5-2018

Association Learning Via Deep Neural Networks

Trevor J. Landeen
Utah State University

ASSOCIATION LEARNING VIA DEEP NEURAL NETWORKS

by

Trevor J. Landeen

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

DOCTOR OF PHILOSOPHY

in

Electrical Engineering

Approved:

UTAH STATE UNIVERSITY
Logan, Utah

2018

ABSTRACT

Association Learning Via Deep Neural Networks

by

Trevor J. Landeen, Doctor of Philosophy

Utah State University, 2018

Major Professor: Jacob Gunther, Ph.D.
Department: Electrical and Computer Engineering

In recent years there has been renewed interest in artificial neural networks. Driving the resurgence is the success of deep learning in many challenging problems. In this dissertation, two independent input association problems are presented and multiple solutions using deep neural networks (DNN) are proposed.

Learning an association between unimodal inputs is posed as an image overlap detection problem. Multiple DNNs using three different output encodings are trained and each model's association performance is evaluated. A modified optimization problem demonstrates the trained DNNs successful learning of the input images' association.

Learning an association between multimodal inputs is presented as an emitter and radio frequency (RF) emission association problem. A DNN is used to determine if a given emitter is the source of any observed RF emissions. Multiple DNNs are trained to make the decisions and then compared to each other.

The analysis on the DNNs from both problems demonstrates the ability of DNNs to learn complicated associations between different observations.

(208 pages)

PUBLIC ABSTRACT

Association Learning Via Deep Neural Networks

Trevor J. Landeen

Deep learning has been making headlines in recent years and is often portrayed as an emerging technology on a meteoric rise towards fully sentient artificial intelligence. In reality, deep learning is the most recent renaissance of a 70 year old technology and is far from possessing true intelligence. The renewed interest is motivated by recent successes in challenging problems, the accessibility made possible by hardware developments, and dataset availability.

The predecessor to deep learning, commonly known as the artificial neural network, is a computational network setup to mimic the biological neural structure found in brains. However, unlike human brains, artificial neural networks, in most cases cannot make inferences from one problem to another. As a result, developing an artificial neural network requires a large number of examples of desired behavior for a specific problem. Furthermore, developing an artificial neural network capable of solving the problem can take days, or even weeks, of computations.

Two specific problems addressed in this dissertation are both input association problems. One problem challenges a neural network to identify overlapping regions in images and is used to evaluate the ability of a neural network to learn associations between inputs of similar types. The other problem asks a neural network to identify which observed wireless signals originated from observed potential sources and is used to assess the ability of a neural network to learn associations between inputs of different types.

The neural network solutions to both problems introduced, discussed, and evaluated in this dissertation demonstrate deep learning's applicability to problems which have previously attracted little attention.

ACKNOWLEDGMENTS

First and foremost, I want to express my gratitude and appreciation to my family for encouraging me and supporting me my entire life, and for their sacrifices made on my behalf. I also want to thank my advisor Dr. Jake Gunther for the countless hours spent helping me during every phase of my research, Sam Whiting for listening to and critiquing my ideas, all of my friends who at times may have felt neglected but continued providing support, and finally KickView Corporation, the U.S. Air Force, and the Utah NASA Space Grant Consortium for providing the funding which has allowed me to focus my time and energy on this research.

Trevor J. Landeen

CONTENTS

Page

ABSTRACT . . . iii

PUBLIC ABSTRACT . . . iv

ACKNOWLEDGMENTS . . . v

LIST OF TABLES . . . ix

LIST OF FIGURES . . . x

ACRONYMS . . . xxiv

1 Introduction . . . 1
- 1.1 Overview of Deep Learning . . . 2
- 1.2 Dissertation Outline . . . 6

2 Image Overlap Detection Introduction . . . 7
- 2.1 Problem Description . . . 7
- 2.2 Related Work . . . 8
  - 2.2.1 Image Overlap . . . 8
  - 2.2.2 Convolutional Neural Networks . . . 9
  - 2.2.3 Visualizing CNN Layers . . . 11
  - 2.2.4 Siamese Neural Networks . . . 13
  - 2.2.5 Additional Related Work . . . 16
- 2.3 Methodology . . . 17
  - 2.3.1 Data format . . . 17
- 2.4 Summary . . . 27

