UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

ROBUST VELOCITY DEALIASING FOR WEATHER RADAR BASED ON CONVOLUTIONAL NEURAL NETWORKS

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By

HYERI KIM

Norman, Oklahoma

2023

ROBUST VELOCITY DEALIASING FOR WEATHER RADAR
BASED ON CONVOLUTIONAL NEURAL NETWORKS

A DISSERTATION APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY THE COMMITTEE CONSISTING OF

Dr. Boonleng Cheong, Co-chair

Dr. Robert D. Palmer, Co-Chair

Dr. Alexander Grigo

Dr. Joseph Havlicek

Dr. Mark Yeary

Dedicated to my beloved family, Sookyeom Kim, Hyundeok Kim,
and Yongdeok Kim

Acknowledgments

First of all, I would like to express my gratitude to God for always being with me. I also want to extend my heartfelt thanks to my beloved family - Sookyeom Kim, Hyundeok Kim, and Yongdeok Kim - for their unwavering love and support. Without their presence in my life, I would not have been able to complete this long journey.

I would like to express my profound gratitude to my advisors Dr. Boonleng Cheong, Dr. Robert D. Palmer, and former advisor Dr. Tian-You Yu for their invaluable guidance, efforts, and constant encouragement throughout my graduate studies. Their mentorship has been instrumental in shaping my academic and professional growth, and I am truly grateful for their contributions to my study. I also appreciate my graduate committee Dr. Joseph P. Havlicek, Dr. Mark Yeary, and Dr. Alexander Grigo for spending their time to help with my study.

I would like to express my appreciation to my colleagues and friends at the Advanced Radar Research Center, especially Cesar Salazar Aquino, Ayano Ueki, Jorge Luis Alva Alarcon, Nim Ccoillo Ramos, Precious Jatau, Jennifer Panoplos, Nawaf Almuqati, Min-Duan Tzeng, Rosalind Agasti, Felipe Moncada, and many others. Thank you for your constant support throughout my studies. I also want to express my gratitude to my friends, especially Eunhyeong Jo, Minseop Song, Sungeun Yi, Sungeun Kim, and Jinhee Seo, for being my friends and supporting me throughout my studies. All of my friends' encouragement and assistance have been invaluable, and I am grateful for their presence in my life. I will pray for their lives to be always blessed

and happy. I am also grateful to my church family at Norman Korean Baptist Church, including pastors and friends. Their prayers and encouragement have been a source of strength and inspiration.

As my advisor said, our academic journey would not have been possible without the help and support of many people. I would like to express my sincere thanks to the professors who have taught and guided us, the janitors who have diligently kept our office spaces clean, the security officers who have greeted us with smiles and ensured our safety, the engineers who have collaborated with us, and the secretaries and managers who are in charge of the laboratory's finance, events, and meetings. Their contributions have been invaluable.

It has been a long journey of over 6 years and the greatest change in my life, as I transitioned from a relatively problem-free existence to facing numerous challenges. When I first came to the United States to study, I struggled with understanding and speaking English, which left me feeling discouraged and isolated. I had to fight loneliness for a long time and lost confidence. However, someone I admire told me something that stuck with me: "This time was necessary for your life, so don't blame yourself." Like his message, I now believe that time has been essential for my personal growth and maturity.

As I stood in line to purchase my graduate gown, I felt weird. Even now, I sometimes ask myself whether I truly deserve a Ph.D. There were moments during my studies when I considered giving up because I thought I doubted my ability to earn this degree. However, I was blessed to have a supportive family, as well as the opportunity to meet great people here at the ARRC, including my advisors and friends, who supported and encouraged me to finish my Ph.D. They always said, "I want you to be happy." Thanks to their support, I am confident that I can expect to finish my Ph.D. Initially, I did not understand why the professors always helped and encouraged me,

but I suddenly realized that there must be something I can do for society, and I would like to give back.

I believe that there is much to learn in order to become an expert. As a result, I am fully aware of the importance of continuous learning and personal growth. I am committed to putting in more effort to become a proud daughter and student to my parents and professors, who have always shown me endless patience and encouragement. I aspire to be someone who can share the love that I have received with the world and make a positive impact wherever I go.

Table of Contents

Acknowledgments **v**

Table of Contents **x**

List of Tables **xi**

List of Figures **xx**

Abstract **xxi**

1 Introduction **1**

2 Weather Radar Fundamentals **8**

2.1 Pulsed Doppler Radar 8

2.2 Radar Range Equation 9

2.2.1 Point Target Radar Range Equation 9

2.2.2 Volume Scattering Radar Range Equation 12

2.2.3 Weather Radar Equation 12

2.3 Radar Variables and Estimators 14

2.3.1 Reflectivity Estimator 14

2.3.2 Mean Radial Velocity Estimator 15

2.3.3 Estimation of Aliasing Range and Aliasing Velocity 18

2.4 Block Diagram 20

2.5 Limitations of Pulsed Doppler Radar – Range-Doppler Ambiguities . . 21

2.5.1 Range-Overlaid Echoes . . . 23
2.5.2 Velocity Aliasing . . . 23
2.6 Existing Range-Doppler Ambiguity Mitigation Strategies . . . 26
2.6.1 Waveform Design Methods . . . 26
2.6.2 Post-Processing Methods . . . 27

3 Overview of Convolutional Neural Networks 32
3.1 Introduction to Machine Learning . . . 32
3.1.1 Neural Networks . . . 34
3.1.2 Deep Learning . . . 36
3.2 Machine Learning Basics . . . 36
3.2.1 Hyperparameters . . . 37
3.2.2 Regression . . . 39
3.2.3 Activation Function . . . 41
3.2.4 Back Propagation . . . 46
3.2.5 Cost Function . . . 48
3.2.6 Dataset . . . 50
3.3 Convolutional Neural Networks . . . 50
3.3.1 Layers . . . 50
3.3.2 Receptive Field . . . 53
3.3.3 Class Segmentation . . . 53

4 Application of Convolutional Neural Networks to Velocity Dealiasing 57
4.1 Overview of Proposed Process . . . 57
4.2 Data Generation . . . 58
4.3 Methodology . . . 60
4.3.1 Pre-Processing . . . 61
4.3.2 Algorithm Description . . . 62
4.4 Training . . . 65

4.4.1 Variables of Optimization 65
4.4.2 Sensitivity Tests 68

5 Results and Discussion **73**
5.1 Evaluation Method and Metrics 73
5.2 Experimental Results 75
5.3 Error Analysis 85
5.4 Discussion 87

6 Conclusions and Future Work **90**

References **94**

Appendix Acronyms **105**

Appendix Index **107**

List of Tables

4.1 The number of scans for training, validation, and test dataset for mostly filled precipitation and sparsely filled precipitation 60

5.1 The population ratio of each label (L_1, L_2, and L_3) of three different evaluation groups: G_1, G_2, and G_3. 75

5.2 Comparisons of the total performance metrics between the proposed CNN method and the conventional region-based dealiasing method. Total performance metrics are calculated by multiplying the different v_{a} group performance metrics and the corresponding weights. 77

5.3 Weights of the different v_{a} groups, i.e., G_1, G_2, and G_3 for L_0, L_1, and L_2. The weight for each group is calculated by the inverse of the number of labels for each evaluation metric. 77

List of Figures

2.1 A block diagram example of a simplified X-band weather radar system. 20

2.2 Doppler dilemma depending on frequency such as S-, C-, and X-band radar system. For a fixed T_{s}, the maximum unambiguous velocity (v_{a}) and the maximum unambiguous range (r_{a}) are determined by the radar frequency, which is the inverse of the wavelength. The red line shows the relation for the S-band radar system, the green line represents the C-band radar system, and the blue line is for the X-band radar system. . 22

2.3 Example situation of the range aliasing in a radar system view toward the target. The target is located beyond r_{a} and range aliasing would occur. 24

2.4 Example diagram of the range overlaid echoes in case of the first-trip target returned with the second pulse. The first-trip target is returned with the low-power when it is overlaid. τ_{s} is the pulse width, and the target echo from pulse#1 is received during the second period, which is overlaid with echoes from pulse#2. 24

2.5 An example PPI images of range overlaid echoes extracted from [52]. Panel (a) is the PPI image of radar reflectivity from RBSL radar and panel (b) is from RMYN radar in South Korea. The left panel (a) displays a PPI collected with a shorter PRT, while the right panel (b) shows a PPI collected with a longer PRT. The dashed line indicates area A, which corresponds to the range overlaid echoes, and it is not visible inside the dotted-dashed circle in panel (b). The squall line area B in panel (a) is also shown in panel (b) inside the dotted-dashed circle. Thus, it is noteworthy that area A is where the range-overlaid echoes are present. 25

2.6 The Doppler spectra with v_{a} = 15 $\mathrm{m\,s^{-1}}$. The left panel shows a spectrum of a target with $v_{\mathrm{r}} = 10\ \mathrm{m\,s^{-1}}$. The right panel shows another with $v_{\mathrm{r}} = 16\ \mathrm{m\,s^{-1}}$ and the measured velocity is aliased to -14 $\mathrm{m\,s^{-1}}$. . 26

2.7 This figure compares the three waveform design methods in a diagram. Figure (a) shows the staggered PRT method, (b) represents the dual PRF method and (c) shows the dual scan method. T_1 is the short PRT and T_2 is the long PRT. 28

2.8 Waveform Design Methods. 29

3.1 A Venn diagram illustrating the different classes of artificial intelligence. A CNN represents a particular type of AI, and it falls within the category of DNN, which belongs to the larger category of NN. NN is the subset of the broader category of artificial intelligence. 35

3.2 An example diagram of a simple NN. It consists of an input layer with two inputs, one hidden layer, and an output layer with one output. . . . 35

3.3 A simple diagram of a DNN. It consists of an input layer with two inputs, three hidden layers, and an output layer. 37

3.4 The concept of the batch size and epoch in one iteration training. Each batch size has 32 samples and a total of 50 batches are included to train one epoch for a total of 1,600 samples. 38
3.5 This figure shows an example of linear regression. The data are fitted to the line $y = 0.9932x + 100.0412$. This line is derived from the least-square fitting method. 40
3.6 This figure illustrates the inappropriate learning rate. A high learning rate would lead to divergence, where the model could fail to map the input data to the latent space with large updates. A low learning rate requires many updates to reach the minimum cost, therefore, it makes the training slow. 41
3.7 These graphs illustrate the different activation functions. Panel (a) shows the sigmoid, panel (b) represents the tanh, panel (c) illustrates the ReLU, and panel (d) shows the leaky ReLU with $\alpha = 0.1$. 43
3.8 This diagram illustrates the example model with logistic regression, which takes three inputs. First, the inputs are summed and then the sigmoid function is utilized as an activation function for the logistic regression. 46
3.9 This figure illustrates the feedforward and backpropagation process. The black line from left to right represents the feedforward process, while the red line from right to left represents the backpropagation process. During the backpropagation process, all the weights and biases are updated. 48

3.10 These figures illustrate the importance of finding an appropriate data-fitting curve. The left panel shows an example of underfitting, the middle panel shows an appropriate fitting curve and the right panel demonstrates an example of overfitting. In each panel, the blue dots represent the data, and the red line shows the fitted curve. A curve that is too simple (underfitting) may not accurately capture the data properties, while a curve that is too detailed would be overfitted on data and may not generalize to unseen data. 51
3.11 This figure shows how the convolutional layer works. The left image is an example of the input data assuming a kernel size of 3 $\times$ 3. 3 $\times$ 3 patches are extracted from the input, the pre-determined convolution filter is convolved pixel-wise, and the calculated result goes to the output. In this example, when we perform an element-wise convolution between the patch and the convolution filter, the resulting value is 27. . 52
3.12 This figure illustrates an example of max pooling. A 2 $\times$ 2 kernel size with a stride 2 is used to downsample the input. The maximum value in each kernel is pooled out. In this example, the value of 6 is pooled from the yellow patch of the left panel, 3 from the red patch, 5 from the green patch, and the blue 5 is pooled from the blue patch of the left panel. 52
3.13 This figure illustrates the concept of the receptive field. From layer 1 in the left panel, the green patches become the center pixel in layer 2 (middle panel). In other words, the receptive field of a green pixel in layer 2 is the green patches (3 $\times$ 3) of layer 1. Similarly, the receptive field of the yellow center pixel in layer 3 is the yellow patches of layer 2 (3 $\times$ 3), which corresponds to the entire image (5 $\times$ 5) of layer 1. Therefore, the receptive field of a yellow center pixel in layer 3 is the entire image of layer 1 (5 $\times$ 5). 54

3.14 U-Net model structure example extracted from [31]. The input image is 572 × 572 and is downsampled to 28 × 28 using a combination of convolutional and max pooling layers. Then, it is upsampled to 388 × 388 using the up-convolutional layers and concatenated with the layers from the downsampling part. 55

4.1 Block diagram of the proposed velocity dealiasing technique using a CNN. Velocity dealiasing is performed by combining the input (aliased) velocity (v_{i}), the aliasing count (L_{p}), and the Nyquist velocity v_{a}. v_{i} passes through the model, which consists of multiple layers of operations, i.e., convolution, pooling, softmax, and prediction. To that end, the technique produces a map that indicates whether a velocity measurement is aliased, the sign, and how many times it is aliased. . . . 63

4.2 Results of the process of velocity dealiasing using the labels predicted by the CNN. In this example, the data were collected from the KTLX radar on 2020-03-08 23:48 UTC. The input velocity v_{i} is obtained by aliasing v_{t} using $v_{\mathrm{a}} = 7\ \mathrm{m\,s^{-1}}$ (top) and $v_{\mathrm{a}} = 17\ \mathrm{m\,s^{-1}}$ (bottom). Z is the radar reflectivity, and v_{p} is the dealiased velocity according to Equation (4.4) 64

4.3 This figure is the distribution of labels with different v_{a} in logarithmic scale: Panel (**a**) represents the distribution of labels on $v_{\mathrm{a}} \in [7, 9]$; panel (**b**) shows on $v_{\mathrm{a}} \in [11, 13]$; and panel (**c**) shows on $v_{\mathrm{a}} \in [21, 23]$. When the v_{a} used is higher (left panel to the right panel), the distribution is more skewed to the L_0. 67

4.4 Comparisons of the trained CNN with different v_{a}, i.e., $v_{\mathrm{a}} = 7\ \mathrm{m\,s^{-1}}$ (blue) $v_{\mathrm{a}} = 12\ \mathrm{m\,s^{-1}}$ (green), $v_{\mathrm{a}} \in [7, 12]$ (red), $v_{\mathrm{a}} \in [7, \nu]$ (purple), and $v_{\mathrm{a}} \in [\nu]$ (orange) for each on the mostly filled precipitation (left), and the sparsely filled precipitation (right). μ_A is the scan averaged accuracy (top) and σ_A is the scan averaged standard deviation (bottom). It is evaluated with the three different v_{a} groups G_1, G_2, and G_3. 69

4.5 Comparisons of the performance of the trained CNN with different template sizes (T), i.e., 32 (blue), 64 (green), 128 (red), and 256 (purple) range gates on mostly filled precipitation (left), and sparsely filled precipitation (right). It is tested with three different v_{a} group, i.e., G_1, G_2, and G_3. 70

4.6 Performance of the CNN algorithm as a function of range with the different T, i.e., 32 (orange), 64 (magenta), 128 (green), and 256 (red) range gates. It is also compared to the conventional region-based dealiasing method (blue dashed line). The first row is μ_A in percentage averaged by the number of scans for the mostly filled precipitation. The second row is also the μ_A but for the sparsely filled precipitation scans. It is analyzed with groups G_1 (left), G_2 (center), and G_3 (right). 71

5.1 Comparison results on velocity dealiasing performance between the proposed CNN method (blue) and the conventional region-based dealiasing method (red). Comparison is performed with mean accuracy (μ_A) (top) and the standard deviation (σ_A) (bottom) in percentage. It is analyzed with the three different v_{a} groups, i.e., G_1, G_2, and G_3. The first column is for the mostly filled precipitation, and the second column is for the sparsely filled precipitation. 76

5.2 An example PPI scan with a mostly filled precipitation. Z is the reflectivity, v_{i} is the input velocity, v_{t} is the ground truth, v_{p} is the dealiased velocity using the *predicted* aliased label from the CNN, and v_{c} is the dealiased velocity using the conventional region-based dealiasing method. The data are synthesized using a 1.32° -EL scan from the KTLX on 4 July 2017 05:38 UTC. This example shows the result of processing a velocity field observed at 7 $\mathrm{m\,s^{-1}}$. For most simple cases such as this, both methods are able to produce an accurate dealiased velocity field. This example shows over 99% accuracy from both CNN and region-based methods. 79

5.3 This figure shows an example PPI scan for isolated storms observed at $v_{\mathrm{a}} = 7\,\mathrm{m\,s^{-1}}$. The data are synthesized using a 1.32°-EL scan from the KTLX on 30 April 2017 19:14 UTC. The CNN method successfully dealiased the scan as it processed the entire scan all at once. The region-based method, however, failed at a number of isolated storms, which are indicated in the yellow circle. In this example, CNN method predicts the 99.5% on L_0, 99.4% on L_1, and 100% on L_2, while the region-based method predicts 77.9%, 67.8%, and 84.4% on L_0, L_1, and L_2 for each. 80

5.4 This figure shows an example PPI scan with isolated storms observed at $v_{\mathrm{a}} = 8\,\mathrm{m\,s^{-1}}$. The data are synthesized using a 0.88°-EL scan from the KTLX on 14 Jan 2017 19:09 UTC. The CNN method successfully dealias with the velocity, it is similar to v_{t}. The region-based method, however, failed to dealias the velocity in a white dashed circle. 81

5.5 Velocity dealiasing PPI result observed with PX-1000 at a 2.6° -EL on 20 May 2013 19:34 UTC . Z is the reflectivity, v_{i} is the input velocity, v_{t} is the true velocity which is used as ground truths, v_{p} is the dealiased velocity with proposed CNN method, and v_{c} is the dealiased velocity with traditional region-based dealiasing method. 82
5.6 Velocity dealiasing PPI result observed with PX-1000 at a 2.6° -EL on 20 May 2013 19:50 UTC. 83
5.7 Enlarged velocity dealiasing PPI result observed with PX-1000 at a 2.6° -EL on 20 May 2013 19:50 UTC. 83
5.8 Velocity dealiasing PPI result observed with PX-1000 at a 4° -EL on 05 May 2022 00:09 UTC. 85
5.9 An example of failed prediction with non-speckle echoes: Panel (**a**) shows the true label, which is synthesized using a 0.88°-EL scan from the KTLX on 16 January 2017 06:33 UTC; Panel (**b**) shows the predicted label; Panels (**c**–**h**) represent the probability of each label from the CNN model. One can see that the green patch near azimuths 0–45° at far ranges is incorrectly predicted. The correct label ($L = 0$), however, has a significant probability value, which would result in a correct prediction if selected. 86
5.10 A similar scan to Figure 5.9 but the CNN model succeeded the prediction of aliasing labels (green patch in panel (b) of Figure 5.9). Panel (**a**) shows the true label, which is synthesized using a 1.32°-EL scan from the KTLX on 16 January 2017 06:33 UTC. In panel (**b**), the green patch near azimuths 0–45° at range gates 180–256 from panel (b) of Figure 5.9 is now correctly identified. Panels (**c**–**h**) represent the probability of each label from the CNN model. 86

5.11 This figure shows the replaced result by the second most probable prediction on failed pixels: Panel (**a**) is the true label; panel (**b**) is the raw predicted label, and panel (**c**) is the same as the middle panel but incorrect labels are replaced by ones with the second highest probability. The value of A increased from 88.1% in (**b**) to 99.6% in (**c**). 87

Abstract

Doppler weather radar is an essential tool for monitoring and warning of hazardous weather phenomena. In weather radar, achieving a longer aliasing range (r_{a}) is crucial for surveillance, and a higher aliasing velocity (v_{a}) is also important to obtain dynamical information of storms unambiguously. However, the desire for longer r_{a} and higher v_{a} creates a conflict because these two parameters are inversely related to the pulse repetition time (PRT). This conflict is known as "Doppler dilemma", as r_{a} and v_{a} cannot be improved simultaneously using a single PRT. This phenomena is more challenging at shorter wavelengths, which means it has a more significant impact on X-band, followed by C-band and S-band.

There are two main approaches to mitigating this issue. The first approach to dealias the velocity is the post-processing method. This method checks for abrupt changes from one end of the v_{a} to another, and a fold is detected when such instances are encountered. The underlying assumption is that the velocity field should be spatially continuous. This approach performs well for wide and spatially continuous storms. However, it still suffers when the storms are isolated within the radar field of view. The second approach is the waveform design method, which utilizes two or more pulse repetition times (PRTs), and the aliased velocities are found by searching for disagreement between two or more velocities observed from different PRTs. Velocity dealiasing is performed by solving a least-common-multiple problem. However, this method still has the inherent limitation of r_{a}. The post-processing method allows the system to operate everything else,

such as ground clutter filter, continuous pulse-pair processing, etc., as waveform design methods require modifications to the existing filters. Therefore, in this study, the main focus will be on the post-processing method, and the key is to detect the aliased velocity accurately, leading to the correct velocity dealiasing.

The detection of aliased velocity can be compared to classification. Raw aliased velocity can be regarded as the input image, and the aliased count can be regarded as label. With advancements in technology, machine learning can be applied to image classification. Convolutional neural networks (CNNs) are widely used for image segmentation, enabling the model to output the same size as the input image. Therefore, in this study, a CNN is utilized to tackle the velocity dealiasing issue. In the training process, the input data comprises aliased velocity and the aliased count (the sign and how many times they are aliased). The best weights and the biases are determined through a fit-and-adjust process. After the training process, the performance is evaluated using unseen test data. The aliased velocity is used as input, and the output is the aliasing count. Velocity dealiasing is performed by combining the input (aliased) velocity, the aliasing count, and the known v_{a}.

For evaluation, the CNN method is compared to the traditional region-based method, which is also a post-processing method in Python ARM Radar Toolkit (Py-ART). Both methods are evaluated on mostly filled precipitation and sparsely filled precipitation. Sensitivity tests are conducted on template size and the v_{a} used to optimize the CNN model to cover the X-band range coverage. This model can be used regardless of v_{a}. Both methods demonstrate similar performance on mostly filled precipitation. However, the CNN method shows better performance on sparsely filled precipitation, as it processes the entire scan at once while the region-based method only processes the limited adjacent area.

The overarching goal of this study is to exploit CNN for velocity dealiasing and to

achieve human-level performance. Through this process, it is expected that the labor-intensive work could be automated.

Chapter 1

Introduction

Radar is an indispensable tool that can be widely used to detect the location of targets and their speed regardless of day or night and can operate under various weather conditions. Radars transmit electromagnetic waves to the targets and receive backscattered signals. A particular design of radar called pulsed Doppler radar can measure the range of a target using pulse-timing techniques, which measures the time between the transmitted and the returned signals from the targets. These radars also estimate the velocity of the targets by measuring the phased change from pulse to pulse [1]. Doppler weather radar is one specific application of pulsed Doppler radar, which is used for monitoring and warning of hazardous weather phenomena [2].

Although an important tool, some limitations exist in weather radar systems, such as atmospheric attenuation, limited range resolution due to bandwidth, temporal resolution, observation range, sensitivity, etc. Attenuation by precipitation is a more severe issue when the radar wavelength is shorter; in other words, an X-band radar system is more attenuated, followed by C-band, and S-band is slightly impacted by precipitation [3]. Range resolution is also limited by the wavelength and bandwidth, the angular resolution is determined by the antenna size, and the temporal resolution is also limited when for a dish-based system, but this limitation can be mitigated by a phased array radar (PAR) [4]. Observation range is also limited by system transmitter power and the

radar sensitivity depends on factors such as size and the shape of antenna, operating frequency, transmitted power, pulse width, etc. [1, 2, 5].

The desire for a large maximum unambiguous range (r_{a}) and a large aliasing velocity (v_{a}, which is also known as Nyquist velocity), is in conflict since these two parameters are oppositely proportional to the pulse repetition time (PRT). This so-called "Doppler dilemma" becomes a more severe issue for shorter-wavelength radars, which means it has more impact on X-band, followed by C-band, and S-band [1, 2, 6].

Next Generation Weather Radar (NEXRAD) is a network of radars consisting of 160 high-resolution S-band Doppler weather radars, including Puerto Rico and the U.S. Virgin Islands are installed and jointly operated by National Weather Service (NWS), the Federal Aviation Administration (FAA), and the U.S. Air Force. In 1988, NEXRAD established the WSR-88D radar systems. WSR stands for Weather Surveillance Radar, 88D refers to 1988 when the first WSR-88D radar system was deployed, and D represents the Doppler capability. S-band was chosen for the WSR-88Ds because it has less impact of attenuation by precipitation and relatively high v_{a} compared to X- and C-band radar systems for a constant r_{a}. However, S-band is relatively expensive due to the large size of the antenna needed to achieve the desired angular resolution. Therefore, smaller radar systems, such as the X- and C-band radar systems, are also employed. S- and C-band radar systems are commonly used for operational purposes around the world, and X-band radar systems are mostly employed for research purposes. X-band systems, such as the PX-1000, are widely used for gap-filling purposes, when there is beam blockage by ground clutter such as in mountainous areas. X-band radar systems are also utilized for field work where a mobile system is necessary for storm/tornado capture [7–9].

The WSR-88D employs a number of Volume Coverage Patterns (VCP) modes for operational purposes, such as clear-air and precipitation modes. Clear-air mode is em-

ployed when there is no precipitation within the radar observation range. It slowly rotates the antenna where the increased dwell time improves the radar sensitivity over the precipitation mode. There are some examples of clear-air modes, which are VCP 31 and 32. These are operated with the same elevation angles. VCP 31 uses a long pulse while VCP 32 employs a short pulse. When precipitation occurs, it does not have to be as sensitive as clear-air mode since the rain has reasonable backscattered signals. There are multiple precipitation VCP modes, such as VCP 12, VCP 212, VCP 215, and so on. VCP 12 has 14 elevation angles and can give dense vertical sampling at lower elevation angles with a reasonable update time, suitable for severe weather. Similar to VCP 12, VCP 212 has 14 elevation angles, but it is suitable for more distant severe weather since it has a longer dwell time compared to VCP 12. VCP 215 is the general surveillance VCP mode with 15 elevation angles and updates every six minutes. For tropical systems, VCP 121 is utilized with nine scans, including multiple lower-elevation angles with different pulse repetition frequency (PRF) values, which is called the split-cut method. In this method, one scan is performed using a long PRT for range estimation, and another scan is performed with a short PRT to enhance velocity estimates [10, 11].

From the backscattered weather signals, spectral moment estimates can be calculated. The three most important spectral moment estimates are signal power, mean Doppler velocity (v_{r}), and spectrum width (σ_{v}). Reflectivity is derived from the signal power, and it is important since the rain rate can be derived from the reflectivity. v_{r} is the air motion toward or away from the radar, and is calculated from the time rate of change of signal phase. The σ_{v} is the measure of the velocity dispersion with the resolution volume of the radar [2, 3, 8]. The phase is directly related to the signal wavelength. If the wavelength is shorter, the phase is easy to be wrapped ($\pm\pi$) compared to a longer wavelength. Therefore, X-band, which has a shorter wavelength, is more challenging in regards to velocity aliasing than S-band radar system, which has a longer

wavelength.

Many algorithms have been devised to mitigate the Doppler dilemma, and are generally grouped into techniques called "Velocity Dealiasing." One approach is based on waveform design where two or more different PRTs are used. Another approach is a post-processing method based on checking the spatial continuity along the azimuth or range direction.

Methods based on waveform design estimate Doppler velocity from different PRTs, and aliasing can be found by searching for any disagreement between the two estimates. Velocity dealiasing is accomplished using their difference by solving a least-common-multiplier (LCM) problem [12]. This family of algorithms includes the dual-PRF, staggered PRT, and the dual scan (which is known as split cut). In dual PRF, the transmitter alternates two "batches" of PRTs [2, 13]. Staggered PRT is the method where the transmitter alternates the two different PRTs every pulse [14–20]. The dual-scan method collects one scan with a short PRT and the other with a long PRT. It should be noted that waveform design methods are limited by the LCM and cannot be extended over the LCM.

