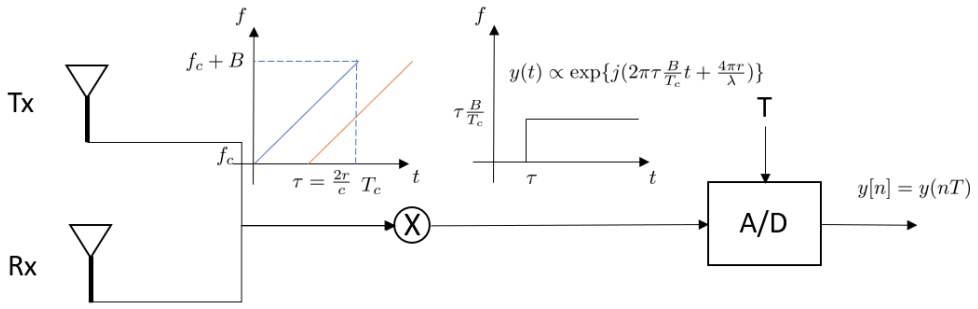


Figure 2-1: How an FMCW radar element works.

The smallest range difference that can be separated by the FMCW radar depends on the bandwidth $r_{res} = \frac{c}{2B}$. In practice, the FMCW radar elements have $B = 2$ GHz, resulting in around 7.5 cm of range resolution. And the maximum range depends on the sampling period T:

$$r_{max} = \frac{cT_c}{2BT}$$

To get N different range samples, we need

$$\begin{aligned} Nr_{res} &= \frac{cT_c}{2BT} \\ N\frac{c}{2B} &= \frac{cT_c}{2BT} \\ T &= \frac{T_c}{N} \end{aligned}$$

Taking the Discrete Fourier Transform (DFT) of the sampled data $y[n]$ produces a range profile as seen in Figure 2-2. The location of the peak seen in the Figure corresponds to the range of the target.

2.1.2 Angle of Arrival (AoA) Estimation

When FMCW elements are arranged with uniform spacing d in a linear array (ULA) with spacing d as in Figure 2-3, a plane wave arriving from angle θ needs to travel an

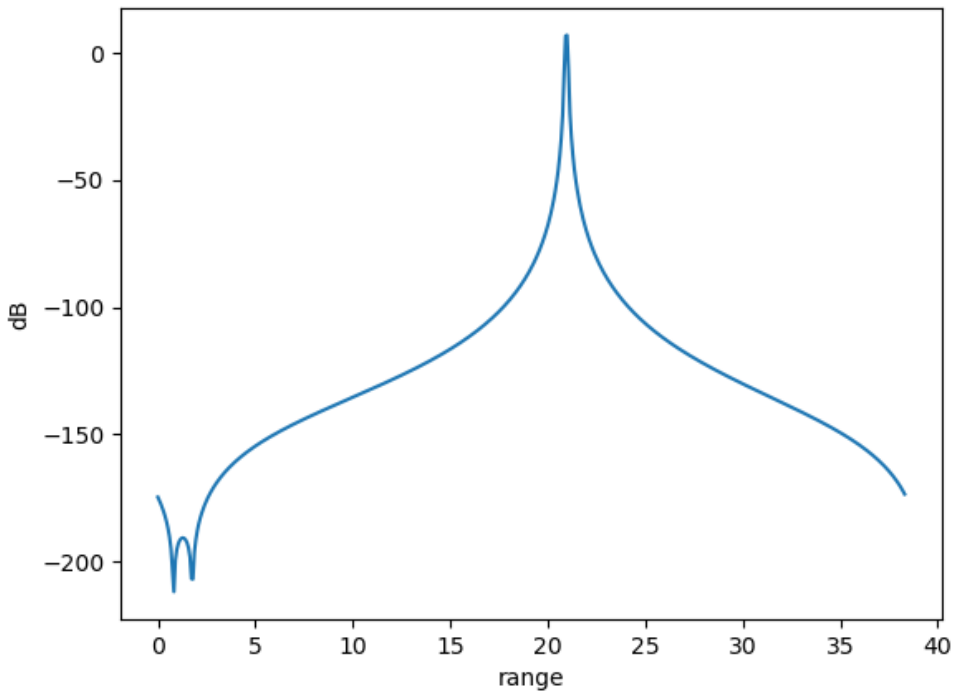


Figure 2-2: DFT of FMCW output given an ideal point reflector at range 21 meters.

additional tiny distance $\Delta = d\cos(\theta)$ to get to the next radar element in Figure 2-3. The additional distance travelled causes a time delay and a phase shift across the ULA. The time delay between two neighboring elements, $\frac{\Delta}{c}$ is practically 0. However, the change in phase, $\frac{4\pi\Delta}{\lambda}$, can be measured by taking DFT across the radar elements. The phase shift only has no ambiguity for all θ between 0 and π when $\frac{4\pi d\cos(\theta)}{\lambda} < 2\pi$, which implies that 180-degree field of view requires $d < \frac{\lambda}{2}$. FMCW radar sensors operate around 77 GHz, so $lambda \approx 4$ mm, and $d \approx 2$ mm. [13] [7].

2.2 Range-Angle Processing Artifacts

2.2.1 Ideal Results

Consider a scene with 4 ideal point reflectors in space at locations shown in Figure 2-4. With unlimited bandwidth B and observation window T_c, then DFT over infinite time samples at each radar element will produce range profiles with arbitrarily small r_{res}. Similarly, DFT over infinite number of radar elements starting at the origin can resolve two AoAs arbitrarily close to each other. Figure 2-5 shows an infinite radar matrix, where the columns are the samples from a single radar element, and the rows are the snapshots of the signals at each radar element at sampling times. Assuming each ideal point reflector in space at an angle θ contributes a phase shift across the array proportional to $\cos(\theta)$ as discussed in section 2.1.2, the 2d-DFT of the infinite radar matrix will generate an ideal range-angle plot shown in Figure 2-5 which contains arbitrarily narrow peaks (delta functions) at locations corresponding to the ranges and angles of the ideal point reflectors.

2.2.2 Effects of Finite Observations

Since it is impossible to have an infinite radar matrix, the effect of having finite observations is the multiplication of infinite radar matrix multiplied by a space-limited

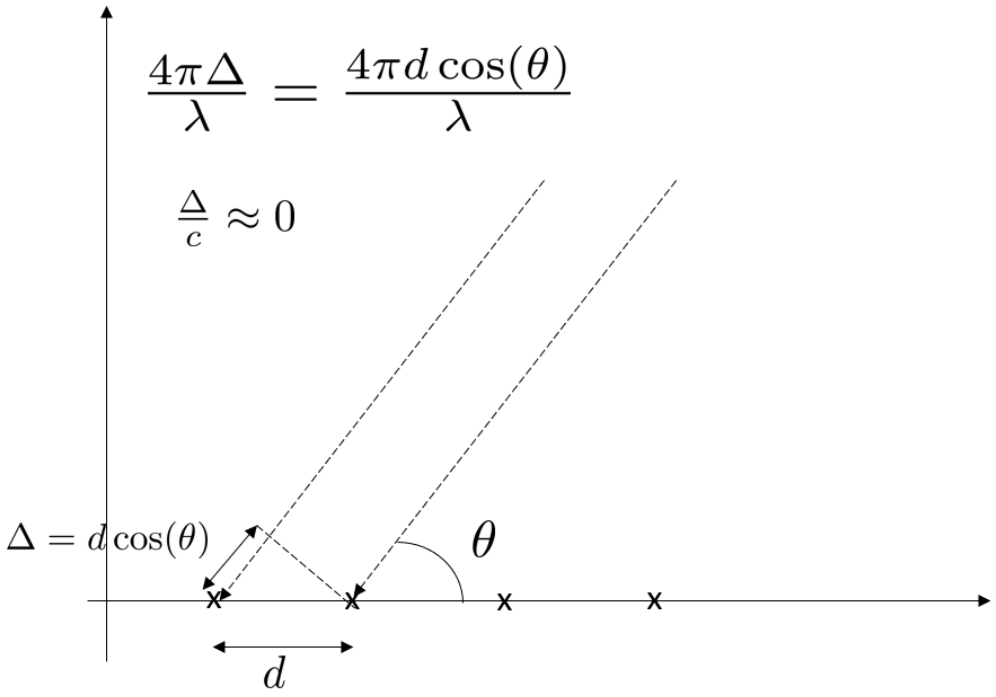


Figure 2-3: Principles of AoA estimation.

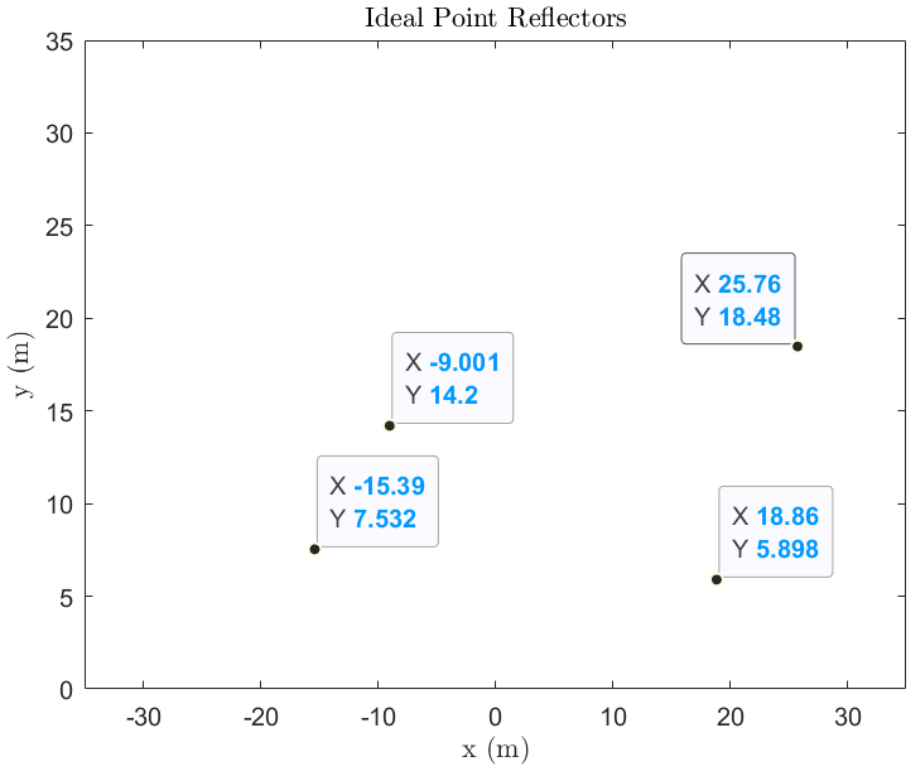


Figure 2-4: Ground truth locations of 4 ideal point reflectors.

windowing function, multiplication leads to convolution in the transform domain. Therefore, the range-angle plot from taking 2d-DFT of a finite matrix would have the result of the ideal range-angle plot convolved with the transform of the windowing function. Some example windowing functions and their transforms are shown in Figure 2-6. The smaller the length of window, the wider the main lobe of the transform of the windowing function, and the harder it is to distinguish two AoAs. Figure 2-7, which shows DFT of range processed signal from 12 radar elements in ULA (left) versus 1024 radar elements (right), illustrates the effect of windowing (limited aperture) on the resolution of AoA. In Figure 2-7, we cannot distinguish two actual AoAs with $\cos(\theta_0) = 0.35$ and $c\cos(\theta_1) = 0.5$ from the left plot, but we can clearly distinguish the two peaks that correspond to the two AoAs ($\cos(\theta_0) = 0.35$ and $c\cos(\theta_1) = 0.5$) from the right plot. Since DFT can be thought of as samples of a single period of the Discrete Time Fourier Transform (DTFT), which is periodic, the result of DFT includes the side-lobes of the aliased copies as seen in both plots in Figure 2-7.

Figure 2-8 shows the range-angle plot obtained from a 512 time samples for each of 1024 radar elements in ULA given the scene described by Figure 2-4. Figure 2-8 includes peaks around the actual $(r, \cos(\theta))$ of each point and side-lobes in the horizontal and vertical direction.

Figure 2-9 shows the range-angle plot based on the response of 12 radar elements in ULA to the same scene described by Figure 2-4. Due to the smaller aperture, the angular resolution is greatly reduced as seen from the much wider peaks in the horizontal direction around the true $(r, \cos(\theta))$ pairs. Figure 2-9 is also affected a lot more by aliasing because the aliased copies of the DFT also have wider peaks and higher side lobes.

2.2.3 Distortion

The range-angle plot given a scene suffers from different levels of distortion at different regions of the field of view. The use of Fourier Transform to estimate AoA assumes that the incident wave is a plane wave. The assumption of plane wave becomes more inaccurate as the target gets closer to the radar array. The dashed lines in Figure 2-10 show the assumed plane wave when we use Fourier Transform to estimate AoA, but the red solid lines show the actual reflected wave paths by the target. The closer the target is to the radar array, the worse the assumption of plane wave is.

When the aperture of the array is much larger than the range resolution, the range difference across the whole array is non-negligible. As a result, 2d-DFT produces more than one horizontal lines corresponding to each actual range in the range-angle plot from the 1024-element array in Figure 2-8 but only one horizontal line corresponding to each actual range in the range-angle plot from the 12-element array in Figure 2-9.

Because of these distortions, when we apply a threshold to the range-angle plot from the 1024-element array in Figure 2-8, by making values above the threshold 1 and the values below the threshold 0, we get patches of different shapes at different locations even though they are all ideal point reflectors at these locations as seen in Figure 2-11.

In all the range-angle plots, the horizontal-axes are $\cos(\theta)$. Given $\Delta > 0$, since the derivative of $|\cos(\theta)$ is small when θ is close to 0 or π but large when θ is close to $\frac{\pi}{2}$, as θ_0 and θ_1 approach to 0 or π, we require larger $|\theta_0 - \theta_1|$ so that $|\cos(\theta_0) - \cos(\theta_1)| > \Delta$. Therefore, the angular resolution is the highest at the center of the radar array's field of view and worst at the edges. Moreover, the physical distance required between two objects at range r with angles θ_0 and θ_1 is $|(r\cos(\theta_0), r\sin(\theta_0) - (r\cos(\theta_1), r\cos(\theta_0))|$, which depends on the range r. When we project range-angle plots into Cartesian coordinates, we see in Figure 2-12 that spatial resolution decreases toward the edge of the radar array's field of view.

All these artifacts we discussed in Section 2.2 make radar data much more difficult to understand and interpret than natural images, and adapting machine learning algorithms that work well on images to process radar signals will have to deal with the effects of these artifacts.

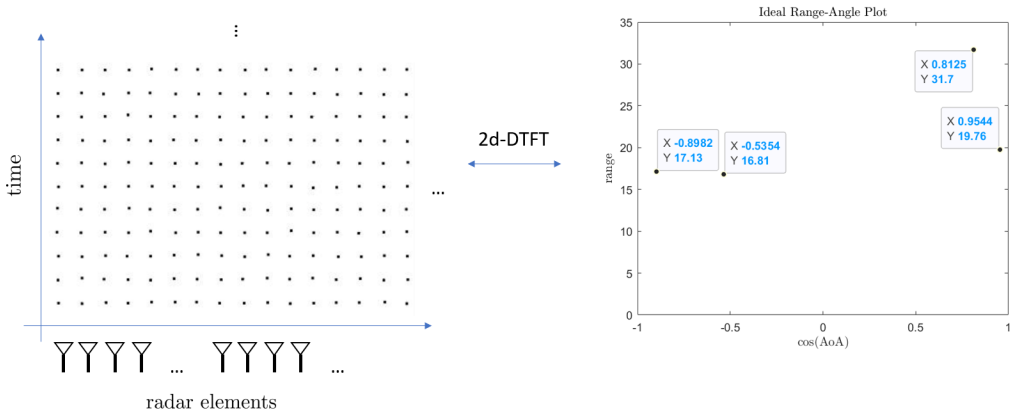


Figure 2-5: Ideal processing result with infinite measurements.

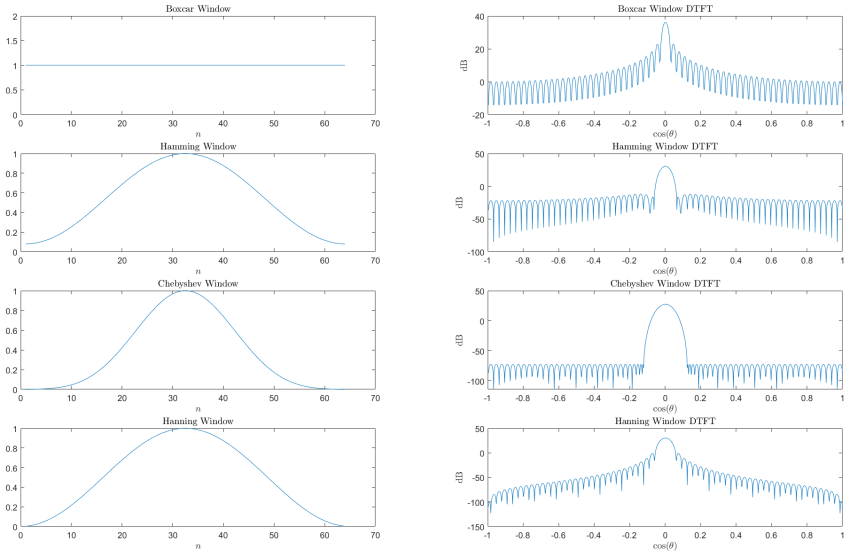


Figure 2-6: Different length-64 windowing functions and their transforms.

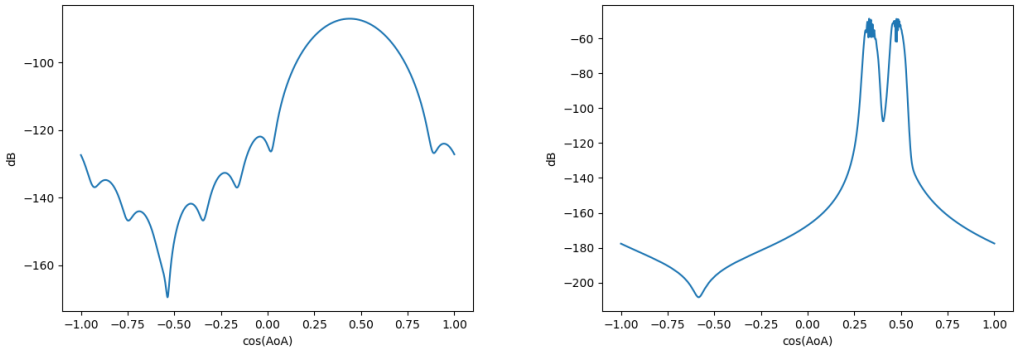


Figure 2-7: DFT of range processed signal from 12 and 1024 radar elements in ULA.