3 Image Overlap Detection Experiments . . . 29
- 3.1 Direct Overlap Mask Encoding . . . 29
  - 3.1.1 Mean Squared Error . . . 30
  - 3.1.2 Penalizing Incorrect Predictions . . . 32
  - 3.1.3 Forcing the Shape of The Predicted Overlapping Region . . . 33
- 3.2 Corners Overlap Mask Encoding . . . 36
- 3.3 Row-Column Overlap . . . 38
  - 3.3.1 Model 1 . . . 40
  - 3.3.2 Model 2 . . . 40
  - 3.3.3 Model 3 . . . 41
  - 3.3.4 Model 4 . . . 41
  - 3.3.5 Model 5 . . . 43
  - 3.3.6 Model 6 . . . 43
  - 3.3.7 Model 7 . . . 44

3.3.8 Model 8 . . . 44
3.3.9 Model 9 . . . 44
3.4 Models Summary . . . 44

4 Image Overlap Detection Results . . . 49
4.1 Figures for Evaluation . . . 50
4.2 All Ones and All Zeros Predicting Models . . . 51
4.3 Training Results . . . 54
4.3.1 Direct Encoding . . . 54
4.3.2 Corner Encoding . . . 69
4.3.3 Row-Column Encoding . . . 73
4.4 Summary . . . 101

5 Image Overlap Detection Network Visualizations . . . 102
5.1 Test Image Space Search . . . 102
5.1.1 Test Image Set Creation . . . 104
5.1.2 Example Search . . . 104
5.1.3 Implemented Test Image Space Search . . . 105
5.2 Offset Image Search Results . . . 108
5.2.1 Perfect Model Results . . . 109
5.2.2 Real Model Results . . . 113
5.3 Weights and Activations Visualizations . . . 117
5.3.1 Row-Column Encoding, Model 2 . . . 118
5.3.2 Corner Encoding Model . . . 119
5.3.3 Direct Encoding, Shape Forcing Model . . . 119
5.3.4 Kernel Discussion . . . 121
5.4 Conclusion . . . 122

6 Emitter and RF Emission Association Introduction . . . 124
6.1 Locating an RF Emitter . . . 125
6.1.1 Time Difference of Arrival Calculation . . . 125
6.1.2 Localization Using TDOAs . . . 126
6.2 Related Work . . . 127
6.3 Summary . . . 128

7 Emitter and RF Emission Association Experiments . . . 129
7.1 Solution Approach . . . 129
7.2 Probability Representation . . . 131
7.3 Generating Datasets . . . 132
7.4 Models Summary . . . 135

8 Emitter and RF Emission Association Results and Discussion . . . 137
8.1 Training Results . . . 140
8.1.1 Base 0 DNN . . . 141
8.1.2 Base 1 DNN . . . 143
8.1.3 Base 2 DNN . . . 143
8.1.4 Large 0 DNN . . . 146
8.1.5 Large 1 DNN . . . 149

8.1.6 Large 2 DNN . . . 151
8.1.7 N_2 DNN . . . 154
8.1.8 N_{50} DNN . . . 155
8.1.9 Final DNN . . . 159
8.1.10 Alternate Geometry DNN . . . 161
8.2 Conclusion . . . 161

9 General Conclusion . . . 165

REFERENCES . . . 167

APPENDICES . . . 172
A Image Overlap Detection Additional Figures . . . 173
A.1 Additional Results . . . 173

CURRICULUM VITAE . . . 184

LIST OF TABLES

Table Page

2.1 Types of overlap and their contributions to the possible orientations. 24

2.2 Training and testing dataset parameters 27

3.1 Summary of image overlap detection DNNs, including the number of neurons in each layer. 48

7.1 Summary of the three main datasets. 134

7.2 Summary of four additional datasets. 135

7.3 Summary of the 10 emitter and RF emission association DNNs. 136

LIST OF FIGURES

Figure Page

1.1 Example of a fully connected network having four input neurons and a single output neuron. 3

1.2 Example CNN with K convolution outputs for K kernels in the layer. . . . 5

2.1 The cardinality of the set of valid overlap sizes, $|V|$, increases as the size of the overlap mask increases. 21

2.2 Illustration of the reference image and test image with no overlapping areas. The overlap is defined relative to the reference image. The source image is from the COCO dataset. 22

2.3 Illustration of complete overlap between the reference image and test image. The overlap is defined relative to the reference image. The source image is from the COCO dataset. 23