The second approach is based on post-processing, which allows use of traditional signal processing algorithms (e.g., clutter filtering, pulse-pair processing, etc.). Waveform-design approaches require the modification of the existing clutter filters [19]. Post-processing methods are performed after moment data estimation, and is based on checks for spatial discontinuity with $2v_{\mathrm{a}}$ along the radial or range to decide whether velocity aliasing has occurred. Aliasing detection is the key to this approach. Once the aliased gate is successfully detected, it can typically be dealiased correctly [21–27].

One popular radar processing software library, Python ARM Radar Toolkit (Py-ART), incorporates a novel region-based velocity dealiasing algorithm [26]. The key assumption of this method is that the "first-guess" field is non-aliased. Therefore, it has

the option to use data from environmental winds (e.g., radiosonde data) to decide the aliasing of the first-guess field. Once the first-guess field is determined, it checks the adjacent radar cells for abrupt velocity changes within a storm cluster and, if aliased, dealiases the velocity by adding $2nv_{\mathrm{a}}$ to the measurement. If a radar scan has multiple isolated storms, it produces multiple first-guess fields and processes each storm individually. If the assumption of the first-guess field is incorrect, it leads to incorrect velocity dealiasing of the connected storm.

Despite numerous studies have been conducted to mitigate the Doppler dilemma, it remains a challenge for the community. For example, the waveform-design methods are limited by their v_{a} and the post-processing method faces the issue of first-guess field estimation. This challenge is especially true in the case of complicated wind fields, such as tornado signal detection, or when the storm is isolated and far away from the radar. Consequently, human intervention is often necessary, which can be cumbersome and time-consuming [28–30].

Detection of aliased velocity can be regarded as an image classification problem. With the help of technology development, the machine learning (ML) technique is widely employed to assist the classification problem. One can see how an ML algorithm can be applied to mitigate velocity aliasing. In principle, an ML algorithm allows the trained model to determine the velocity aliased count, the number of folds, and the direction. Additionally, through the semantic segmentation method, ML also allows the identification of the range gates where velocity dealiasing will be required. Therefore, it is expected that ML would replace what human intervention provides, i.e., identification of the velocity aliasing region and classification of the velocity aliasing count. ML can perform these two tasks in one pass, much like what a human is capable of. ML is a subset of AI (Artificial Intelligence), which is a rule-based system. ML is the data-fitting method, which is able to learn via iterative fit-and-adjust training. In ML,

model parameters (i.e., weights and biases) are optimized through the iterative training process to minimize a cost function. Each couple of weight and bias is a neuron, and multiple neurons form a neural network (NN) (1 layer), and more than one layer makes the model train the more complicated features, which is called deep neural networks (DNN). Deep learning is the training process of DNN. A single-layer NN is similar to the current technique (one threshold); however, the DNN is more complex, with the promise of better performance. Convolutional Neural Networks (CNN) is a type of deep learning methods to handle two or more dimensional data, and it is suitable for image classification. U-Net is one of the CNN architecture type, which is proposed by [31]. It is an U-shaped architecture with successive layers so called "fully convolutional network." It consists of encoding (convolution and pooling) and decoding (upsampling and de-convolution) part to provide the same output size as the input size. Using the U-Net, it can segment the image and gives the output at each pixel [31–35].

CNN is the concatenated layers of filters that operate like the convolution operator, and is widely applied in image processing since it has strength in classification by extracting common features. CNN-based image classification can produce a single label that represents the whole image, e.g., facial recognition [36–38], or an output image that indicates multiple labels (segments) within an image, e.g., medical diagnosis [39], object recognition, speech recognition [40, 41], and so on.

CNNs are also applied in meteorological data processing. For example, classify spatially localized climate patterns from Community Atmospheric Model v5 (CAM5) simulation [42], detection of cold and warm fronts from reanalysis data [43]. It is also applied in classifying the tropical cyclone intensity from satellite images [44], prediction of the probability of severe hail [45], detecting the bird roosts from combined radar products [46], and so on.

Like these studies, CNNs can be applied to the Doppler dilemma issue [47]. Detec-

tion of aliased velocity can be transferred to the image classification or segmentation problem. In the human-level, if we know the aliased number and the direction of the aliased velocity, it can be dealiased with this information and v_{a}. In the same way, once the training is completed and the aliased velocity is induced to the trained CNN model, it would give the aliased number and direction as an output label. The promise of using a CNN is to achieve human-level performance. Through this process, it is expected that the labor-intensive task of velocity aliasing could be automated.

This dissertation is organized as follows. Chapter 2 describes weather radar fundamentals, followed by Chapter 3 with an overview of convolutional neural networks. Chapter 4 provides the application of Convolutional Neural Networks to Velocity Dealiasing, and Chapter 5 describes the results and discussion of the proposed algorithm. Finally, Chapter 6 concludes and describes thoughts on future work.

Chapter 2

Weather Radar Fundamentals

In this chapter, weather radar fundamentals are explained. First, the concept of pulsed Doppler radar is described. The radar range equation for the point target is derived and expanded to the volumetric targets and the weather radar. Radar variables such as reflectivity, and mean radial velocity will be derived. The range and velocity aliasing concept, which is the "Doppler dilemma" is introduced and two main approaches to mitigate this issue will be discussed.

2.1 Pulsed Doppler Radar

Pulsed Doppler radars measure the range and the velocity. It determines the range of a target by pulse-timing techniques and measures the target's radial velocity by using the Doppler effect of the returned signal. Doppler effect is the change in frequency of a wave in a relative location to an observer with moving toward (-) or forward (+) from the wave source. The basic idea of "Doppler radar" is to compare the frequency change of the radar signal from a moving target to the frequency of the original signal. A continuous wave (CW) should generate the transmitted and received signals differently to distinguish them since it continuously transmits and receives. However, for the pulsed waveform, with a pulse-timing technique, it transmits the pulse and pauses the time to

receive the returned signal from the target.

In this chapter, it would describe the weather radar fundamentals including the radar range equation for the point target and the volumetric scattering, and it is applied to derive the weather radar equation. Range and velocity estimation is also explained, it is expanded to the range and velocity ambiguity, and the general attempts to mitigate the problem are also described.

2.2 Radar Range Equation

In this section, the radar range equation is derived for point targets and the volumetric scatter, and it is expanded to the weather targets, such as rain echoes which are a type of volumetric scatters.

2.2.1 Point Target Radar Range Equation

The point target radar range equation is derived from the power spectral density for the point target. When a pulse of energy is emitted from a radar, it is dispersed to the surface of targets. For a spherical target with radius r, the surface of a sphere is $4\pi r^2$; therefore, the power density can be derived by dividing the transmitted power (P_t) by the surface of a sphere. The radius r of the sphere is the range of the target from the radar.

$$S_{\mathrm{t}}(r) = \frac{P_{\mathrm{t}}}{4\pi r^2} \tag{2.1}$$

Antenna gain (G) is the radiation intensity of the antenna in a specific direction over

the radiation intensity of the antenna of an isotropic source and it is as shown:

$$G = \frac{4\pi A}{\lambda^2}. \tag{2.2}$$

Therefore, the power spectral density from the directive antenna can be calculated with the power spectral density from the isotropic antenna multiplied by antenna gain.

$$S_{\mathrm{t}}(r) = \frac{P_{\mathrm{t}} G}{4\pi r^2} \tag{2.3}$$

The radiated power density of the reflected signal at the radar is shown in Equation (2.4). From Equation (2.3), the radar cross section (RCS), represented by σ (m^2), is multiplied to Equation (2.3) and divided by the received area, which is also the surface of a sphere.

$$S(r_{\mathrm{t}}, r_{\mathrm{r}}) = \frac{P_{\mathrm{t}} G}{4\pi r_{\mathrm{t}}^2} \left(\frac{\sigma}{4\pi r_{\mathrm{r}}^2} \right) \tag{2.4}$$

Received power (P_{r}) can be derived by Equation (2.5). It is the power density at the radar, which is in Equation (2.4), multiplied by the effective area of the receiving antenna (A_e). P_r is shown below:

$$P_{\mathrm{r}} = \frac{P_{\mathrm{t}} G}{4\pi r_{\mathrm{t}}^2} \left(\frac{\sigma A_{\mathrm{e}}}{4\pi r_{\mathrm{r}}^2} \right), \tag{2.5}$$

where the A_e can be expressed in terms of the antenna gain and the wavelength λ and it is shown in Equation (2.6).

$$A_{\mathrm{e}} = \frac{G\lambda^2}{4\pi} \tag{2.6}$$

This equation is substituted into Equation (2.5) to obtain Equation (2.7) and it is the radar range equation for the point target.

$$P_{\mathrm{r}} = \frac{P_{\mathrm{t}} G^2 \lambda^2 \sigma}{64\pi^3 r^4} \tag{2.7}$$

Once we consider the loss (L), received power P_{r} is divided by L since the P_{r} is attenuated by L and it is in linear scale. The loss-considered received power can be derived as follows:

$$P_{\mathrm{r}} = \frac{P_{\mathrm{t}} G^2 \lambda^2 \sigma}{64\pi^3 r^4 L}. \tag{2.8}$$

Additionally, signal-to-noise (SNR), which is the measure to detect a given target at a given range can be expressed by considering the P_{r} and the noise (N). For NEXRAD, a target is considered to be present when the SNR is higher than 3 dB. Noise can be calculated in Equation (2.9) and the SNR can be expressed as Equation (2.10).

$$N = k B_{\mathrm{n}} T_{\mathrm{s}}, \tag{2.9}$$

where k is the Boltzmann's constant, which is ($1.38 \times 10^{-23}\ J \cdot K^{-1}$), T_{s} is the system noise temperature, and B_n is the noise bandwidth of the receiver.

$$\frac{S}{N} = \frac{P_{\mathrm{r}}}{N} \tag{2.10}$$

Thus, Equation (2.11) can be obtained by substituting P_{r} to Equation (2.7) and N to Equation (2.9).

$$\frac{S}{N} = \left(\frac{P_{\mathrm{t}} G^2 \lambda^2 \sigma}{64\pi^3 r^4 L}\right)\left(\frac{1}{k B_{\mathrm{n}} T_{\mathrm{s}}}\right) = \frac{P_{\mathrm{t}} G^2 \lambda^2 \sigma}{64\pi^3 r^4 k T_{\mathrm{s}} B_{\mathrm{n}} L} \tag{2.11}$$

2.2.2 Volume Scattering Radar Range Equation

In this section, the radar range equation is extended to the volumetric target. For the spherical targets, the total backscattering cross-sectional area of targets within the radar sample volume σ can be expressed in Equation (2.12).

$$\sigma = V\Sigma\sigma_i, \tag{2.12}$$

where the sample volume (V) is defined in Equation (2.13) [48].

$$V = \pi\frac{r\theta}{2}\frac{r\phi}{2}\frac{h}{2}, \tag{2.13}$$

where ϕ is the vertical beam width of antenna pattern, θ is the horizontal beam width, and h is the height.

The volume of a radar pulse by Probert-Jones [49] using a Gaussian shape for beam pattern is the following:

$$V = \frac{\pi r^2\theta\phi h}{16\ln(2)}. \tag{2.14}$$

In Equation (2.7), σ is replaced to Equation (2.12) using Equation (2.14) and result in following Equation (2.15):

$$P_{\mathrm{r}} = \frac{P_{\mathrm{t}}G^2\lambda^2\theta\phi h\Sigma\sigma_i}{1024\ln(2)\pi^2\mathrm{r}^2}. \tag{2.15}$$

2.2.3 Weather Radar Equation

If a sphere is small compared to the wavelength of the radar, the scattering behavior of the target is considered to be in the Rayleigh regime and the RCS can be defined as Equation (2.16) [50].

$$\sigma_i = \frac{\pi^5 |K|^2 \sum_{i=1}^{n/vol} D_i^6}{\lambda^4}, \tag{2.16}$$

where D is the rain drop diameter and $|K|^2$ is defined as follows:

$$|K|^2 = \left| \frac{\epsilon - 1}{\epsilon + 2} \right|^2, \tag{2.17}$$

where ϵ is the permittivity, and K is the dielectric constant, which is often expressed as $|K|^2$ since it is typically used for power-related quantities. For water, the value of $|K|^2$ is 0.93; whereas for ice, it is 0.2.

Since weather radar systems primarily target rain echoes, which are volumetric in nature, a radar cell is considered to be a volumetric target. The key assumption is that the entire radar cell is uniformly filled with rain. A new expression of σ can be derived by replacing V and σ_i in Equation (2.12) with Equations (2.14) and (2.16), respectively.

$$\sigma = V \sum_{i=1}^{n/vol} \sigma_i = \left(\frac{\pi r^2 \theta \phi h}{16 \ln(2)} \right) \left(\frac{\pi^5 |K|^2 \sum_{i=1}^{n/vol} D_{\mathrm{i}}^6}{\lambda^4} \right) \tag{2.18}$$

A weather-specific radar equation, referred to as the weather radar equation, can be derived by replacing σ in Equation (2.7) with Equation (2.18). The weather radar equation is expressed in Equation (2.19):

$$P_{\mathrm{r}} = \left(\frac{P_{\mathrm{t}} G^2 \lambda^2}{64 \pi^3 r^4} \right) \sigma = \frac{P_{\mathrm{t}} G^2 \pi^3 \theta \phi h |K|^2 \Sigma D_{\mathrm{i}}^6}{1024 \ln(2) \lambda^2 \mathrm{r}^2}. \tag{2.19}$$

2.3 Radar Variables and Estimators

2.3.1 Reflectivity Estimator

Reflectivity (η) is the total backscattering cross-sectional area per unit volume. In general, for the distributed target, η is defined as:

$$\eta = \int_0^\infty \sigma_{\mathrm{b}}(D) N(D)\, dD, \tag{2.20}$$

where D is the diameter of a water sphere, $\sigma_{\mathrm{b}}(D)$ is the expected backscattering cross section for a hydrometeor of D, and $N(D)$ is the particle size distribution, which is the number density in the resolution volume between D and $D + dD$.

For the spherical water drops that are small compared to the wavelength by assuming the Rayleigh distribution, the backscattering cross-section can be approximated as shown:

$$\sigma_{\mathrm{b}}(D) \approx \frac{\pi^5}{\lambda^4} |K_{\mathrm{w}}|^2 D^6, \tag{2.21}$$

where K_{w} is the dielectric constant of water. Using Equation (2.21), $\sigma_b(D)$ is substituted into Equation (2.20) to obtain:

$$\eta = \frac{\pi^5}{\lambda^4} |K_{\mathrm{w}}|^2 \int_0^\infty D^6 N(D)\, dD. \tag{2.22}$$

From the Equation (2.22), the integral part is the linear radar reflectivity factor (z) in $\mathrm{mm}^6\mathrm{m}^{-3}$, which is shown below:

$$z = \int_0^\infty D^6 N(D)\, dD. \tag{2.23}$$

z can be converted to logarithmic scale relative to 1 $(\mathrm{mm}^6\mathrm{m}^{-3})$ as shown in Equa-

tion (2.24). Z is the logarithmic reflectivity factor measured in dBZ.

$$Z\,(\mathrm{dBZ}) = 10\log_{10}\left(\frac{z}{1\,\mathrm{mm}^6\,\mathrm{m}^{-3}}\right) \tag{2.24}$$

2.3.2 Mean Radial Velocity Estimator

Mean radial velocity is derived from the echo voltage equation, which is Equation (2.25) as below:

$$V(t,r) = |A|\exp\left[j2\pi ft - j\left(\frac{4\pi}{\lambda}r + \frac{4\pi}{\lambda}v_{\mathrm{r}}t\right) + j\psi_{\mathrm{t}} + j\psi_{\mathrm{s}}\right], \tag{2.25}$$

where A is the amplitude, $\frac{4\pi}{\lambda}r$ is the two-way path, $\frac{4\pi}{\lambda}v_{\mathrm{r}}t$ is the mean radial velocity, also known as the Doppler velocity, ψ_{t} is the transmitting phase, and ψ_{s} is the scattering phase.

This voltage model (Equation (2.25)) can be simplified to Equation (2.26) and it is described below:

$$V(kT_{\mathrm{s}}) = s_k\exp\left[-j\left(\frac{4\pi}{\lambda}\right)v_{\mathrm{r}}kT_{\mathrm{s}}\right], \tag{2.26}$$

where kT_{s} corresponds to t in Equation (2.25), k is the sample index, which is 0, 1, ..., $M-1$. T_{s} is the pulse repetition time, s_k is the total constant of the k^{th} sample. v_{r} can be expressed in terms of f_{d} and it is described as follows:

$$v_{\mathrm{r}} = -\frac{\lambda f_{\mathrm{d}}}{2}. \tag{2.27}$$

Equation (2.27) can be also expressed for Doppler frequency f_{d} and it is shown as follows:

$$f_{\mathrm{d}} = -\frac{2v_{\mathrm{r}}}{\lambda}. \tag{2.28}$$

Since w_{d} is equivalent to $2\pi f_{\mathrm{d}} = -\frac{4\pi v_{\mathrm{r}}}{\lambda}$, Equation (2.26) can be expressed with w_{d} and it is shown as below:

$$V(kT_{\mathrm{s}}) = s_k exp[jw_{\mathrm{d}}kT_{\mathrm{s}}]. \tag{2.29}$$

If white noise, which is from the internal electronic components, atmospheric sources, and so on, is considered, it becomes:

$$V(kT_{\mathrm{s}}) = s_k \exp(jw_{\mathrm{d}}kT_{\mathrm{s}}) + n_k. \tag{2.30}$$

Doppler spectrum of $V(kT_{\mathrm{s}})$ can be obtained by estimating its Auto Correlation Function (ACF). The Doppler velocity can then be estimated from the ACF at lag 1. The ACF is defined as below:

$$\hat{R}(lT_{\mathrm{s}}) = \mathrm{E}[V^*(kT_{\mathrm{s}})V((k+l)T_{\mathrm{s}})] \tag{2.31}$$

$$= \mathrm{E}[(s_k^* e^{jw_{\mathrm{d}}kT_{\mathrm{s}}} + n_k^*)(s_k e^{jw_{\mathrm{d}}(k+l)T_{\mathrm{s}}} + n_{k+l})] \tag{2.32}$$

$$= \mathrm{E}[(s_k^* s_{k+l})e^{jw_{\mathrm{d}}lT_{\mathrm{s}}}] + \mathrm{E}(n_k^* n_{k+l}) + \mathrm{E}(n_k s_{k+l} e^{jw_{\mathrm{d}}(k+l)T_{\mathrm{s}}}) + \mathrm{E}(s_k^* e^{-jw_{\mathrm{d}}kT_{\mathrm{s}}} n_{k+l}), \tag{2.33}$$

where E is the expected value and it means the weighted average. Since the signal (s) and the noise (n) are uncorrelated, $\mathrm{E}[s \cdot n] = 0$; therefore, Equation (2.33) becomes:

$$\hat{R}(lT_{\mathrm{s}}) = \mathrm{E}\left[(s_k^* s_{k+l})e^{jw_{\mathrm{d}}lT_{\mathrm{s}}}\right] + \mathrm{E}(n_k^* n_{k+l}), \tag{2.34}$$

where the first term $\mathrm{E}[(s_k^* s_{k+l})e^{jw_{\mathrm{d}}lT_{\mathrm{s}}}]$ is defined as follows:

$$\begin{cases} \mathrm{E}[s_k^* s_k] = s, \text{for } l = 0 \\ \mathrm{E}[s^* s_{k+l}] = s\rho(lT_{\mathrm{s}}), \text{in general.} \end{cases} \tag{2.35}$$

Therefore, if l is non-zero, then the final form of ACF is shown as follows:

$$\hat{R}(lT_{\mathrm{s}}) = s\rho(lT_{\mathrm{s}})\exp(jw_{\mathrm{d}}lT_{\mathrm{s}}), \tag{2.36}$$

where the $\rho(lT_{\mathrm{s}})$ for the Gaussian spectrum correlation coefficient is given by [2]:

$$\rho(lT_{\mathrm{s}}) = \exp\left[-8\left(\frac{\pi\sigma_v lT_{\mathrm{s}}}{\lambda}\right)^2\right]. \tag{2.37}$$

The ACF estimate can be expressed in a different form by substituting the $\rho(lT_{\mathrm{s}})$ term in Equation (2.36) with Equation (2.37), as the following:

$$\begin{aligned}\hat{R}(lT_{\mathrm{s}}) &= s\exp\left[-8\left(\frac{\pi\sigma_v lT_{\mathrm{s}}}{\lambda}\right)^2\right]\exp(jw_{\mathrm{d}}lT_{\mathrm{s}}) \\ &= s\exp\left[-8\left(\frac{\pi\sigma_v lT_{\mathrm{s}}}{\lambda}\right)^2\right] exp\left[-j\left(\frac{4\pi}{\lambda}\right)v_{\mathrm{r}}lT_{\mathrm{s}}\right].\end{aligned} \tag{2.38}$$

If $l = 1$ and δ_l = 0, then $\hat{R}(lT_{\mathrm{s}})$ becomes Equation (2.39).

$$\hat{R_1}(T_{\mathrm{s}}) = s\exp\left[-8\left(\frac{\pi\sigma_v T_{\mathrm{s}}}{\lambda}\right)^2\right]\exp\left[-j\left(\frac{4\pi}{\lambda}\right)v_{\mathrm{r}}T_{\mathrm{s}}\right] \tag{2.39}$$

Therefore, the argument, which is the phase component of $\hat{R_1}(T_{\mathrm{s}})$, is defined as the following:

$$arg(\hat{R_1}(T_{\mathrm{s}})) = -\frac{4\pi v_{\mathrm{r}}T_{\mathrm{s}}}{\lambda}. \tag{2.40}$$

The estimator for the mean radial velocity at T_{s} can be expressed as follows:

$$\hat{v_{\mathrm{r}}} = -\frac{\lambda}{4\pi T_{\mathrm{s}}}arg(\hat{R_1}(T_{\mathrm{s}})). \tag{2.41}$$

2.3.3 Estimation of Aliasing Range and Aliasing Velocity

For a stationary target, v_{r} is 0. Therefore, Equation (2.25) can be simplified to the following:

$$V(t,r) = |A| \exp\left[j2\pi f\left(t - \frac{2r}{c}\right) + j\psi_{\mathrm{t}} + j\psi_{\mathrm{s}}\right]. \tag{2.42}$$

Equation (2.42) is composed of in-phase and quadrature components of voltage as follows:

$$I(t) = \frac{|A|}{\sqrt{2}} u\left(t - \frac{2r}{c}\right) \cos\left(\frac{4\pi r}{\lambda} - \psi_{\mathrm{t}} - \psi_{\mathrm{s}}\right), \tag{2.43}$$

$$Q(t) = -\frac{|A|}{\sqrt{2}} u\left(t - \frac{2r}{c}\right) \sin\left(\frac{4\pi r}{\lambda} - \psi_{\mathrm{t}} - \psi_{\mathrm{s}}\right), \tag{2.44}$$

where u is the unit step function and it is defined as follows:

$$u(t) = \begin{cases} 1, & t \geq 0 \\ 0, & t < 0, \end{cases} \tag{2.45}$$

where t is the time. For convenience, the phase component of Equations (2.43) and (2.44) is expressed as follows:

$$\psi_e(t) = -\frac{4\pi r}{\lambda} + \psi_{\mathrm{t}} + \psi_{\mathrm{s}}, \tag{2.46}$$

where r changes in time whereas ψ_{t} and ψ_{s} are independent in time. Therefore, the time rate of phase change is the Doppler angular frequency, the unit is in radians per second. Doppler angular frequency is shown in the following equation:

$$\frac{d\psi_e}{dt} = -\frac{4\pi}{\lambda}\frac{dr}{dt} = -\frac{4\pi}{\lambda} v_{\mathrm{r}} = \omega_{\mathrm{d}}, \tag{2.47}$$

where ω_{d} is equal to $2\pi f_{\mathrm{d}}$.

From the "Nyquist Sampling Theorem" [51], sampling frequency $f_{\mathrm{s}} = 1/T_{\mathrm{s}}$ must be twice over the highest frequency in the signal.

$$f_{\mathrm{s}} > 2f_{\mathrm{N}}, \tag{2.48}$$

where f_{N} is the highest frequency in the signal and f_{s} is the inverse of PRT (T_{s}). Therefore, it can be expressed as Equation (2.49):

$$f_{\mathrm{N}} < \frac{f_{\mathrm{s}}}{2} = \frac{1}{2T_{\mathrm{s}}}. \tag{2.49}$$

If f_{d} is higher than f_{N}, the frequency will be aliased. Therefore, f_{d} should be lower than f_{N} to avoid aliasing.

$$|f_{\mathrm{d}}| < |f_{\mathrm{N}}| \tag{2.50}$$

Equation (2.50) can be expressed using Equation (2.28) and Equation (2.49).

$$\left| -\frac{2}{\lambda} v_{\mathrm{r}} \right| < \left| \frac{1}{2T_{\mathrm{s}}} \right|, \tag{2.51}$$

$$|v_r| < \left| \frac{\lambda}{4T_{\mathrm{s}}} \right| \tag{2.52}$$

From Equation (2.52), the aliasing velocity v_{a} is defined as the following:

$$v_{\mathrm{a}} = \pm \frac{\lambda}{4T_{\mathrm{s}}}. \tag{2.53}$$

The aliasing range, which represents the maximum unambiguous range, can be determined by considering the two-way path of the target. Therefore, it can be expressed

as the speed of light (c) times pulse repetition time (T_s) divided by two.

$$r_\mathrm{a} = \frac{cT_\mathrm{s}}{2} \tag{2.54}$$

2.4 Block Diagram

An overview of a simplified weather radar system architecture will be provided in this section. An example of an X-band radar system block diagram is shown in Figure 2.1:

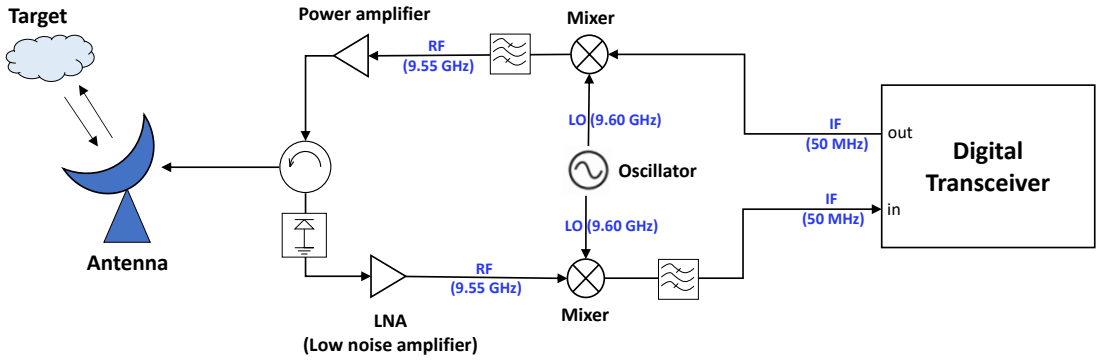


Figure 2.1: A block diagram example of a simplified X-band weather radar system.

In Figure 2.1, the digital transceiver generates a pulse with an intermediate frequency (IF). IF is mixed with a continuous wave from the local oscillator (LO), and the resulting pulses are with two frequencies: $f_\mathrm{LO} + f_\mathrm{IF}$ and $f_\mathrm{LO} - f_\mathrm{IF}$. A bandpass filter (BPF) is used to pass the pulse with $f_\mathrm{LO} - f_\mathrm{IF}$ and BPF removes the pulse with $f_\mathrm{LO} + f_\mathrm{IF}$. The resulting frequency is known as the radio frequency (RF). The RF pulse is then amplified by the power amplifier and routed to the antenna through the circulator. The antenna sends out pulses and receives backscattered signals from the targets. A limiter is added to prevent the receiver from being damaged by strong signals. The transmitted signal is attenuated at receive, so a low noise amplifier (LNA) is used to amplify the signal with minimal additional noise. The received backscattered signals are

mixed with the local oscillator again to return to the pulse with IF for signal processing. The resulting signal is then passed through the BPF, which passes only the pulse with $f_{\mathrm{LO}} - f_{\mathrm{IF}}$. The pulse with IF goes into the digital transceiver, which includes the signal processor. In the signal processor, the received pulses are sampled and integrated to generate moment data, such as reflectivity, velocity, etc. Sampling the received signal is performed at regular intervals of time to determine the target's position and Doppler shift. Before generating the moment data, quality control is performed, including tasks such as speckle removal, clutter filtering, sensoring, attenuation correction, and so on. The processed radar signals are then displayed on a screen.