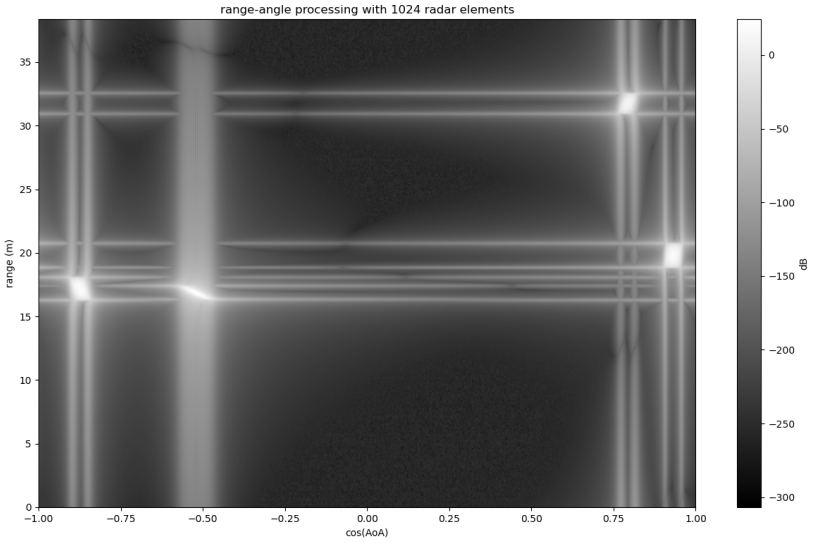


Figure 2-8: Range-angle plot based on the response of 1024 radar elements in ULA.

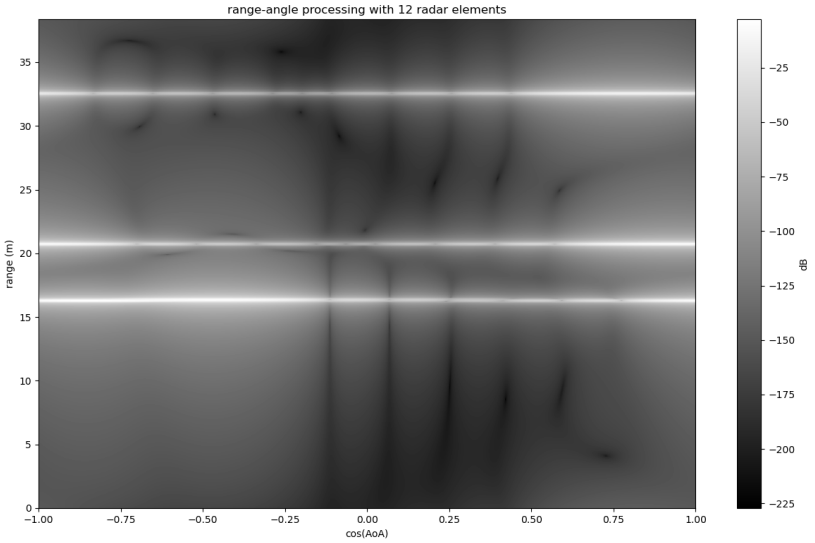


Figure 2-9: Range-angle plot based on the response of 12 radar elements in ULA.

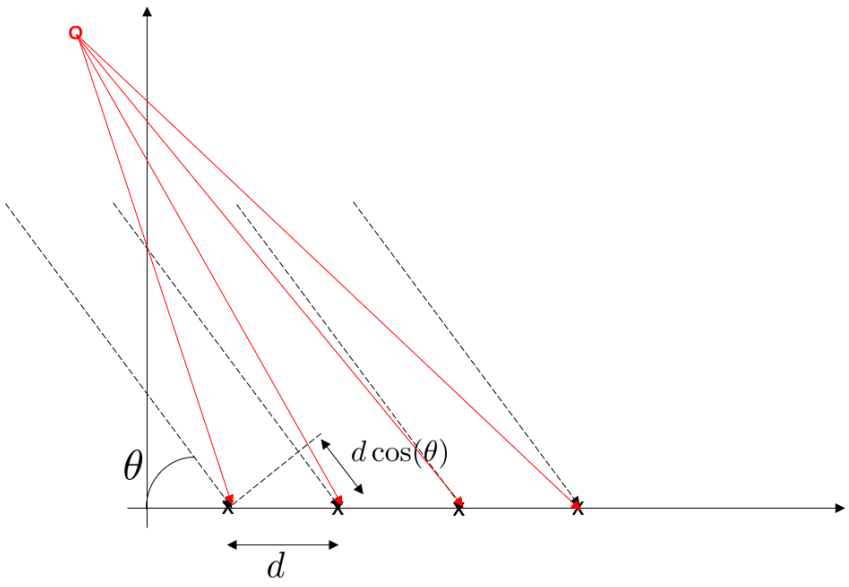


Figure 2-10: Plane wave assumption.

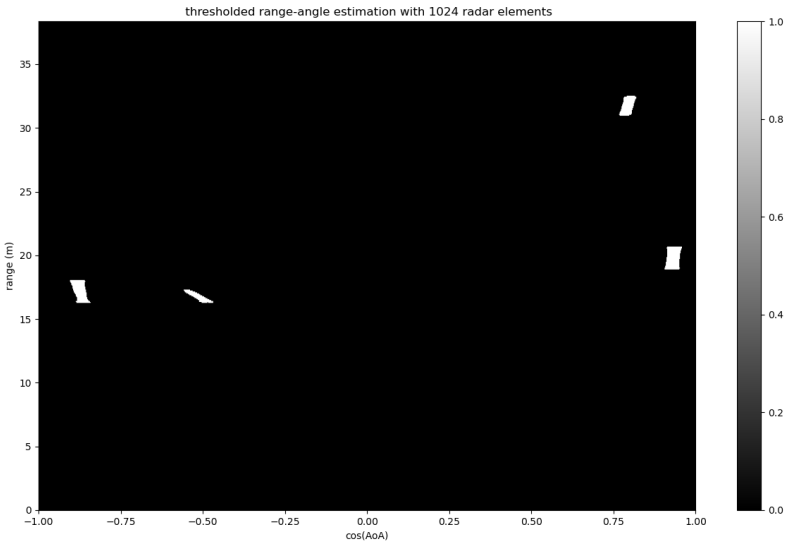


Figure 2-11: Range-angle plot after applying threshold.

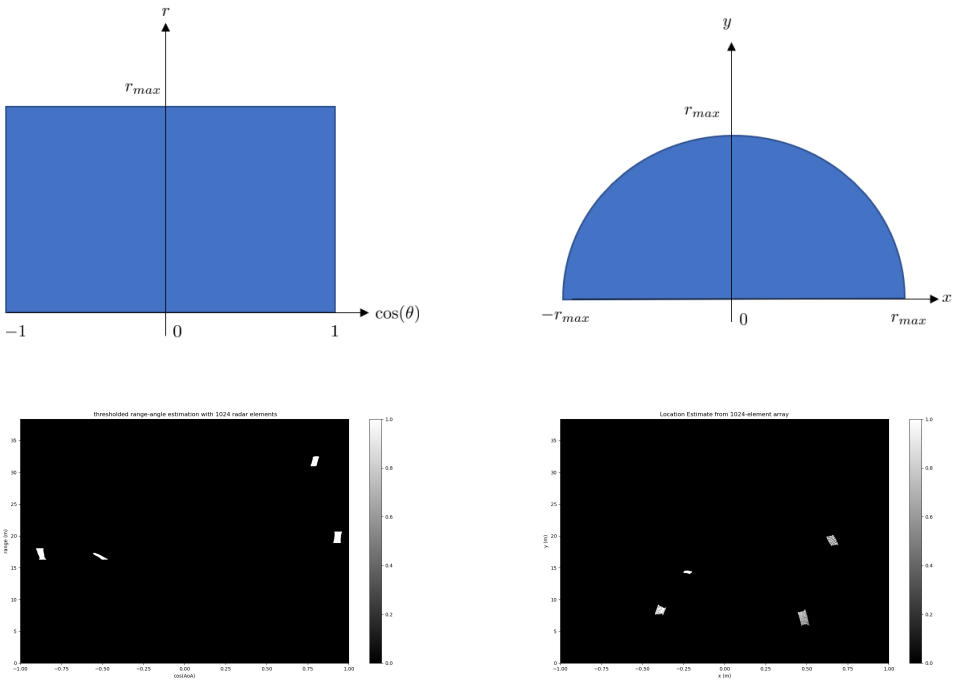


Figure 2-12: Artifacts of range-angle plot projected into Cartesian coordinates.

Chapter 3

Methods

In this chapter, we describe the methods we adopted to solve the problem formulated in Section 1.3. First, we discuss how we chose the network architecture we adopted for training. Next, we present the details of how we simulated the radar sensor array response to ideal point reflectors. Finally, we describe the post processing done on the simulated data to get the inputs to the neural network and training process, which are normalized log-magnitude and phase information of range-angle plots.

3.1 Network Architecture

Results from extensive research have shown that deep convolutional neural networks perform well on tasks such as de-noising, in-painting, and super-resolution for images [11] [14]. The high performance of the neural networks on natural images can be attributed to two properties of image data. First, images generally satisfy spatial translational invariance, because, if a set of pixels represents an object, the same set of pixels would represent the same object even if they are at a different location in the image. Second, the correlation between two pixels generally only depends on the distance between the two pixels, as pixels closer together tend to be more highly correlated with each other, while pixels farther apart have less to do with each other.

By contrast, range-angle plots from radar data satisfy neither of the two properties because of the artifacts of the range-angle processing discussed in Section 2.2. Range angle plots do not have spatial translational invariance because even the same ideal point reflector produces produce different images at different locations as shown in Figure 2-11. The correlation between two points in the range-angle plot also depend on more than just the distance between them. For example, the correlation between signals at range indices $n_1 = 0$ and $n_1 = 1$ is going to be different from the correlation between signals at signals at range indices $n_1 = 255$ and $n_1 = 256$. Natural images are also typically not sparse in the way that the range-angle plots are.

Therefore, a neural network architecture that works well on natural images does not necessarily work well on radar data. For example, the UNet architecture, which works well on image super-resolution, does not perform well on radar data. We experimented with a UNet architecture by adapting the UNet architecture used for biomedical image segmentation as shown in Figure 3-1 [3]. We trained the network on thresholded range-angle plots in Cartesian coordinates like the example shown in Figure 3-2. We found that the weights would quickly converge to some very small value so that the output is close to zero. Since the thresholded data is very sparse, the difference between the thresholded data and the zero matrix is so small that the UNet tends to get trapped at this local minimum. An example of the result of applying the UNet on a sample of training data after convergence is shown in Figure 3-2.

Generative Adversarial Networks (GANs) and Variational AutoEncoders (VAE) have been shown to perform well on Bayesian inference tasks by modeling the prior distribution[15] [16]. Assuming that there is some unobserved, low-dimensional latent variable $\mathbf{z}$, which could be entirely fictitious, and function $g(\cdot)$, such that $\mathbf{x} = g(\mathbf{z})$, GANs and VAEs model the prior for the high-dimensional vector $\mathbf{x}$ by learning the function $g(\cdot)$ from training samples.

Since the problem of radar signal super-resolution was also formulated as a Bayesian

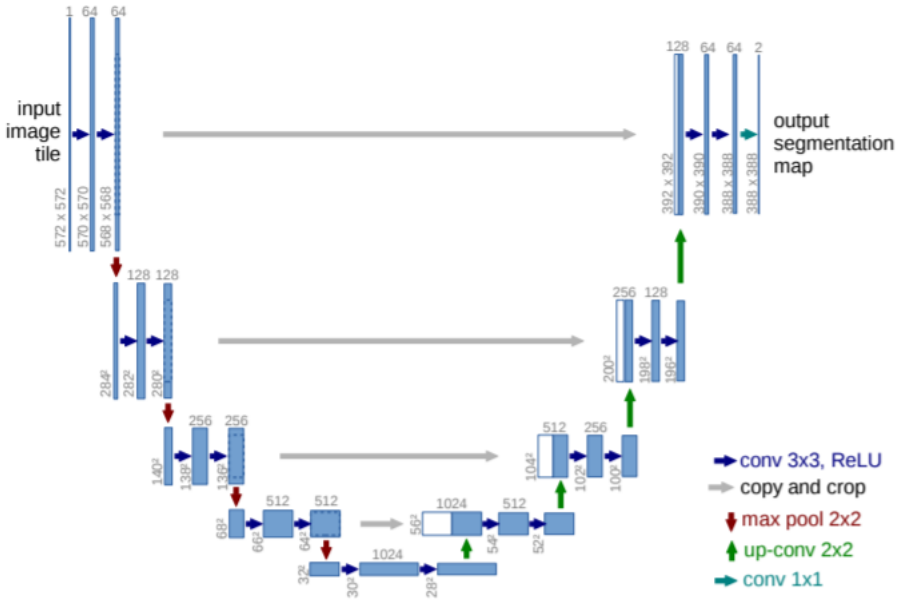


Figure 3-1: UNet architecture used in [3]

inference task, an encoder plus decoder architecture as commonly found in Auto Encoders is a natural choice. Although the output of the full-aperture array $\mathbf{y_{full}}$ contains 512x1024 complex numbers, the scene can be fully described by the locations of the ideal point reflectors, which has a much lower dimension. If a scene has 500 point reflectors, then we only need to know 2x500 real numbers to fully reconstruct the scene. Therefore, given enough training data, an encoder can learn to map the small-aperture subarray information to a latent variable $\mathbf{z}$, which can be interpreted as the "locations" of the reflectors, and the decoder can learn to generate the large-aperture array output given the "locations."

Since many works that use machine learning to process range-velocity data still have convolutional layers as their basic building blocks [4] [12][17], we also rely on convolutional layers as building blocks in our network architecture. The resulting architecture shown in Figure 3-3 has 6 downsampling layers in the encoder portion and 6 upsampling layers in the decoder portion. The downsampling layer consists of a convolution layer with 64 3x3 filters with dilation 1 in parallel with a convolution

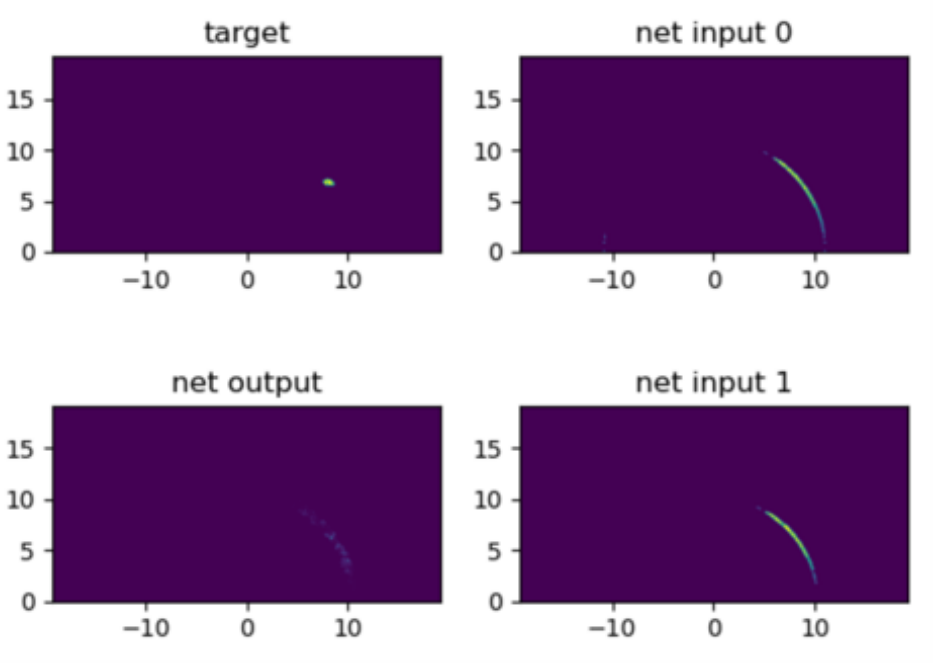


Figure 3-2: UNet results on training data.

layer with 64 5x5 filters with dilation 1 and followed by batch-normalization and max-pooling. We use two filters of different sizes to capture features on different scales at each level. The upsampling layer consists of 64 covnvtranspose layer followed by batch-normalization and ReLU. It is important to use circular padding for convolution layers because results of DFT are circular symmetric.

3.2 Data Generation

3.2.1 FMCW Simulation

For an FMCW element located at $\mathbf{a}$ that operates as in 2-1, its response to an ideal point reflector located at $\mathbf{d}$ is given by equation 3.1.

$$y(t) = \frac{\sigma P_{out}}{|\mathbf{d}-\mathbf{a}|^2} \exp\{j(2\pi \frac{2S|\mathbf{d}-\mathbf{a}|}{c} t + \frac{4\pi|\mathbf{d}-\mathbf{a}|}{\lambda})\} \tag{3.1}$$

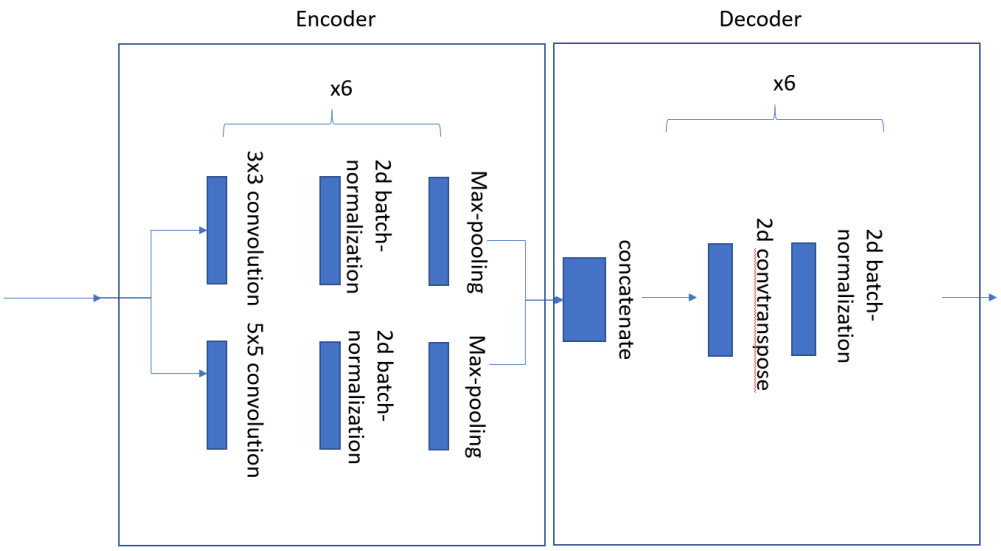


Figure 3-3: Network Architecture

where σ is the reflectivity of the point source, which is set to 1 given our assumption of ideal reflectors. The output power P_{out} by Tx antenna was chosen arbitrarily to be 100 Watts for all simulations. The actual values for the other parameters listed in Table 3.1 are chosen to match the values used for FMCW elements deployed for vehicles.