2.4 Illustration of complete column overlap between the reference image and test image. The overlap is defined relative to the reference image. The source image is from the COCO dataset. 23

2.5 Illustration of the square overlap between the reference image and test image. The overlap is defined relative to the reference image. The source image is from the COCO dataset. 25

2.6 Illustration of the general case of overlap between the reference image and test image. The overlap is defined relative to the reference image. The source image is from the COCO dataset. 25

2.7 The cardinality of the set of all possible overlap orientations, $|0_{total}|$, increases as the size of the overlap mask increases. 26

2.8 Histogram of generated training dataset. The dataset is uniformly distributed over the set V. 28

2.9 Histogram of generated testing dataset. The dataset is uniformly distributed over the set V. 28

3.1 Overview of the DNN structure using the direct overlap mask encoding. Two images are input into the siamese neural network and a vector $\mathbf{m}$ representing the overlap is output. 31

3.2 The direct encoding DNN's specific siamese neural network details. The number of neurons in each layer is indicated below the layer. 31

3.3 Illustration of the corner overlap encoding. One corner is provided by $\mathbf{a} = [a_0,\ a_1,\ a_2,\ ,a_3]$ and the other by $\mathbf{b} = [b_0,\ b_1,\ \dots\ ,b_{99}]$. In this example, $a_3 = 1$ and $b_k = 1$. The region between them is filled in to get the overlap mask. 37

3.4 Overview of the DNN structure using the corner overlap mask encoding. Two images are input into the siamese neural network and the vectors **a** and **b** indicating which corners of the overlapping region are output. 38

3.5 The architecture of the siamese network using the corner encoding. The number of output neurons in each layer is indicated by the number below the layer. 39

3.6 Example of the row-column overlap mask encoding. The elements of **r** are used to indicate whether or not overlap occurs in each specific row. Similarly for the elements of **c**, each element indicates if overlap exists in its represented column. 39

3.7 The overview of the DNN structure using the row-column overlap mask encoding. Two images are input into the siamese neural network and the vectors **r** and **c** indicating which rows and columns contain the overlapping region are output. 40

3.8 The architecture of the first siamese network (Model 1) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer. 41

3.9 The architecture of the second siamese network (Model 2) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer. 42

3.10 The architecture of the third siamese network using (Model 3) the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer. 42

3.11 The architecture of the fourth siamese network (Model 4) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer. 43

3.12 The architecture of the fifth siamese network (Model 5) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer. 43

3.13 The architecture of the sixth siamese network (Model 6) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer. 45

3.14 The architecture of the seventh siamese network (Model 7) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer. 45

3.15 The architecture of the eighth siamese network (Model 8) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer. 46

3.16 The architecture of the ninth siamese network (Model 9) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer. 46

4.1 Histogram of prediction accuracies for the DNN predicting all ones. 52

4.2 Histogram of prediction accuracies for the DNN predicting all zeros. 52

4.3 The average prediction accuracy for the possible sizes of overlapping regions for the DNN predicting all ones. 53

4.4 The average prediction accuracy for the possible sizes of overlapping regions for the DNN predicting all zeros. 53

4.5 Detailed histogram of prediction accuracies and the area of the overlapping regions for the all ones prediction models. The number of occurrences of each is indicated by the pixel intensity. 55

4.6 Detailed histogram of prediction accuracies and the area of the overlapping regions for the all zeros prediction models. The number of occurrences of each is indicated by the pixel intensity. 56

4.7 Histogram of the prediction accuracies for the DNN with the mean squared error loss function. 56

4.8 The average prediction accuracy for the possible sizes of overlapping regions for the DNN with the mean squared error loss function. 57

4.9 Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN with the mean squared error loss function. The number of occurrences of each is indicated by the pixel intensity. 57

4.10 Sample prediction of the DNN with the mean squared error loss function. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. . . . 59

4.11 Histogram of the prediction accuracies for the DNN trained to emphasize the correct prediction of the non-overlapping regions. 59

4.12 The average prediction accuracy for the possible sizes of overlapping regions for the DNN trained to emphasize the correct prediction of the non-overlapping regions. 61

4.13 Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN trained to emphasize the correct prediction of the non-overlapping regions. The number of occurrences of each is indicated by the pixel intensity. 61

4.14 Sample prediction of the DNN trained to emphasize the correct prediction of the non-overlapping regions. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 62