2.5 Limitations of Pulsed Doppler Radar – Range-Doppler Ambiguities

In Section 2.3.3, r_{a} and v_{a} are defined. However, there is a limitation between the r_{a} and v_{a} since these are in a trade-off relation as illustrated in Equation (2.55). The aliasing range, r_{a}, is directly proportional to the PRT (T_{s}), but the aliasing velocity, v_{a}, is inversely proportional to the T_{s}. This competing relation is known as the *Doppler dilemma*, also referred to as the *range-Doppler ambiguity*.

From Equation (2.53) and Equation (2.54), the multiplication of the variables v_{a} and r_{a} is a constant, as shown in the following equation.

$$v_{\mathrm{a}} r_{\mathrm{a}} = \frac{c\lambda}{8}. \tag{2.55}$$

where λ is the wavelength.

As shown in Equation (2.55), the Doppler dilemma is a more severe issue in shorter wavelengths. That is, the product of v_{a} and r_{a} becomes lower as λ gets shorter. Fig-

ure 2.2 shows the relation between the aliasing velocity, v_{a}, and the aliasing range, r_{a}, for S-, C-, and X-band radar systems. The S-band radar system has a longer wavelength than C- and X-band radar system, followed by C-band, and then the X-band radar system. The red line represents the r_{a} and v_{a} relation of an S-band radar system, the green line corresponds to a C-band radar system, and the blue line corresponds to an X-band radar system. Once we assume the r_{a} as $100\ \mathrm{km}$ (shown as black dashed line), v_{a} of X-band radar system is about $11\ \mathrm{m\,s^{-1}}$ while S-band radar system is approximately $38\ \mathrm{m\,s^{-1}}$. Since the velocity of typical storms is less than $38\ \mathrm{m\,s^{-1}}$, the "Doppler dilemma" is a more severe issue in the X-band radar system.

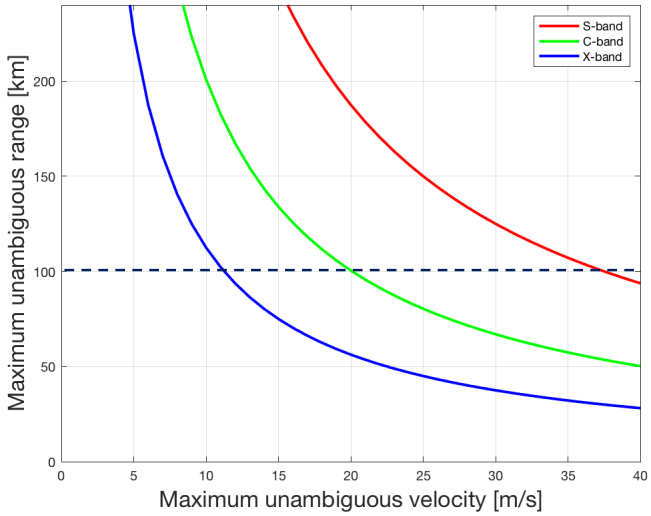


Figure 2.2: Doppler dilemma depending on frequency such as S-, C-, and X-band radar system. For a fixed T_{s}, the maximum unambiguous velocity (v_{a}) and the maximum unambiguous range (r_{a}) are determined by the radar frequency, which is the inverse of the wavelength. The red line shows the relation for the S-band radar system, the green line represents the C-band radar system, and the blue line is for the X-band radar system.

2.5.1 Range-Overlaid Echoes

In this section, range-overlaid echoes will be discussed. When pulses are transmitted and all echoes are within the r_{a}, the unambiguous ranges of the target are as described in Equation (2.54). However, when the target is located beyond the r_{a}, as shown in Figure 2.3, the radar still receives the signals from the target, but the location is ambiguous and overlaid within r_{a} [2, 52]. This phenomenon is illustrated in Figure 2.4 when the first pulse is transmitted, but the target signal is not returned during T_{s} since it is beyond the r_{a}. It is returned along with the returned signals from the second pulse and the true location of the target is in between r_{a} and $2r_{\mathrm{a}}$. This configuration is called "range overlaid echoes" since the target signal is overlapped to the second-pulse echoes even if it is from the first pulse.

An example of range overlaid echoes in Plan Position Indicator (PPI) is shown in Figure 2.5. The left panel shows the result with the short T_{s}, which has a relatively shorter r_{a} compared to the right panel (b). Each circle represents a 50 km range interval. In panel (a), the outer circle (100 km) corresponds to the dotted-dashed circle in panel (b). The squall line area B in panel (a) is also shown in panel (b). However, the range overlaid echoes in area A of panel (a) are not visible in panel (b).

2.5.2 Velocity Aliasing

As explained in Equation (2.55), v_{a} is limited under the condition of the fixed radar wavelength (λ) and the r_{a}. Therefore, if the measured velocity v_r is higher than v_{a}, the velocity will be aliased.

Figure 2.6 shows two example spectra with $v_{\mathrm{a}} = 15\mathrm{m\,s^{-1}}$. If the target velocity is $10\ \mathrm{m\,s^{-1}}$, which is below v_{a}, the Doppler spectrum is non-aliased as shown in the left panel of Figure 2.6. However, if target velocity is $16\ \mathrm{m\,s^{-1}}$, it is higher than v_{a}. In this

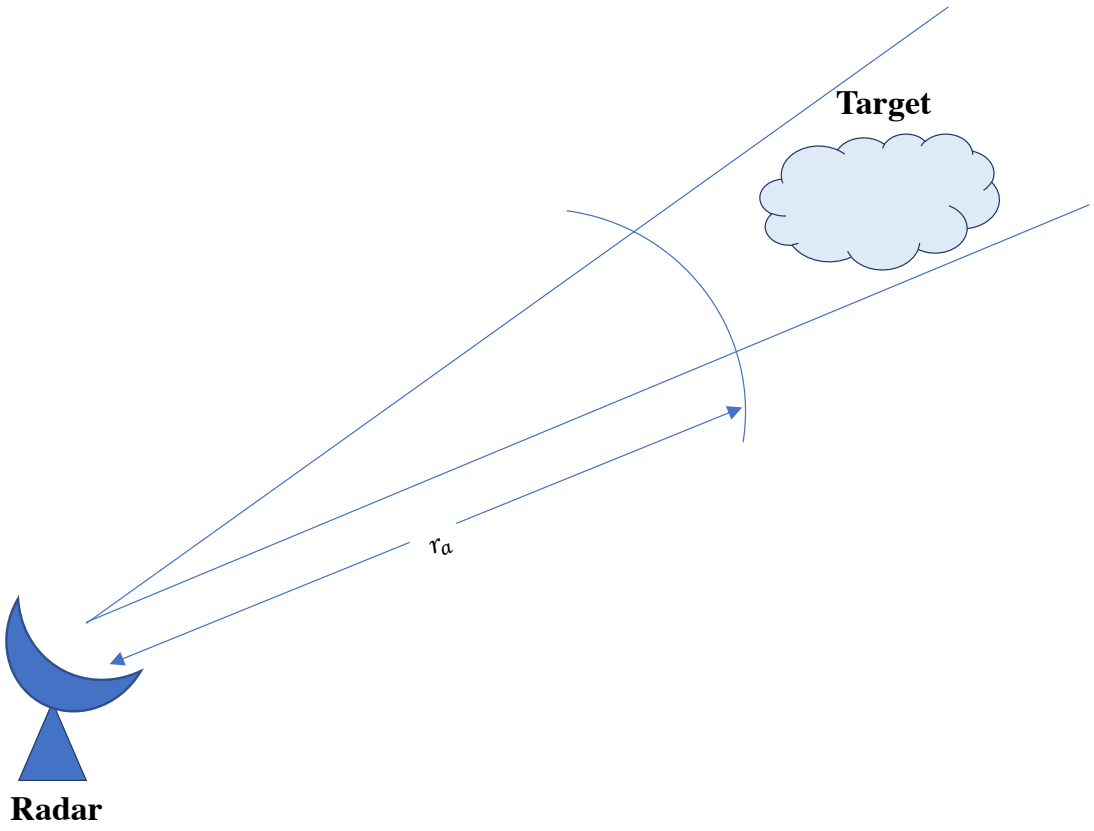


Figure 2.3: Example situation of the range aliasing in a radar system view toward the target. The target is located beyond r_{a} and range aliasing would occur.

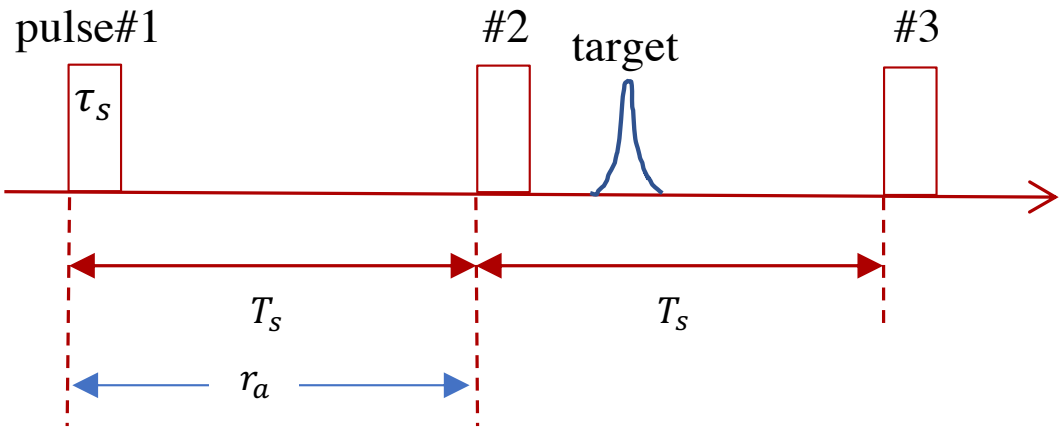


Figure 2.4: Example diagram of the range overlaid echoes in case of the first-trip target returned with the second pulse. The first-trip target is returned with the low-power when it is overlaid. τ_{s} is the pulse width, and the target echo from pulse#1 is received during the second period, which is overlaid with echoes from pulse#2.

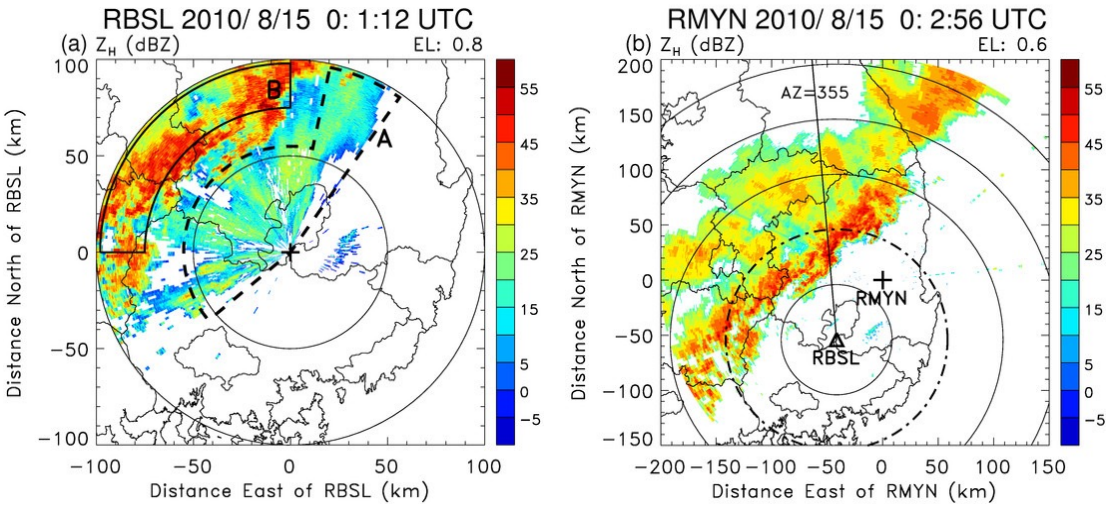


Figure 2.5: An example PPI images of range overlaid echoes extracted from [52]. Panel (a) is the PPI image of radar reflectivity from RBSL radar and panel (b) is from RMYN radar in South Korea. The left panel (a) displays a PPI collected with a shorter PRT, while the right panel (b) shows a PPI collected with a longer PRT. The dashed line indicates area A, which corresponds to the range overlaid echoes, and it is not visible inside the dotted-dashed circle in panel (b). The squall line area B in panel (a) is also shown in panel (b) inside the dotted-dashed circle. Thus, it is noteworthy that area A is where the range-overlaid echoes are present.

case, the Doppler spectrum is aliased and the target velocity appears as $-14\ \mathrm{m\,s^{-1}}$ as shown in the right panel.

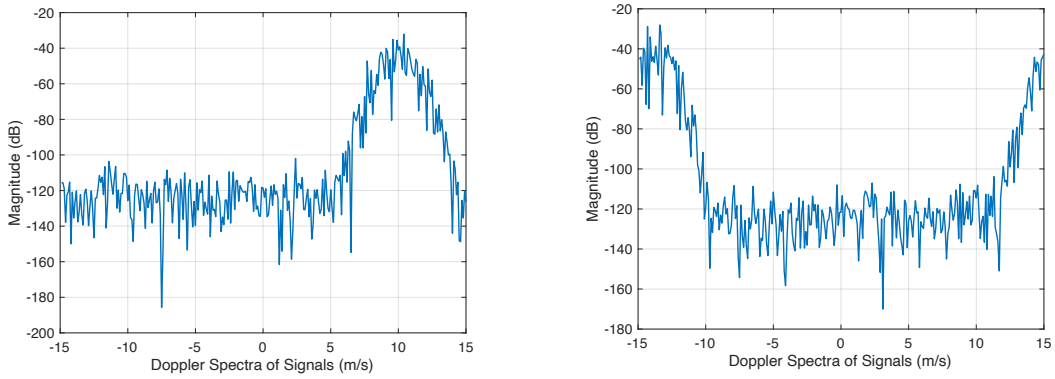


Figure 2.6: The Doppler spectra with v_{a} = 15 $\mathrm{m\,s^{-1}}$. The left panel shows a spectrum of a target with $v_{\mathrm{r}} = 10\ \mathrm{m\,s^{-1}}$. The right panel shows another with $v_{\mathrm{r}} = 16\ \mathrm{m\,s^{-1}}$ and the measured velocity is aliased to -14 $\mathrm{m\,s^{-1}}$.

2.6 Existing Range-Doppler Ambiguity Mitigation Strategies

Many studies have been conducted to mitigate the "Doppler dilemma", which can be divided into two main approaches: the waveform design method and the post-processing method.

2.6.1 Waveform Design Methods

Two or more PRT values are used for waveform design methods to dealias the velocity. The aliased velocities are found by searching for the disagreements between the two measurements with different PRT values. With the waveform design methods, velocity dealiasing is executed by solving the LCM problem. Pre-defined dealiasing rule is applied depending on the velocity difference between two (or multiple) measurements.

There are three typical ways in waveform design methods such as the staggered PRT, the dual PRF, and the dual scan method (which is also known as the split-cut

method). The basic idea of the staggered PRT method is to interleave two pulses with different T_{s} values as shown in Figure 2.7 (a) [14–20]. The dual PRF method is shown in Figure 2.7 (b). In this mode, the radar collects a radial by splitting it into two halves; each half uses the different periods and generates the radial with each T_{s} and dealiases the velocity by finding the velocity difference between two halves and solving for the unaliased velocity [2, 13]. Figure 2.7 (c) shows the dual scan method, which is known as the split-cut method. The radar scans the same elevation angle twice with different T_{s} values; one scan with a short T_{s} value is utilized for the high v_{a} and another scan with a high T_{s} is used for the long r_{a}. In operational use, the scan with the short T_{s} is used as it is, however, it can be technically expanded to the LCM of two v_{a} from each scan. Dealiasing is performed by a pre-determined rule based on the difference between two v_{r} from scans with different T_{s}.

2.6.2 Post-Processing Methods

The second approach is the post-processing method. The key idea of this method is continuity checking. In this method, aliased velocities are detected by searching the velocity disagreements (typically $\sim 2n\, v_{\mathrm{a}}$, where n is -2, -1, 1, 2, and so on). Consequently, the dealiasing is performed by adding or subtracting $2n\, v_{\mathrm{a}}$. The key assumption is that the initial velocity measurement, e.g., first cell, first range gate, first azimuth, etc., is non-aliased. An illustrative example is shown in Figure 2.8 where one can see that there is a clear discontinuity along the radial, which can be used to detect velocity aliasing.

Numerous studies have investigated velocity dealiasing to determine the discontinuity. It started from a one-dimensional dealiasing, which checks the continuity along the radial [21]. Checking discontinuity along the radial finds the first meaningful range gate

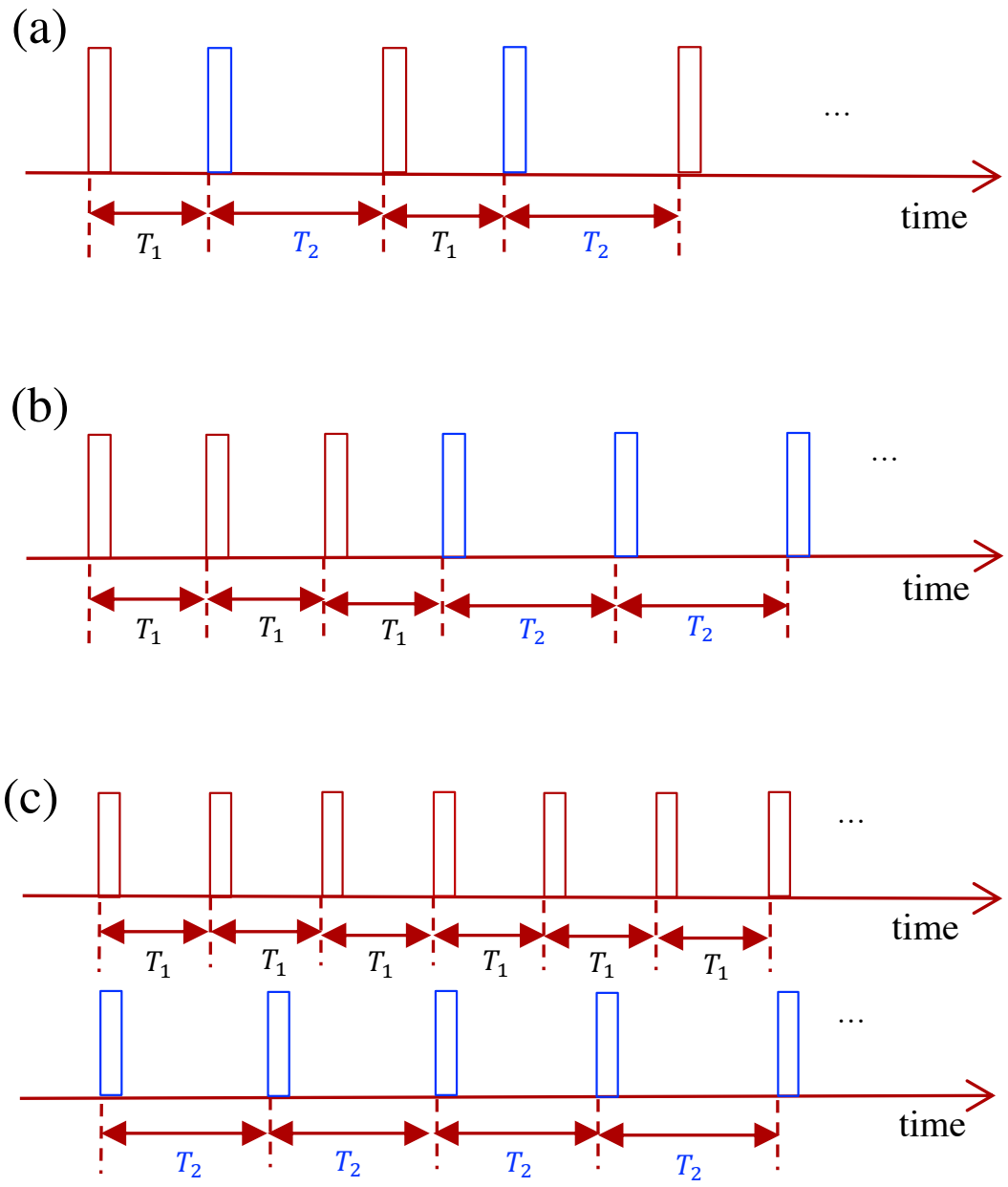
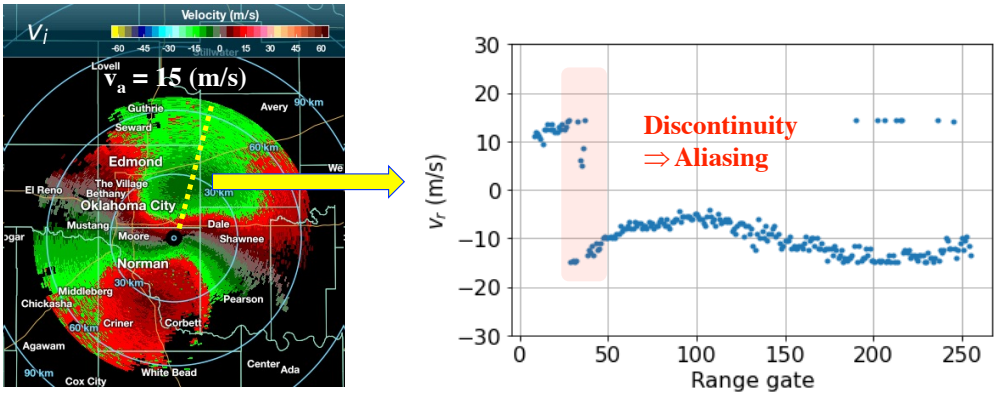


Figure 2.7: This figure compares the three waveform design methods in a diagram. Figure (a) shows the staggered PRT method, (b) represents the dual PRF method and (c) shows the dual scan method. T_1 is the short PRT and T_2 is the long PRT.



Mostly filled precipitation

Figure 2.8: Waveform Design Methods.

and uses its velocity as the reference velocity, which is assumed non-aliased. However, the reference velocity could lead to error propagation in case the assumption is incorrect. The environmental wind aids to determine the aliasing of initial velocity measurement [22], especially on the disjointed storm. Since the temporal and spatial resolution could be different in the wind profile from radiosonde, it could result in poor performance when the wind field is non-uniform and changes rapidly. Thus, VAD (Velocity Azimuth Display) was proposed to represent the environmental wind field [24, 53, 54]. VAD shows the radar display of the mean Doppler velocity at a specific range gate as the antenna rotates the 360 ° at a fixed elevation angle. Later, the one-dimensional continuity checking technique was expanded to two or higher-dimensional dealiasing methods. The higher dimension is more beneficial by introducing neighboring cells in azimuth, elevation, and time, thus, resulting in mitigation of the high dependency on external data sources such as sounding [23, 27, 55]. The method proposed by [25] utilizes a two-dimensional multi-pass scheme to find the velocity discontinuity. This method searches the reference velocity in two directions, i.e., clockwise and counter-clockwise with strict criteria for the first dealiasing of less likely aliased radials. Later

on, it gradually relieves the threshold to dealias the more complex velocities. In the data assimilation field, velocity dealiasing is performed on a four-dimensional radar data assimilation system "VDRAS" (Variational Doppler Radar Analysis System) by using three-dimensional wind fields from the objective analysis or the VDRAS analysis. It dealiases Doppler velocity at each grid point. Interpolated numerical weather prediction (NWP) model data and the radar VAD profiles are on the VDRAS grid and blended using a Barnes interpolation technique [56]. Most of the velocity dealiasing algorithms are limited since they focus on typical storms with a measured velocity ranging from 20 to 36 $\mathrm{m\,s^{-1}}$. Motivated to apply velocity dealiasing beyond this range, the ADTH (Automated Dealiasing for Typhoon and Hurricane) was developed [57]. In this method, the first reference radial is critical, as the subsequent dealiasing depends on the radial velocity that has been previously corrected. The algorithm begins by identifying the radial velocity that includes the zero Doppler line, as it is less likely to be aliased. Once the reference is determined, a two-dimensional multi-pass scheme is applied to dealias the velocity.

A novel region-based velocity dealiasing algorithm was proposed by [26]. The critical assumption of this method is that the first-guess field is non-aliased. Then, it checks the adjacent radar cells for abrupt velocity changes within a storm cluster and, if aliased, dealiases the velocity by adding $2nv_{\mathrm{a}}$ to the velocity measurement. Problems occur when the radar scan has multiple isolated storms, especially when the isolated storms are far from the radar. Multiple isolated storms require multiple attempts of estimating the first-guess field velocity aliasing. If the first-guess field is incorrect, it leads to incorrect velocity dealiasing of the connected storm. Therefore, this algorithm has an option to utilize the environmental background wind, which can be used to help the first-guess field estimation. This method is a part of the Py-ART, which is a software library of high-level radar data processing algorithms.

Although many attempts of mitigating the Doppler dilemma, the Doppler dilemma continues to be a challenge to the meteorological radar community. The waveform-design methods are still constrained by v_{a} after the LCM problem. The post-processing method has the first-guess field estimation problem especially when the storm is isolated and far from the radar. As a result, human intervention is often needed, which is a laborious and time-consuming process [28–30].

Chapter 3

Overview of Convolutional Neural Networks

In this chapter, an overview of AI, ML, NN, deep learning, and CNN will be provided. Especially the fundamental theory of machine learning including hyperparameters, and cost functions, will be explained.

3.1 Introduction to Machine Learning

AI is a term used when computers can do things that normally only humans can do, i.e., reasoning, problem-solving, planning, perception, learning, acting, and others [58]. AI is a rule-based system and its purpose is to create a machine that can think and act like humans. AI is widely applied including machine learning, deep learning, natural language processing, computer vision, robotics, and cognitive computing. These techniques can learn from the data and can be used in predictions, making decisions, etc.

ML is another way to arrive at the rules set to implement an AI. Unlike general programming, in which an engineer builds a model or method, the model finds their optimum weights and biases by its own fit-and-adjust process in ML. ML is widely utilized in various areas, especially in the application of classification. ML is a data-fitting method, where the model parameters, i.e., weights and biases, are optimized by

an iterative training process.

In general, ML can be classified into supervised, semi-supervised, and unsupervised learning depending on the existence of ground truth. In supervised learning, the model learns the methods using the labeled data (ground truth), which means that both input data and output labels are used in a training process. This learning method maps the input features to the known output, and finally, it makes a prediction for unseen data [59–61]. In unsupervised learning, the model learns the patterns from the unlabeled data without any help from experts, therefore, it has to find its own structure and representations. Supervised learning sometimes limits the training ability for the lack of ground truth, time-consuming, and expensive to generate the true labels by experts. Therefore, unsupervised learning can be advantageous because it does not require labeled data for training. However, it can be challenging to evaluate the performance of training results since there is no ground truth available for comparison, and the algorithm may require a large dataset to accurately identify the patterns [59, 62–64]. Semi-supervised learning is a type of ML that is the combined concept of supervised and unsupervised learning. One popular approach to semi-supervised learning is using "pseudo labeling" [65]. The first step is similar to supervised learning, where the model is pre-trained with the labeled data. The second step involves generating "pseudo labels" for the unlabeled data using the pre-trained model. In the next step, the data is thresholded using the softmax probability, which allows the model to filter out the low probability data, i.e., only high confidence data remain. The final step involves training the model with the labeled data and the pseudo-labeled data. This learning method is used when the true label is small, but a larger number of unlabeled data are available [66–68].