Given L point sources located at $\{\mathbf{d_0}, ..., \mathbf{d_{L-1}}\}$ and N antennas at locations $\{\mathbf{a_0}, ..., \mathbf{a_{N-1}}\}$, the signal received by element at $\mathbf{a_n}$ at time t is given by equation 3.2

$$y_n(t) = \sum_{l=0}^{L-1} \frac{P_{out}}{|\mathbf{d_l} - \mathbf{a_n}|^2} \exp\{j(2\pi \frac{2S|\mathbf{d_l} - \mathbf{a_n}|}{c} t + \frac{4\pi|\mathbf{d_l} - \mathbf{a_n}|}{\lambda})\} \tag{3.2}$$

where $S = \frac{B}{T_c}$

After sampling with period $T_s = \frac{T_c}{N}$ with $N = 512$ as in equation 2.1, the measurements from the full array $\mathbf{y_{full}}$ is a 512-by-1024 matrix, where $\mathbf{y_{full}}[n_1, n_2]$ is given

Table 3.1: FMCW radar simulation parameter values

Name	Symbol	Value
Power of output	P_{out}	100 Watts
Speed of Light	c	299792458 m/s
Bandwidth	B	2 GHz
Center Frequency	f	77 GHz
Wavelength	λ	$\frac{c}{f} = 3.896$ mm
Radar element spacing	d	$\frac{\lambda}{2} = 1.948$ mm
Range Resolution	r_{res}	$\frac{c}{2B} = 7.5$ cm
Maximum Range	r_{max}	$512 r_{res} = 38.4$ m

by equation 3.3

$$\begin{aligned} \mathbf{y_{full}}[n_1, n_2] &= y_{n_2}(t)|_{t=n_1 T_s} \\ &= \sum_{l=0}^{L-1} \frac{P_{out}}{|\mathbf{d_l} - \mathbf{a_{n_2}}|^2} \exp\{j(2\pi \frac{2S|\mathbf{d_l} - \mathbf{a_{n_2}}|}{c} \frac{n_1 B}{512S} + \frac{4\pi|\mathbf{d_l} - \mathbf{a_{n_2}}|}{\lambda})\} \end{aligned}$$

$$\mathbf{y_{full}}[n_1, n_2] = \sum_{l=0}^{L-1} \frac{P_{out}}{|\mathbf{d_l} - \mathbf{a_{n_2}}|^2} \exp\{j(\frac{4\pi n_1 B}{512} \frac{|\mathbf{d_l} - \mathbf{a_{n_2}}|}{c} + \frac{4\pi|\mathbf{d_l} - \mathbf{a_{n_2}}|}{\lambda})\} \tag{3.3}$$

The raw data from the two sub-arrays are given by

$$\mathbf{y_0}[n_1, n_2] = \begin{cases} \mathbf{y_{full}}[n_1, n_2] & 0 \leq n_2 < 12 \\ 0 & otherwise \end{cases}$$

$$\mathbf{y_1}[n_1, n_2] = \begin{cases} \mathbf{y_{full}}[n_1, n_2 + 1012] & 0 < n_2 < 12 \\ 0 & otherwise \end{cases}$$

3.2.2 Post Processing

Since the loss function in equation 1.3 measures the squared error between the output to the neural network and some post-processed version of $\mathbf{y_{full}}$, $P(\mathbf{y_{full}})$, it is desirable

for $P(\mathbf{y_{full}})$ to be in the form that we can easily interpret without further processing. Making the input data to the neural network $P'(\mathbf{y_0})$ and $P'(\mathbf{y_1})$ as similar to the target data $P(\mathbf{y_{full}})$ as possible is helpful for speeding up training. Additionally, for the neural network to converge, it is also helpful to have the input data bounded by a known value.

Therefore, we first use Fourier analysis to get range-angle plots from $\mathbf{y_{full}}$, $\mathbf{y_0}$, and $\mathbf{y_1}$, then we normalize the log-magnitude of range-angle plots to be within range $[0, 1]$. We further incorporate back in the phase information to the input data so that we can reconstruct y_i given $P'(\mathbf{y_i})$ up to some scaling factor caused by normalization and phase unwrapping.

2.1.2.1. Fourier Analysis

We estimate the range-angle plots from all 1024 elements as in Equation 3.4.

$$\mathbf{Y_{full}}[k_1, k_2] = \sum_{n_2=0}^{1023}(\sum_{n_1=0}^{511} \mathbf{y_{full}}[n_1, n_2]w_1[n_1]\exp\{-j\frac{2\pi n_1 k_1}{512}\})w_2[n_2]\exp\{-j\frac{2\pi k_2 n_2}{1024}\} \tag{3.4}$$

where $w_1[\cdot]$ and $w_2[\cdot]$ are Hanning windows of lengths 512 and 1024. The first index k_1 in tracks the range from the center of the array. The second index k_2 tracks $cos(\theta)$, where θ is the angle relative to the center of the radar array whose locations are at $\mathcal{A}_{full}$.

Similarly, for $i = 0, 1$

$$\mathbf{Y_i}[k_1, k_2] = \sum_{n_2=0}^{1023}(\sum_{n_1=0}^{511} \mathbf{y_i}[n_1, n_2]w_1[n_1]\exp\{-j\frac{2\pi n_1 k_1}{512}\})w_2'[n_2]\exp\{-j\frac{2\pi k_2 n_2}{1024}\} \tag{3.5}$$

where $w_2'[n]$ for $0 \leq n < 12$ is a length-12 Hanning window and 0 for $n \geq 12$. Figure 3-4 shows the results of Fourier analysis of the full aperture array and the small subarrays. The ground truth scene is shown in the top left quadrant. $\mathbf{Y_{full}}$ is

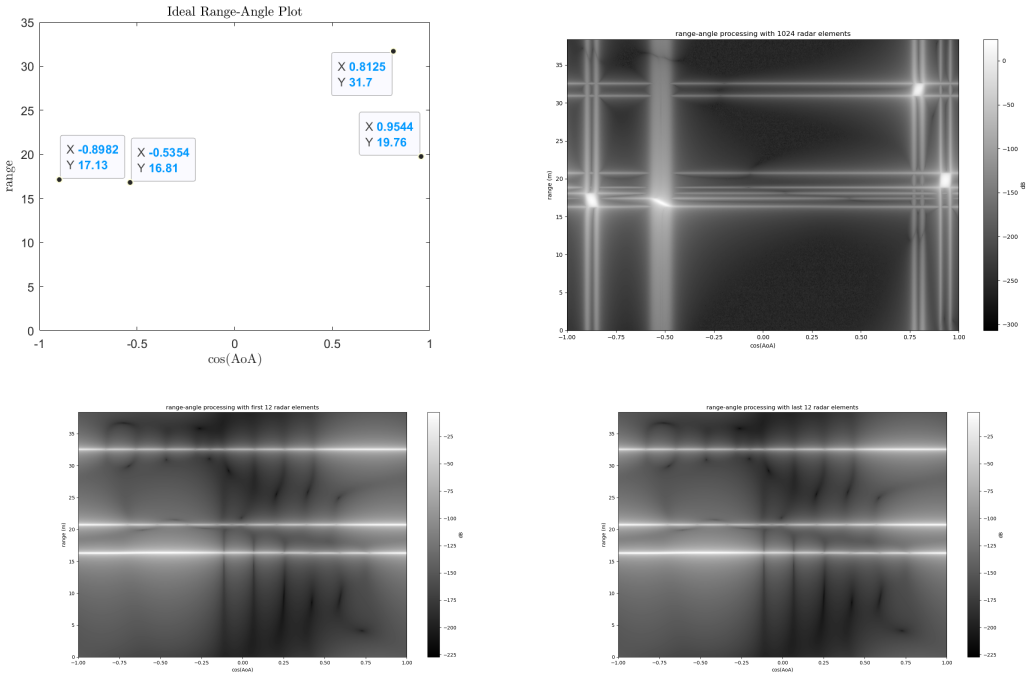


Figure 3-4: Fourier analysis of large and small aperture arrays.

visualized in the top right quadrant. $\mathbf{Y_0}$ and $\mathbf{Y_1}$ are visualized in the bottom two half.

2.1.2.2. Normalization

Since the power output was chosen arbitrarily in the simulation, the data can be normalized to be in the range $[0, 1]$. Normalizing the log-magnitude also makes sense because, based on equation 3.1, the reflected power decreases as $\frac{1}{r^2}$, echoes from points closer to the array do not dominate over the echoes from points farther away on the decibel scale. Therefore, the target data $P(\mathbf{y_{full}})$ from equation 1.3 is $\mathbf{\Lambda_{full}}$, where

$$\mathbf{\Lambda_{full}}[k_1, k_2] = \frac{\log|\mathbf{Y_{full}}[k_1, k_2]| - \min(\log|\mathbf{Y_{full}}[k_1, k_2]|)}{\max(\log|\mathbf{Y_{full}}[k_1, k_2]|) - \min(\log|\mathbf{Y_{full}}[k_1, k_2]|)}$$

Figure 3.2.2 shows an the normalized version of the range-angle plot from the top right of Figure 3-4.

Since the objective is to infer more information from the observed data, we want

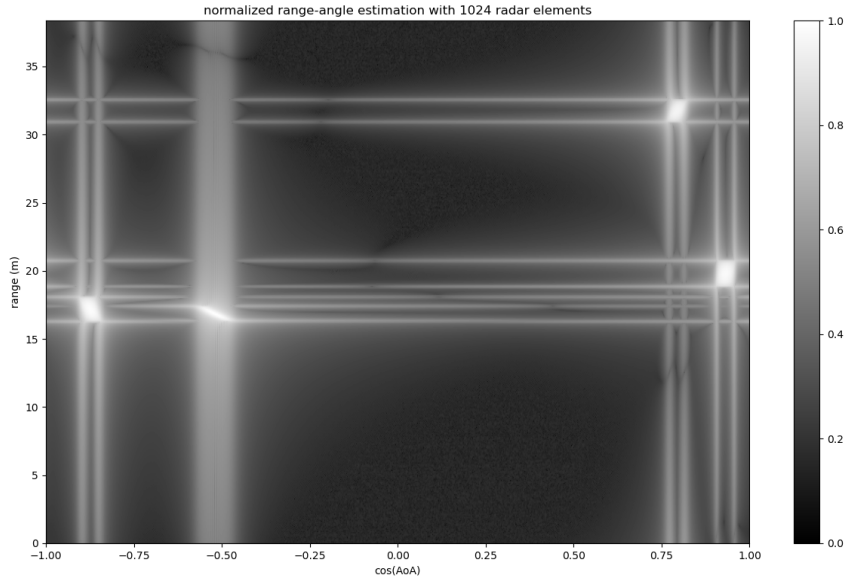


Figure 3-5: Normalized Range-Angle plot from 1024-element array data.

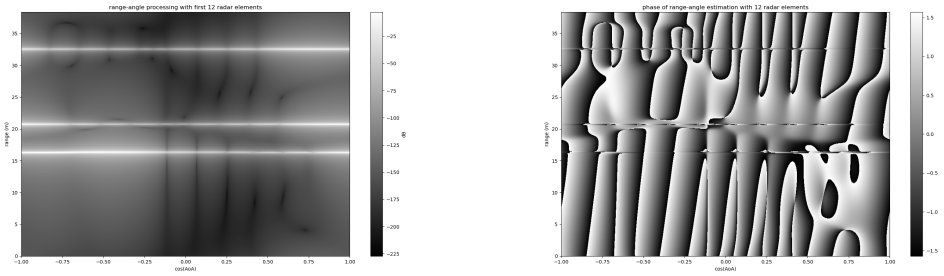


Figure 3-6: Magnitude and phase of the range-angle plot from 12-element array.

avoid discarding anything from the sub-array data, especially the phase information of the range angle data $\mathbf{Y_i}$. Figure 3-6 shows the magnitude of the range-angle plot given first 12 elements of the full aperture array, $|\mathbf{Y_0}|$, and the accompanying phase, $\angle|\mathbf{Y_0}| = \tan^{-1} \frac{\mathbf{Real\{Y_i\}}}{\mathbf{Real\{Y_i\}}}$.

Since the phase plot looks very different from the log-magnitude plot, it is not a ideal to directly give both as inputs to the neural network. Also, phase is only relevant when magnitude is non-zero. Therefore, the inputs to the neural network

$P'(\mathbf{y_i})$ for $i = 0, 1$ is $\mathbf{\Lambda_i}$, where

$$\mathbf{\Lambda'_i} = [\mathbf{\Lambda_i} \cos(\angle \mathbf{Y_i}), \mathbf{\Lambda_i} \sin(\angle \mathbf{Y_i})]$$

$$\cos(\angle \mathbf{Y_i}[k_1, k_2]) = \frac{\mathbf{Re}\{\mathbf{Y_i}[k_1, k_2]\}}{|\mathbf{Y_i}[k_1, k_2]|}$$

$$\sin(\angle \mathbf{Y_i}[k_1, k_2]) = \frac{\mathbf{Im}\{\mathbf{Y_i}[k_1, k_2]\}}{|\mathbf{Y_i}[k_1, k_2]|}$$

Figure 3-7 shows $\cos(\angle \mathbf{Y_0})$, $\sin(\angle \mathbf{Y_0})$, and $\mathbf{\Lambda'_0}$.

The input to the neural network is then a 4x512x1024 tensor:

$$[\mathbf{\Lambda'_0}, \mathbf{\Lambda'_1}] = [\mathbf{\Lambda_0} \cos(\angle \mathbf{Y_0}), \mathbf{\Lambda_0} \sin(\angle \mathbf{Y_0}), \mathbf{\Lambda_1} \cos(\angle \mathbf{Y_1}), \mathbf{\Lambda_1} \sin(\angle \mathbf{Y_1})]]$$

Given the post-processing steps, inputs to the neural network $\mathbf{\Lambda'_i}$ contains values in the range $[-1, 1]$ and magnitude proportional to the log-magnitude of $\mathbf{Y_i}$. Given $\mathbf{\Lambda'_i}$, we can reconstruct the normalized log magnitude $\mathbf{\Lambda_i} = \mathbf{\Lambda_i} \sqrt{\cos^2(\angle \mathbf{Y_i}) + \sin^2(\angle \mathbf{Y_i})}]$. We can also recover the phase of $\mathbf{Y_i}$ within a range over 2π. Since $\ln(\mathbf{Y_i}) = \ln(|\mathbf{Y_i}| \exp\{j \angle \mathbf{Y_i}\}) = \ln |\mathbf{Y_i}| + j \angle \mathbf{Y_i} + 2\pi k$ for any integer n, given $\ln |\mathbf{Y_i}|$ and $\angle \mathbf{Y_i}$, we can reconstruct $\mathbf{Y_i}$ up to a scaling factor from normalization and phase unwrapping, and $\mathbf{y_i}$ is the inverse DFT of $\mathbf{Y_i}$. This means that the input to the neural network preserves most of the information from the observed data.

3.3 Summary of Methods

We have a neural network whose architecture is described in 3-3. The input to the neural network has dimension 4x512x1024, and the output of neural network has dimension 1 x 512 x 1024. At the narrowest point, the data has dimension 64x16x32.

The flow chart in Figure 3-8 shows the training algorithm. At each iteration, we first simulate $\mathbf{y_{full}}$ given a scene with ideal point reflectors. We assume that we

observe $\mathbf{y_0}$ and $\mathbf{y_1}$, which are parts of the full array response $\mathbf{y_{full}}$. Then we process the observed data from the full array response to get $P(\mathbf{y_{full}}) = \mathbf{\Lambda_{full}}$ and process the small aperture array response to get $P'(\mathbf{y_i}) = \mathbf{\Lambda'_i} = [\mathbf{\Lambda_i} \cos(\angle \mathbf{Y_i}), \mathbf{\Lambda_{full}} \sin(\angle \mathbf{Y_i})]$ for $i = 0, 1$. The neural network then takes as input $[\mathbf{\Lambda'_0}, \mathbf{\Lambda'_1}]$ and produces output $\mathbf{\Lambda_{out}}$. The output of the neural network $\mathbf{\Lambda_{out}}$ is compared to the post-processed full-array data $\mathbf{\Lambda_{full}}$ to compute loss and update weights of the encoder and decoder simultaneously.

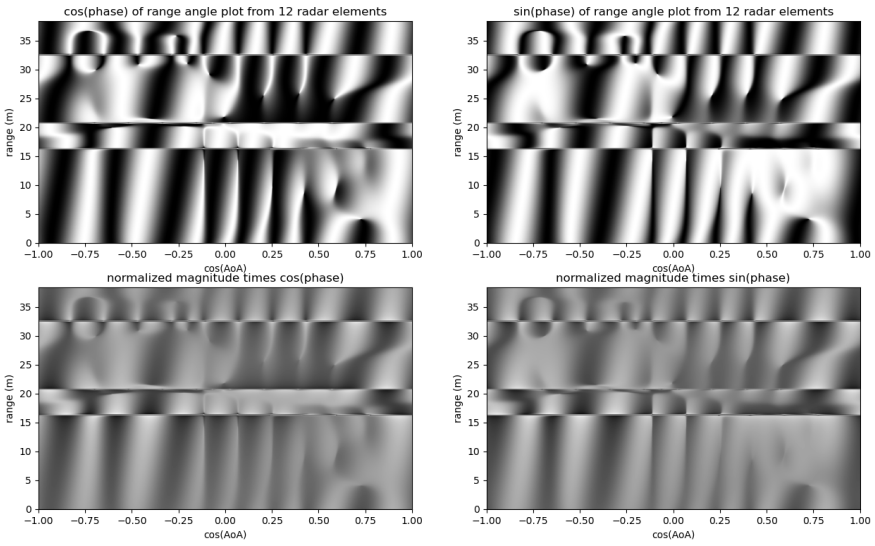


Figure 3-7: $\cos(\angle \mathbf{Y_0})$, $\sin(\angle \mathbf{Y_0})$, and $\mathbf{\Lambda'_0}$.

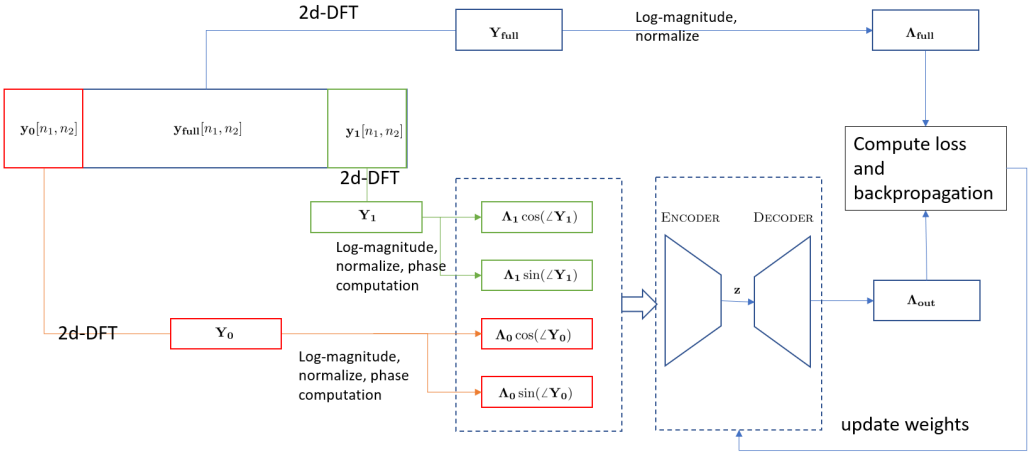


Figure 3-8: Summary of our setup and methods.

Chapter 4

Experiments and Results

This chapter first outlines two of the experiments we performed and presents the results of improvements from only using 12-element sub-array data.

4.1 Experiment 1

4.1.1 Data set

In this experiment, the neural network's weights were optimized through training on a data set which consists of 1000 example scenes with variable numbers of ideal point reflectors in noise-free vacuum. The training data set $\mathcal{D}$ was generated with the following steps for $L = 1 : 4$.