4.15 Histogram of the prediction accuracies for the DNN trained to emphasize the correct prediction of the overlapping regions. 63

4.16 The average prediction accuracy for the possible sizes of overlapping regions for the DNN trained to emphasize the correct prediction of the overlapping regions. 63

4.17 Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN trained to emphasize the correct prediction of the overlapping regions. The number of occurrences of each is indicated by the pixel intensity. 64

4.18 Sample prediction of the DNN trained to emphasize the correct prediction of the overlapping regions. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 64

4.19 Histogram of the prediction accuracies for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 5$ 66

4.20 The average prediction accuracy for the possible sizes of overlapping regions for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 5$. 66

4.21 Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 5$. The number of occurrences of each is indicated by the pixel intensity. 67

4.22 Sample prediction of the DNN trained to predict the correct shape of the overlapping region. $\lambda = 5$. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 68

4.23 Histogram of the prediction accuracies for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 10$ 69

4.24 The average prediction accuracy for the possible sizes of overlapping regions for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 10$ 70

4.25 Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 10$. The number of occurrences of each is indicated by the pixel intensity. 70

4.26 Sample prediction of the DNN trained to predict the correct shape of the overlapping region. $\lambda = 10$. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 71

4.27 Histogram of the prediction accuracies for the DNN implementing the corner overlap mask encoding. 72

4.28 The average prediction accuracy for the possible sizes of overlapping regions for the DNN implementing the corner overlap mask encoding. 72

4.29 Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN implementing the corner overlap mask encoding. The number of occurrences of each is indicated by the pixel intensity. 74

4.30 Sample prediction of the DNN implementing the corner overlap mask encoding. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 74

4.31 Histogram of the prediction accuracies for row-column overlap encoding model 1. 76

4.32 The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 1. 76

4.33 Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 1. The number of occurrences of each is indicated by the pixel intensity. 77

4.34 Sample prediction for row-column overlap encoding model 1. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 77

4.35 Histogram of the prediction accuracies for row-column overlap encoding model 2. 79

4.36 The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 2. 79

4.37 Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 2. The number of occurrences of each is indicated by the pixel intensity. 80

4.38 Sample prediction for row-column overlap encoding model 2. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 80

4.39 Histogram of the prediction accuracies for row-column overlap encoding model 3. 82

4.40 The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 3. 82

4.41 Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 3. The number of occurrences of each is indicated by the pixel intensity. 83

4.42 Sample prediction for row-column overlap encoding model 3. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 83

4.43 Histogram of the prediction accuracies for row-column overlap encoding model 4. 85

4.44 The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 4. 85

4.45 Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 4. The number of occurrences of each is indicated by the pixel intensity. 86

4.46 Sample prediction for row-column overlap encoding model 4. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 86

4.47 Histogram of the prediction accuracies for row-column overlap encoding model 5. 88

4.48 The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 5. 88

4.49 Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 5. The number of occurrences of each is indicated by the pixel intensity. 89

4.50 Sample prediction for row-column overlap encoding model 5. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 89

4.51 Histogram of the prediction accuracies for row-column overlap encoding model 6. 91

4.52 The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 6. 91

4.53 Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 6. The number of occurrences of each is indicated by the pixel intensity. 92

4.54 Sample prediction for row-column overlap encoding model 6. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 92

4.55 Histogram of the prediction accuracies for row-column overlap encoding model 7. 93

4.56 The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 7. 94

4.57 Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 7. The number of occurrences of each is indicated by the pixel intensity. 94

4.58 Sample prediction for row-column overlap encoding model 7. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 95

4.59 Histogram of the prediction accuracies for row-column overlap encoding model 8. 96

4.60 The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 8. 96

4.61 Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 8. The number of occurrences of each is indicated by the pixel intensity. 98

4.62 Sample prediction for row-column overlap encoding model 8. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 98

4.63 Histogram of the prediction accuracies for row-column overlap encoding model 9. 99

4.64 The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 9. 99

4.65 Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 9. The number of occurrences of each is indicated by the pixel intensity. 100

4.66 Sample prediction for row-column overlap encoding model 9. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black. 100

5.1 Given the reference image and mask M, the test image satisfying the specified overlap is outlined in blue. The DNN is used to find the test image. 103

5.2 Illustration of the test image space search. The reference image and mask are given. All possible test images are processed using a DNN. The loss is computed between the predicted masks and the true mask, and the test image yielding the lowest loss is chosen. 104