A NN is made up of artificial neurons. A simple model is just a node with a weight and a bias term. This simplest unit is often referred to as a neuron. While an NN has one hidden layer, DNN is composed of more than two hidden layers, which can have a

hundred or thousand hidden layers. Increasing the depth of the model allows the model to learn and extract more complex features, which can lead to improved performance on a given task at a certain point. However, if the model becomes too deep, it can lead to overfitting issues, where the model becomes too specialized to the training data and does not generalize to new, unseen data.

A CNN is a two-dimensional data filtering process to extract the features from input data. The term convolution refers to the two-dimensional filters that operate like a sliding window. The mathematical operation used in CNNs is convolution, which represents how one function is affected by another function. Therefore, it is suitable for image processing such as classification and segmentation to extract the features between input data and the convolution filter. More details will be explained in Section 3.3.

The Venn diagram in Figure 3.1 illustrates the relation between AI, NN, DNN, and CNN. CNN belongs to the DNN, which is a type of NN, and NN is a subset of AI.

3.1.1 Neural Networks

A simple NN is shown in Figure 3.2. Each weight and bias pair is a neuron, and multiple neurons form an NN (1 layer). This NN is composed of the input layer, hidden layer, and output layer. The input and output can be described as:

$$y = \mathrm{w}x + \mathrm{b}, \tag{3.1}$$

where y represents the output, x represents the input, w and b are the weight and bias, respectively. A design with more than one hidden layer is considered a DNN.

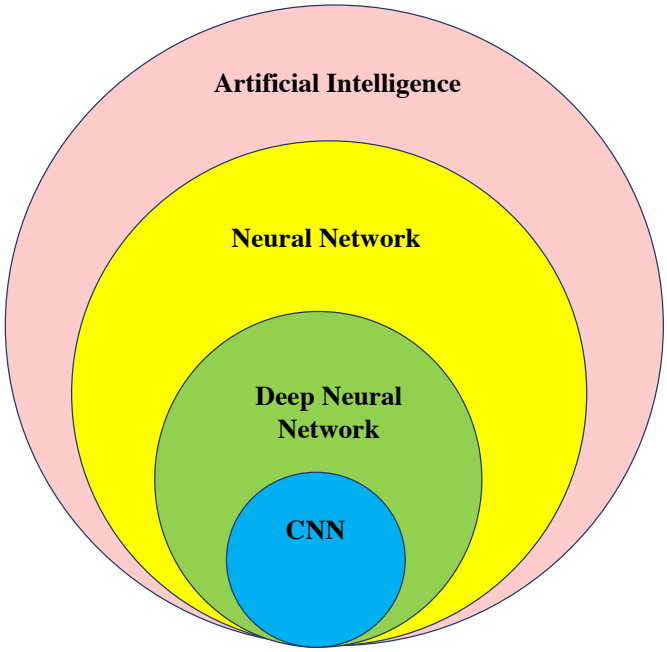


Figure 3.1: A Venn diagram illustrating the different classes of artificial intelligence. A CNN represents a particular type of AI, and it falls within the category of DNN, which belongs to the larger category of NN. NN is the subset of the broader category of artificial intelligence.

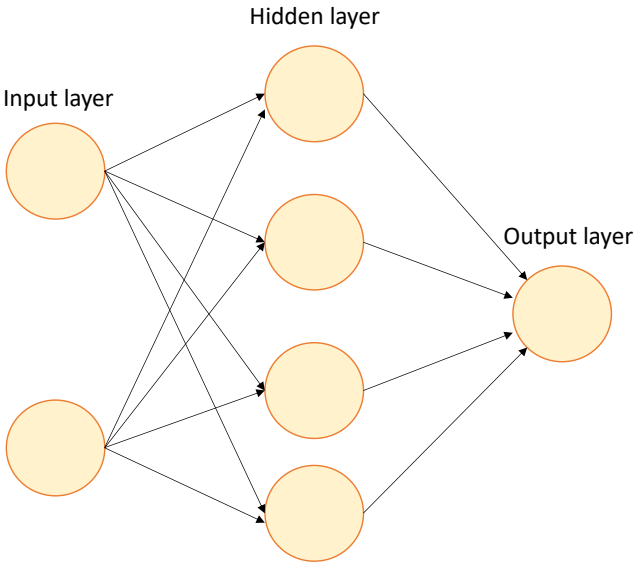


Figure 3.2: An example diagram of a simple NN. It consists of an input layer with two inputs, one hidden layer, and an output layer with one output.

3.1.2 Deep Learning

Deep learning is the training process of a DNN [32]. An example of a DNN is shown in Figure 3.3. The process to obtain the weights and biases is often referred to as ML. Therefore, the goal of ML is to optimize the weights and biases of the model. During the training process, the weights and biases are updated to minimize the error between the true label and the predicted output. Minimizing the error refers to the optimization process. The fitting curve, which is the combination of the weights and biases, is updated by the gradient descent algorithm. The gradient descent algorithm computes the gradients of the error, which is the slope, on the weights and biases to find the minimum error. This algorithm updates the weights and biases when the set of input data is induced to the model until the error reaches the minimum value. In other words, data fitting finds the best set of weights and biases for a machine learning model by a fit-and-adjust process to minimize the error between the true label and the predicted output.

Deep learning can be applied in various fields including developing autonomous vehicles to help their vehicle perceive and detect obstacles, healthcare for medical diagnoses, and gaming for the intelligent response when playing with machines such as AlphaGo.

3.2 Machine Learning Basics

In this section, the high-level parameters, which are considered hyperparameters, such as epoch, batch size, and kernel size, will be discussed. The regression method will also be described including linear regression, the logistic regression, and it will be expanded to the activation function of deep learning models. Since machine learning is the fit-and-adjust process, backpropagation is one important process, which performs like a

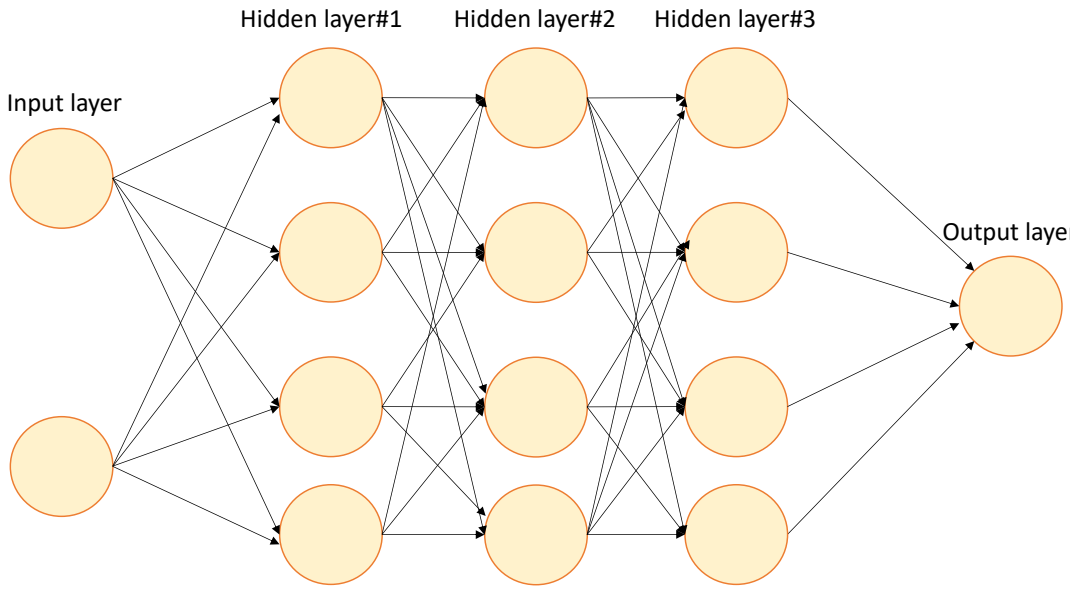


Figure 3.3: A simple diagram of a DNN. It consists of an input layer with two inputs, three hidden layers, and an output layer.

feedback process, to adjust the model weights and the biases. Through the backpropagation process, the weights and biases are optimized to minimize the cost function using the gradient descent algorithm. At the end of the section, the role of training, validation, and test dataset will be also discussed.

3.2.1 Hyperparameters

Batch size is defined as the number of samples in one neural update, where one neural refers to the weight and bias. It determines how many samples or images from the training data are used to update the model's parameters, which are the weights and biases. An epoch refers to a complete pass of the entire training dataset through multiple updates. For example, as shown in Figure 3.4, if the training data contains 1,600 samples and the batch size is set to 32, assuming no samples are reused during a cycle, the model would be updated (trained) 50 times in order to process the entire training data once. The batch size and the number of epochs are important hyperparameters to set since

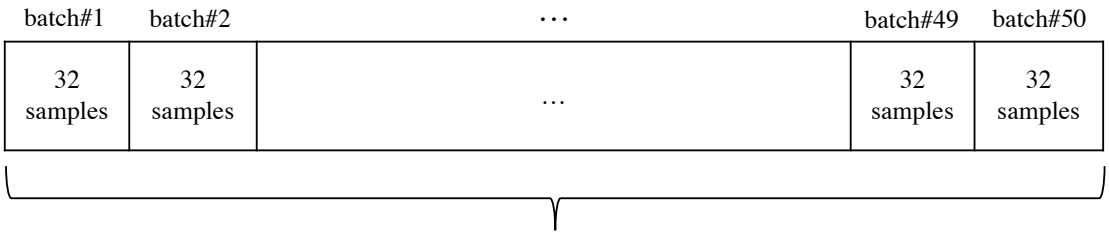


Figure 3.4: The concept of the batch size and epoch in one iteration training. Each batch size has 32 samples and a total of 50 batches are included to train one epoch for a total of 1,600 samples.

they can affect the learning efficiency and the performance of the model. A small batch size would lead to unstable parameter updates, on the other hand, a large batch size would slow down the training time to process one batch and sometimes, it is over the computation power. Therefore, it is important to select the appropriate batch size and it is experimentally determined depending on training conditions. The first batch size trial is commonly started with 32 samples and it is adjusted depending on the dataset and the performance updates. Training with a small number of epochs can result in underfitting, whereas a large number of epochs can lead to overfitting. Therefore, finding the well-balanced batch size and the number of epochs is an important task in training and it can be empirically obtained through experimentation.

Kernel size should also be discussed in the choice of model hyperparameter. A kernel is a filter. It is a small matrix that is used to extract the features from an input image. A larger kernel size can decrease the computational time, but it may lead to loss of details, while a smaller kernel size can detect fine details. The appropriate kernel size is determined through experimentation, the common choice of kernel size is 3×3 or 5×5.

3.2.2 Regression

As explained in Section 3.1.2, data fitting is the key to the training process. The goal of regression is to find the best-fit function between the input (independent variables) and the output label (dependent variables) [69]. The best fit refers to the appropriate fit, which should not be underfitting, leading to low accuracy even on the training dataset, nor overfitting, which is highly focused on the training dataset, which could lead to failure in unseen data prediction. The most common types of regression in deep learning are linear regression and logistic regression.

The linear regression model is the statistical approach to modeling the relationship between the independent and dependent variables [61]. In an NN, the input data can be one or more, however, the model is implemented using a single hidden layer and has a linear activation function. Since linear regression is the data fitting method to fit the linear curve, it can be shown in Figure 3.5 as an example. The fitting curve is defined in Equation (3.2).

$$y_i = \sum_{i=1}^{\mathrm{n}} \mathrm{w}x_i + \mathrm{b}, \tag{3.2}$$

where w is the weight, b is the bias, x is the input, and n is the total number of samples. The goal of the regression is to find w and b that best fit the entire dataset of x_i and y_i as a whole. One typical way to solve linear regression is using the least-squares method. It is the data-fitting method, therefore, the curve is determined as the best fit for a set of data. It provides the relationship between the input (x) and the output (y). Another way to find a solution is by the gradient descent method. The w and b in Equation (3.2) are updated when the sample pairs of x_i and y_i are induced for every batch.

The gradient descent method minimizes the cost function resulting in optimized w and b for each layer. Figure 3.6 compares the high and low learning rates, which will

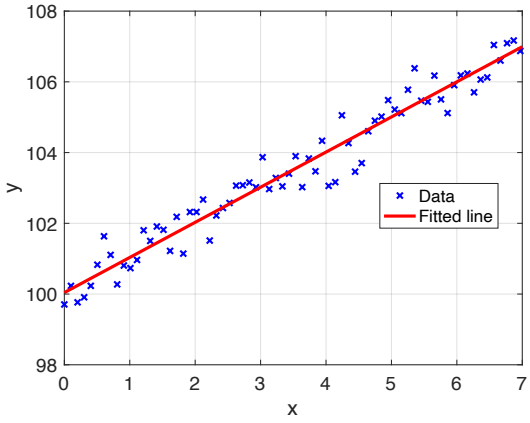


Figure 3.5: This figure shows an example of linear regression. The data are fitted to the line $y = 0.9932x + 100.0412$. This line is derived from the least-square fitting method.

be discussed in Section 3.2.4. In this method, the independent variable (x-axis) is the weight w (or bias b), and the dependent variable (y-axis) is the cost function. The cost function will be discussed in Section 3.2.5. The training starts with the random w and b, and these are updated to reduce the cost function. The gradient of the cost function on w or b is used to update the weight and the bias by reducing the cost function and the gradient is also gradually reduced, therefore, this method is named as the gradient descent algorithm. The update of the w and b are performed by Equation (3.3) and Equation (3.4) for each.

$$\mathrm{w}_{\mathrm{new}} = \mathrm{w}_{\mathrm{old}} - \alpha \frac{\delta}{\delta \mathrm{w}} C, \tag{3.3}$$

$$\mathrm{b}_{\mathrm{new}} = \mathrm{b}_{\mathrm{old}} - \alpha \frac{\delta}{\delta \mathrm{w}} C, \tag{3.4}$$

where C represents the cost function and it is related to the loss. The cost function will be discussed in Section 3.2.5.

Linear regression is the data fitting method for the linear relation between input

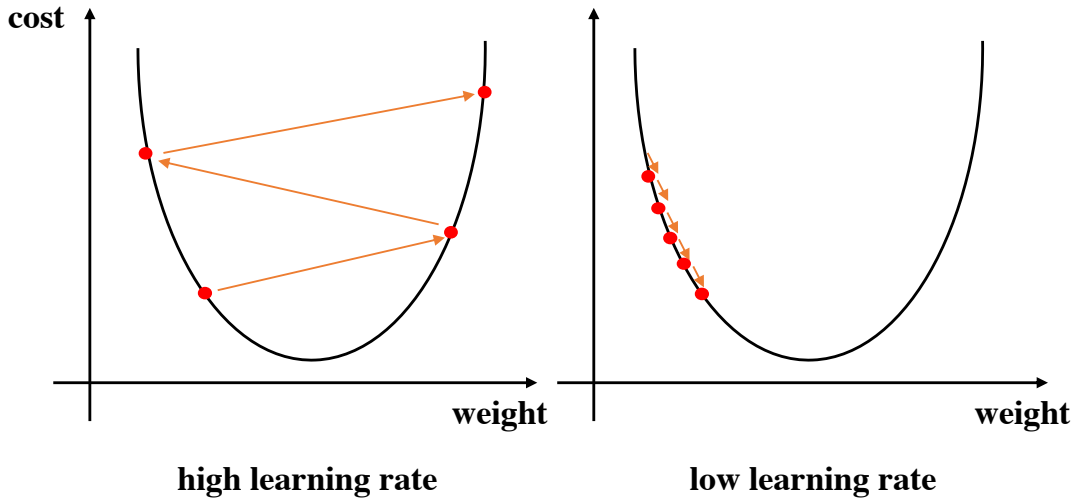


Figure 3.6: This figure illustrates the inappropriate learning rate. A high learning rate would lead to divergence, where the model could fail to map the input data to the latent space with large updates. A low learning rate requires many updates to reach the minimum cost, therefore, it makes the training slow.

and output outcomes. Therefore, it can be used in a simple linear network. However, the complex features including the image classification and detection of the velocity aliasing would have the non-linearity between the image features. Therefore, "logistic regression" is introduced to deal with the non-linear features. This regression is used to analyze the binary or categorical outcome. It models the probability of the dependent variable being in a particular category using a logistic (or activation) function to map the input values to the output probability [70]. The activation function will be explained in the following section.

3.2.3 Activation Function

An activation function is a mathematical function that maps an input value into an output probability in the range between 0 and 1. It operates like a switch or a thresholding function that converts an input value to an output value that indicates on or off, hence,

the term activation. The goal of the activation function is to enable the model to learn the complex features by introducing non-linearity. There are several commonly used activation functions, i.e., sigmoid function, Rectified Linear Unit (ReLU), Hyperbolic Tangent (tanh), and leaky ReLU. Additionally, the softmax function is often used for multi-class classification tasks [60].

A sigmoid function maps any input to the range of 0 to 1, as shown in Figure 3.7(a). It is commonly used in binary or categorical classification problems, and the goal is to predict a probability for each class. The equation is as follows:

$$f(x) = \frac{1}{1 + e^{-x}}, \tag{3.5}$$

where the x is the input value. However, the use of the sigmoid function can lead to a significant issue known as the "vanishing gradient" problem. As more layers are added with the sigmoid function, the gradients of the loss function can approach zero, making it difficult to update the model's parameters and leading to training difficulties.

A hyperbolic tangent (tanh) function maps input to the range of -1 to 1 as shown in Figure 3.7(b) and expressed in Equation (3.6). This function can be useful in image classification where the output can be mapped between -1 to 1. For example, it can be used to label the aliased velocity, as Doppler velocity can have a negative or positive direction. However, it is not generally used in CNNs due to the vanishing gradient issue, which is similar to the issue encountered with the sigmoid function.

$$f(x) = \tanh(x) \tag{3.6}$$

A rectified linear unit (ReLU) function returns the input if the input of the activation function is positive, and returns 0 if the input is negative, as shown in Figure 3.7(c). ReLU can be expressed in Equation (3.7).

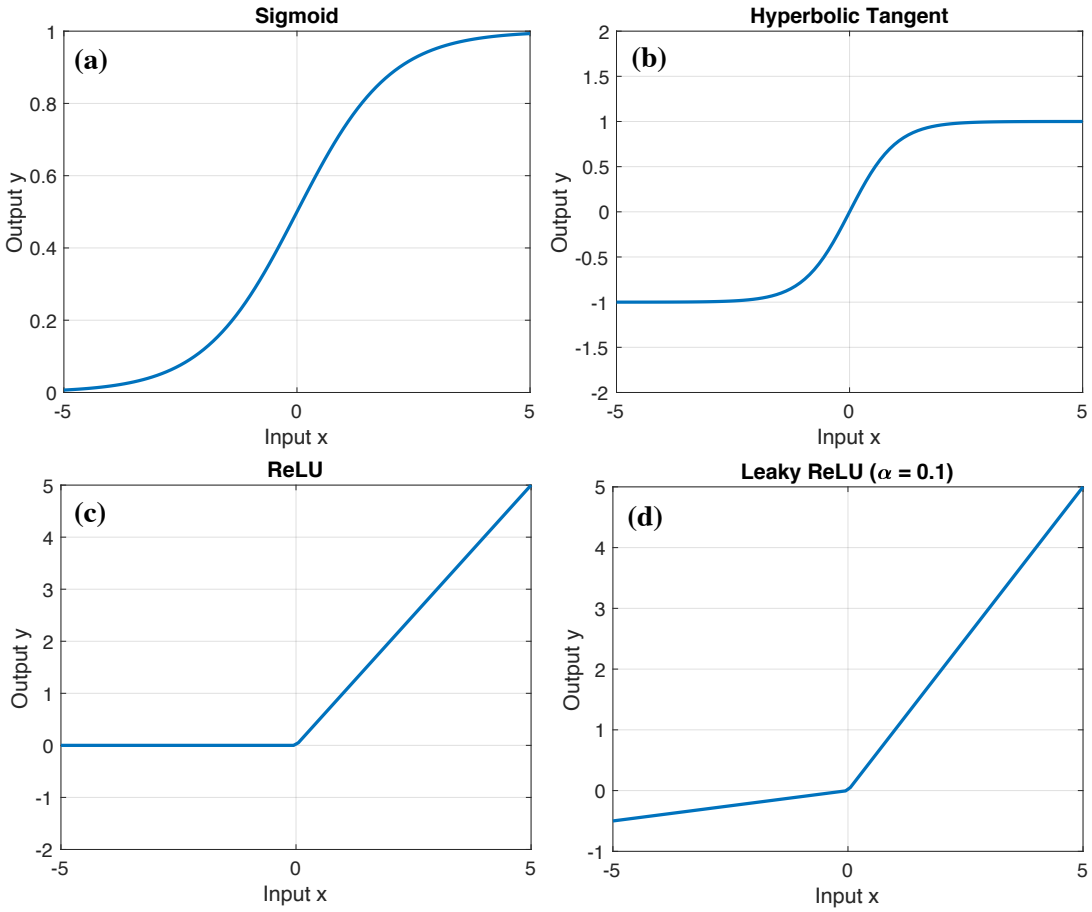


Figure 3.7: These graphs illustrate the different activation functions. Panel (a) shows the sigmoid, panel (b) represents the tanh, panel (c) illustrates the ReLU, and panel (d) shows the leaky ReLU with $\alpha = 0.1$.

$$f(x) = \max(0, x) \tag{3.7}$$

ReLU introduces the non-linearity into the NN by setting negative input to zero and it can help avoid the vanishing gradient issue. This problem can occur when the gradient of the cost function on weights is extremely small in the early layers of the network, making it difficult to update the weights or slowing down the learning process. In a sigmoid function, when the input is large, the gradient of the cost function is small, therefore, it may occur this vanishing gradient issue. However, ReLU has a gradient of 1 for the positive input and 0 for the negative input, it can keep the gradient and avoid the vanishing gradient issue. Moreover, ReLU is computationally efficient since it only requires the simple comparison of input value with zero, making it widely used in CNN architectures.

Since the ReLU has the "dying ReLU" problem, which has 0 as an output for the negative input, "Leaky ReLU" is introduced. It protects the "dying ReLU" problem by introducing the small slope for negative inputs, and it is shown in Figure 3.9(d), where the α (slope) sets up as 0.1, and the equation is as follows:

$$f(x) = \max(\alpha x, x). \tag{3.8}$$

In multi-label classification, the softmax function ($s(y)$), which is expressed in Equation (3.9), is commonly used by mapping the inputs to a probability distribution over the multiple classes. It maps the output value in the range between 0 and 1 and the sum of all outputs is equal to 1, therefore, the output value can correspond to the probability of each class.

$$s(y_i) = \frac{e^{y_i}}{\sum_{j=1}^{M} e^{y_j}}, \tag{3.9}$$

where M is the number of classes.

In the early days of NN, the sigmoid function is widely used as an activation function. However, as mentioned, it has a limitation for the vanishing gradient problem, which means that when the input is significantly small or large, the gradient of the function (slope) would be significantly small, and it leads to the vanishing of the gradient. To relieve this issue, the hyperbolic tangent (tanh) function is introduced by extending the range to -1 to 1 whereas the sigmoid function maps the output to 0 to 1. It somewhat helps the vanishing gradient problem, but it still exists. Therefore, the $ReLU$ function is introduced to mitigate this issue by giving zero to the negative input and one to the positive input. Moreover, since it is computationally efficient, it is widely used as an activation function. For the multi-label classification, the softmax function is widely used since it provides the probability as outputs for each class, and the maximum value is chosen as the output label.

An example model with logistic regression using the activation function can be depicted in Figure 3.8, and the output (y) can be defined in Equation (3.10). In this figure, three inputs are multiplied with weights, and then the bias is added. The resulting value is passed through the activation function, which maps the summed input to the output probability.

$$y = \sigma(\sum_{i=1}^{\mathrm{n}} \mathrm{w}_i x_i + \mathrm{b}), \tag{3.10}$$

where σ is the activation function. Non-linear functions such as the sigmoid function can expand the model's ability to cover the non-linearity.

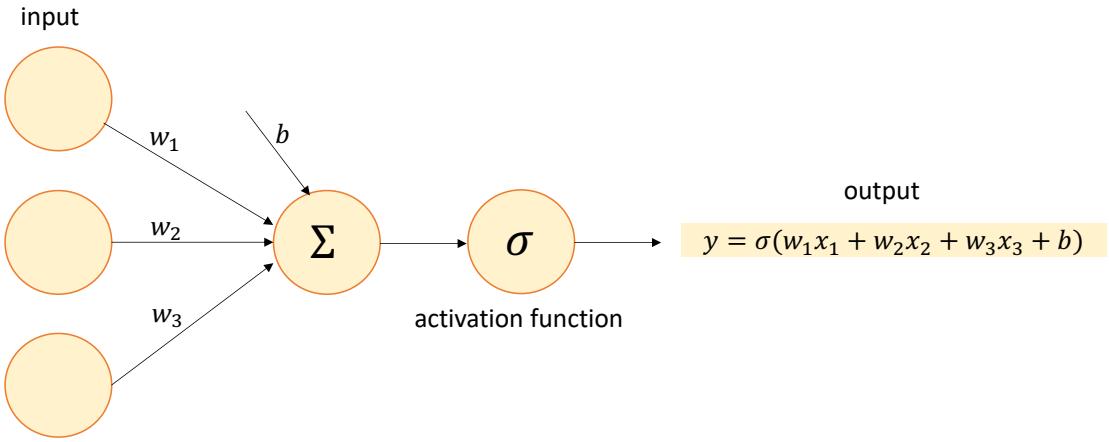


Figure 3.8: This diagram illustrates the example model with logistic regression, which takes three inputs. First, the inputs are summed and then the sigmoid function is utilized as an activation function for the logistic regression.

3.2.4 Back Propagation

As mentioned before, one approach to solving linear regression is to find the best fit for the input and output data by updating the weights and biases. This is achieved based on the chain rule, which states that the derivative of a composite function can be obtained by multiplying the derivative of the outer function by the derivative of the inner function. The example of the chain rule is shown in Equation (3.11).

$$\frac{\delta f}{\delta w} = \frac{\delta f}{\delta g}\frac{\delta g}{\delta w} \tag{3.11}$$

The chain rule can be used to calculate the gradient of the loss function with respect to the parameters, which is the concatenation of the multiple weights or biases, from the right to the left side. This is the backward feedback process, which is called back-propagation. The weights and biases are updated through the backpropagation based on the chain rule. Updating is performed to reduce the error, which is the difference between the true label (y_{t}) and the predicted output (y_{p}).

There is another factor that affects the updating of weights and biases, which is

known as the learning rate (α). Therefore, the learning rate should be optimized. A large α would lead to divergence, while a small α could make the training speed slow. Hence, the appropriate α should be determined empirically. The common α range is 0.1 to 0.0001, depending on the factors such as the size of the training dataset, the complexity of the network, etc. A good starting α for testing is 0.01. which can be adjusted depending on the loss.

During training, the model starts with random weights and biases, the loss might be reduced significantly in the early epochs. However, if the loss does not decrease sufficiently in the early epochs, it may indicate that the learning rate is too small. In this case, increasing the value of α can improve the model's performance. Sometimes, even if the loss has saturated, the model's accuracy may still be significantly low. This could be due to a high learning rate, which can cause the cost function to diverge. To address this issue, it is often helpful to decrease the learning rate. If the learning rate is not the issue, it is possible to be a data problem or a structural problem with the model.

Data issue includes poor data quality or not enough data availability, which can limit the training. An inappropriate network structure can also hinder the training process of the model. For example, complex features might be difficult to be trained with a simple network. In this case, adding more layers or changing the network structure would be helpful to increase the performance.

The backpropagation process is illustrated in Figure 3.9, the feedforward process is performed by following the black line from left to right, while the backpropagation process is operated in the opposite direction, from right to left, by following the red line. During the backpropagation process, weights and biases are updated using Equation (3.3) and Equation (3.4) respectively based on the chain rule.