1. Create empty list of locations.

2. For l = 1: L, independently sample a range value r from normal distribution centered at $r_{max}/2$ with standard deviation $0.3r_{max}$ and an angle value θ from normal distribution with zero mean and unit variance. Clip the range and angle values to be inside the full radar array's vield of view. Add the $[r\cos(\theta), r\sin(\theta)]$ to the list of locations.

3. Assume ideal point reflector at each location in the list created in step 1, generate $\mathbf{\Lambda_{full}}$, $\mathbf{\Lambda_0'}$, and $\mathbf{\Lambda_0'}$ as described in Section 3.2.

4. repeat steps 1 - 3 for 200 times.

4.1.2 Training Parameters

The network was trained for 100 epochs, and training data was grouped into batch-size of 8. The weights were modified at each iteration with Stochastic Gradient Descent (SGD) with learning rate 0.1, and momentum 0.9.

4.2 Experiment 2

4.2.1 Data set

In experiment 2, the network was trained on a much larger and more diverse data set than in experiment 1. We generated the training dataset $\mathcal{D}$ with $|\mathcal{D}| = 10000$ by repeating the following steps for $L = 1 : 4$.

1. Create an empty list of locations,

2. For l = 1: L,

   (a) sample random variable u from uniform distribution over $[0, 1)$

   (b-i) if $u < 0.5$, independently sample a range value r from normal distribution centered at $r_{max}/2$ with standard deviation $0.3r_{max}$ and an angle value θ from normal distribution with zero mean and unit variance. Clip the range and angle values to be inside the full radar array's vield of view. Add the $[r\cos(\theta), r\sin(\theta)]$ to the list of locations.

   (b-ii) else, sample x_c from Uniform distribution over $[-0.8r_{max}, 0.8r_{max}]$ and y_c from uniform distribution over $[0, 0.8r_{max}]$ add location $[x_c, y_c + nr_{res}]$ for

$n = 1 : 75$ to the list of locations. Some of the locations might not be inside the radar's field of view and the reflections from those locations produce aliased versions on the other side of the range-angle plot.

3. Assume ideal point reflector at each location in the list created in step 1, generate $\mathbf{\Lambda_{full}}$, $\mathbf{\Lambda_0'}$, and $\mathbf{\Lambda_0'}$ as described in Section 3.2.

4. repeat steps 1 - 3 for 2000 times.

The result is that 50% of the targets in the data set are isolated reflectors, and the other 50% are clusters of reflectors in a vertical line.

4.2.2 Training Parameters

The network was again trained for 100 epochs. The training data was grouped into batch-size of 32. The weights were modified at each iteration also with Stochastic Gradient Descent (SGD) with learning rate 0.1, and momentum 0.9.

4.3 Results

The performance of the networks trained in Section 4 were evaluated on 3 different test data sets, each with 100 samples not used for training or validation in either experiment.

- Test Set 1: Single points. The data in this data set are the responses of radar data to scenes with a single point source in space.

- Test Set 2: Multiple Points. The scenes that produced the data contain 3 to 6 isolated ideal point reflectors in space.

- Test Set 3: Mixed Objects. The scenes contain 3 to 6 "objects." Each object has 50% chance of being an isolated point reflector, and 50% chance of being a

Table 4.1: Squared-Error Loss

Dataset/Experiment	Experiment 1	Experiment 2	Baseline
Test Set 1	7657.9339	1815.0327	20565.3039
Test Set 2	14264.3491	1341.2179	11334.9906
Test Set 3	12936.4824	1613.2859	17044.0797

cluster of point reflectors in a vertical line. We always have at least 1 cluster of points and 1 isolated point reflector that has the same range as part of the cluster.

The averaged squared-error losses over the 3 test data sets from the two experiments are summarized in Table 4.1. The last column "Baseline" shows the squared error between $\mathbf{\Lambda_0}$ and $\mathbf{\Lambda_{full}}$. The results from Experiment 2 is much improved from the results in Experiment 1. On average, given range-angle plots from two 12-element sub-arrays, the output of the neural network trained in Experiment 2 is more than 10 times closer in terms of squared-error to the range-angle plot from 1024-element array data than the range-angle data from either 12-element sub-array.

$$\mathbf{L}(\mathcal{D}, f) = \frac{1}{|\mathcal{D}|} \sum_{i=0}^{|\mathcal{D}|-1} ||\mathbf{\Lambda_{full}^{(i)}} - f([\mathbf{\Lambda_0'^{(i)}}, \mathbf{\Lambda_1'^{(i)}}]))||_2^2$$

4.3.1 Results from Experiment 1

Figure 4-1 shows some example results of testing the network on samples from Set 2. Images labeled as "net input 0" and "net input 1" show the normalized log-magnitude of the range-angle plots obtained from small aperture sub-arrays $\mathbf{\Lambda_0}$ and $\mathbf{\Lambda_1}$. Images labeled as "target" show the normalized log-magnitude of the range-angle plots obtained from the large aperture 1024-element arrays, $\mathbf{\Lambda_{full}}$, and images labeled

as "net output" are the outputs of the network trained in Experiment 1.

These results show that the network is able to learn the relationship between $\mathbf{\Lambda_0'}$ and $\mathbf{\Lambda_1'}$, and locate the peaks in each range. Even though the distance between the two sets of radar elements is not an input to the network, the network is able to infer how the input images are related to each other from the training set. This task is non-trivial because the difference in range of a point reflector perceived by the two small arrays and the phase difference all depend on the actual range and AoA.

When we test the network on samples from Set 3, we see that the output is very blurred and inaccurate around the clusters as seen in Figure 4-2. However, in most cases, the network can still reconstruct the isolated point sources. The failure to reconstruct the vertical line clusters is expected since they are not samples from the distribution that generated the training data.

4.3.2 Results from Experiment 2

Some example results of testing the network trained in Experiment 2 on Set 2 are shown in Figure 4-4. When the scene only contains a few point reflectors, the network can detect and locate them as long as they are sufficiently separated. The ability to resolve points very close to each other, though still limited, is improved compared to the results from Experiment 1. Figure 4-3 shows a side by side comparison of an Experiment 2 result versus an Experiment 1 result. The network trained in Experiment 1 cannot accurately resolve the two points at range 31m, while the new network from Experiment 2 is able to accurately reconstruct the two points. Even though only a small portion of the training data in Experiment 2 contains 3 or more point reflectors, the network is still able to easily reconstruct the scenes as shown in Figure 4-4. Even when the scene contains only a single point reflector, the network is able to accurately predict its location, as seen in Figure 4-5. This result shows that the training data set contains enough samples for the model to learn these low-probability events.

Figure 4-6 shows some results of testing the network on samples from Set 3. Because most of the test data from Set 3 are from the same distribution as the training and validation data, outputs of the network are far less blurry and more accurate around vertical line clusters compared to the results of previous experiment in Figure 4-2. This improvement shows that the neural network performs very well when the assumed prior distribution, which we used to generate the training data, matches the "actual" prior distribution that generates the testing data.

4.3.3 Probabilities of Detection and False Alarm

In the context of radar, the probability of detection P_D and the probability of false alarm P_{FA} are important metrics of performance. Given network output $\mathbf{\Lambda_{out}}$, the target data $\mathbf{\Lambda_{full}}$, and thresholds $\eta = 0.7$ and $\eta' = 0.9$ on the network output and the target data, the predicted reflector locations are $\mathbf{\Lambda_{out}} > \eta$ and the target result is $\mathbf{\Lambda_{full}} > \eta'$. Then the probability of detection, P_D, is given by equation 4.1. Similarly, the probability of false alarm, P_{FA}, is given by equation 4.2. Figure 4-7 shows two examples of the results of applying threshold η on the output of the neural network trained in experiment 2 and applying threshold η' on the target data $\mathbf{\Lambda_{full}}$.

$$P_D = \mathbb{P}[\mathbf{\Lambda_{out}} > \eta | \mathbf{\Lambda_{full}} > \eta'] = \frac{\mathbb{P}[\mathbf{\Lambda_{out}} > \eta \cap \mathbf{\Lambda_{full}} > \eta']}{\mathbb{P}[\mathbf{\Lambda_{full}} > \eta']} \tag{4.1}$$

$$P_{FA} = \mathbb{P}[\mathbf{\Lambda_{out}} > \eta | \mathbf{\Lambda_{full}} < \eta'] = \frac{\mathbb{P}[\mathbf{\Lambda_{out}} > \eta \cap \mathbf{\Lambda_{full}} < \eta']}{\mathbb{P}[\mathbf{\Lambda_{full}} < \eta']} \tag{4.2}$$

Without using the neural networks, the probability of detection and probability of false alarm using only the data from the first 12-element sub-array are given by equations 4.3 and 4.4. Figure 4-8 shows the results of applying threshold η' on $\mathbf{\Lambda_0}$

and $\mathbf{\Lambda_{full}}$. These values serve as a baseline for comparison.

$$P_D = \mathbb{P}[\mathbf{\Lambda_0} > \eta' | \mathbf{\Lambda_{full}} > \eta'] = \frac{\mathbb{P}[\mathbf{\Lambda_0} > \eta' \cap \mathbf{\Lambda_{full}} > \eta']}{\mathbb{P}[\mathbf{\Lambda_{full}} > \eta']} \tag{4.3}$$

$$P_{FA} = \mathbb{P}[\mathbf{\Lambda_0} > \eta' | \mathbf{\Lambda_{full}} < \eta'] = \frac{\mathbb{P}[\mathbf{\Lambda_0} > \eta' \cap \mathbf{\Lambda_{full}} < \eta']}{\mathbb{P}[\mathbf{\Lambda_{full}} < \eta']} \tag{4.4}$$

The P_D and P_{FA} values using the networks trained in the two experiments and using the baseline method are summarized in Tables 4.2 and 4.3.

The probability of detection, P_D over Test Set 1 using the network trained in Experiment 2 is 98.98%, only slightly better than the P_D of 99.90% using the network trained in Experiment 1. However, the probability of false alarm, P_{FA} over Test Set 1 using the network trained in Experiment 2 is 0.20%, which is a significant improvement over the $P_{FA} = 0.37\%$ with the network trained in Experiment 2. With the baseline method of applying a threshold on $\mathbf{\Lambda_0}$, the P_D over Test Set 1 is only 12.40%, with $P_{FA} = 0.2\%$. The low probability of detection using the baseline method can be explained by the fact that the distance from the point reflector to the center of the large-aperture array is different from the distance to the center of the first 12-element sub-array, and that difference is often larger than the range-resolution. The neural networks, on the other hand, were trained to predict the actual location of the point reflector.

Over test set 3, the P_D value from Experiment 1 results is about 55.7%, which

Table 4.2: Probability of Detection

Dataset/Experiment	Experiment 1	Experiment 2	Baseline
Test Set 1	98.90%	98.98%	12.4%
Test Set 2	72.55%	82.44%	6.89%
Test Set 3	58.77%	91.97%	67.53%

Table 4.3: Probability of False Alarm

Dataset/Experiment	Experiment 1	Experiment 2	Baseline
Test Set 1	0.44%	0.37%	0.2%
Test Set 2	0.63%	0.56%	0.56%
Test Set 3	2.51%	1.02%	5.57%

is expected since test set 3 is generated from a very different distribution from the training set in Experiment 1. By contrast, P_D from Experiment 2 results is much higher at 91.97% compared to both P_D with the baseline method and from Experiment 1.

Overall, the P_D values from both Experiment 1 and Experiment 2 are significantly higher than the P_D achieved with the baseline method. The results from Experiment 2 have P_{FA} values no larger than results using the baseline method.

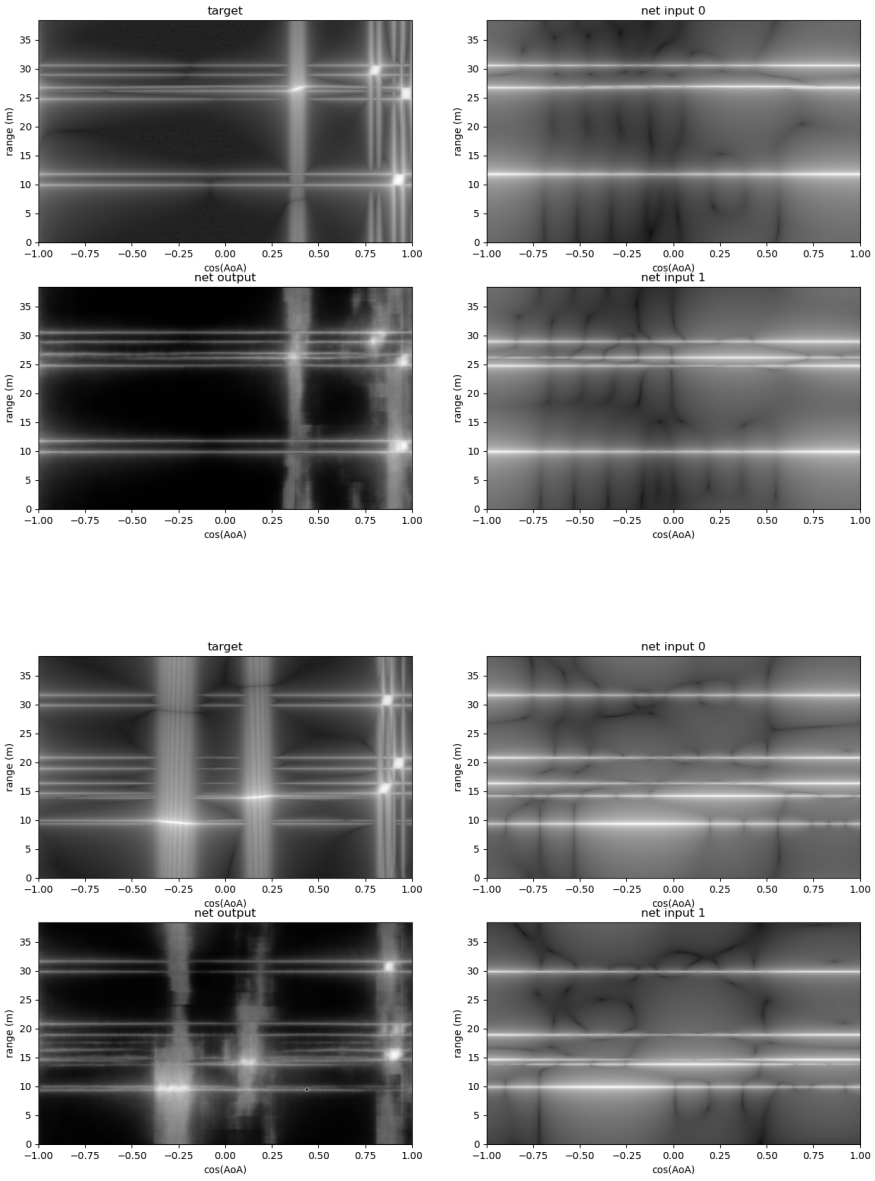


Figure 4-1: Results from Experiment 1 on Multiple Points.

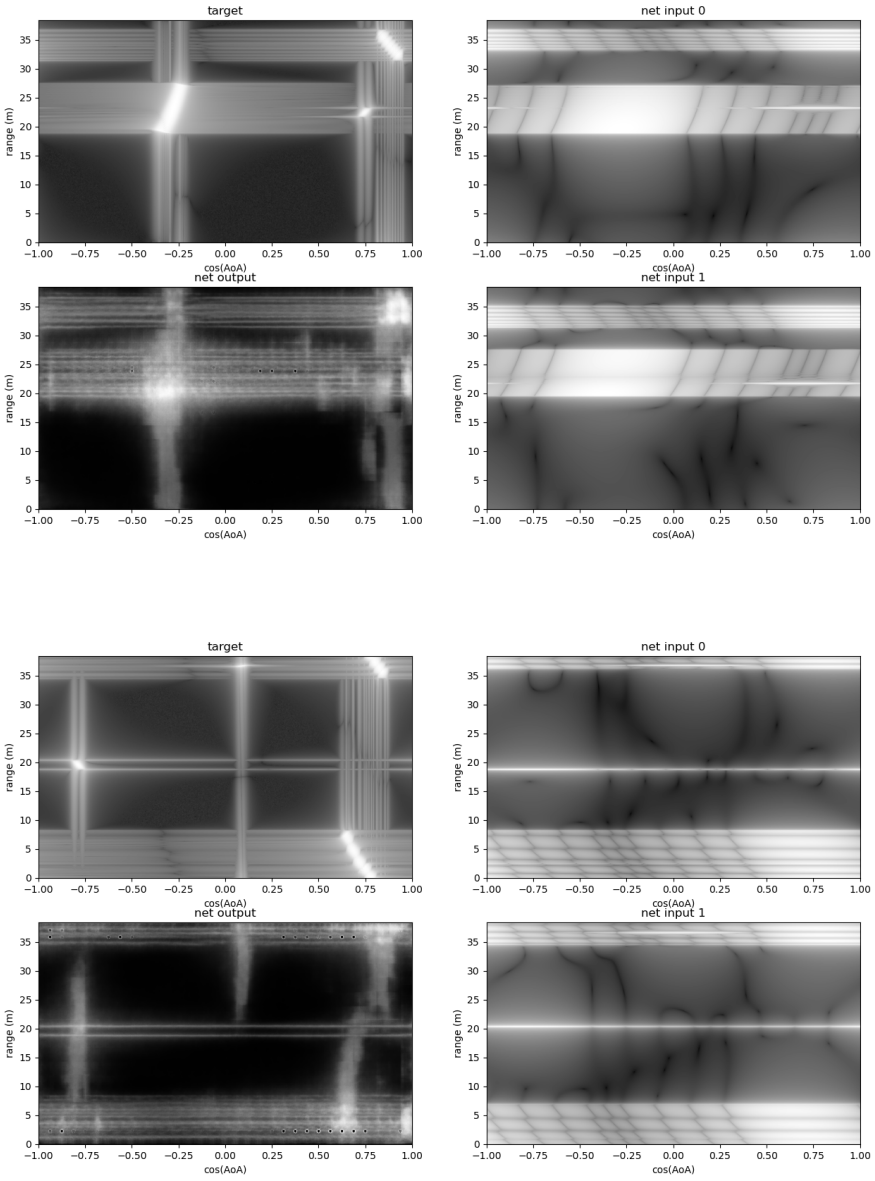


Figure 4-2: Results from Experiment 1 on Mixed Objects.

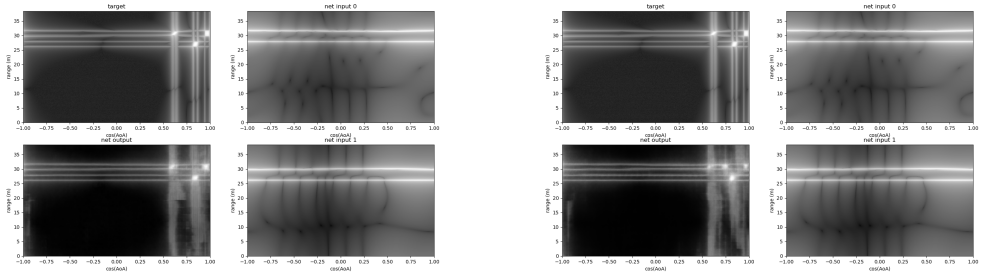


Figure 4-3: Comparison of results from Experiment 1 (left) with result from Experiment 2 (right).