As explained earlier, machine learning is the fit-and-adjust process using backpropagation to find optimized weights and biases. The optimization is done by minimizing

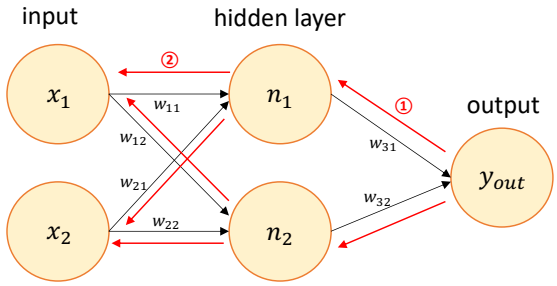


Figure 3.9: This figure illustrates the feedforward and backpropagation process. The black line from left to right represents the feedforward process, while the red line from right to left represents the backpropagation process. During the backpropagation process, all the weights and biases are updated.

the cost function, which is also referred to as the loss function.

3.2.5 Cost Function

In this section, the cost function will be discussed. The training goal is to iteratively adjust the model parameters through a fit-and-adjust process. The cost function, also known as the loss function, measures the difference between the true value and the predicted output value. Examples of cost functions include Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), cross-entropy, and others. Cross-entropy is commonly used in DNN, particularly for multi-label segmentation.

Intuitively, entropy measures the uncertainty of a field [71], and cross-entropy measures the uncertainty between the two variables, which correspond to the predicted label and the true label. Cross-entropy has been shown to be a viable loss function for segmentation problems, e.g., [31, 72]. A cross-entropy of zero indicates that the predicted label is the same as the true label. Therefore, minimizing cross-entropy means making the model closer to perfect predictions. During the training process, the cost function (loss) decreases until convergence, which is defined as the successive change of the

performance that is less than a preset threshold. A properly trained CNN model should be able to produce good predictions in general.

$$C = D(s(y), L) = -\sum_{i,k} L_i(k) \log(s(y_i(k))), \tag{3.12}$$

where L is the true label, i represents the aliasing label, k is the cell index, and $s(y)$ represents the output of the softmax classifier.

Since the model parameters are trained by minimizing the loss of the more populated label, the less populated labels may not be well-trained in the training process. In the real world, data often exhibit class imbalance, where certain labels have much fewer samples than others. For example, when training a model for aliased velocity, one or more of the aliased velocities may be less populated compared to non-aliased velocity.

In this case, class weight can help to better train the less populated label by giving it more weight during the training process. Therefore, weighted cross-entropy loss, which gives the different class weights for each label, can be employed to equalize the imbalanced data. The weighted cross-entropy loss (C_{w}) is defined as follows:

$$C_{\mathrm{w}} = -\sum_{i} \mathrm{w}_i L_i \log(s(y_i)), \tag{3.13}$$

where w_i is the class weight of label i, which is defined as the inverse of the population ratio of the label. However, in cases where the data distribution is extremely skewed, using class weights may not be a viable solution as it could have a negative impact on the training of the more populated label. For instance, if the training dataset contains 500 cats and only one dog, the class weight assigned to cats versus a dog is 1 to 500. In this scenario, the model would try to fit one dog equally as the other 500 cats, resulting

in poor performance in identifying cats. Therefore, even if the class weights are implemented, a sufficient amount of data on less populated labels should still be provided to ensure the model can learn their characteristics.

3.2.6 Dataset

The training dataset is used to update the model parameters, which are the weights and biases. The validation dataset is used to determine the training stopping point, which prevents the model from overfitting to the training dataset. Examples of underfitting, appropriate fitting, and overfitting are shown in Figure 3.10 using the same data in all three panels. The left panel shows the underfitting curve, the middle panel shows an appropriate fitting curve, and the right panel represents the overfitting curve. The test dataset is utilized to evaluate the performance of the model. It is important to note that these three datasets are mutually exclusive and should not overlap.

3.3 Convolutional Neural Networks

CNN is a type of DNN widely used for tasks such as two-dimensional image classification and three-dimensional object recognition in video processing. The structure of a CNN model consists of a combination of convolutional layers, pooling layers, activation function, and so on [33]. Typical CNN models ingest input data as width by height by depth.

3.3.1 Layers

A convolutional layer is a collection of concatenated two-dimensional filters used to extract the common features through the convolution operation. It is the core layer of CNN, and it learns the meaningful representations of the input data [35]. Figure 3.11

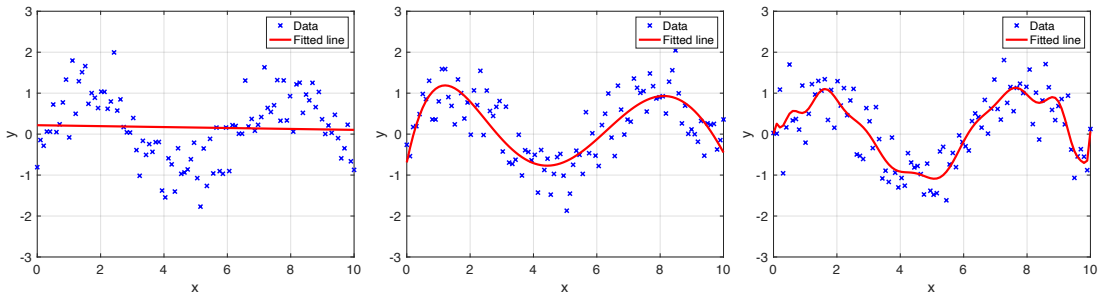


Figure 3.10: These figures illustrate the importance of finding an appropriate data-fitting curve. The left panel shows an example of underfitting, the middle panel shows an appropriate fitting curve and the right panel demonstrates an example of overfitting. In each panel, the blue dots represent the data, and the red line shows the fitted curve. A curve that is too simple (underfitting) may not accurately capture the data properties, while a curve that is too detailed would be overfitted on data and may not generalize to unseen data.

shows an example of a convolutional layer with kernel size (3×3) and stride 1. Here, the stride refers to the number of pixels that shifts over the input matrix during an operation such as convolution. The input patch is element-wise multiplied by the convolution filter.

The pooling layer performs the downsampling by reducing the spatial size of the input while retaining the important features. A 2×2 window is commonly used to downsample the input. Average pooling and max pooling are widely used pooling methods. Average pooling extracts the average value in the window, while max pooling extracts the maximum value from the window, as shown in Figure 3.12. The maximum value is extracted from each kernel. Pooling layers help reduce the complexity of the network and make the representation more invariant to small translations in the input [37].

A fully connected layer performs the classification task and gives the output of the final predictions. It takes the feature maps learned by the previous layers and makes the prediction based on the entire input [37]. Another type of layer, named the dropout layer, randomly drops out some neurons during training to prevent overfitting [73]. The

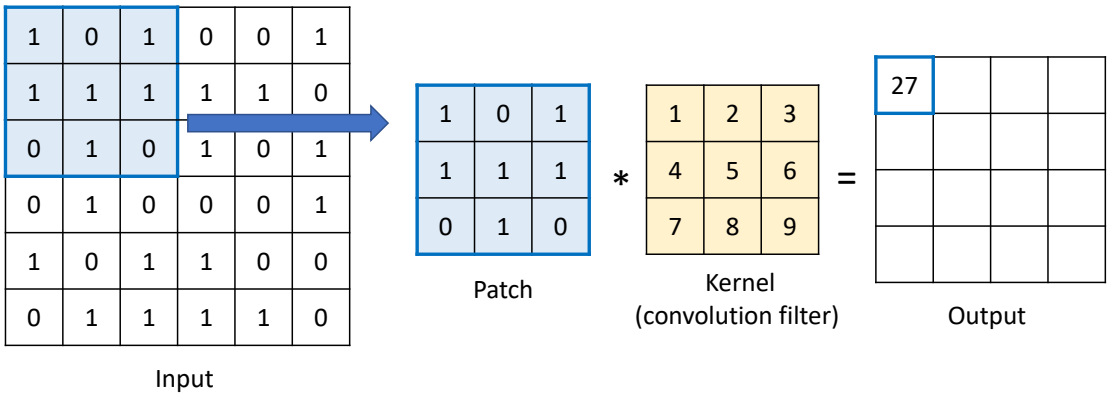


Figure 3.11: This figure shows how the convolutional layer works. The left image is an example of the input data assuming a kernel size of 3×3. 3×3 patches are extracted from the input, the pre-determined convolution filter is convolved pixel-wise, and the calculated result goes to the output. In this example, when we perform an element-wise convolution between the patch and the convolution filter, the resulting value is 27.

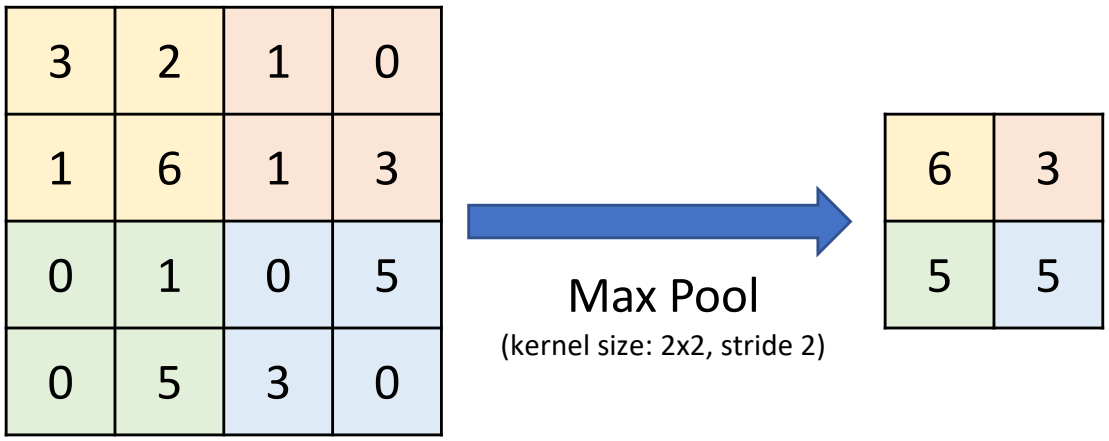


Figure 3.12: This figure illustrates an example of max pooling. A 2×2 kernel size with a stride 2 is used to downsample the input. The maximum value in each kernel is pooled out. In this example, the value of 6 is pooled from the yellow patch of the left panel, 3 from the red patch, 5 from the green patch, and the blue 5 is pooled from the blue patch of the left panel.

activation function is also considered as one type of the CNN layer and it is explained in Section 3.2.3.

3.3.2 Receptive Field

The receptive field refers to the region of the input space that directly affects a neuron's output. It is the field that the neuron is "looking at" to produce its output. For example, in Figure 3.13, with a kernel size of 3×3, the yellow cell in layer 3 includes the information from the entire cell of Layer 1. Therefore, in a training process, key features can be captured while the input size is reduced. The receptive field should be considered when designing the model structure for the number of convolutional and pooling layers. Typically, if the receptive field is already larger than the input size, additional layers do not have any impact, which helps in deciding the appropriate model structure.

3.3.3 Class Segmentation

As mentioned above, CNNs are widely used in image classification. During training, an input image is fed into the network along with the ground truth label. The trained model provides the output scores for each class, and the class is determined as the one with the highest score or probability. Since training begins with random weights and biases, the model gradually learns to identify features by reducing the loss, which is the difference between the true label and the predicted output during training.

After training, the optimized weights and biases are applied to new, unseen data, and the model can determine the class of unseen images. In addition to single-label classification, pixel-wise prediction is also available by using the deconvolution process. The deconvolution process is used to recover the original image size from a downsampled, low-resolution image by convolving an upsampled image with the corresponding

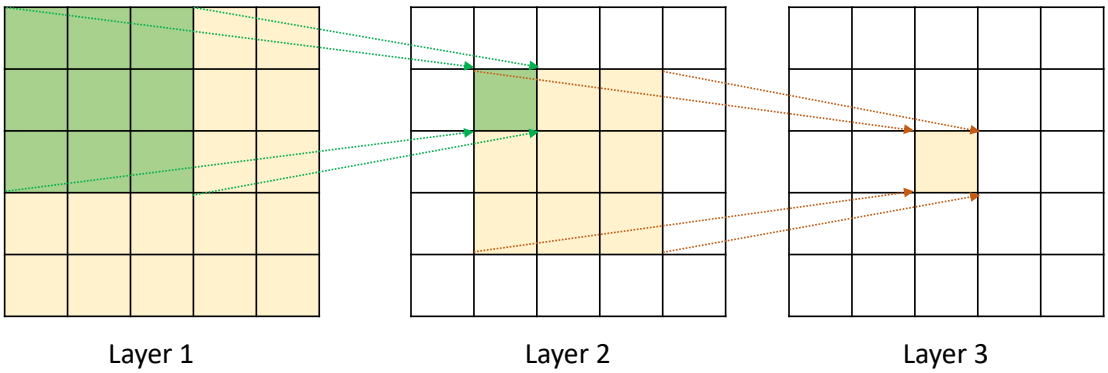


Figure 3.13: This figure illustrates the concept of the receptive field. From layer 1 in the left panel, the green patches become the center pixel in layer 2 (middle panel). In other words, the receptive field of a green pixel in layer 2 is the green patches (3×3) of layer 1. Similarly, the receptive field of the yellow center pixel in layer 3 is the yellow patches of layer 2 (3×3), which corresponds to the entire image (5×5) of layer 1. Therefore, the receptive field of a yellow center pixel in layer 3 is the entire image of layer 1 (5×5).

level of the image from the downsampling process and concatenating the result. Pixel-wise prediction can provide an output label for each pixel, making it useful for image segmentation.

One possible model to enable deconvolution is the U-Net structure. The U-Net model mostly consists of convolution and pooling layers and segments the image by capturing edges, lines, corners, and other features. As the structure becomes deeper, the number of filters increases while the image size becomes smaller, enabling more accurate classification at a certain point by considering the receptive field and template size.

To achieve the same output size as the input, a U-Net architecture can be used for semantic segmentation using a CNN [31]. The U-Net architecture consists of an encoding and decoding process. As shown in Figure 3.14, the left side is the downsizing part, which is the encoding process that extracts features from the input. The right side is the decoding part, which recovers the same output size as the input. It is composed

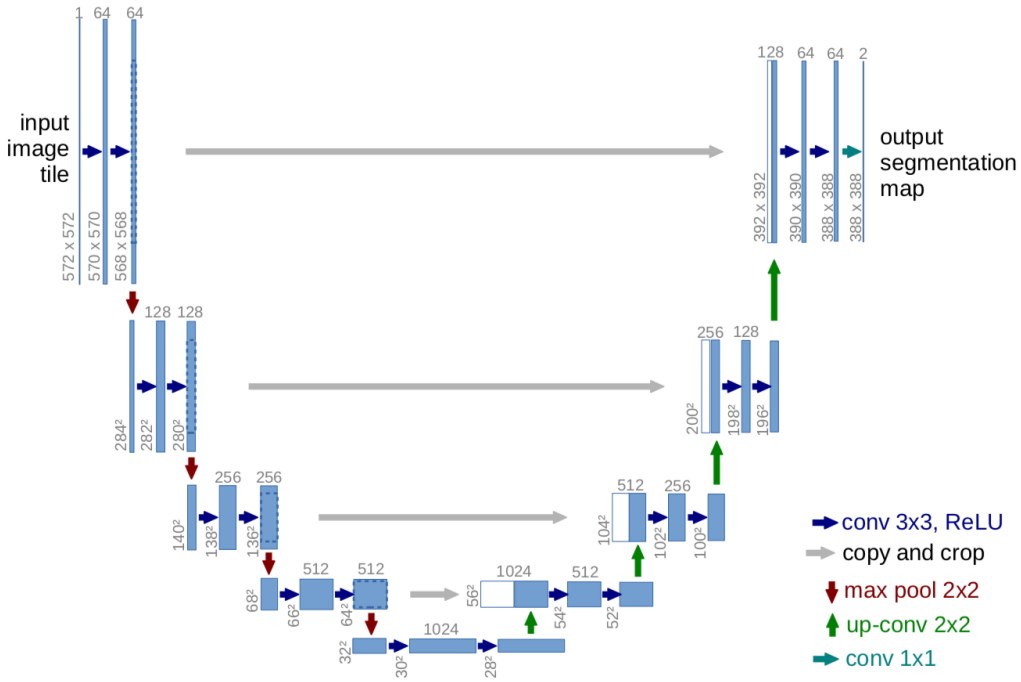


Figure 3.14: U-Net model structure example extracted from [31]. The input image is 572 $\times$ 572 and is downsampled to 28 $\times$ 28 using a combination of convolutional and max pooling layers. Then, it is upsampled to 388 $\times$ 388 using the up-convolutional layers and concatenated with the layers from the downsampling part.

of multiple convolutional layers and max pooling layers by considering the receptive fields. The output is a map of labels generated from the features learned by the encoder at a lower resolution and projected onto the pixel level at a higher resolution.

In the context of U-Net, upsampling using an output along with the input from the previous layers at the same depth provides a mechanism to generate a feature map of the same size at the same depth. The last layer turns the output from the last deconvolutional layer into labels through a process that is similar to the mode process, i.e., the value at which the probability is the highest. A softmax function is used to provide a normalized output, which can be used as the probability of each label. It is commonly employed for multi-label classification [33].

As mentioned in Chapter 1, the detection of velocity aliasing can be treated as

image segmentation, and CNNs are widely used in image classification. The aliased velocity can be used as the input image, and the aliased count (i.e., the number of folds) and direction can be treated as the output labels. The U-Net structure allows class segmentation and provides pixel-wise predictions, which means that predictions are performed at each pixel, and the predicted output size can be the same as the input data. Once the aliased count is predicted as the output at each range gate, velocity dealiasing can be performed using the aliased count, the input aliased velocity, and the aliasing velocity (v_{a}). Chapter 4 presents the application of CNNs to velocity dealiasing.

In the next chapter, the application of CNN to velocity dealiasing will be discussed including pre-processing, velocity dealiasing using CNN, and the training method. This will also include the variables of optimization and the results of sensitivity tests.

Chapter 4

Application of Convolutional Neural Networks to Velocity Dealiasing

In this chapter, the application of CNN to velocity dealiasing will be discussed. An overview of proposed method, pre-processing including data generation, velocity dealiasing using CNN, and training method will be explained.

4.1 Overview of Proposed Process

In this study, a novel velocity dealiasing method based on a CNN construct is proposed to tackle the velocity aliasing challenge, which is a long-standing problem in the weather radar community. Image segmentation can be performed through a CNN, where concatenated layers of filters operate like a convolution operation. CNN-based image classification can produce a single label that represents the whole image, or an output image that indicates multiple labels (segments) within an image. In this context, the detection of velocity aliasing can be compared to a segmentation (pixel classification) problem. The label indicates whether a velocity cell is aliased and, by extension, how many times the velocity is aliased. To train the supervised CNN, ground truth data are required where the raw aliased velocity field is used as input. And the output is a flag field indicating whether the velocity is aliased and how many times it is aliased. There-

fore, the detection of aliasing can be converted into a labeling problem. As mentioned earlier, once correct aliasing information is detected, dealiasing is straightforward.

In the process of designing and implementing a CNN model, a set of training data, which consists of the raw velocity fields and the corresponding ground truth, i.e., labels indicating whether a segment of velocity is aliased, is necessary for supervised learning. There are a number of options to generate ground truth data. The most traditional method is hand-dealiasing the raw aliased velocity field, which is extremely labor intensive involving hours of expert human intervention. Another option uses a longer-wavelength radar, such as the S-band NEXRAD radar velocity, as ground truth data and generates the aliased velocity field through artificial aliasing. X-band systems typically have lower v_a than S-band systems while preserving a moderate coverage, approximately 60–90 km. For example, the PX-1000 radar typically operates with a PRF = 2,000 Hz to provide $r_{\mathrm{a}} = 75$ km, and $v_{\mathrm{a}} = 15.7\ \mathrm{m\,s^{-1}}$. L_1 can occur when v is within $\pm 47.1\ \mathrm{m\,s^{-1}}$ while L_2 can occur when v is within $\pm 78.5\ \mathrm{m\,s^{-1}}$, which can sufficiently cover most velocity measurements of meteorological echoes. For these reasons, we chose the latter option for generating our data, which is used for training the CNN to identify $L = 0$ (non-aliased), $L \in [-1, 1]$ (once aliased), and $L \in [-2, 2]$ (twice aliased label).

4.2 Data Generation

As mentioned, artificial aliasing based on S-band data is performed to generate the input velocity fields for training the CNN model. Input data are the simulated X-band radar velocity field (v_{i}) from the NEXRAD S-band radar by a set of simple rules and its aliased count L, which indicates how many times the velocity is aliased. In the context

of this study, $L \in [-2, -1, 0, 1, 2]$ is produced according to the following rules:

$$L = \begin{cases} -2, & v < -3v_a \\ -1, & -3v_a \leq v < -v_a \\ 0, & -v_a \leq v < v_a \\ 1, & v_a \leq v < 3v_a \\ 2, & v > 3v_a \end{cases} \tag{4.1}$$

In short, v_{i} can be defined as shown.

$$v_i = v_t - 2v_a L \tag{4.2}$$

For example, if $v_{\mathrm{t}} = 16\ \mathrm{m\,s^{-1}}$, and $v_{\mathrm{a}} = 15\ \mathrm{m\,s^{-1}}$, then $v_{\mathrm{i}} = -14\ \mathrm{m\,s^{-1}}$ and $L = 1$ since it is once aliased and detected in the positive direction. If $v_{\mathrm{t}} = -29\ \mathrm{m\,s^{-1}}$, and $v_{\mathrm{a}} = 8\ \mathrm{m\,s^{-1}}$, then $v_{\mathrm{i}} = 3\ \mathrm{m\,s^{-1}}$ and $L = -2$ since it is twice aliased and detected in the negative direction.

The number of scans for training, validation, and test dataset is shown in Table 4.1. To diversify the datasets, scans from four different years and five different NEXRAD radar sites are collected. Furthermore, data are collected by considering the area where precipitation fills the scan. Although a qualitative categorization, the cases are separated into "mostly filled" precipitation and "sparsely filled" precipitation. Generally, these cases correspond to stratiform and convective precipitation, respectively. The training dataset consists of 1,872 scans, which are made out of 624 cases from three elevation angles, i.e., 0.5° , 0.9° , and 1.3° . It comprises 240 scans of mostly filled precipitation and 1,632 scans of sparsely filled precipitation. Unaliased velocity fields from the NEXRAD KTLX, KFWS, KICT, KLSX, and KLOT radar sites in 2018 are

Table 4.1: The number of scans for training, validation, and test dataset for mostly filled precipitation and sparsely filled precipitation

	Training	Validation	Test
Mostly filled precipitation	240	75	102
Sparsely filled precipitation	1,632	240	393
Total	1,872	315	495

employed as the training dataset. The validation dataset includes 315 scans with 75 scans of mostly filled precipitation and 240 scans of sparsely filled precipitation (105 cases with three elevation angles), and the scans from the NEXRAD KTLX radar site in 2019 are collected. The test dataset has a total of 495 scans with 102 scans of mostly filled precipitation and 393 scans of sparsely filled precipitation, which is 135 cases with three elevation angles, and the scans are collected from the NEXRAD KTLX radar site in 2017 and 2020.

The imbalanced number of scans between two precipitation types are collected to cover both mostly filled precipitation and sparsely filled precipitation. More sparsely filled precipitation scans are collected over mostly filled precipitation because their complexity is quite different. The mostly filled precipitation cases are characterized by being spatially continuous and having relatively simple features, which means the features can be trained with a lower number of scans. On the other hand, the sparsely filled precipitation is more complex and includes spatially discontinuous storms, which requires diverse training data to represent the complex features.

4.3 Methodology

In this section, pre-processing and post-processing will be discussed. The key is converting the velocity dealiasing problem to a labeling problem. Velocity dealiasing is

performed with the measured velocity fields using the labels.

4.3.1 Pre-Processing

Input data are generated by considering the PX-1000 coverage, which has an inherent limitation of the number of training data. Unlike typical machine learning that employs more than 100,000 training samples (scans), our collected training data are approximately 2,000 scans. To overcome this limitation, so-called data augmentation was performed to generate additional training data. Data augmentation is a process to diversify the data by slightly modified copies of existing data [74–77].

As mentioned in Chapter 3, typical CNN models ingest input data as width by height by depth. One can think of the width by height as the size of an image and the depth as the number of images that are related and processed simultaneously, such as the red, green, and blue components of a color image. In the context of velocity dealiasing, the depth is just one as the velocity is processed alone. The dimension D of the input array of the velocity is:

$$D = n_{\mathrm{a}} \times n_{\mathrm{r}} \times 1, \tag{4.3}$$

where n_a is the number of azimuths and n_r is the number of range gates. Since it is a labeling process, the output arrays also share the same dimension as the input.

Input and output data are generated to cover $60\ \mathrm{km}$, which is a typical range coverage of X-band radars, e.g., [9, 78, 79]. With the NEXRAD range resolution of $250\ \mathrm{m}$, 256 range gates result in a range coverage of $64\ \mathrm{km}$. The number of azimuths is resampled by a 2° spacing with 180 radials resulting in a full 360° coverage. The input array is expanded to 256 azimuths by copying the adjacent azimuths to produce a continuous input field. That is, a scan has azimuths 284, 286, ..., 358, 0, 2, 4, ..., 358, 0, 2, ..., 74. This is done instead of simply setting the first and last parts to zeros. During

the processing stage, the middle portion, i.e., 0, 2, 4, ..., 358, is extracted as the final output for velocity dealiasing.

With the collected dataset, the wind direction has a general bias as the selected NEXRAD radars are located where there is a non-zero mean of the wind in terms of climate [80]. In order to overcome this general wind bias, which could result in a CNN model to develop a bias, a data augmentation to randomly rotate the radar scans and randomly negate the velocity values is performed. Essentially, the mean wind is randomized (removed) as a set in the hopes to train a CNN model to handle velocity fields with different wind directions.

Data normalization is also applied when the model input was generated from v_{i}. There are two main purposes: one is to eliminate the bias, and the other is to map the input velocity to a range that is independent of specific v_{a} values. It is performed by dividing the v_{i} by the v_{a}.

4.3.2 Algorithm Description

With the generated input data, evaluation is performed with the trained model (the training process will be discussed in Section 4.4). A block diagram is shown in Figure 4.1. At first, input (aliased) velocity (v_{i}) passes through the model with optimized parameters. It produces the predicted label (aliased count) L_{p} as output. The model is comprised of two processes: encoding and decoding. In the encoding process, the convolutional and pooling (downsampling) layers are stacked to extract the features by convolution and downsampling the input data. In the encoding process, the azimuths and range gates of input data are reduced while the number of feature maps is increased. In a decoding process, on the other hand, the number of azimuths and range gates of input data is increased by deconvolution and up-conversion (upsampling) layers. Finally,

the softmax classifier and a one-hot coding method are applied to generate a map of L_{p} as output. Once the L_{p} is generated, velocity dealiasing is accomplished by combining the input velocity v_{i}, the predicted label L_{p}, and the Nyquist velocity v_{a} as shown in the following:

$$v_p = v_i + 2v_a L_p. \tag{4.4}$$

For convenience, three sets of labels will be used:

$$\begin{aligned} L_0 &= L \in [0] \\ L_1 &= L \in [-1, 1] \\ L_2 &= L \in [-2, -2]. \end{aligned} \tag{4.5}$$

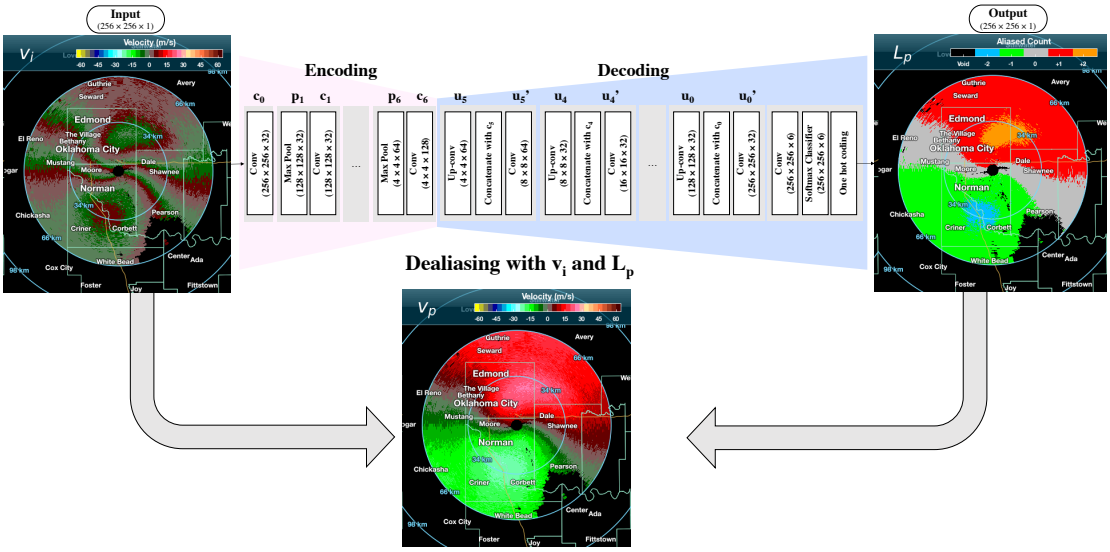


Figure 4.1: Block diagram of the proposed velocity dealiasing technique using a CNN. Velocity dealiasing is performed by combining the input (aliased) velocity (v_{i}), the aliasing count (L_{p}), and the Nyquist velocity v_{a}. v_{i} passes through the model, which consists of multiple layers of operations, i.e., convolution, pooling, softmax, and prediction. To that end, the technique produces a map that indicates whether a velocity measurement is aliased, the sign, and how many times it is aliased.