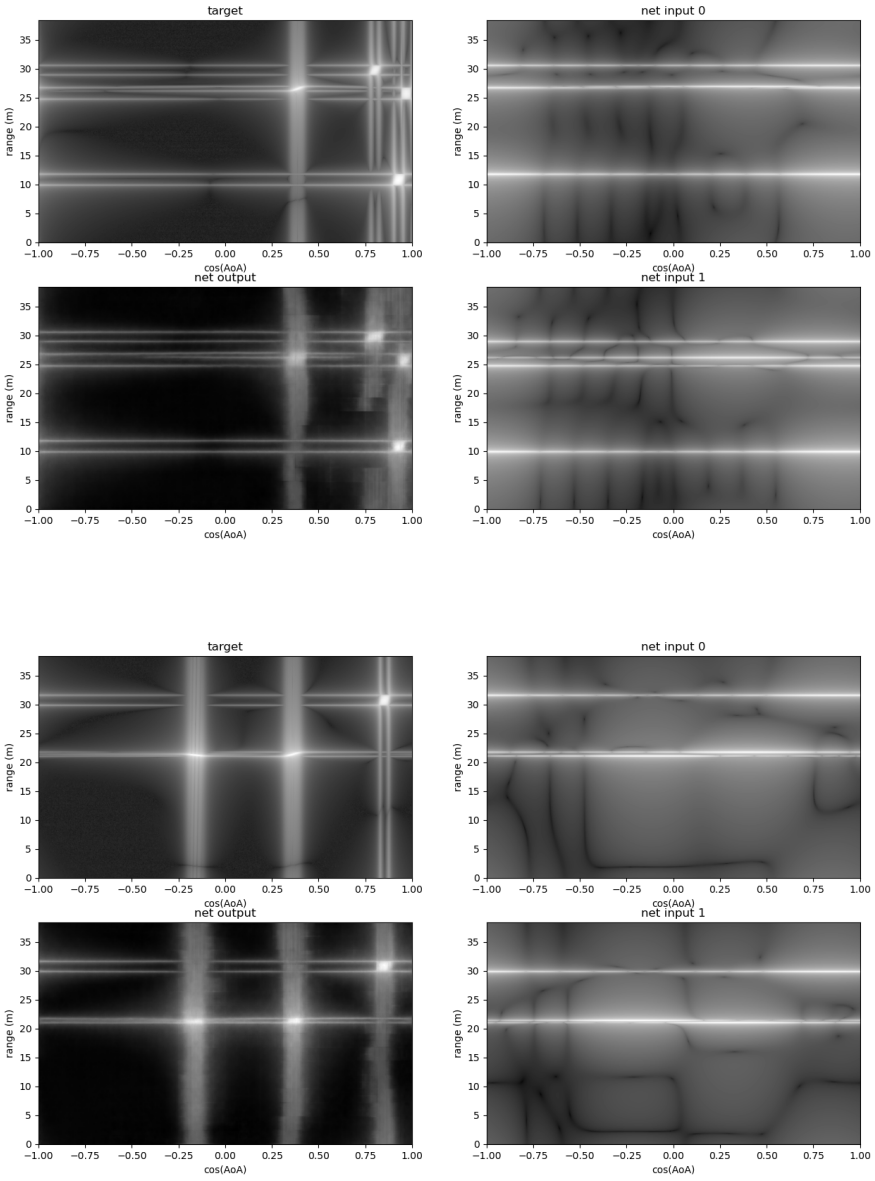


Figure 4-4: Results from Experiment 2 on Multiple Points.

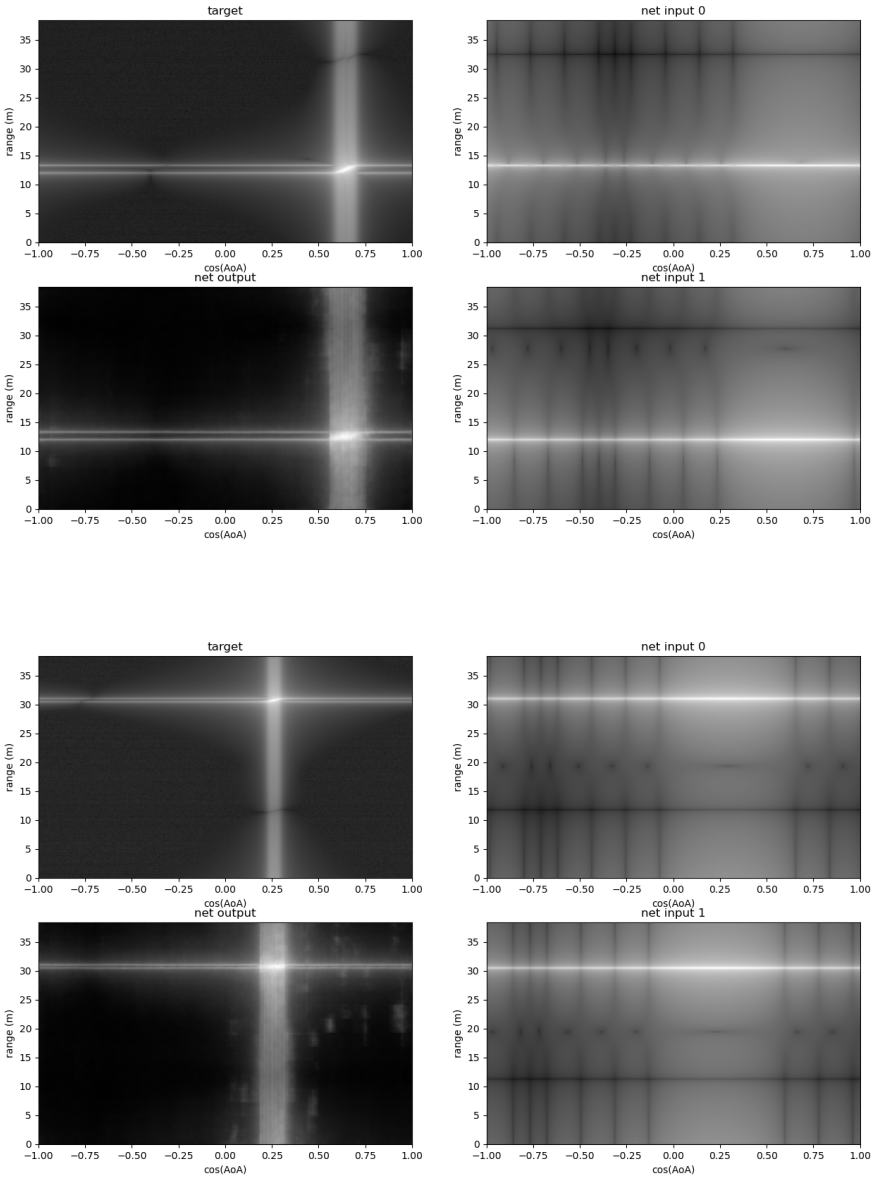


Figure 4-5: Results from Experiment 2 on Single Points.

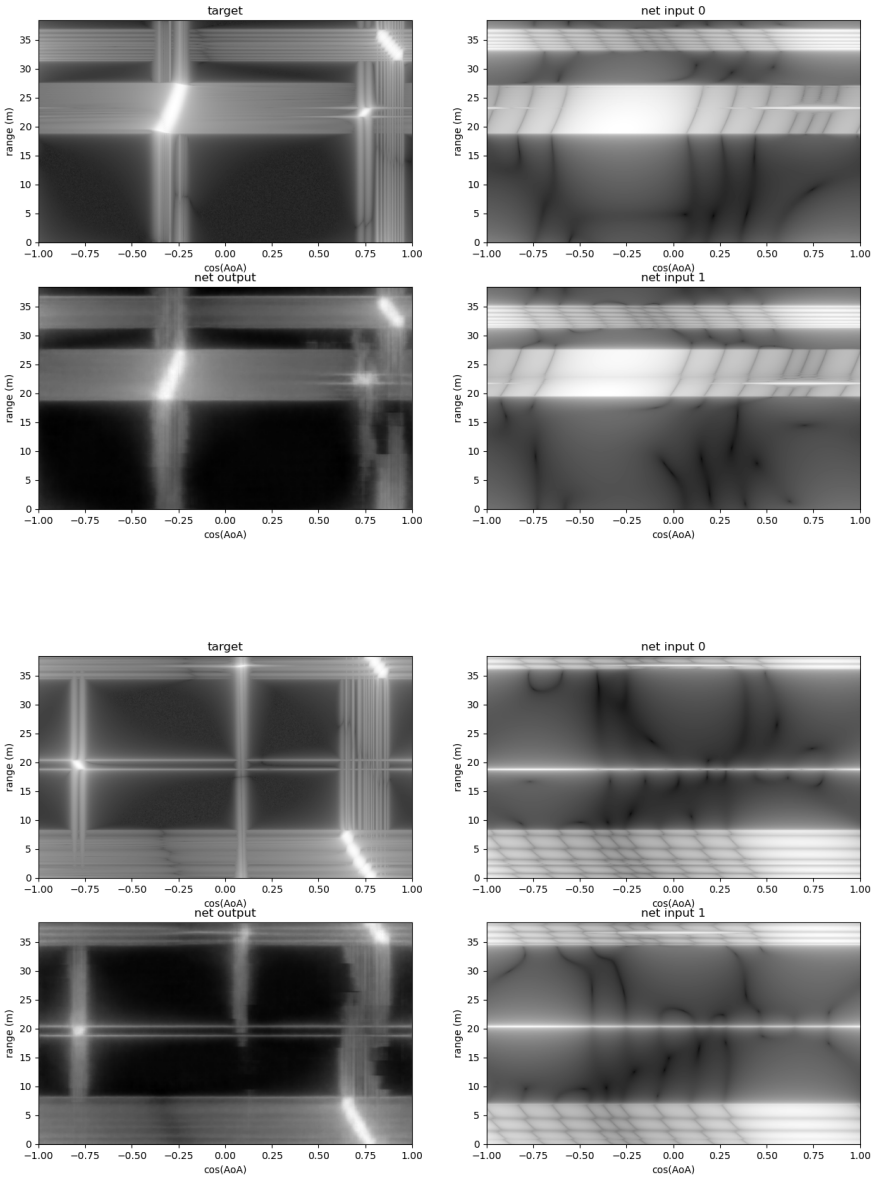


Figure 4-6: Results of Experiment 2 on Mixed Objects.

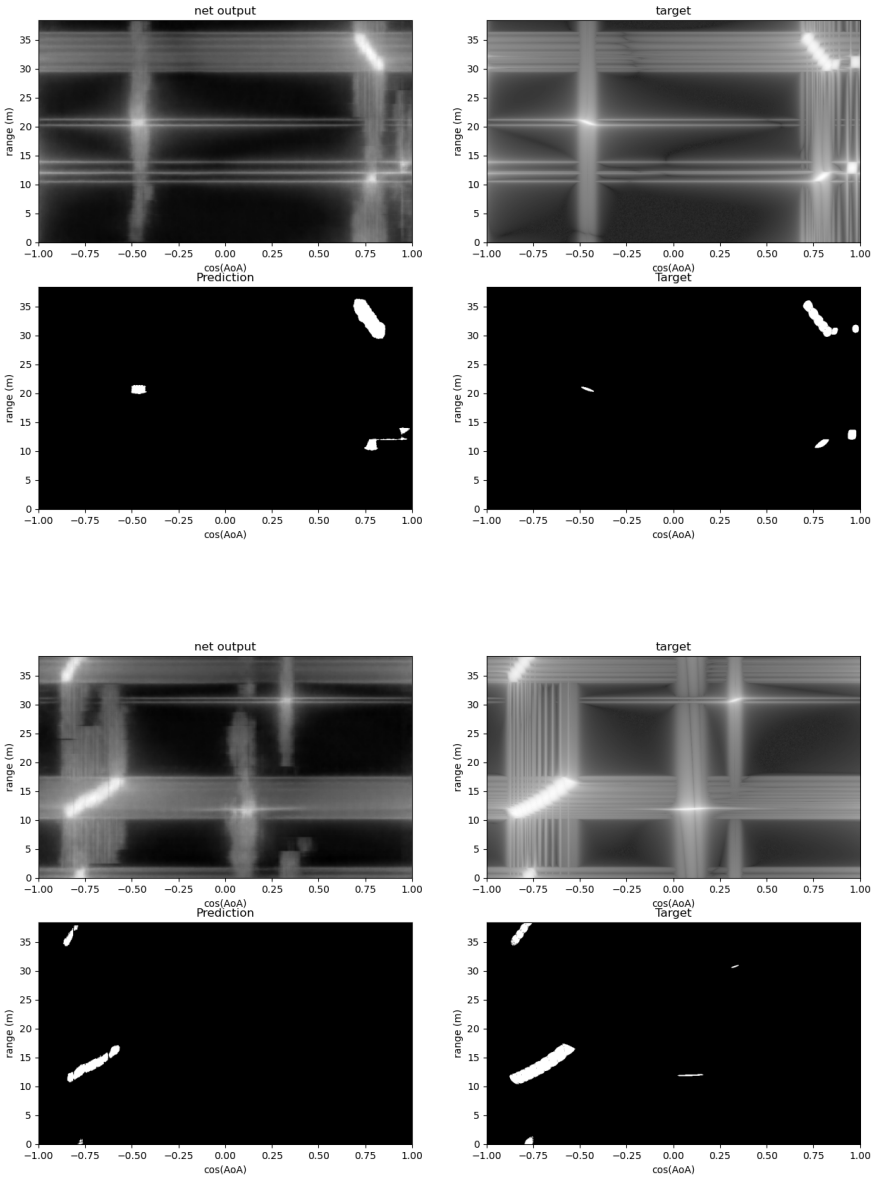


Figure 4-7: Results of Experiment 2 on test set 3 before and after threshold.

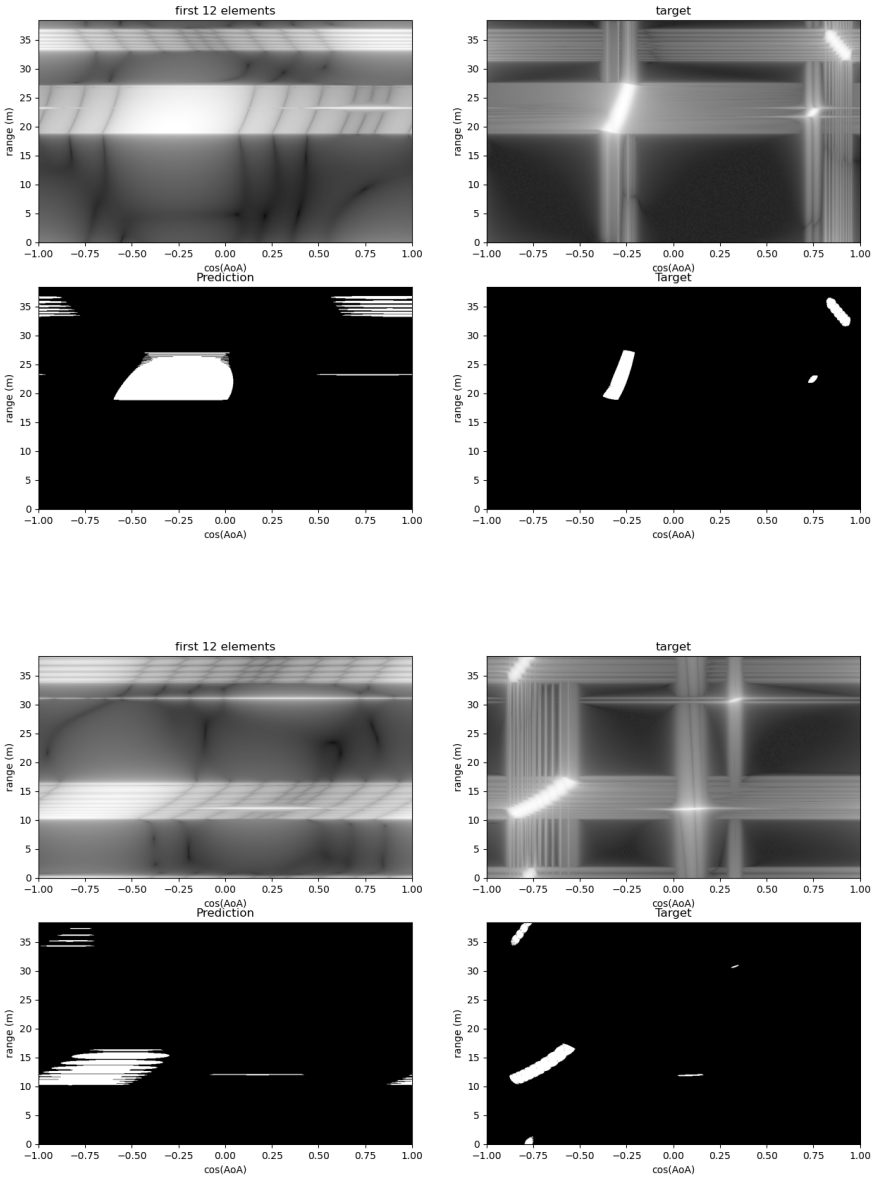


Figure 4-8: Baseline prediction results with a 12-element sub-array.

Chapter 5

Discussion

This chapter discusses how the choices made in this thesis regarding loss function, network architecture, and training data affect the results presented in Section 4.3.

5.1 Effects of Loss Function

As expected, the network from Experiment 2 has averaged loss almost 1/10 of the loss of the network from Experiment 1. This result matches the assumption that training on more data generally leads to better performance.

From a visual perspective, the network from Experiment 1 does not perform as well on samples from Set 3 as it does on samples from Set 1 or Set 2, as seen in Figures 4-1 and 4-2, yet, from Table 4.1, the average loss of Experiment 1 on Set 3 is lower than its averaged loss on set 2. This implies that the loss function is not most suited for measuring the ability of the network to perform the task of super-resolution. The squared error is not a good measure for resolution comparison. For example, consider signal $y[n] = \delta[n-200]+\delta[n-250]$ and two estimates $\hat{y}_1[n] = \delta[n-201]+\delta[n-251]$ and $\hat{y}_2[n] = \sqrt{\frac{1}{205}}(1-\delta[n-200]-\delta[n-250])$ as shown in Figure 5-1. The squared errors of the two estimates, $||y-\hat{y_1}||_2^2$ and $||y-\hat{y}_2||_2^2$, are both 4, but $\hat{y_1}[n]$ is a much more desirable approximation to $y[n]$ than $\hat{y_2}[n]$. However, a neural network trained to

minimize the squared error might converge at a local minimum that produces signals like $\hat{y}_2$. Signals like $\hat{y}_2$ are also much easier to produce via convolution than signals like $\hat{y}_1$. This effect is especially pronounced in the results of testing the network from Experiment 1 on test set 3.

Another reason that squared error as a loss function is not ideal is that it discriminates against sparse signals. For example, for signals $y_1[n]$, $y_2[n]$ shown in Figure 5-2, if a network is trained to generate $\hat{y}_i = 1.2y_i[n]$, then the squared error $||y_2 - \hat{y}_2||_2^2$ is twice the squared error $||y_2 - \hat{y}_2||_2^2$. Therefore, when a network is trained on data with varying levels of sparsity, as in Experiment 2, the errors for more sparse signals are weighted less. This can be seen from results of Figure 5-3, where the two points in the blue box seen in the image labeled as "target image" are mistakenly reconstructed as a vertical line cluster in the box in the image labeled as "net output".