Figure 4.2 shows the velocity dealiasing process with two synthetic velocity fields, each with a different v_{a}, using a trained CNN model. In this example, the velocity field

is mostly filled with precipitation. A homogeneous wind field and wide continuous storm can be seen in this scan. The first column shows the radar reflectivity Z, and the second column is the ground truth data (v_{t}) from the S-band radar velocity field. The third column is the input (aliased) velocity (v_{i}), which is manually aliased by v_{a} using Equation (4.1). The fourth column is the predicted aliased label (L_{p}), and the last column is the dealiased velocity using Equation (4.4). In the top row, v_{i} is aliased by $v_{\mathrm{a}} = 7\ \mathrm{m\,s^{-1}}$, where L_{p} includes 0 (non-aliased), 1 (once aliased), and 2 (twice aliased). However, in bottom row, when v_{i} is aliased by $v_{\mathrm{a}} = 17\ \mathrm{m\,s^{-1}}$, L_{p} does not include 2 (twice aliased). In this simple case, regardless of different v_{a}, the velocity dealiasing results are similar to v_{t}, illustrating the efficacy of using a CNN model to correctly label the velocity aliasing count. More details and the evaluation results will be explained later in Section 4.4.

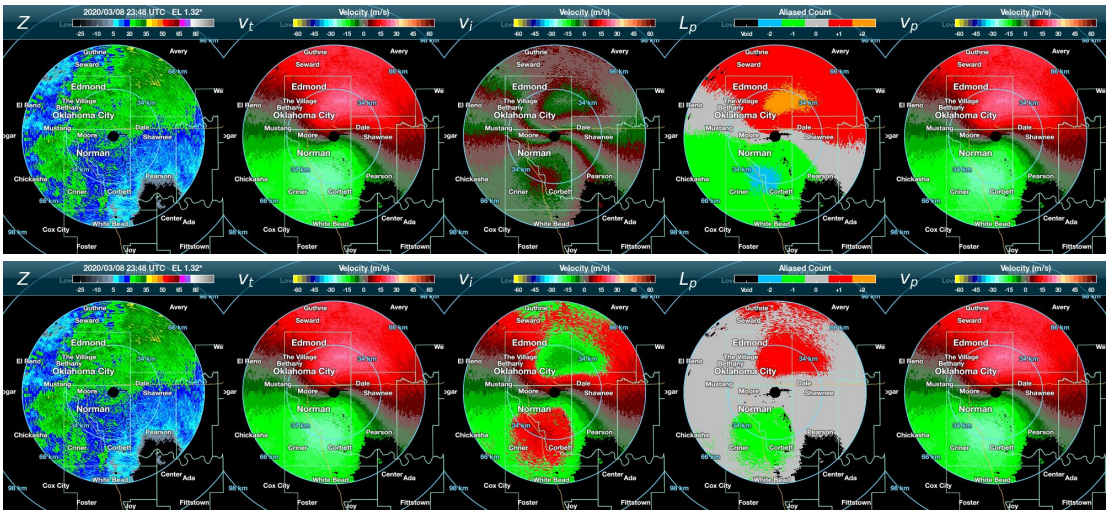


Figure 4.2: Results of the process of velocity dealiasing using the labels predicted by the CNN. In this example, the data were collected from the KTLX radar on 2020-03-08 23:48 UTC. The input velocity v_{i} is obtained by aliasing v_{t} using $v_{\mathrm{a}} = 7\ \mathrm{m\,s^{-1}}$ (top) and $v_{\mathrm{a}} = 17\ \mathrm{m\,s^{-1}}$ (bottom). Z is the radar reflectivity, and v_{p} is the dealiased velocity according to Equation (4.4)

4.4 Training

Training is performed by updating the internal model parameters, i.e., as weights and biases, to reduce the loss by using a fit-and-adjust process, which is also known as the gradient-descent algorithm. Weighted cross-entropy loss is employed as a cost function to handle the imbalances of the data. For example, when v_{a} is higher, the number of L_1 and L_2 are smaller while the number of L_0 is larger. Weighted cross-entropy is defined as the multiplication of the class weight and the cross-entropy of each label, where the class weight is defined as the inverse of the population ratio. The model parameters are trained by reducing the loss of the more populated labels and, hence, the less populated labels may not be well-trained. Intuitively, cross-entropy measures the dissimilarity between the output label and the true label, which is known as the ground truth data [81]. The cost function (loss) is decreasing during the training process and it is set to stop at convergence, which is defined as the successive change of the performance that is less than a preset threshold. As output, it produces the aliased label L_{p}.

Various training variables such as different v_{a}, different template sizes T, and the different number of layers can be optimized, and these will impact the performance of the trained model.

4.4.1 Variables of Optimization

In a training process, there are model and training variables. Different combinations of variables produce different results. The model hyperparameters are empirically determined. They include kernel sizes, learning rate, and training variables such as v_{a} and T. Training variables were determined through sensitivity tests and will be discussed in detail later in Section 4.4.2. Through a series of trial and error, the utilized model hyperparameters are kernel size of three by three, and a learning rate of 0.001. A total

of 32 layers are used, which are comprised of the encoding part (seven convolutional layers and six pooling layers) and the decoding part (seven convolutional layers, six up-convolutional layers, and six concatenated layers).

The purpose of training is to force the model to learn the aliasing concept rather than to remember specific scans from a given v_{a} value. For doing that, training is performed by combining the multiple v_{a} values and is then compared to the training with single v_{a} to determine similarities. The single v_{a} training is executed with two different v_{a}, i.e., $v_{\mathrm{a}} = 7\ \mathrm{m\,s^{-1}}$ and $12\ \mathrm{m\,s^{-1}}$. Since we limit the maximum v_{t} to $33\ \mathrm{m\,s^{-1}}$, artificial aliasing with $v_{\mathrm{a}} = 7\ \mathrm{m\,s^{-1}}$ includes L_2. Also, $v_{\mathrm{a}} = 12\ \mathrm{m\,s^{-1}}$ alone was used since it has the highest once-aliased population when maximum v_{t} is limited to $33\ \mathrm{m\,s^{-1}}$, which is the raw v_{a} of the NEXRAD S-band radar. However, this model cannot predict L_2 since this label is not included in the training. Three different combined v_{a} methods are trained. One is performed by combining $v_{\mathrm{a}} = 7\ \mathrm{m\,s^{-1}}$ and $12\ \mathrm{m\,s^{-1}}$, which is named $v_{\mathrm{a}} \in [7, 12]$. The second option, which is named $v_{\mathrm{a}} \in [\nu]$, is a training with v_{a} set to ν, which is a random variable that has a uniform distribution in between 7 and $23\ \mathrm{m\,s^{-1}}$.

During the training process, the population of a label dictates the number of adjustments that are made to identify that particular label. Logically, the labels with the highest counts would be fitted the best. However, the goal is to achieve similar performance among all labels. One way to overcome this imbalance is by using a so-called class weight, which is inversely proportional to the population ratio of the labels, effectively undoing the imbalanced adjustments caused by the population distribution. Panel (a) of Figure 4.3 shows the distributions of labels on $v_{\mathrm{a}} \in [7, 9]$, panel (b) is on $v_{\mathrm{a}} \in [11, 13]$, and panel (c) is on $v_{\mathrm{a}} \in [21, 23]$. Training v_{a} is limited to $23\ \mathrm{m\,s^{-1}}$ to avoid becoming highly biased toward identifying L_0 and the performance of identifying L_1 is penalized. Since the class weight of this dataset is $1 : 6.17 : 555$, the training could be highly bi-

ased toward L_2, which has a lower population. Focusing toward identifying L_2 results in poorer performance of identifying L_0 and L_1. The class weight helps equalize the skewed distribution. However, when the population is extremely skewed, it could lead to an overfitting issue. For example, if the input data has 500 L_0 and only one L_1, the class weight $L_0 : L_1 = 1 : 500$. The model would be trained to fit the one L_1 equally as the other 500 L_0, resulting in a net loss, i.e., negatively impacting the performance of identifying L_0.

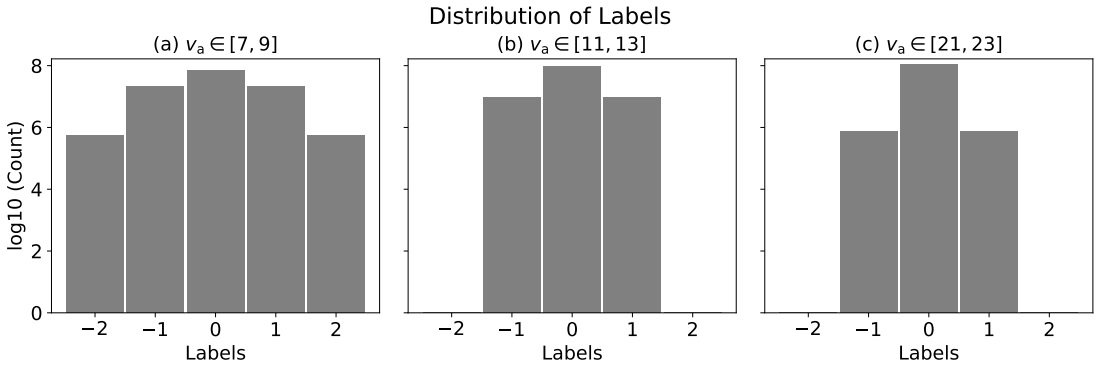


Figure 4.3: This figure is the distribution of labels with different v_a in logarithmic scale: Panel (**a**) represents the distribution of labels on $v_\mathrm{a} \in [7, 9]$; panel (**b**) shows on $v_\mathrm{a} \in [11, 13]$; and panel (**c**) shows on $v_\mathrm{a} \in [21, 23]$. When the v_a used is higher (left panel to the right panel), the distribution is more skewed to the L_0.

Therefore, another v_a random training method, $v_\mathrm{a} \in [7, \nu]$, is introduced to include the more collected L_2 data instead of increasing class weights. It is performed by fixing $v_\mathrm{a} = 7\ \mathrm{m\,s^{-1}}$ and selecting one other random v_a (ν) between $v_\mathrm{a} = 8$ and $23\ \mathrm{m\,s^{-1}}$. In other words, two sets of data are trained together. One set is trained with $v_\mathrm{a} = 7\ \mathrm{m\,s^{-1}}$, and another set is trained with one random v_a between $v_\mathrm{a} = 8$ and $23\ \mathrm{m\,s^{-1}}$. The class weight of this method is set as $1 : 2.61 : 77.0$, where the class weight of the twice-aliased label is reduced.

As mentioned before, the template size T is defined as the number of azimuths n_a by range gates n_r. For simplicity, $n_\mathrm{a} = 256$ is fixed, and four different n_r are trained and

evaluated by concatenating them to generate the 256 range gates. The trained number of range gates are described as: 32 range gates cover the $8\ \mathrm{km}$, 64 range gates cover the $16\ \mathrm{km}$, 128 range gates cover the $32\ \mathrm{km}$, and 256 range gates cover the $64\ \mathrm{km}$.

4.4.2 Sensitivity Tests

Training variables are evaluated on v_a in three configurations: G_1, G_2, and G_3. G_1 is the combined performance of $v_\mathrm{a} = 7$ and $9\ \mathrm{m\,s^{-1}}$, G_2 is the combined performance of $v_\mathrm{a} = 11$ and $13\ \mathrm{m\,s^{-1}}$, and G_3 is the combined performance of $v_\mathrm{a} = 21$ and $23\ \mathrm{m\,s^{-1}}$.

In G_1, the population ratio of $L_0 : L_1 : L_2$ is equal to $110 : 48 : 1$, and the corresponding class weight ratio is $1 : 2.28 : 110$. In G_2, $L_0 : L_1 = 6.89 : 1$, and the corresponding class weight is the inverse of the population ratio, i.e., $1 : 6.89$. In G_3, $L_0 : L_1 = 119 : 1$, and the corresponding class weight ratio is $1 : 119$.

Figure 4.4 shows the performance with different training v_a and evaluation using G_1, G_2, and G_3. In G_1, $v_\mathrm{a} = 7$ and $v_\mathrm{a} \in [7, \nu]$ show the highest performance among five different models since both include the $v_\mathrm{a} = 7\ \mathrm{m\,s^{-1}}$ in training, which has the largest number of aliased labels (L_1 and L_2) in training. However, in G_3, which includes the least aliased label since it has the highest v_a, training with $v_\mathrm{a} \in [7, \nu]$ shows the lower σ_A than $v_\mathrm{a} = 7$ model. $v_\mathrm{a} \in \nu$ model shows a relatively poor performance than others for its extremely skewed class weights ($1 : 6.17 : 555$), especially on L_2. The raw population of L_2 is deficient. Therefore, it diminishes the non-aliased and once-aliased training performance by highly focusing the model parameters optimization on the twice-aliased label. From this experiment, $v_\mathrm{a} \in [7, \nu]$ model is chosen for our final training v_a condition for not to be biased toward one specific v_a.

Figure 4.5 shows the μ_A and σ_A (scan average) as a function of T. The left panels provide results on the mostly filled precipitation scans and the right panels correspond

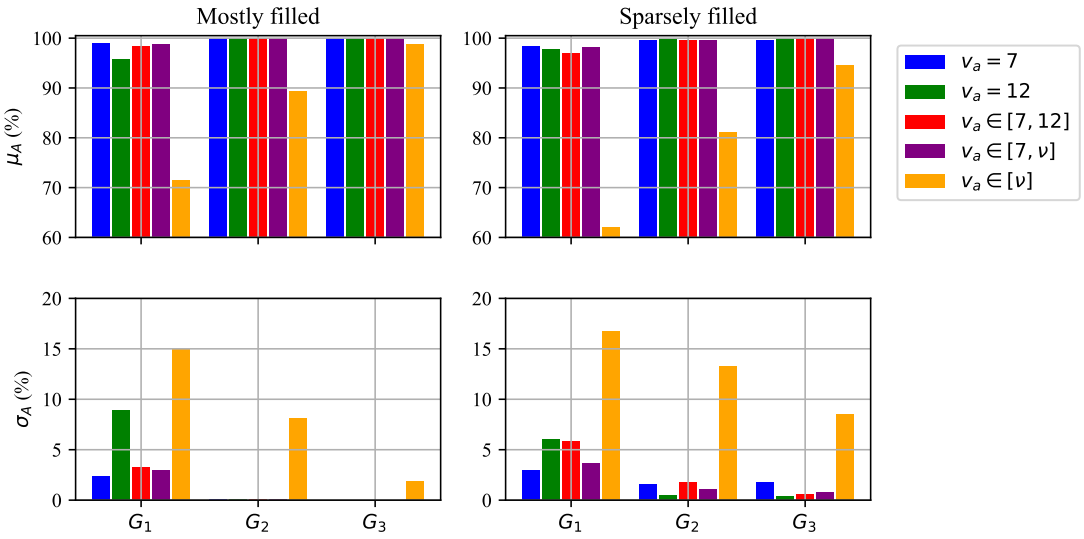


Figure 4.4: Comparisons of the trained CNN with different v_a, i.e., $v_\mathrm{a} = 7\ \mathrm{m\,s^{-1}}$ (blue) $v_\mathrm{a} = 12\ \mathrm{m\,s^{-1}}$ (green), $v_\mathrm{a} \in [7, 12]$ (red), $v_\mathrm{a} \in [7, \nu]$ (purple), and $v_\mathrm{a} \in [\nu]$ (orange) for each on the mostly filled precipitation (left), and the sparsely filled precipitation (right). μ_A is the scan averaged accuracy (top) and σ_A is the scan averaged standard deviation (bottom). It is evaluated with the three different v_a groups G_1, G_2, and G_3.

to the sparsely filled precipitation scans. In general, using a larger T produces better results (higher μ_A and lower σ_A), with the only exception on mostly filled precipitation with $T = 128$ and $T = 256$, which are in reverse order, but the difference is less than 0.5%. In G_2 and G_3, all four different template sizes show similar performances. This is because the mostly filled precipitation scans are spatially continuous and have relatively simple features in contrast to the sparsely filled precipitation. It is noteworthy that training with $T = 32$, which is relatively short-range coverage, also shows high performance since it still has 360° coverage and a more homogeneous wind field, which makes it easier to predict the aliased label. Mostly filled precipitation is less impacted by template size since these are spatially continuous and mostly filled. In contrast, more spatially complicated cases can be negatively impacted by template size. A larger template size covers a wider area, and it is beneficial for predicting the aliased label L.

Figure 4.6 shows the performance of the trained model using different template

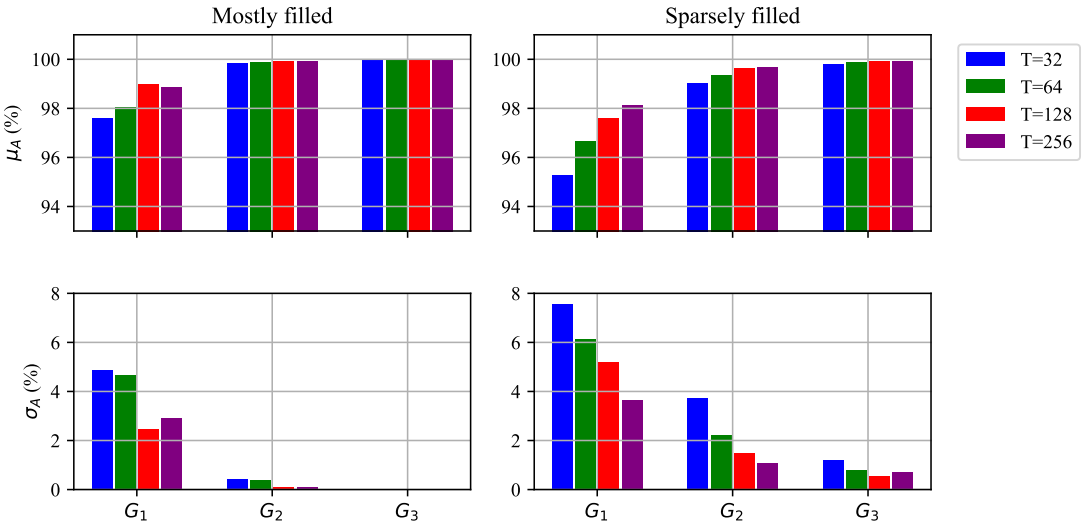


Figure 4.5: Comparisons of the performance of the trained CNN with different template sizes (T), i.e., 32 (blue), 64 (green), 128 (red), and 256 (purple) range gates on mostly filled precipitation (left), and sparsely filled precipitation (right). It is tested with three different v_{a} group, i.e., G_1, G_2, and G_3.

sizes as a function of range. The top panels are the μ_A (%) on the mostly filled precipitation scans, and the bottom panels are the μ_A (%) on the sparsely filled precipitation scans. The evaluation results for G_1, G_2, and G_3 are provided in the left, middle, and right panels, respectively. For the mostly filled precipitation, in G_1, in range gates 0–127, all four template sizes show similar performances. However, in range gates 128–256, the performances are shown in this order: $\mu_A(128) > \mu_A(256) > \mu_A(64) > \mu_A(32)$. In G_2 and G_3, all four template sizes show the similar performances. For the sparsely filled precipitation, in G_1, the performance is shown in this order: $\mu_A(256) > \mu_A(128) > \mu_A(64) > \mu_A(32)$. It is noteworthy that the performance reduction is shown at each template boundary, unlike the mostly filled precipitation result. The sparsely filled precipitation scans include the non-uniform wind field, spatially discontinuous, and isolated storms. For these scans, more information would certainly help determine the dealiasing decisions. $T = 128$ and $T = 256$ are similarly

performed, however, $T = 128$ shows the boundary reduction on its template boundary and $T = 256$ can be done in one prediction to cover the 64 km while $T = 128$ needs two prediction. In G_2, both $T = 256$ and $T = 128$ exhibit similar best performance, followed by $T = 64$, and $T = 32$. Similar to G_1, there is a reduction in performance at the template boundary of $T = 128$, while such reduction is not observed in $T = 256$. In G_3, all four template sizes show similar performances. For both mostly filled and sparsely filled precipitation scans, the improved performance is observed as tendency when evaluation v_a is higher.

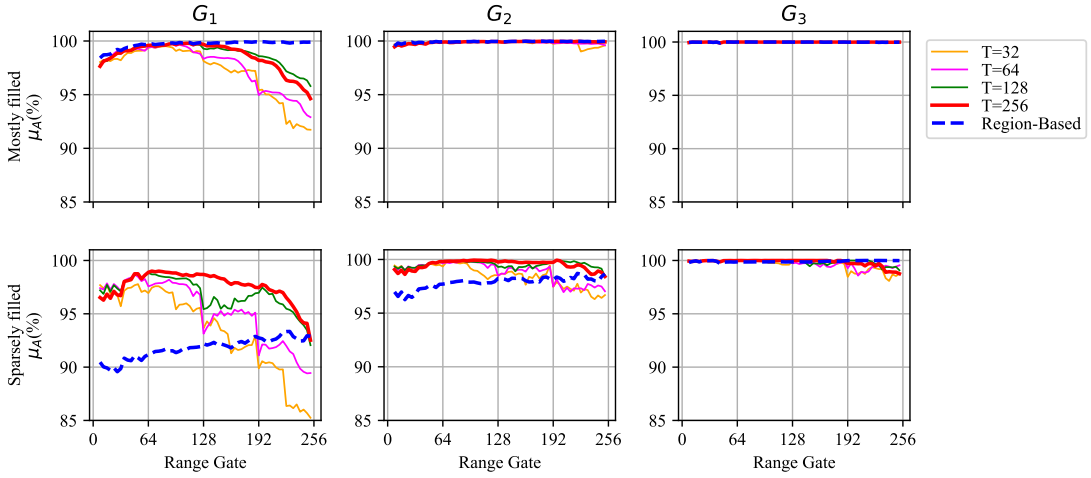


Figure 4.6: Performance of the CNN algorithm as a function of range with the different T, i.e., 32 (orange), 64 (magenta), 128 (green), and 256 (red) range gates. It is also compared to the conventional region-based dealiasing method (blue dashed line). The first row is μ_A in percentage averaged by the number of scans for the mostly filled precipitation. The second row is also the μ_A but for the sparsely filled precipitation scans. It is analyzed with groups G_1 (left), G_2 (center), and G_3 (right).

In Figure 4.6, velocity dealiasing results with different template sizes as a function of the range are compared against the more traditional region-based method [26]. The region-based method has the option to utilize the environmental background wind for referencing the first-guess field velocity estimation. With the goal of a fair comparison between the CNN and region-based methods, no environmental background

wind is used. With the mostly filled precipitation scans, the region-based dealiasing method properly dealiases the velocity and shows stable performance with range compared to the CNN method. However, with the sparsely filled precipitation scans, the region-based dealiasing method performs poorly at the initial range gate compared to the $T = 256$ model, and it does not show any measurable performance reduction at far ranges, while the CNN method shows significant performance reduction. In the region-based method, estimating the velocity of the first-guess field is important. If it correctly predicts the velocity of the first-guess field, the performance is consistent along the range. However, if it fails to estimate the velocity aliasing of the first-guess field, it leads to failing the aliasing prediction of the entire storm cell. For sparsely filled precipitation, which includes multiple isolated storms, a larger number of first-guess field predictions is required than in a single-storm case. When the storm is isolated and far from the radar, it is much more difficult to estimate the first-guess field even for a human-expert implementation. For the higher v_{a} groups (G_2 and G_3), overall performance is gradually improved and the performance differences among different template sizes are also reduced. Although class weight helps equalize the less populous labels to be trained by weighting them higher, the distribution of the labels in the evaluation sets is different. That is, G_3 contains more L_0 than G_2 and G_1. Therefore, evaluation using G_3 results in higher overall performance than G_2 and G_1. The same explanations can be applied to comparisons between G_1 and G_2.

In the next chapter, the proposed velocity dealiasing algorithm will be quantitatively compared against the region-based method. The comparison is performed with final parameters such as $T = 256$ and $v_{\mathrm{a}} \in [7, \nu]$ and evaluated with μ_A and σ_A. Qualitative analyses will be performed with a large set of radar scans and error analysis and discussion will be conducted.

Chapter 5

Results and Discussion

In this chapter, the final training parameters are determined through a sensitivity test. The evaluation will be performed using these parameters, and the statistical results will then be analyzed.

5.1 Evaluation Method and Metrics

The training parameters are determined empirically through the sensitivity tests in Chapter 4. To ensure sufficient data coverage, a template size (T) of 256 (azimuth gates) $\times$ 256 (range gates) is selected. For each iteration process, combined v_{a}, which is $7\,\mathrm{m\,s^{-1}}$ and one random v_{a} between $v_{\mathrm{a}} = 8$ and $23\,\mathrm{m\,s^{-1}}$, is used. The training class weight is used as $1 : 2.61 : 77.0$, which is also empirically determined.

Evaluation is performed with *synthesized* X-band radar velocity field based on data from the NEXRAD S-band radar, which provides readily available ground truth data. However, false labels can persist even after applying filters to mitigate clutter and undesirable echoes, such as planes and biological echoes with high velocity, which can negatively affect the evaluation of meteorological echoes. Occasionally, the CNN output appears more meteorologically "natural" than the "ground truth" data from NEXRAD S-band radar. This is because the so-called ground truth data can be contaminated by

non-meteorological echoes (e.g., ground clutter, low-SNR clear-air echoes, aircraft), as the training process is performed with the non-masked data. In our experience, most humans would prefer the CNN output compared to ground truth data from NEXRAD S-band radar velocity field as the CNN output is spatially continuous and has less speckle noise. Unlike the training process, the evaluation is performed only on the masked area of precipitation. The use of non-meteorological echoes including clear air data could be useful for training on complicated storms, such as sparsely filled precipitation, as it provides wider coverage since CNN is less prone to noise or bad pixels. However, as explained above, non-meteorological echoes or speckles from false labels are not considered during the evaluation process, as our primary interest is only precipitation. For a fair comparison, the following common masking conditions were applied on both CNN and region-based methods: Hydrometeor Classification Algorithm (HCA) is used to exclude non-precipitation echoes, including biological echoes, the residue of ground clutters, and clear echoes. SNR thresholding is used to filter poor-quality data, and reflectivity mask is utilized to filter data outside of $0 < Z < 80\,\mathrm{dBZ}$, which is pre-applied as part of the algorithm for region-based method.