Alternative loss functions that we considered include the Wasserstein loss, also known as the Earth-Mover (EM) distance, binary cross entropy (BCE) loss, peak signal-to-noise ratio (PSNR), MSE loss normalized with respect to sparsity, or probability of detection [16]. However, these loss functions are either difficult to implement, like Wasserstein loss, or are difficult to converge to a desired minimum.

5.2 Effects of Network Architecture

The network architecture as described in Figure 3-3 is convolutional in nature, while convolutional layers work very well at capturing local statistics and applying local statistics to tasks locally, they are not great at capturing some global characteristic for very localized tasks. As seen in Figure 5-4, which shows the normalized log-magnitude of the range-angle plot from one of the small aperture arrays, $\mathbf{\Lambda}_0$, for two different scenes. The left image corresponds to a scene with a single ideal point reflector at $[0, 20m]$, and the right image corresponds to a scene with a single point

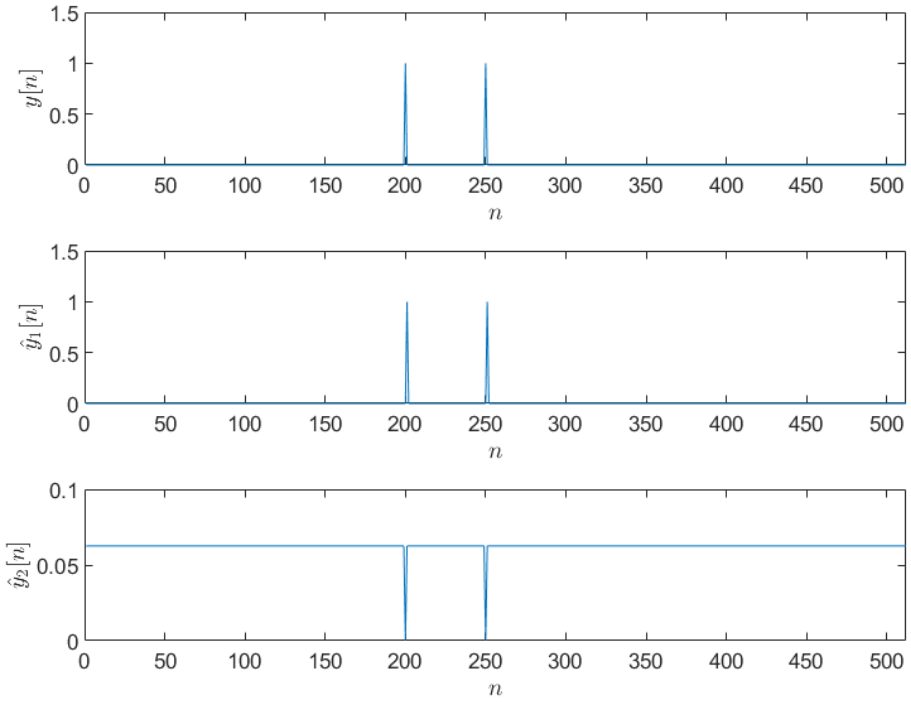


Figure 5-1: Squared error is a bad measure for resolution.

reflector at $[20\cos(\frac{\pi}{2}-0.1)m, 20\sin(\frac{\pi}{2}-0.1)m]$. Slight changes in angle can result in phase changes in the radar array whose effects ripple through the whole image even after taking the Fourier transform. While the two images in Figure 5-4 are virtually indistinguishable in the center region, there are noticeable differences near the edges and at range 13m, where the image on the left shows a dark line that cannot be seen in the image on the right. Ideally, we would want the network to capture these global characteristics and apply them to only the regions around where the small aperture radar signal is non-zero. The convolutional structure is likely not efficient at using the information in the surrounding area to inform the super-resolution within a small region of interest. Moreover, convolution produces undesirable artifacts that are not characteristic of radar signals, such as blurring as seen in Figures 4-6. An important direction to explore in the future is finding a better architecture that works in a more intuitive way given our goal and our data.

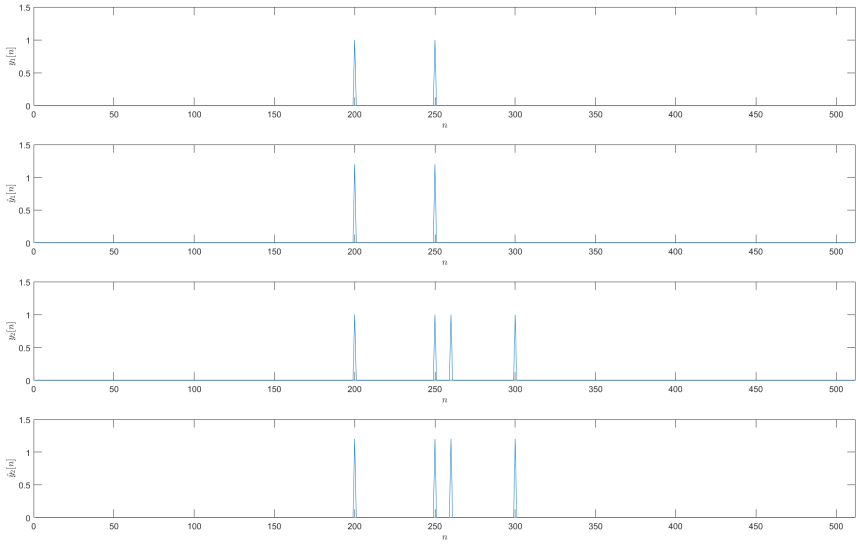


Figure 5-2: Squared error discriminates against sparse signals.

5.3 Effects of Data Distribution

Even though we do not make explicit assumptions about sparsity, we implicitly impose sparsity constraints by limiting our training data to those that are sparse. The effect on the neural network is that the network learns the distribution from which the training data is sampled from. When the testing data is sampled from an entirely different distribution, the model does not perform well as seen in the results of testing Model 1 on the kind of data Model 2 is trained on, and testing Model 2 on data with different shapes of clusters. Figures 5-5 show some examples of testing the network from Experiment 2 on scenes with clusters of points that are from a Gaussian distribution around a center point. Since the network has only seen clusters in the shape of a vertical line, it detects the Gaussian clusters but reconstructs them as vertical lines. Therefore, it is important for the training data to have statistics that match those seen in real life.

While ideal point reflectors do not exist in real life, radar sensors on vehicles will commonly encounter pedestrians, other cars, and railings. Even though we trained

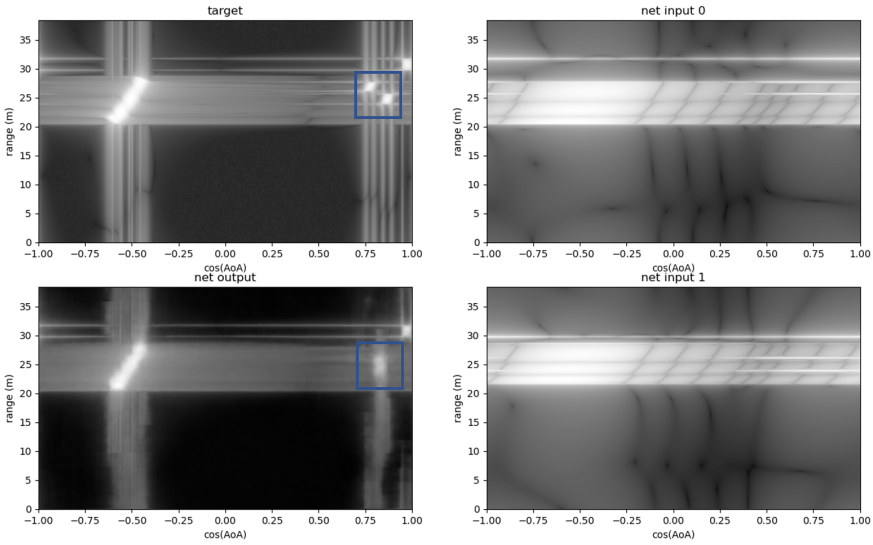


Figure 5-3: Mistaking points for cluster.

our models on ideal point reflectors, we can think of the isolated point of reflectors as a "spherical cow" kind of representation for pedestrians and the points clustered in a vertical line as representations of the railings and sides of cars or trucks. Results of Model 1 and Model 2 both show that the accuracy of our reconstruction is as good as the assumption we make about the prior distribution of the ground truth scenes. If we had access to the realistic data taken from the road, we would be able to estimate the true distribution of the number of "pedestrians" and "railings" and "cars" from actual measured data instead of assuming uniform or Gaussian distribution. Since we have the ability to sample from arbitrary distribution with Monte Carlo methods, we could have trained the model on a set of data that is more representative of what the radar array on cars could encounter. Part of the difficulty with using Machine Learning based algorithms on radar data is the lack of standard libraries of labeled radar data, but as interest grows in radar, and the use of radar becomes more widespread in autonomous vehicles, such libraries may begin to emerge, and sampling synthetic data from a realistic probability distribution will become feasible.

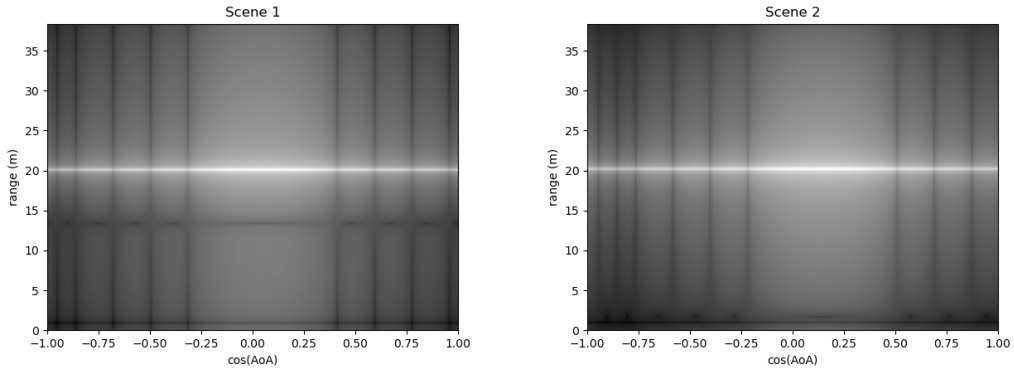


Figure 5-4: Small change, big difference.

5.4 Limitations to Super-resolution

We use inputs from small aperture arrays from the two ends of the full array because we hope that having a wider total aperture will allow us to do better than having a single 12-element array or a 24-element array. In most cases, this assumption is true because having a wide total aperture is like looking at the scene from two different perspectives. Having a new perspective generally provides a little more information. However, since the total aperture (2 meters) is much larger than the range resolution (0.075 meters) and a non-negligible fraction of the maximum range (34 meters), the small aperture arrays on two ends will have some parts of field of view that do not overlap as illustrated in Figure 5-6. If a point is out of the field of view of one of the small aperture arrays or is occluded by another cluster, then the perspective from this small aperture array does not add any information. Therefore, we rely solely on one of the arrays. In Figure 5-7, the point in the box is occluded by the cluster of points from the point of view of the small aperture array on the left, and we do not expect the neural network to be able to reconstruct this point.

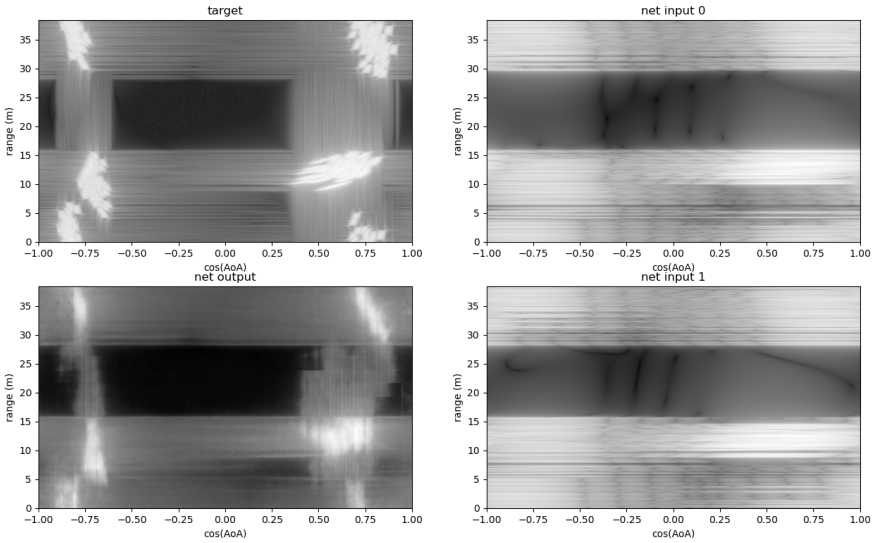


Figure 5-5: Results from Experiment 2 on Gaussian clusters.

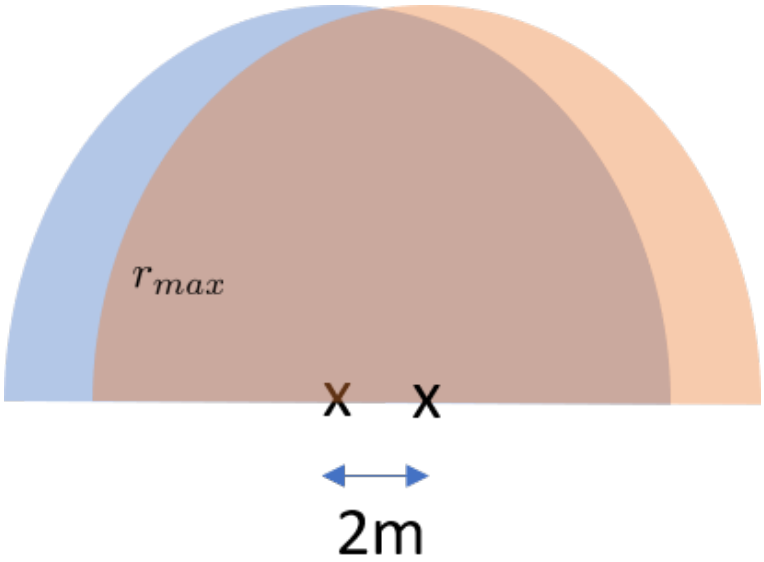


Figure 5-6: FOV of subarrays.

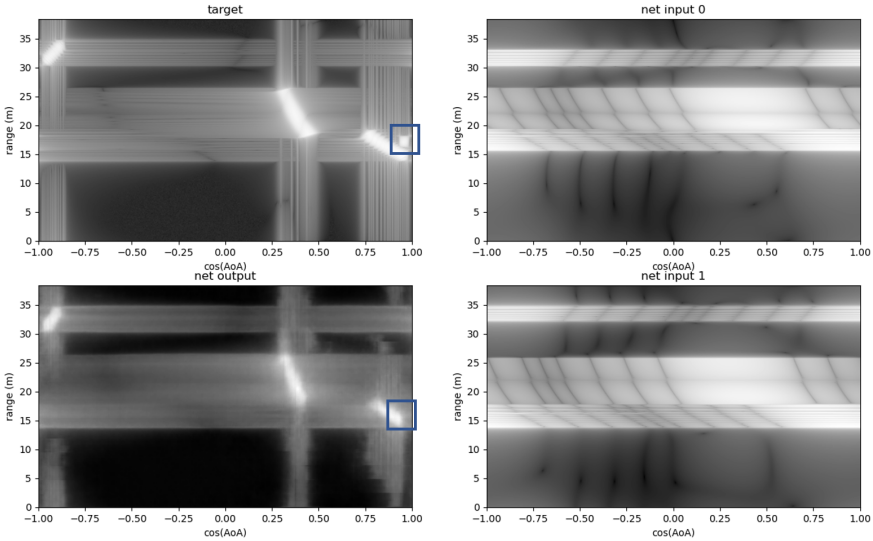


Figure 5-7: Limitations to super-resolution.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we investigated the problem of super-resolution given radar signals from two 12-element arrays spaced across a wider aperture. Instead of trying to solve the problem with traditional signal processing methods, we formulated this problem from a inference perspective and adopted a machine learning approach to solve the problem.

We trained a neural network with an encoder-plus-decoder architecture that encodes the inputs, which are normalized log-magnitude and phase information from the small-aperture 12 element arrays, into a lower dimensional tensor, which the decoder decodes as outputs that approximate the normalized log-magnitude of range-angle information from large-aperture 1024-element arrays. The final neural network is able to reconstruct locations of isolated points and clusters that would be too close to be resolved by the small-aperture arrays alone.

The results from this work show that, by learning to model the underlying distribution that generates scenes, we can achieve much higher resolution with radar signals from practically realizable small aperture arrays placed across a wide aperture than

traditional signal processing methods allow. The neural network improves the processing of the sub-array signals significantly both in terms of squared error compared to the full aperture results and in terms of probabilities of detection and false alarm. This work also shows the limitations of adapting convolutional neural networks that perform well on natural images to radar data. While fully convolutional networks and UNet do not work well with radar signals, GANs and VAEs show much greater capabilities of adapting to the statistics of radar signals.

However, our work in this thesis still leaves much room for improvement. There are still many limitations to the performance of our neural network. We also made strong assumptions about our input data since we generated our training and testing data by assuming ideal point reflectors in noise-free vacuum and ignoring additional physics. Nonetheless, this work lays foundation for very promising results for radar signal super resolution and bodes a bright future for the use of radar sensors in the automotive industry. Moreover, this work starts the investigation into applying machine learning methods to the super-resolution of sparse signals that are similar to radar signals.

6.2 Future Work

We plan to further investigating the problem by exploring in the following directions.

6.2.1 Different Data for Training and Testing

Ideally, we would like to train the network with real radar data from roads, and use LIDAR sensors to create a ground truth scene.

We initially hoped to use Ansys High Frequency Structure Simulator (HFSS) with Shooting and Bouncing Rays + (SBR+) solution type to simulate FMCW radar responses for training the neural network. The simulation takes into account geometry,

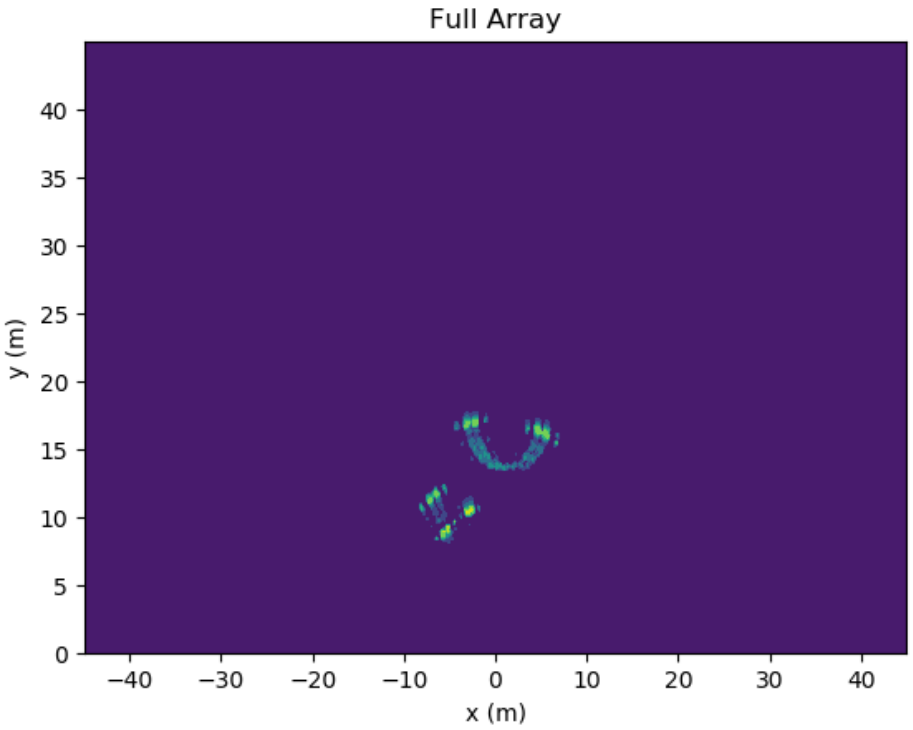


Figure 6-1: HFSS SBR+ simulation example.

diffraction at corners, and surface physics. However, it took 40 minutes on average to simulate a the response of a 1024-virtual-element array to a scene with simple geometric objects even with MIMO implementation. Figure 6-1 shows the radar image produced with HFSS when the scene contains a cylinder and a cube oriented at an angle. While HFSS SBR+ simulation is far more realistic than our simulation that assumes ideal point reflectors, it was unfortunately prohibitively time consuming to use the software to generate thousands of examples with the computing resources available.

With more time, we plan to simulate radar data with HFSS SBR+ given the same specifications in Table 3.1 and test the networks trained in Experiment 2 on the new simulated data.

6.2.2 Wasserstein GAN

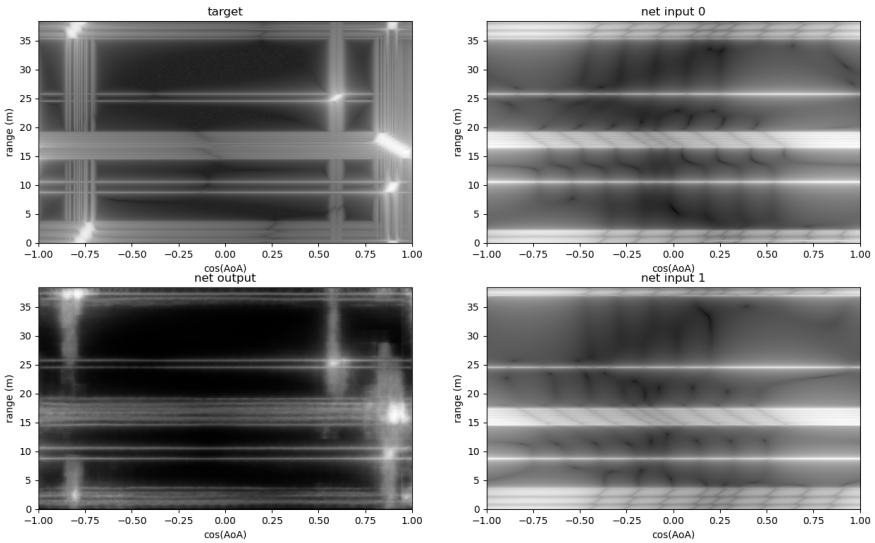
The Wasserstein Loss, also known as the earth-mover distance, measures the difference between two probability distributions. If we think of two probability distributions as two piles of dirt with equal mass, and we have an algorithm that makes the two piles of dirt, or two distributions identical by repeatedly taking some mass of "dirt" from the first pile and moving it some distance, then we can compute the sum of mass of dirt moved multiplied by the distance it was moved by. The Wasserstein loss is the minimum over all possible algorithms.

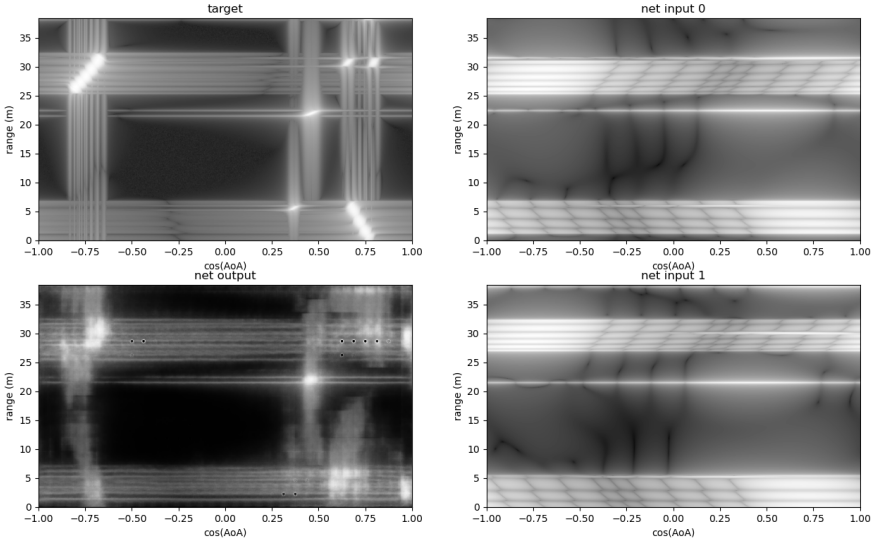
[16] proposes an algorithm to train a GAN with the Wasserstein loss, and we would like to further explore the possibility of adapting the algorithm to add a regularization term to the loss function.

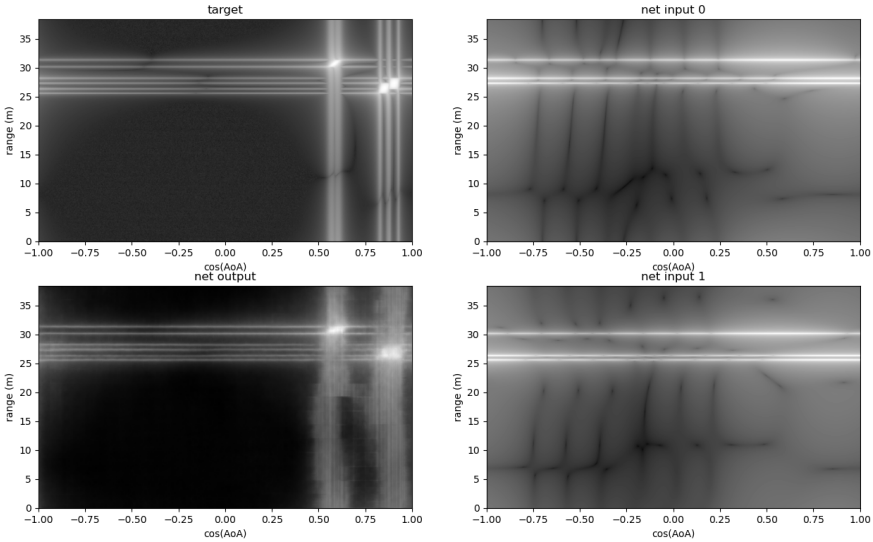
Appendix A

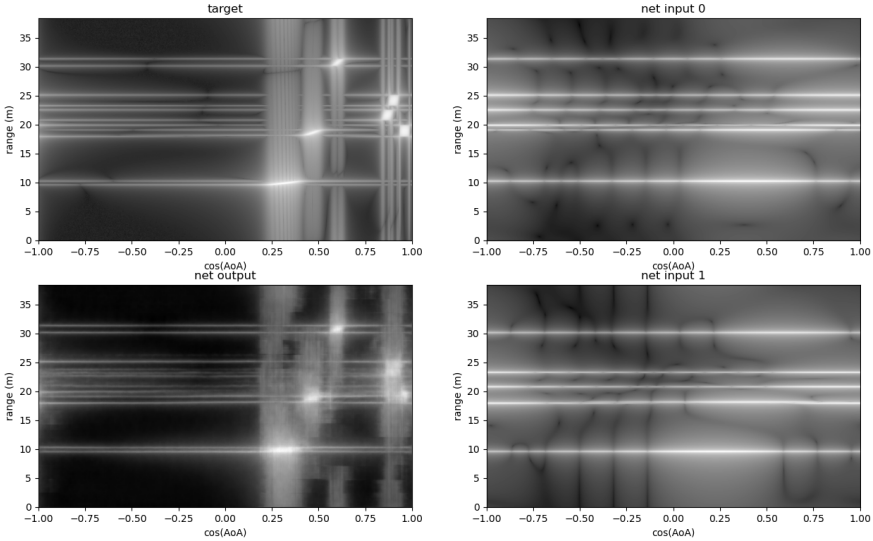
Additional Results

A.1 Additional Results from Experiment 1

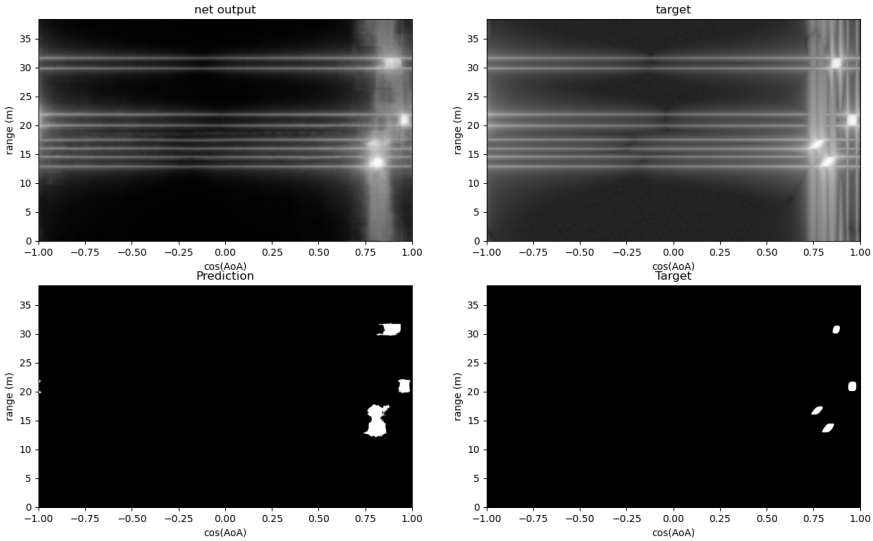


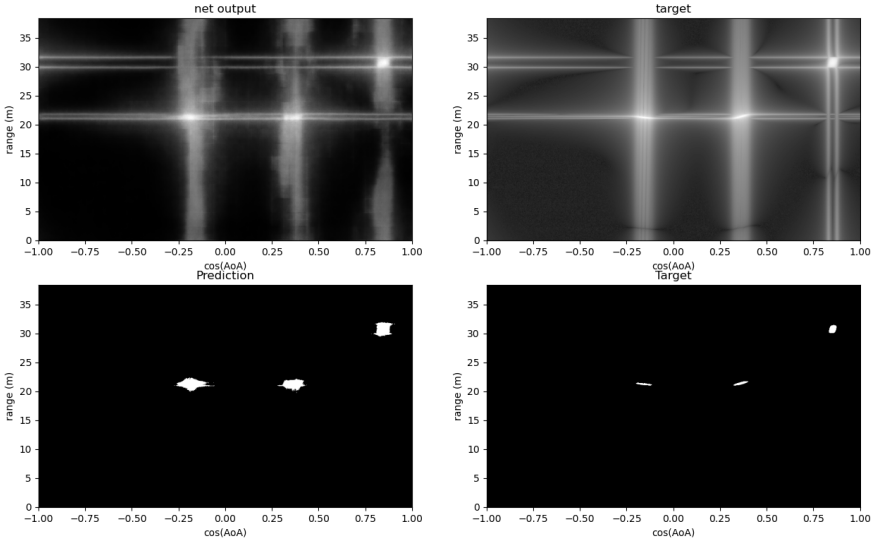

target
net input 0
net output
net input 1
range (m)
cos(AoA)

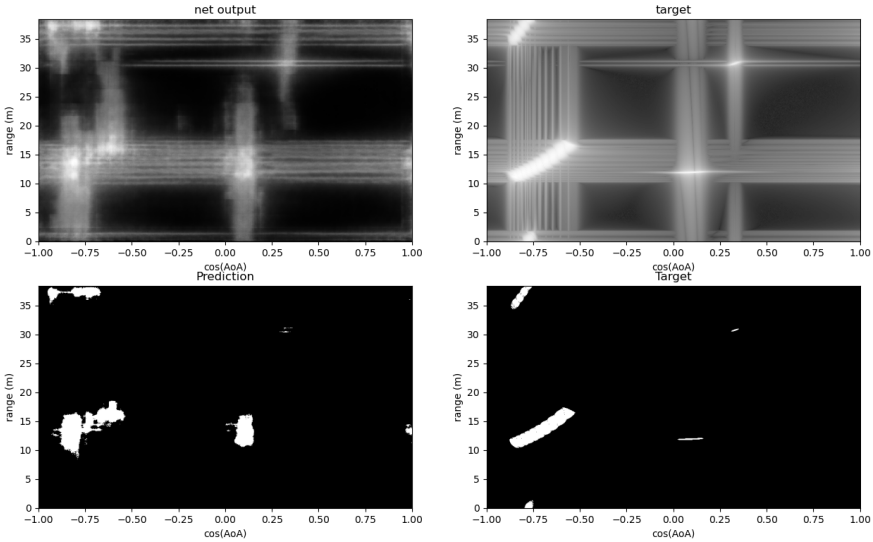

target
net input 0
net output
net input 1
range (m)
cos(AoA)



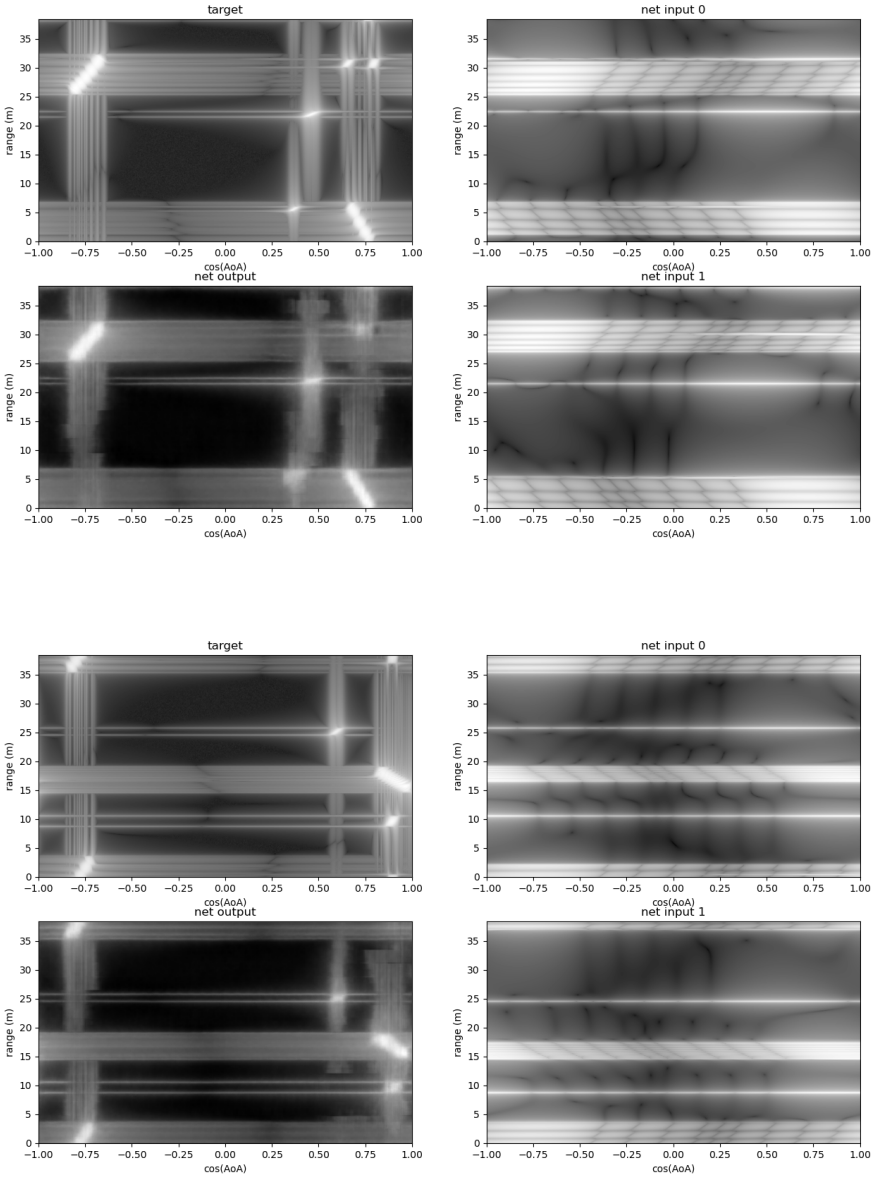
A.1.1 Results from Experiment 1 After Applying Threshold

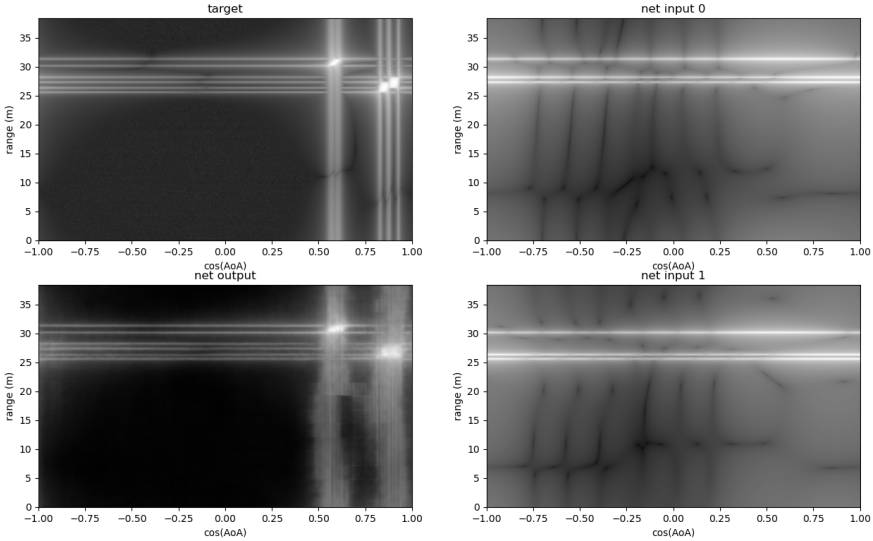


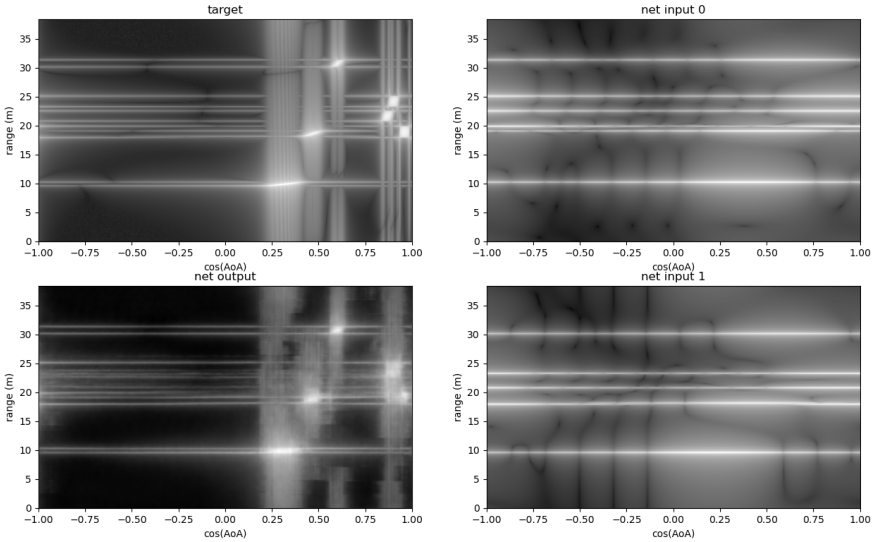

net output
target
Prediction
Target
range (m)
cos(AoA)