During the training process, all scans are trained together. However, for evaluation, the data are split into two categories based on the amount of precipitation filling: mostly filled and sparsely filled precipitation, where the classification is manually performed. For evaluation, the test data are further divided into three groups called G_1, G_2, and G_3. It is based on the v_a used since it is challenging to see the impact of less-populated labels when analyzing all labels simultaneously, particularly when the v_a is higher. G_1 includes the performance of $v_\mathrm{a} = 7$ and $9\ \mathrm{m\,s^{-1}}$, G_2 includes $v_\mathrm{a} = 11$ and $13\,\mathrm{m\,s^{-1}}$, while G_3 comprises $v_\mathrm{a} = 21$ and $23\,\mathrm{m\,s^{-1}}$. The separation of groups is based on label population, where G_1 includes L_2, G_2 has a higher proportion of L_1, and it does not include L_2, and G_3, which are mainly includes L_0. The label population ratio is

Table 5.1: The population ratio of each label (L_1, L_2, and L_3) of three different evaluation groups: G_1, G_2, and G_3.

	L_0	L_1	L_2
G_1	109	47.8	1
G_2	6.89	1	0
G_3	119	1	0

presented in Table 5.1.

To evaluate performance, μ_A and σ_A are measured, where A represents the accuracy of a single scan. The accuracy is calculated as the ratio of the number of correctly predicted cells to the total valid number of cells in one scan. μ_A is the mean A averaged by the number of scans, while σ_A represents the scan-averaged standard deviation of A to check how the accuracy varies and is dispersed. In addition, since the overall performance could mitigate the performance on the less-populated label, the accuracy for each label (L_0, L_1, and L_2) is also calculated.

5.2 Experimental Results

As explained in Chapter 2, the region-based method has the option to utilize the environmental background wind to aid the first-guess field; however, in this study, environmental background wind is not utilized in order to provide a fair comparison. In Figure 5.1, μ_A and σ_A of CNN and the region-based dealiasing methods from each v_{a} group are compared. In mostly filled precipitation scans, in G_1, the CNN method has the lower μ_A and higher σ_A than the region-based method. In G_2 and G_3, it shows the similar performance on both methods of μ_A and σ_A. However, in sparsely filled precipitation scans, in G_1 and G_2, the CNN method has a higher μ_A and lower σ_A than the region-based method. In G_3, μ_A is similar on both methods, but the σ_A is still lower on

CNN than the region-based method. Typical X-band radars that are set up to provide a 60-km coverage ($r_{\mathrm{a}} = 60$ km) have a v_{a} at approximately $15\ \mathrm{m\,s^{-1}}$. For that configuration, performance on G_2 is most representative. It includes a reasonable amount of L_1 and shows high performance on both mostly filled and sparsely filled precipitation scans.

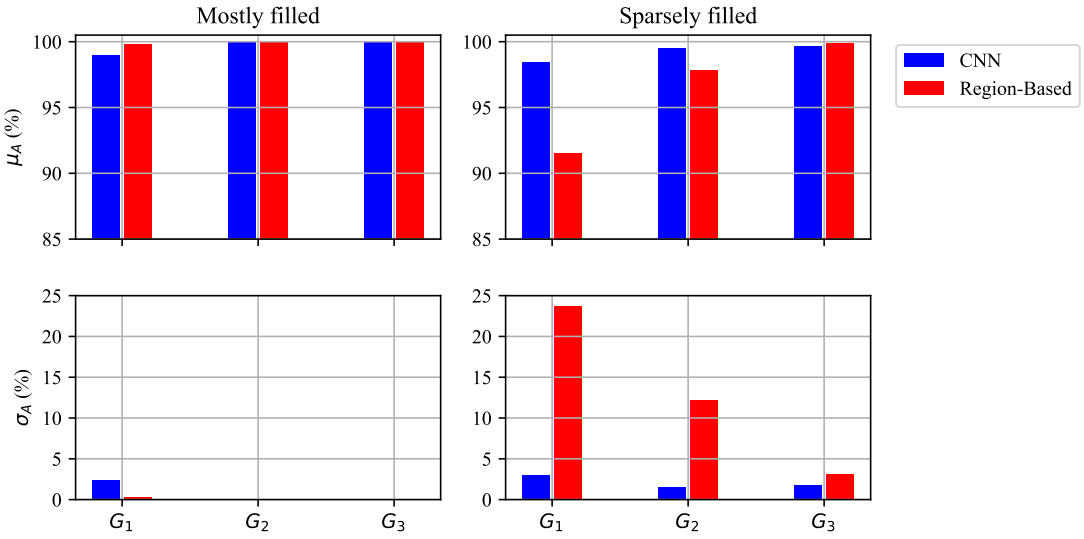


Figure 5.1: Comparison results on velocity dealiasing performance between the proposed CNN method (blue) and the conventional region-based dealiasing method (red). Comparison is performed with mean accuracy (μ_A) (top) and the standard deviation (σ_A) (bottom) in percentage. It is analyzed with the three different v_{a} groups, i.e., G_1, G_2, and G_3. The first column is for the mostly filled precipitation, and the second column is for the sparsely filled precipitation.

In Table 5.2, the overall performances, which is the weighted sum of G_1, G_2, and G_3, are calculated with Equation (5.1) on each aliased label, i.e., non-aliased (L_0), once-aliased (L_1) and twice-aliased (L_2).

$$\Pi_A = \sum_i \sum_{l \in L} w^{(G_i,l)} \mu_A^{(G_i,l)}, \tag{5.1}$$

where $w(G_i, l)$ represents the weight of labels in group G_i and set l, which are iterated

Table 5.2: Comparisons of the total performance metrics between the proposed CNN method and the conventional region-based dealiasing method. Total performance metrics are calculated by multiplying the different v_{a} group performance metrics and the corresponding weights.

		Π_A	L_0	L_1	L_2
Mostly filled	CNN	99.58	99.62	97.89	93.44
	Region-Based	99.90	99.87	98.54	94.30
	Difference	-0.32	-0.25	-0.65	-0.86
Sparsely filled	CNN	99.23	98.39	96.85	84.96
	Region-Based	96.39	96.88	91.45	74.39
	Difference	2.84	1.51	5.39	10.57

Table 5.3: Weights of the different v_{a} groups, i.e., G_1, G_2, and G_3 for L_0, L_1, and L_2. The weight for each group is calculated by the inverse of the number of labels for each evaluation metric.

	v_{a} Group	L_0	L_1	L_2
Mostly filled	$G1$	0.25	0.61	1.00
	$G2$	0.33	0.35	-
	$G3$	0.42	0.04	-
Sparsely filled	$G1$	0.28	0.74	1.00
	$G2$	0.35	0.25	-
	$G3$	0.38	0.01	-

through all groups (G_1, G_2, and G_3) and all label sets (L_0, L_1, and L_2).

The weights, which show the population ratio of each group based on the v_{a} used, are shown in Table 5.3, where the sum of the weights is '1' for each label, and the overall performance of the CNN and region-based methods are compared against each other.

For the mostly filled precipitation scans, both methods achieve similarly high Π_A ($>$99%), with the region-based method exhibiting slightly better performance. The performance difference between the two methods is similar with a discrepancy of less

than 1% for each label. For the sparsely filled precipitation scans, which has a higher complexity than the mostly filled precipitation scans, the CNN method achieves higher Π_A compared to the region-based method. The overall accuracy Π_A for both methods is higher than the group specific accuracy ($\mu_A(G = G_1)$) since Π_A is derived with more elements in L_0 as the set includes groups G_2 and G_3. The overall performance for the three aliasing labels is in the order of $\Pi_A(L = L_0) > \Pi_A(L = L_1) > \Pi_A(L = L_2)$, meaning that the CNN model is more effective in identifying non-aliased regions, followed by once-aliased regions, and then twice-aliased regions. Identifying twice-aliased regions requires correct identification of once-aliased regions that are adjacent. As such, it is not surprising that $\Pi_A(L = L_2)$ is lower than $\Pi_A(L = L_1)$. The same reasoning can be applied to $\Pi_A(L = L_0) > \Pi_A(L = L_1)$.

Case studies are now conducted to demonstrate the effectiveness of two dealiasing methods under conditions with mostly filled and sparsely filled precipitation scans. The studies are conducted using NEXRAD S-band radar velocity data and PX-1000 X-band radar velocity data.

The image displayed in Figure 5.2 presents an example of a PPI scan with mostly filled precipitation, as shown in Z. The evaluated v_{a} is 7 $\mathrm{m\,s^{-1}}$, which includes L_0, L_1, and L_2. The wind field is spatially continuous, which can be seen in ground truth v_{t}. Both CNN and region-based approaches are able to dealias the velocity in v_{i} with a precision of over 99%, since the storm is extensive and spatially continuous, which contains sufficient information to determine aliasing also for a human. Such storms are typical for widespread stratiform precipitation events. The CNN and region-based method results are shown in v_{p} and v_{c} for each.

Figure 5.3 provides an example of how a CNN method can be used to dealias sparsely filled precipitation scan, which is indicative of convective storms. This case is also synthesized with 7 $\mathrm{m\,s^{-1}}$ of v_{a} and it contains multiple isolated storms, as shown

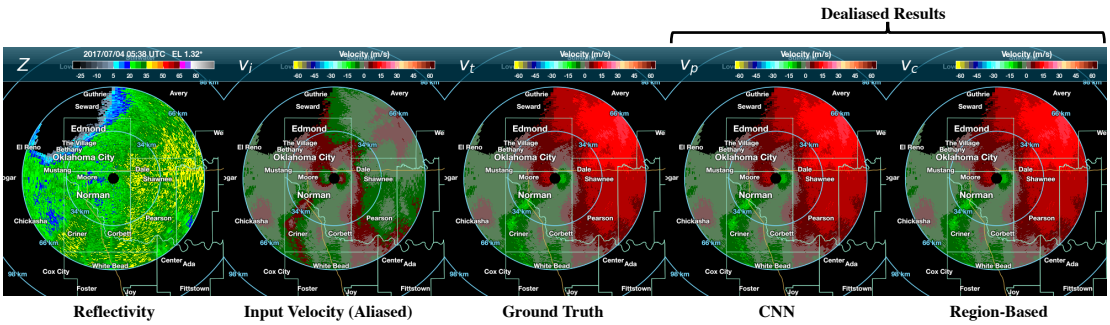


Figure 5.2: An example PPI scan with a mostly filled precipitation. Z is the reflectivity, v_{i} is the input velocity, v_{t} is the ground truth, v_{p} is the dealiased velocity using the *predicted* aliased label from the CNN, and v_{c} is the dealiased velocity using the conventional region-based dealiasing method. The data are synthesized using a 1.32° -EL scan from the KTLX on 4 July 2017 05:38 UTC. This example shows the result of processing a velocity field observed at 7 $\mathrm{m\,s^{-1}}$. For most simple cases such as this, both methods are able to produce an accurate dealiased velocity field. This example shows over 99% accuracy from both CNN and region-based methods.

in Z and v_{t}. In v_{i}, the wind field is discontinuous, and extremely challenging to distinguish the aliased area even for a human. The CNN method (v_{p}) is able to successfully dealias most of the isolated storms, whereas the region-based method fails at multiple isolated storms. It is because the region-based method assumes that the first-guess field is non-aliased, and it leads to failure to decide the aliasing of the whole isolated storm in case of failure estimating the first-guess field, which is shown in the yellow circle of v_{c}. The assumption that the first-guess field is non-aliased can be problematic, as illustrated in this example. The CNN model has a wider view than the region-based method, which aids in aliasing decision-making.

In the domain of CNN processing, there is a notion of the receptive field, which is the region that each particular CNN layer is looking at [82]. The receptive field in a CNN is a two-dimensional processing window that is created by multiple layers of convolution. As one would expect, more successive convolution results in a wider processing region. With the CNN model that processes each radar cell through multiple

layers, the receptive field is wide. In the proposed CNN architecture, the receptive field includes the whole PPI, which is the whole radar coverage. This wide view enables the model to comprehend the overall structure of a storm and identify areas of aliasing, similar to how a human would perceive it. By training the CNN to identify large-scale features through this wide view, it becomes capable of identifying patterns that are not apparent when considering only individual radar cells.

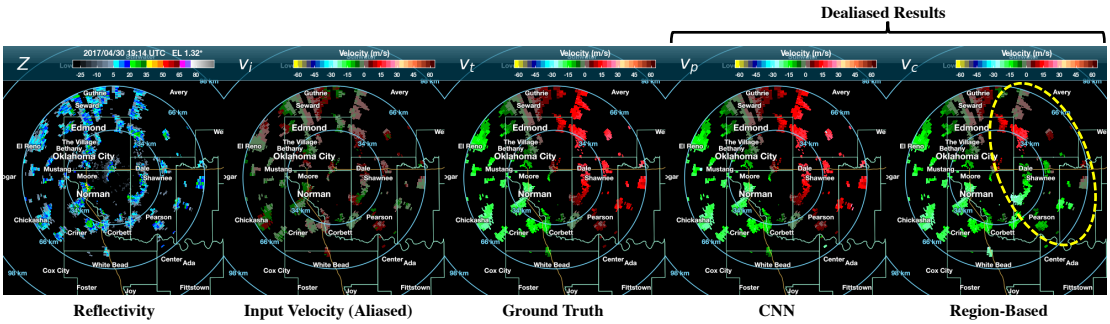


Figure 5.3: This figure shows an example PPI scan for isolated storms observed at $v_{\mathrm{a}} = 7\,\mathrm{m\,s^{-1}}$. The data are synthesized using a 1.32°-EL scan from the KTLX on 30 April 2017 19:14 UTC. The CNN method successfully dealiased the scan as it processed the entire scan all at once. The region-based method, however, failed at a number of isolated storms, which are indicated in the yellow circle. In this example, CNN method predicts the 99.5% on L_0, 99.4% on L_1, and 100% on L_2, while the region-based method predicts 77.9%, 67.8%, and 84.4% on L_0, L_1, and L_2 for each.

Another example of an isolated storm PPI scan synthesized from the NEXRAD S-band radar is presented in Figure 5.4. It also contains multiple isolated storms, which are shown in Z. In v_{i}, an isolated storm in the white dashed circle is aliased. As with the previous example, the dealiased velocity using CNN (v_{p}) is similar to the ground truth v_{t}. However, the region-based method failed to perform the velocity dealiasing at the same location, as shown in v_{c}.

Figure 5.5 shows the dealiasing result obtained from the PX-1000 X-band radar velocity, which is the target of this study. In this example, v_{a} is set to 15.7 $\mathrm{m\,s^{-1}}$ and the scan contains two large storms. The bottom isolated storm includes the aliased

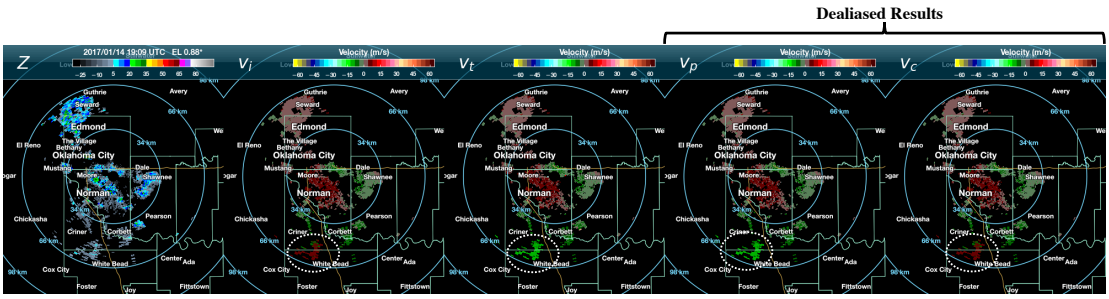


Figure 5.4: This figure shows an example PPI scan with isolated storms observed at $v_{\mathrm{a}} = 8\,\mathrm{m\,s^{-1}}$. The data are synthesized using a 0.88°-EL scan from the KTLX on 14 Jan 2017 19:09 UTC. The CNN method successfully dealias with the velocity, it is similar to v_{t}. The region-based method, however, failed to dealias the velocity in a white dashed circle.

velocity in v_{i}, whereas the top storm does not. With the proposed CNN method, the aliased storm is successfully dealiased as shown in v_{p} with the precision of 99.6%, 92.7% for each L_0 and L_1. L_2 does not exist in this case. In contrast, the region-based method inappropriately dealiases the entire bottom isolated storm, as shown in dealiased velocity (v_{c}). This failure can be attributed to the inaccurate estimation of the first-guess field, as explained above in the failed isolated storm dealiasing.

One key feature of this study is that the proposed model is trained using X-band radar velocity data that were derived from S-band data. The results show the potential of applying CNN unfolding technique on X-band radar velocity fields for wide and spatially continuous storms. This is because the model was trained on features extracted from wide and spatially continuous storm scans, which enables the model to learn the features from the data regardless of different range resolutions. However, the feature of small-scale and rapidly changing storms can be different depending on the range resolution. Therefore, training the small-scale and rapidly changing storm features would help to improve the performance to cover that features.

Another PX-1000 example is shown in Figure 5.6. As shown in Z, this example

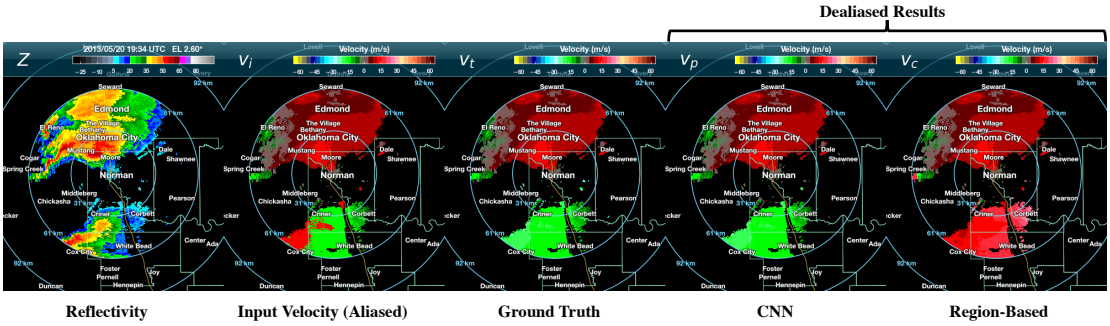


Figure 5.5: Velocity dealiasing PPI result observed with PX-1000 at a 2.6° -EL on 20 May 2013 19:34 UTC . Z is the reflectivity, v_{i} is the input velocity, v_{t} is the true velocity which is used as ground truths, v_{p} is the dealiased velocity with proposed CNN method, and v_{c} is the dealiased velocity with traditional region-based dealiasing method.

includes a "hook echo," which is indicative of a mesocyclone with an embedded tornado. Compared to v_{t}, while both velocity dealiasing methods successfully dealias the velocity on wide and spatially continuous storms, as shown in an orange dashed circle at the bottom part of the PPI scan (v_{p} and v_{c}), there is incorrect dealiasing in the white dashed circle on both CNN and region-based methods. To examine the features of this signal in more detail, an enlarged figure is shown in Figure 5.7. In v_{i}, there are three aliasing parts: one is from the bottom wide and spatially continuous storm, one is in the yellow dashed circle, which is also part of a wide and continuous storm, and another is in the white dashed circle, which is the hook echo and an important tornadic signal. The bottom storm and the storm in the yellow dashed circle seems correctly dealiased on both methods. However, in the white dashed circle, the CNN method failed to predict the aliasing label because this feature is not sufficiently trained with S-band NEXRAD data. The region-based method also failed to dealias the velocity because the velocity of this storm is rapidly changed, which could also be challenging for the continuity checking method, even though it is able to correctly estimate the first-guess field aliasing. Therefore, further training with X-band radar data will be expected to handle small-scale and rapidly changing echoes that may not be captured with S-band

data.

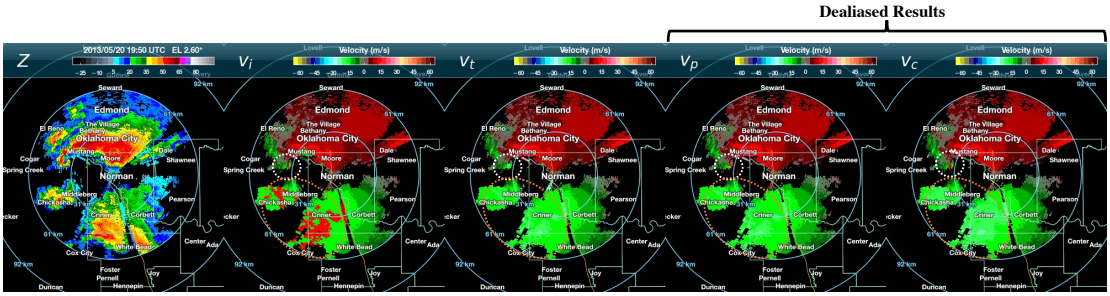


Figure 5.6: Velocity dealiasing PPI result observed with PX-1000 at a 2.6° -EL on 20 May 2013 19:50 UTC.

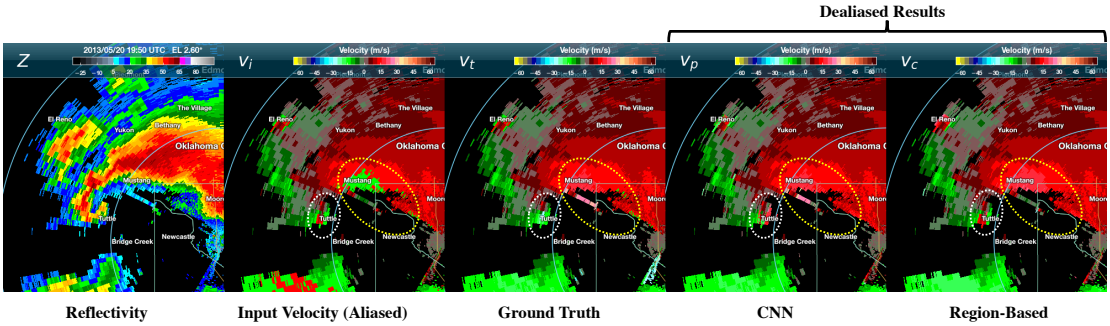


Figure 5.7: Enlarged velocity dealiasing PPI result observed with PX-1000 at a 2.6° -EL on 20 May 2013 19:50 UTC.

Figure 5.8 shows another example of a PPI scan obtained from the PX-1000 X-band radar, without ground truth (v_{t}). As previously mentioned, the availability of ground truth is significantly important for supervised training. Additionally, it is also important in the evaluation of the algorithm's performance because the lack of ground truth limits quantitative evaluation. This scan is the velocity dealiasing PPI result observed with PX-1000 at a 4° -EL on 05 May 2022 00:09 UTC. In this scan, the v_{a} of 11.6 $\mathrm{m\,s^{-1}}$ is utilized.

In Z, multiple isolated storms are visible, with the convective storm at the center. In v_{i}, the aliased isolated storms and aliased area inside the center convective storm can

be observed. Since ground truth is not available, the performance evaluation is limited. However, the inter-comparison of v_{p} and v_{c} shows that v_{c} still includes aliasing parts after velocity dealiasing on the white and yellow dashed circle, unlike the v_{p}, which shows a consistent pattern. In v_{c}, the isolated storms inside the yellow dashed circle cannot be evaluated without ground truth since the velocity of the entire isolated storm is continuous. However, the storm in the white dashed circle shows a discontinuity in the dealiasing result. The storm in the white dashed circle of Z has light precipitation, far away from the radar center, and therefore, it seems to have a low SNR, and signals are unstable and discontinuous, resembling interference. This indicates that the continuity-checking method can be failed inside the storm when the signal is unstable, not just for the incorrect estimation of the first-guess field.

The lack of ground truth is the main concern when using the CNN method. In terms of future work, there are two considerations. One approach is to manually generate the ground truth by hand-dealiasing, but this would require a significant amount of human labor since the training dataset requires a large amount of data. Another consideration is semi-supervised learning, as mentioned in Chapter 3. Training small-scale and rapidly changing signals such as tornadoes is challenging with S-band radars since even these longer-wavelength radars can be aliased. The semi-supervised learning enables the model to learn with the combined datasets of labeled and unlabeled data at the same time. Therefore, improved performance is expected with training combined dataset with labeled data, which are synthesized X-band data from S-band radar data, and unlabeled data, which are X-band radar data from X-band radar system.

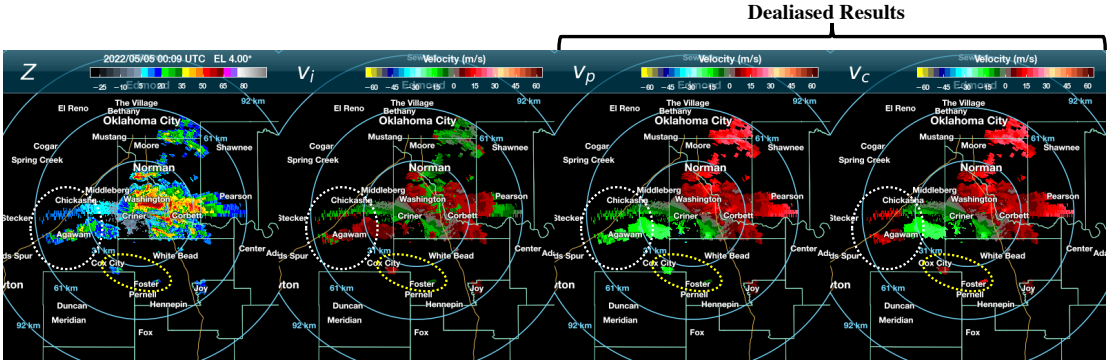


Figure 5.8: Velocity dealiasing PPI result observed with PX-1000 at a 4° -EL on 05 May 2022 00:09 UTC.

5.3 Error Analysis

In this section, error analysis is conducted by separating the measurement type as speckles and non-speckles. The separation process involves grouping together the error pixels that are connected. If a group has less than a threshold (here, 10 pixels are used as a threshold), it is considered a speckle. Otherwise, it is classified as a non-speckle.

An incorrect prediction is presented in Figure 5.9, where panel (a) displays the true labels, and panel (b) shows the predicted labels generated using the CNN model. Panels (c)–(h) represent the softmax classifier outputs, which normalize a vector to the [0, 1] range for each label, resembling a probability distribution of a random variable. Notably, significant overlaps exist between the labels '-1' and '0'. Some '0' labels are incorrectly classified as '-1' labels.

In this example, the otherwise correct label '0' has the second highest probability. In order to obtain more insight into the incorrect prediction, a similar scan taken at a similar time from a different elevation angle is examined. In Figure 5.10, the predicted label is generally accurate, and it overlaps somewhat between the label '-1' and '0'. A region with a low probability, lighter than surrounding pixels, is evident in the correct

label ’0’, and the second-highest probability, which is darker than the adjacent pixels, is seen in the incorrect label ‘-1’.

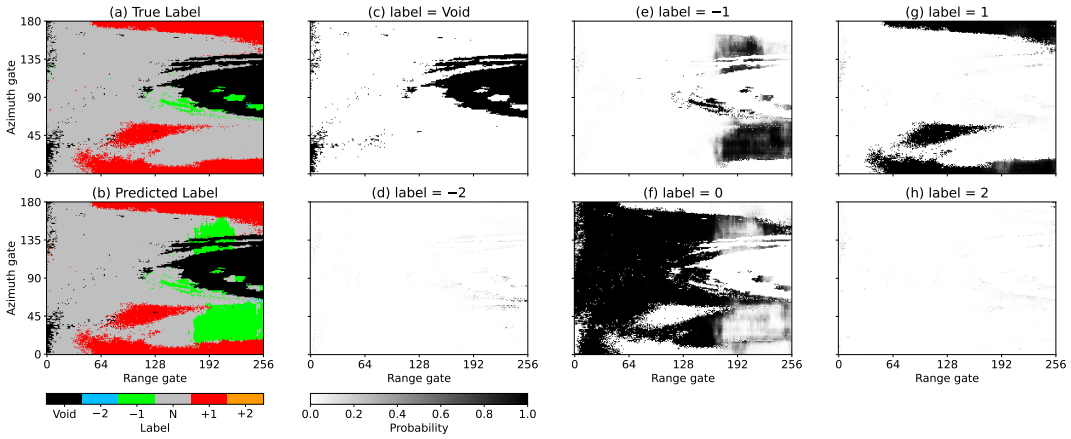


Figure 5.9: An example of failed prediction with non-speckle echoes: Panel (**a**) shows the true label, which is synthesized using a 0.88°-EL scan from the KTLX on 16 January 2017 06:33 UTC; Panel (**b**) shows the predicted label; Panels (**c**–**h**) represent the probability of each label from the CNN model. One can see that the green patch near azimuths 0–45° at far ranges is incorrectly predicted. The correct label ($L = 0$), however, has a significant probability value, which would result in a correct prediction if selected.