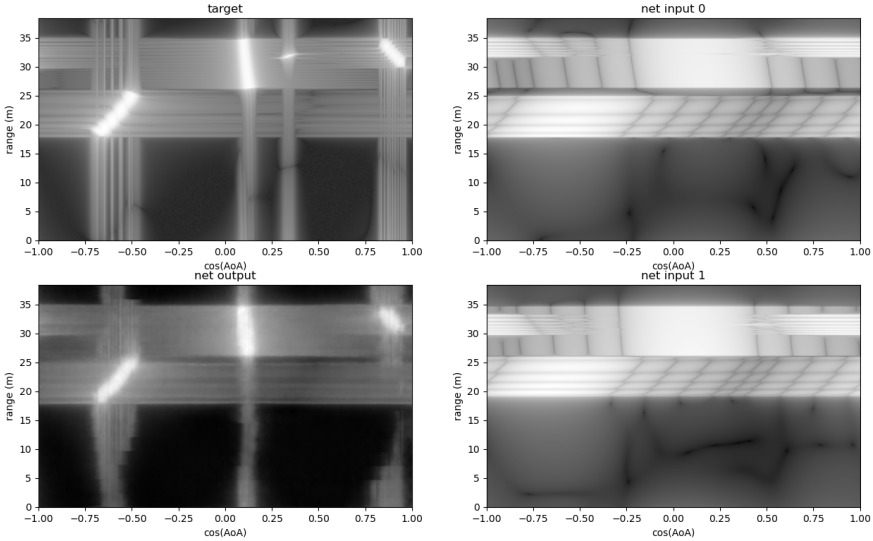

net output
target
Prediction
Target
range (m)
cos(AoA)

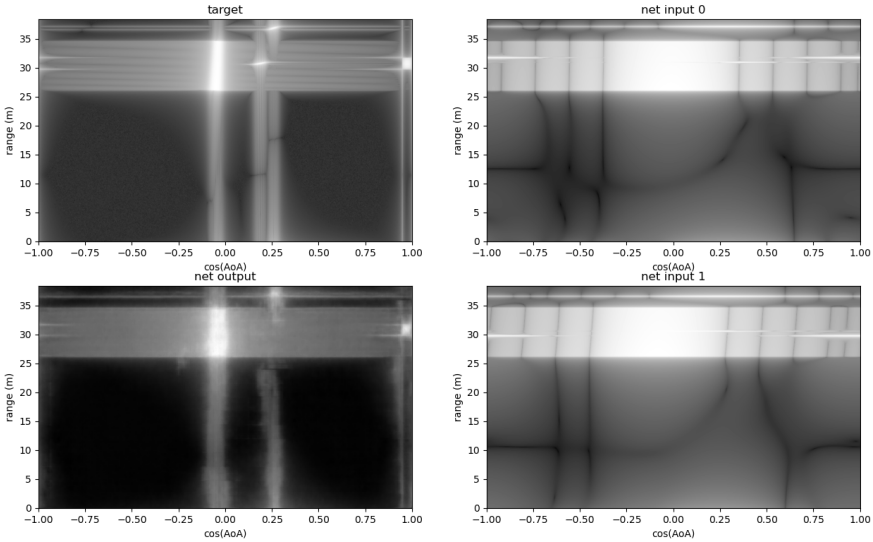
A.2 Additional Results from Experiment 2

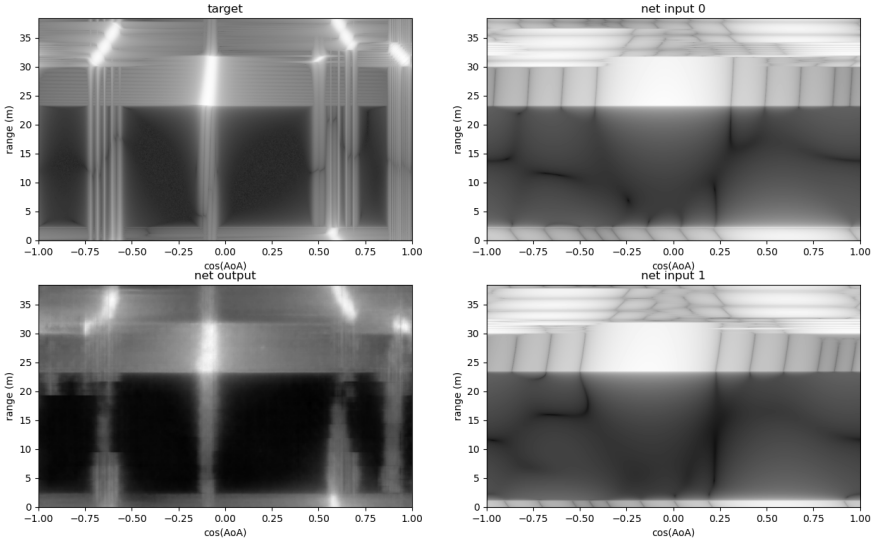


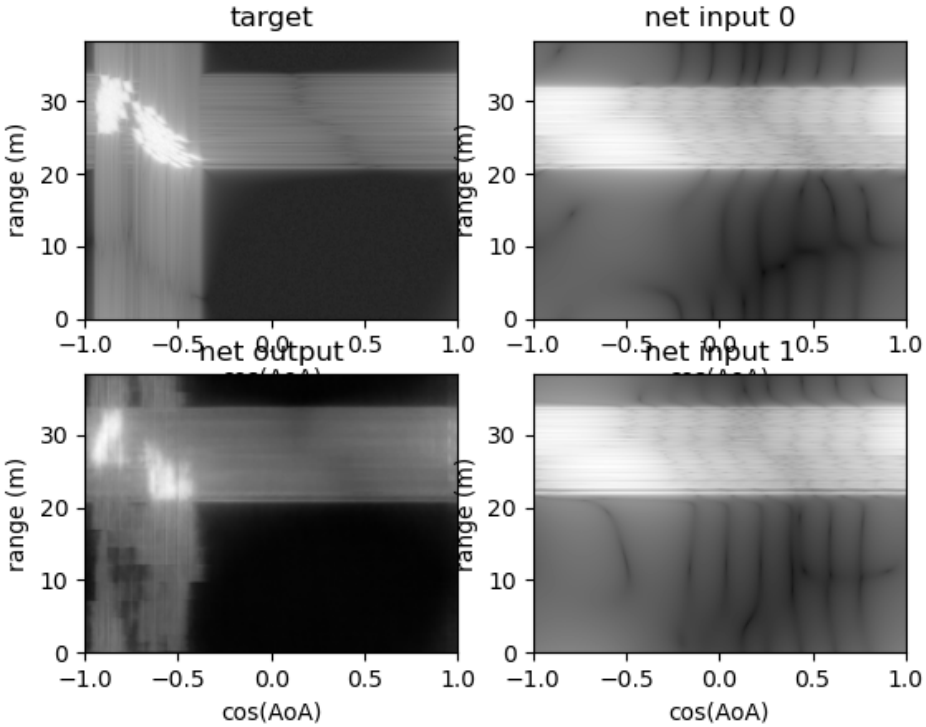

target
net input 0
net output
net input 1
range (m)
cos(AoA)

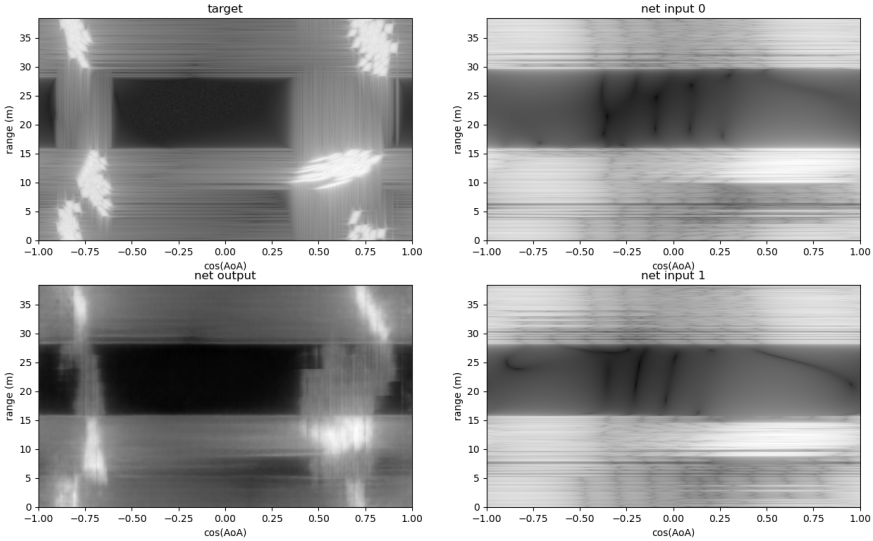

target
net input 0
net output
net input 1
range (m)
cos(AoA)


target
net input 0
net output
net input 1
range (m)
cos(AoA)

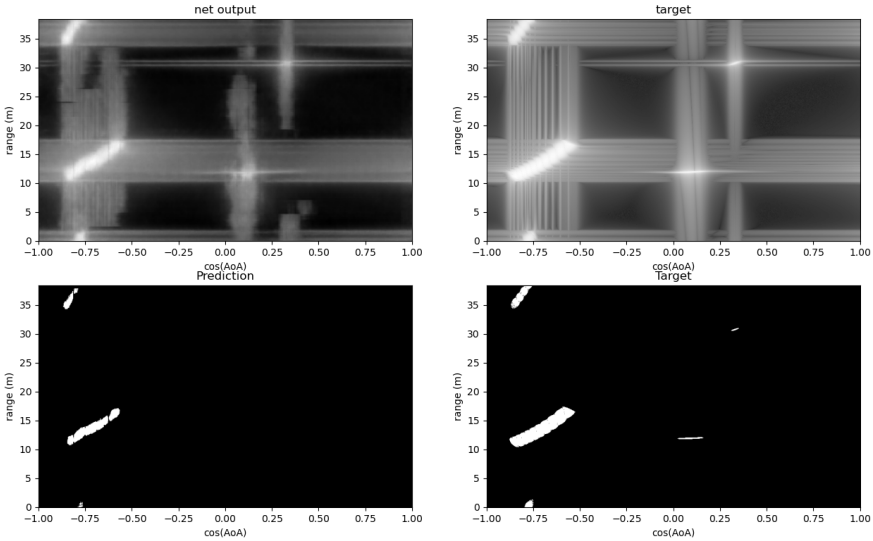

target
net input 0
net output
net input 1
range (m)
cos(AoA)

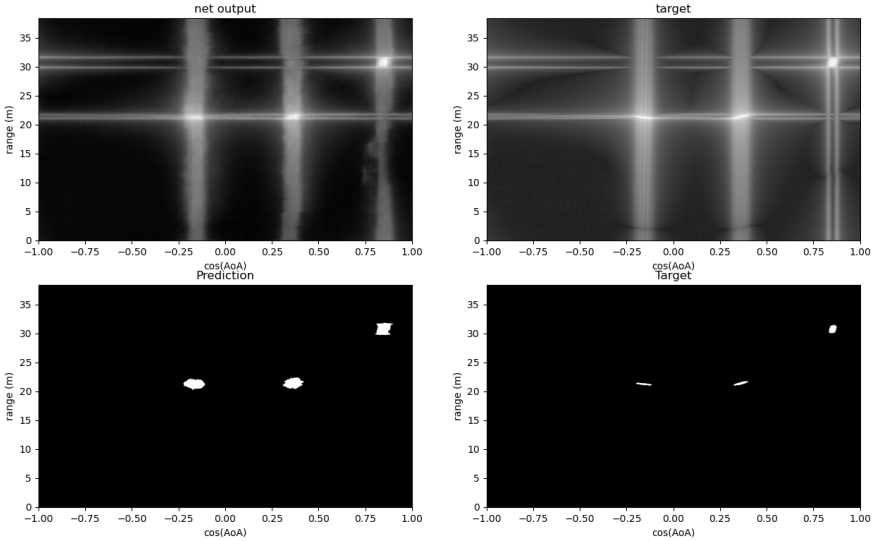

target
net input 0
net output
net input 1
range (m)
cos(AoA)

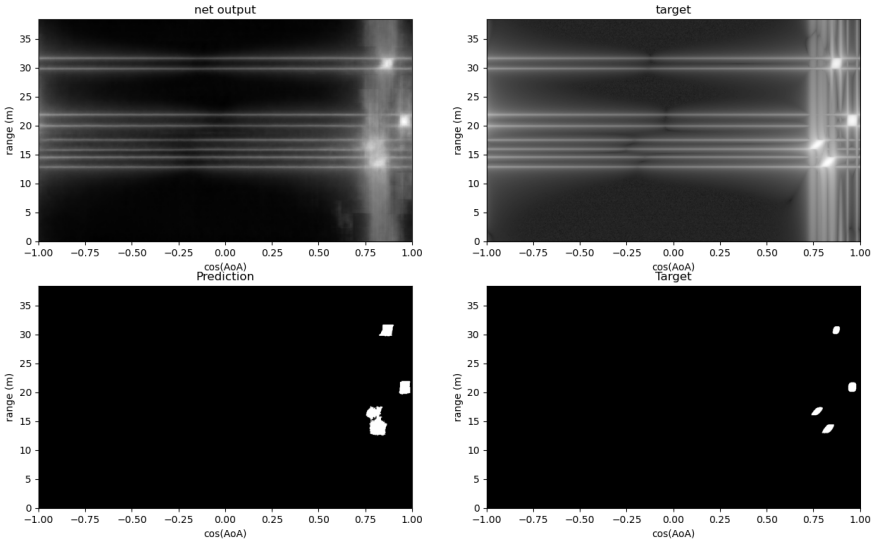

target
net input 0
net output
net input 1
range (m)
cos(AoA)

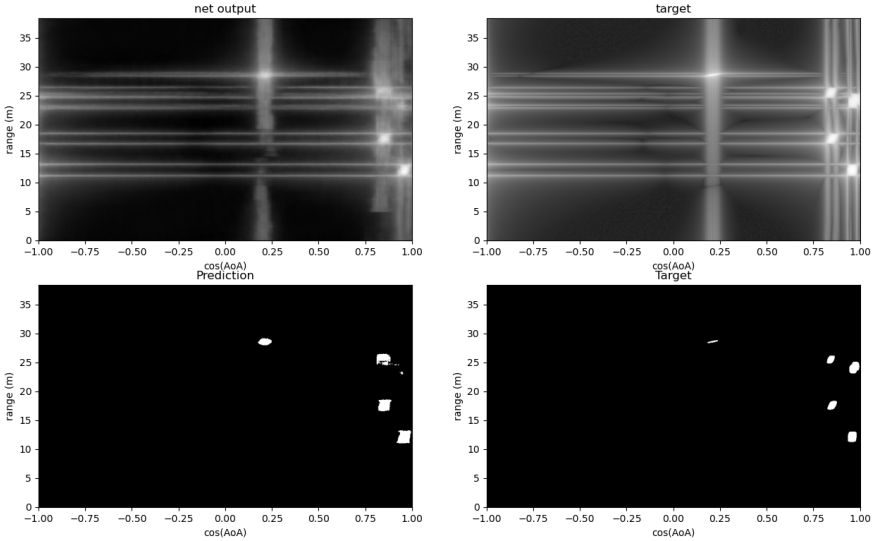


A.2.1 Results from Experiment 2 After Applying Threshold




net output
target
Prediction
Target
range (m)
cos(AoA)


net output
target
Prediction
Target
range (m)
cos(AoA)


net output
target
Prediction
Target
range (m)
cos(AoA)
0
5
10
15
20
25
30
35
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

Appendix B

Relevant Code

The Github repository containing code used in this thesis can be found

https://github.com/JasmineJin/cow_experiment.git

Bibliography

[1] S. Ipek, "LIDAR Perception Challenges." [Online]. Available: https://www.analog.com/en/technical-articles/lidar-perception-challenges.html

[2] J. D. Krieger, Y. Kochman, and G. W. Wornell, "Multi-coset sparse imaging arrays," *IEEE Transactions on Antennas and Propagation*, vol. 62, no. 4, pp. 1701–1715, 2014.

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[4] A. P. Sligar, "Machine learning-based radar perception for autonomous vehicles using full physics simulation," *IEEE Access*, vol. 8, pp. 51 470–51 476, 2020.

[5] A. Yellepeddi, "Seeing Farther with LIDAR Using Tracking." [Online]. Available: https://www.analog.com/en/technical-articles/seeing-farther-with-lidar-using-tracking.html

[6] D. McCarthy, "Radar, the Car's Virtual Eye." [Online]. Available: https://www.analog.com/media/en/technical-documentation/tech-articles/radar-the-cars-virtual-eye.pdf

[7] C. I. Radar, A. Manager, and S. Rao, "The fundamentals of millimeter wave sensors The fundamentals of millimeter wave sensors 2," Tech. Rep., 2017.

[8] L. Zhao, L. Wang, L. Yang, A. M. Zoubir, and G. Bi, "The Race to Improve Radar Imagery: An overview of recent progress in statistical sparsity-based techniques," *IEEE Signal Processing Magazine*, vol. 33, no. 6, pp. 85–102, 11 2016.

[9] G. V. Tsoulos, *Adaptive Antennas for Wireless Communications*. IEEE Press, 2001, ch. 2.

[10] A. Beck and M. Teboulle, "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems." [Online]. Available: http://www.cs.cmu.edu/afs/cs/Web/People/airg/readings/2012_02_21_a_fast_iterative_shrinkage-thresholding.pdf

[11] W. Yang, X. Zhang, Y. Tian, W. Wang, and J.-H. Xue, "Deep Learning for Single Image Super-Resolution: A Brief Review," 8 2018. [Online]. Available: http://arxiv.org/abs/1808.03344http://dx.doi.org/10.1109/TMM.2019.2919431

[12] J. Gao, B. Deng, Y. Qin, H. Wang, and X. Li, "Enhanced Radar Imaging Using a Complex-valued Convolutional Neural Network."

[13] *Arrays and Spatial Filters*. John Wiley & Sons, Ltd, 2002, ch. 2, pp. 17–89. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/0471221104.ch2

[14] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep Image Prior," 11 2017. [Online]. Available: http://arxiv.org/abs/1711.10925http://dx.doi.org/10.1007/s11263-020-01303-4

[15] D. Patel and A. A. Oberai, "Bayesian Inference with Generative Adversarial Network Priors," 7 2019. [Online]. Available: https://arxiv.org/abs/1907.09987

[16] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," 2017.

[17] G. Zhang, F. Wenger, and H. Li, "Object Detection and 3D Estimation via an FMCW Radar Using a Fully Convolutional Network." [Online]. Available: https://www.researchgate.net/publication/331088590