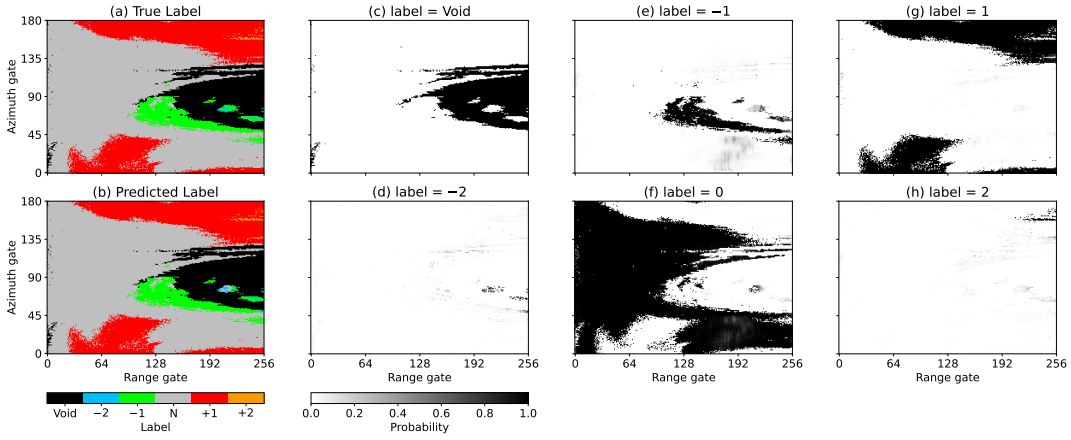


Figure 5.10: A similar scan to Figure 5.9 but the CNN model succeeded the prediction of aliasing labels (green patch in panel (b) of Figure 5.9). Panel (**a**) shows the true label, which is synthesized using a 1.32°-EL scan from the KTLX on 16 January 2017 06:33 UTC. In panel (**b**), the green patch near azimuths 0–45° at range gates 180–256 from panel (b) of Figure 5.9 is now correctly identified. Panels (**c**–**h**) represent the probability of each label from the CNN model.

Substituting the incorrect pixels with labels having the second highest probability would have increased the accuracy from 88.1% to 99.7%. The replaced outcomes are displayed in Figure 5.11, where panel (a) shows the true label, panel (b) indicates the original predicted label and panel (c) is the replaced label from panel (b) using labels with the second most probable prediction for failed ones. Among the total of 495 test data scans, 168 non-speckle scans were examined, and 80.9% of them had the correct label as the second most probable prediction. Although it is challenging to explain false predictions due to the complexity of the CNNs, it is evident that the CNN model's performance could improve considerably if the second most probable predictions were selected under such circumstances. However, it must be emphasized here that this cannot be recovered in practice. This example is presented here only to illustrate the potential for improvements and to provide insights into future possibilities.

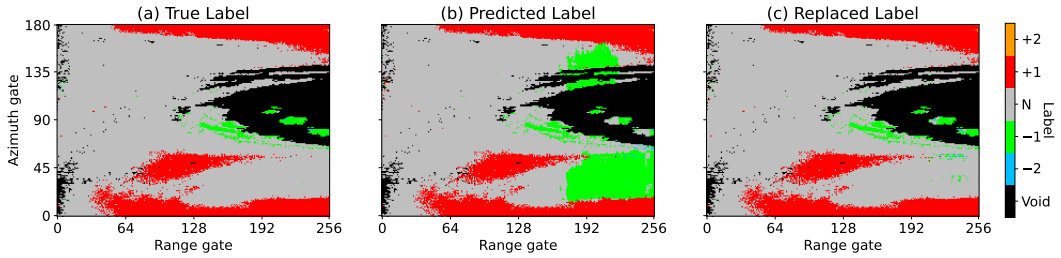


Figure 5.11: This figure shows the replaced result by the second most probable prediction on failed pixels: Panel (**a**) is the true label; panel (**b**) is the raw predicted label, and panel (**c**) is the same as the middle panel but incorrect labels are replaced by ones with the second highest probability. The value of A increased from 88.1% in (**b**) to 99.6% in (**c**).

5.4 Discussion

Generating a training dataset is arguably the most crucial step in developing a successful deep-learning model. One method of designing a deep-learning model involves incorporating the Nyquist velocity and mean wind as part of the input metadata. In

essence, knowing the scan elevation and mean wind enables us to roughly anticipate where aliasing might occur, making the use of these two variables useful in identifying aliasing locations. However, in this study, a different strategy was taken. That is, data normalization and augmentation. In our view, both methods achieve similar outcomes. The data normalization would eliminate the need for Nyquist velocity to be included as a part of metadata, while the data augmentation including rotating the PPI and negating the velocity values eliminates the need for mean wind, allowing the model to function without these variables.

Wind speed changes rapidly as altitude varies, and the possibility of aliasing can vary depending on scan elevation and range. One could argue that all scan elevations should be included in the training datasets. However, as mentioned previously, our hope is to let the CNN model comprehend the concept of aliasing rather than just memorizing particular patterns. Similar to how humans learn the aliasing concept, having all scan elevations is not necessary. Nonetheless, further research to examine the real-world outcome may be valuable in the future.

If the CNN model fails to identify certain velocity discontinuity features, replacing the output with the second-highest probability label can improve the accuracy from 88.1% to 99.7%. This indicates that there is room for improvement in the recognition of such features. While there is currently no obvious solution at the moment, recovering some of these errors may lead to significant enhancement in overall performance.

Applying the CNN method to X-band radar velocity synthesized from S-band radar velocity remains challenging when it comes to training for small-scale and rapidly changing signals, such as tornadoes. This is because S-band data may not capture these small-scale and rapidly changing signals adequately. However, a potential solution is to use semi-supervised learning, as explained in Chapter 3. This approach allows us to train unlabeled data using a pseudo-label approach. The model can be trained using

the labeled velocity, which is S-band radar with the ground truth. This is the same process as supervised learning. The generated model predicts the pseudo-label for the unlabeled data. After the quality control of the pseudo-label by thresholding with the softmax output value, the pseudo-label, and labeled velocity can be trained again. The benefit of this approach is that it enables us to train unlabeled data, which can include X-band radar data with limited availability of ground truth, or even S-band aliased velocity data that includes severe storm features.

Finally, the computational expense is a potential issue when utilizing a deep learning model. The training time of our design is on the order of tens of hours with a supercomputer while the inference time is only a fraction of a second, making it feasible for real-time applications.

Chapter 6

Conclusions and Future Work

In this study, velocity dealiasing using a CNN method is proposed, implemented, and evaluated. For the training, input velocity, and true label fields are generated from the non-aliased NEXRAD S-band radar velocity field. The velocity field is artificially aliased to produce the aliased velocity fields that simulate the effect of using X-band radar. However, the collected data tends to have an inherent mean bias due to the regional dominant wind direction. To overcome this issue, data augmentation is performed by rotating the velocity field in azimuth and negating the sign. The class weight is also used to balance the less frequently occurring labels, which are mostly the aliased labels. The optimization aims to minimize the difference between the true and the predicted labels during the training using the cross-entropy as the cost function. The velocity dealiasing using a CNN method is performed with input velocity v_{i}, predicted output label L_{p}, and the Nyquist velocity v_{a}. A sensitivity test was conducted to determine the best training conditions, which resulted in the selection of a template size of $T = 256$ and training $v_{\mathrm{a}} \in [7, \nu]$.

Evaluation is performed by comparison to the region-based method, which is a part of the Py-ART software collection. The evaluation is divided into three groups, denoted as G_1, G_2, and G_3, where each group represents a different set of velocity values (v_{a}). It is analyzed on precipitation scan characteristics (mostly filled and sparsely filled).

These groupings provide insights into how the algorithm performs in real-world scenarios when certain distributions of aliasing conditions are present. Specifically, G_1 represents a severe aliasing condition, G_2 represents a typical aliasing condition from an X-band radar, and G_3 represents a collection of velocity fields that are the easiest to process.

When dealing with mostly filled precipitation, both the CNN and region-based methods are able to generate the dealiasing label and produce the corresponding dealiased velocity fields, with a performance difference of less than 1 %. This demonstrates that the CNN method can be effectively employed in situations where precipitation is mostly filled. However, for sparsely filled precipitation, the CNN method performs significantly better than the region-based method.

The performance difference can be attributed to the discontinuity of the storms, which poses a challenge for the region-based method as it struggles to produce the first-guess fields correctly. Although utilizing external wind measurement could aid this process, this study did not employ this approach to ensure fairness in comparisons. In contrast, the CNN model, which has a receptive field covering the entire scan, can process the entire scan in one shot. Through the large collection of velocity fields in the training dataset, one can surmise the CNN model has learned what a proper velocity field and the corresponding aliasing label should look like. Consequently, it can produce the correct labels despite the discontinuity of the storms. This level of data comprehension and processing is what a human would do during a hand dealiasing process.

Case studies were conducted on synthetic X-band (that were derived from S-band NEXRAD radar) velocity and native X-band PX-1000 radar velocity field. For the synthetic X-band (derived from S-band) radar, both the CNN and region-based methods performed well for mostly filled scans. However, for sparsely filled scans, the CNN

performs well, while the region-based method failed for some isolated storms since they failed to estimate the first-guess field. For the X-band PX-1000 radar, the CNN model showed high performance for wide and spatially continuous storms since these features were included in the training dataset, whereas the region-based method has a possibility of failure when estimating the first-guess field. For the small scales and rapidly changing storms like tornadoes, the current model can not sufficiently cover the features due to the different range resolutions and the limited dataset of those features in training. Therefore, further study may be required to cover these features.

Several future works are suggested by the findings of this study. First, there are cases where the CNN model can encounter failures. This study shows that more than 80% of the errors (from the non-speckle echoes) could be eliminated if they were identified as the label with the second highest probability. Further investigation might be required to recover this type of error into the correct prediction.

Secondly, size dependency should be addressed in future improvements. With the current implementation, interleaving can be used to handle the full-resolution data. That is, for example, odd azimuths and even azimuths are processed separately since a complete 360° scan can be covered with 180 2° radials. Of course, discontinuity can occur whenever they disagree. In addition, for spatially rich weather phenomena, this method may miss small-scale features or predict incorrect labels since the interleaving method does not increase spatial resolution. Thus, addressing size dependency is crucial to enable this method to handle inputs of varying sizes to use this method regardless of scan strategy.

Additional research would be recommended for future work to predict the aliasing labels of small-scale and rapidly changing storms. To capture these features, it is recommended to include the X-band radar velocity from X-band radar system in training dataset. One potential approach is to use semi-supervised learning, which involves

training the model with a combined dataset that includes labeled X-band radar data synthesized from S-band radar system such as NEXRAD and unlabeled X-band radar data from X-band radar system such as PX-1000.

References

[1] M. I. Skolnik, *Introduction to Radar Systems, Third Edition*. McGraw-Hill Professional, 2001.

[2] R. J. Doviak and D. Zrnic, *Doppler Radar and Weather Observations*. Dover Publications, Inc., 1993.

[3] V. N. Bringi and V. Chandrasekar, *Polarimetric Doppler Weather Radar: Principles and Applications*. Cambridge University Press, 2001.

[4] R. Mailloux, *Phased Array Antenna Handbook, Third Edition*. 2017.

[5] A. Zahrai and D. Zrnić, "The 10-cm-Wavelength Polarimetric Weather Radar at NOAA's National Severe Storms Laboratory," *Journal of Atmospheric and Oceanic Technology*, vol. 10, no. 5, pp. 649–662, 1993.

[6] D. Zrinc and P. Mahapatra, "Two Methods of Ambiguity Resolution in Pulse Doppler Weather Radars," *IEEE Transactions on Aerospace and Electronic Systems*, no. 4, pp. 470–483, 1985.

[7] T. D. Crum and R. L. Alberty, "The WSR-88D and the WSR-88D Operational Support Facility," *Bulletin of the American Meteorological Society*, vol. 74, no. 9, pp. 1669–1688, 1993.

[8] E. E. Gossard, G. EE, and S. RG, "Radar Observation of Clear Air and Clouds," 1983.

[9] B. L. Cheong, R. Kelley, R. D. Palmer, Y. Zhang, and T.-Y. Yu, "PX-1000: A Solid-State Polarimetric X-band Weather Radar and Time–Frequency Multiplexed Waveform for Blind Range Mitigation," *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 11, pp. 3064–3072, Nov. 2013.

[10] NOAA, "JetStream Max: Volume Coverage Patterns (VCPs)," 2022, Available at https://www.weather.gov/jetstream/vcp_max.

[11] National Research Council, *Weather Radar Technology Beyond NEXRAD*. Washington, DC: The National Academies Press, 2002, Available at https://nap.nationalacademies.org/catalog/10394/weather-radar-technology-beyond-nexrad. DOI: 10.17226/10394.

[12] M. I. Skolnik, *Radar Handbook*. McGraw-Hill Education, 2008.

[13] H. Sauvageot, *Radar Meteorology*. Artech House on Demand, 1992.

[14] D. Sirmans *et al.*, "Extension of Maximum Unambiguous Doppler Velocity by Use of Two Sampling Rates.," 1976.

[15] R. J. Doviak, D. S. Zrnic, and D. S. Sirmans, "Doppler Weather Radar," *Proceedings of the IEEE*, vol. 67, no. 11, pp. 1522–1553, 1979.

[16] P. Joe and P. May, "Correction of Dual PRF Velocity Errors for Operational Doppler Weather Radars," *Journal of Atmospheric and Oceanic Technology*, vol. 20, no. 4, pp. 429–442, 2003.

[17] P. Tabary, F. Guibert, L. Perier, and J. Parent-du-Chatelet, "An Operational Triple-PRT Doppler Scheme for the French Radar Network," *Journal of Atmospheric and Oceanic Technology*, vol. 23, no. 12, pp. 1645–1656, 2006.

[18] G. Gray, B. Lewis, J. Vinson, and F. Pratte, "A Real-Time Implementation of Staggered PRT Velocity Unfolding," *Journal of Atmospheric and Oceanic Technology*, vol. 6, no. 1, pp. 186–187, 1989.

[19] M. Sachidananda and D. Zrnić, "Clutter Filtering and Spectral Moment Estimation for Doppler Weather Radars Using Staggered Pulse Repetition Time (PRT)," *Journal of Atmospheric and Oceanic Technology*, vol. 17, no. 3, pp. 323–331, 2000.

[20] S. M. Torres, Y. F. Dubel, and D. S. Zrnić, "Design, Implementation, and Demonstration of a Staggered PRT Algorithm for the WSR-88D," *Journal of Atmospheric and Oceanic Technology*, vol. 21, no. 9, pp. 1389–1399, 2004.

[21] P. S. Ray and C. Ziegler, "De-aliasing First-Moment Doppler Estimates," *Journal of Applied Meteorology*, vol. 16, no. 5, pp. 563–564, 1977.

[22] L. Hennington, "Reducing the Effects of Doppler Radar Ambiguities," *Journal of Applied Meteorology*, vol. 20, no. 12, pp. 1543–1546, 1981.

[23] W. R. Bergen and S. C. Albers, "Two-and Three-Dimensional De-aliasing of Doppler Radar Velocities," *Journal of Atmospheric and Oceanic technology*, vol. 5, no. 2, pp. 305–319, 1988.

[24] P. Tabary, G. Scialom, and U. Germann, "Real-Time Retrieval of the Wind from Aliased Velocities Measured by Doppler Radars," *Journal of Atmospheric and Oceanic technology*, vol. 18, no. 6, pp. 875–882, 2001.

[25] J. Zhang and S. Wang, "An Automated 2D Multipass Doppler Radar Velocity Dealiasing Scheme," *Journal of Atmospheric and Oceanic Technology*, vol. 23, no. 9, pp. 1239–1248, 2006.

[26] J. J. Helmus and S. M. Collis, "The Python ARM Radar Toolkit (Py-ART), a Library for Working with Weather Radar Data in the Python Programming Language," *Journal of Open Research Software*, vol. 4, 2016.

[27] M. D. Eilts and S. D. Smith, "Efficient Dealiasing of Doppler Velocities Using Local Environment Constraints," *Journal of Atmospheric and Oceanic technology*, vol. 7, no. 1, pp. 118–128, 1990.

[28] D. J. Bodine, R. D. Palmer, and G. Zhang, "Dual-Wavelength Polarimetric Radar Analyses of Tornadic Debris Signatures," *Journal of Applied Meteorology and Climatology*, vol. 53, no. 2, pp. 242–261, 2014.

[29] J. M. Kurdzo, D. J. Bodine, B. L. Cheong, and R. D. Palmer, "High-Temporal Resolution Polarimetric X-band Doppler Radar Observations of the 20 May 2013 Moore, Oklahoma, Tornado," *Monthly Weather Review*, vol. 143, no. 7, pp. 2711–2735, 2015.

[30] J. L. Houser, H. B. Bluestein, and J. C. Snyder, "A Finescale Radar Examination of the Tornadic Debris Signature and Weak-Echo Reflectivity Band Associ-

ated with a Large, Violent Tornado," *Monthly Weather Review*, vol. 144, no. 11, pp. 4101–4130, 2016.

[31] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2015, pp. 234–241.

[32] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[33] A. Karpathy, "Stanford University CS231n: Convolutional Neural Networks for Visual Recognition," 2017, Available at https://cs231n.stanford.edu/syllabus.html.

[34] Y. LeCun *et al.*, "Handwritten Digit Recognition with a Back-Propagation Network," *Advances in Neural Information Processing Systems*, vol. 2, 1989.

[35] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[36] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep Face Recognition," 2015.

[37] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[38] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[39] P. Arena, A. Basile, M. Bucolo, and L. Fortuna, "Image Processing for Medical Diagnosis Using CNN," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 497, no. 1, pp. 174–178, 2003.

[40] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Černockỳ, "Strategies for Training Large Scale Neural Network Language Models," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, IEEE, 2011, pp. 196–201.

[41] G. Hinton *et al.*, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[42] E. Racah, C. Beckham, T. Maharaj, S. E. Kahou, M. Prabhat, and C. Pal, "ExtremeWeather: A Large-Scale Climate Dataset for Semi-Supervised Detection, Localization, and Understanding of Extreme Weather Events," in *Advances in Neural Information Processing Systems*, 2017, pp. 3402–3413.

[43] R. Lagerquist, A. McGovern, and D. J. Gagne II, "Deep Learning for Spatially Explicit Prediction of Synoptic-Scale Fronts," *Weather and Forecasting*, vol. 34, no. 4, pp. 1137–1160, 2019.

[44] A. Wimmers, C. Velden, and J. H. Cossuth, "Using Deep Learning to Estimate Tropical Cyclone Intensity from Satellite Passive Microwave Imagery," *Monthly Weather Review*, vol. 147, no. 6, pp. 2261–2282, 2019.

[45] D. J. Gagne, S. E. Haupt, D. W. Nychka, and G. Thompson, "Interpretable Deep Learning for Spatial Analysis of Severe Hailstorms," *Monthly Weather Review*, vol. 147, no. 8, pp. 2827–2845, 2019.

[46] C. Chilson, K. Avery, A. McGovern, E. Bridge, D. Sheldon, and J. Kelly, "Automated Detection of Bird Roosts Using NEXRAD Radar Data and Convolutional Neural Networks," *Remote Sensing in Ecology and Conservation*, vol. 5, no. 1, pp. 20–32, 2019.

[47] H. Kim and B. Cheong, "Robust Velocity Dealiasing for Weather Radar Based on Convolutional Neural Networks," *Remote Sensing*, vol. 15, no. 3, p. 802, 2023.

[48] R. E. Rinehart, *Radar for Meteorologists*. University of North Dakota, Office of the President, 1991.

[49] J. Probert-Jones, "The Radar Equation in Meteorology," *Quarterly Journal of the Royal Meteorological Society*, vol. 88, no. 378, pp. 485–495, 1962.

[50] L. J. Battan, "Radar Observation of the Atmosphere," Tech. Rep., 1973.

[51] H. Nyquist, "Certain Topics in Telegraph Transmission Theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, no. 2, pp. 617–644, 1928.

[52] S.-G. Park, J.-H. Kim, J.-S. Ko, and G. Lee, "Identification of Range Overlaid Echoes Using Polarimetric Radar Measurements Based on a Fuzzy Logic Ap-

proach," *Journal of Atmospheric and Oceanic Technology*, vol. 33, no. 1, pp. 61–80, 2016.

[53] K. Browning and R. Wexler, "The Determination of Kinematic Properties of a Wind Field Using Doppler Radar," *Journal of Applied Meteorology*, vol. 7, no. 1, pp. 105–113, 1968.

[54] J. Gong, L. L. Wang, and Q. Xu, "A Three-Step Dealiasing Method for Doppler Velocity Data Quality Control," *Journal of Atmospheric and Oceanic technology*, vol. 20, no. 12, pp. 1738–1748, 2003.

[55] C. N. James and R. A. Houze, "A Real-Time Four-Dimensional Doppler Dealiasing Scheme," *Journal of Atmospheric and Oceanic Technology*, vol. 18, no. 10, pp. 1674–1683, 2001.

[56] E. Lim and J. Sun, "A Velocity Dealiasing Technique Using Rapidly Updated Analysis from a Four-Dimensional Variational Doppler Radar Data Assimilation System," *Journal of Atmospheric and Oceanic Technology*, vol. 27, no. 7, pp. 1140–1152, 2010.

[57] G. He, J. Sun, and Z. Ying, "An Automated Velocity Dealiasing Scheme for Radar Data Observed from Typhoons and Hurricanes," *Journal of Atmospheric and Oceanic Technology*, vol. 36, no. 1, pp. 139–149, 2019.

[58] S. J. Russell, *Artificial Intelligence a Modern Approach*. Pearson Education, Inc., 2010.

[59] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning*. Springer, 2006, vol. 4.

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

[61] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009, vol. 2.

[62] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[63] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," *arXiv preprint arXiv:1511.06434*, 2015.

[64] B. R. Kiran, D. M. Thomas, and R. Parakkal, "An Overview of Deep Learning Based Methods for Unsupervised and Semi-Supervised Anomaly Detection in Videos," *Journal of Imaging*, vol. 4, no. 2, p. 36, 2018.

[65] D.-H. Lee, "Pseudo-Label: The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks," in *Workshop on Challenges in Representation Learning, ICML*, vol. 3, 2013, p. 896.

[66] D. P. Kingma, S. Mohamed, D. Jimenez Rezende, and M. Welling, "Semi-Supervised Learning with Deep Generative Models," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[67] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "Mixmatch: A Holistic Approach to Semi-Supervised Learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[68] S. Laine and T. Aila, "Temporal Ensembling for Semi-Supervised Learning," *arXiv preprint arXiv:1610.02242*, 2016.

[69] S. Weisberg, *Applied Linear Regression*. John Wiley & Sons, 2005, vol. 528.

[70] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*. John Wiley & Sons, 2013, vol. 398.

[71] C. E. Shannon, "A Mathematical Theory of Communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[72] S. Jadon, "A Survey of Loss Functions for Semantic Segmentation," in *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, IEEE, 2020, pp. 1–7.

[73] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a Simple Way to Prevent Neural Networks from Overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[74] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding Data Augmentation for Classification: When to Warp?" In *2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, IEEE, 2016, pp. 1–6.

[75] L. Perez and J. Wang, "The Effectiveness of Data Augmentation in Image Classification Using Deep Learning," *arXiv preprint arXiv:1712.04621*, 2017.

[76] A. Mikołajczyk and M. Grochowski, "Data Augmentation for Improving Deep Learning in Image Classification Problem," in *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, IEEE, 2018, pp. 117–122.

[77] C. Shorten and T. M. Khoshgoftaar, "A Survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[78] D. McLaughlin *et al.*, "Short-Wavelength Technology and the Potential For Distributed Networks of Small Radar Systems," *Bull. Amer. Meteor. Soc.*, vol. 90, no. 12, pp. 1797–1817, Dec. 2009.

[79] A. Pazmany and H. Bluestein, "Mobile Rapid Scanning X-Band Polarimetric (RaXPol) Doppler Radar System," in *34th Conf. Radar Meteor.*, Amer. Meteor. Soc., Williamsburg, VA, Oct. 2009, 8A.2.

[80] *Average Wind Speeds - Map Viewer*, Available at https://www.climate.gov/maps-data/dataset/average-wind-speeds-map-viewer.

[81] I. Csiszár, "I-Divergence Geometry of Probability Distributions and Minimization Problems," *The Annals of Probability*, pp. 146–158, 1975.

[82] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the Effective Receptive Field in Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

Acronyms

ACF Auto correlation function

AI Artificial intelligence

ARRC Advanced Radar Research Center

BPF Bandpass filter

CAM5 Community Atmospheric Model v5

CNN Convolutional Neural Network

CW Continuous wave

DNN Deep neural network

FAA Federal Aviation Administration

HCA Hydrometeor classification algorithm

IF Intermediate frequency

LNA Low noise amplifier

LCM Least common multiple

LO Local oscillator

MAE Mean absolute error

ML Machine learning

MSE Mean squared error

NEXRAD Next Generation Weather Radar

NN Neural network

NOAA National Oceanic and Atmospheric Administration

NWS National Weather Service

OU The University of Oklahoma

PAR Phased array radar

PPI Plan position indicator

PRF Pulse repetition frequency

PRT Pulse repetition time

Py-ART Python ARM radar toolkit

RCS Radar cross section

ReLU Rectified linear unit

RF Radio frequency

RMSE Root mean squared error

SNR Signal-to-noise ratio

VCP Volume coverage pattern

WSR-88D Weather Surveillance Radar - 88 Doppler

Index

Aliasing, 2–5, 7, 8, 18, 19, 21–24, 27, 29, 30, 41, 49, 55–58, 63, 64, 66, 72, 78–80, 82, 84, 86, 88, 91, 92

Artificial intelligence, 5, 32, 34, 35

Convolutional neural network, 6, 7, 32, 34, 35, 42, 44, 49, 50, 53, 54, 56–58, 61–64, 69–82, 84–88, 90–92

Cross-entropy loss, 48, 65, 90

Dealiasing, 4, 5, 7, 26, 27, 29, 30, 56–58, 60–64, 70–72, 75–85, 90, 91

Deep neural network, 6, 33–37, 48, 50

Least-common-multiplier, 4, 26, 27, 31

Machine learning, 5, 32, 33, 36

Maximum unambiguous range, 2, 19, 21–24, 27

Neural network, 6, 33–35, 39, 44, 45

NEXRAD, 2, 11, 58–62, 66, 73, 74, 78, 80, 82, 90, 91, 93

Nyquist velocity, 2, 4, 5, 7, 19, 21–23, 26, 27, 30, 31, 56, 58, 59, 62–78, 80, 81, 83, 90

Plan position indicator, 23, 25, 79–83, 85, 88

Pulse repetition frequency, 3, 4, 26–28, 58

Pulse repetition time, 2–4, 11, 15–17, 19–23, 25–28

PX-1000, 2, 58, 61, 78, 80–83, 85, 91–93

Python ARM Radar Toolkit, 4, 30, 90

Radar range equation, 8–10, 12, 13

Region-based, 4, 30, 71, 72, 74–82, 90–92

U-Net, 6, 54–56

Weather radar, 1, 2, 7–9, 13, 20, 57
Weighted cross-entropy loss, 49, 65
WSR-88D, 2