5.3 The true test image. The mask corresponds to the overlap between the red reference image and the blue test image. The source image is from the COCO dataset. 106

5.4 New test image is shifted down and right from the true test image. The mask corresponds to the overlap between the red reference image and the blue test image. The source image is from the COCO dataset. 106

5.5 New test image is shifted up and right from the true test image. The mask corresponds to the overlap between the red reference image and the blue test image. The source image is from the COCO dataset. 107

5.6 New test image is shifted up and left from the true test image. The mask corresponds to the overlap between the red reference image and the blue test image. The source image is from the COCO dataset. 107

5.7 Average loss results of the stochastic image search for the perfect prediction DNN implementing the direct encoding. 110

5.8 Average prediction accuracy results of the stochastic image search for the perfect prediction DNN implementing the direct encoding. 111

5.9 Average loss results of the stochastic image search for the perfect prediction DNN implementing the corner encoding. 112

5.10 Average prediction accuracy results of the stochastic image search for the perfect prediction DNN implementing the corner encoding. 112

5.11 Average loss results of the stochastic image search for the perfect prediction DNN implementing the corner encoding. 114

5.12 Average prediction accuracy results of the stochastic image search for the perfect prediction DNN implementing the corner encoding. 114

5.13 Row-Column encoding, model 2: The average loss results of the stochastic test image space search. 116

5.14 Row-Column encoding, model 2: The average prediction accuracy results of the stochastic test image space search. 116

5.15 Corner encoding: The average loss results of the stochastic test image space search. 116

5.16 Corner encoding: The average prediction accuracy results of the stochastic test image space search. 116

5.17 Direct encoding, emphasis on prediction overlap region shape, $\lambda = 10$: The average loss results of the stochastic test image space search. 118

5.18 Direct encoding, emphasis on prediction overlap region shape, $\lambda = 10$: The average prediction accuracy results of the stochastic test image space search. 118

5.19 Image used to generate the activations. 119

5.20 Row-Column encoding, model 2: Convolution kernels. 120

5.21 Row-Column encoding, model 2: Activations. Each activation is produced by the kernel in the same position. 120

5.22 Corner encoding: Convolution kernels. 120

5.23 Corner encoding: Activations. Each activation is produced by the kernel in the same position. 120

5.24 Direct encoding, emphasis on prediction overlap region shape, $\lambda = 10$: Convolution kernels. 122

5.25 Direct encoding, emphasis on prediction overlap region shape, $\lambda = 10$: Activations. Each activation is produced by the kernel in the same position. . . 122

6.1 The emitter and RF emission problem involves identifying if any of the emitters detected in the region of operation (black square) are the sources of the observed RF emissions. The RF emissions are observed using three antennas with known locations. 124

6.2 Example TDOA calculation where two antennas observe the same transmitted signal τ seconds apart. 126

6.3 TDOA localization involves finding the intersection of multiple isochrones. Three antennas provide two sets of independent TDOAs and each set is symmetric between its antennas. Only the pair which intersect in the operating region are depicted. 128

7.1 Overview of the processing chain. The video frames are preprocessed to obtain optical features and the RF data are preprocessed to obtain the RF features and both features are used as inputs of a DNN. The focus of this research is the DNN and the format of the output $P(Inputs)$. 130

7.2 Desired probability behavior. The probability decay rate decreases as α decreases. 133

8.1 Prediction error surface for the base 0 DNN. The region of operation is approximately 350 by 280 meters. 142

8.2 Point Responses for the base 0 DNN. Each point response is shown over the entire region of operation. 142

8.3 ROC Curves for the base 0 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 144

8.4 The prediction error surface for the base 1 DNN. The region of operation is approximately 350 by 280 meters. 144

8.5 Point Responses for the base 1 DNN. Each point response is shown over the entire region of operation. 145

8.6 ROC Curve for the base 1 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 145

8.7 Prediction error surface for the base 2 DNN. The region of operation is approximately 350 by 280 meters. 147

8.8 Point Responses for the base 2 DNN. Each point response is shown over the entire region of operation. 147

8.9 ROC Curve for the base 2 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 148

8.10 Prediction error surface for the large 0 DNN. The region of operation is approximately 350 by 280 meters. 148

8.11 Point Responses for the large 0 DNN. Each point response is shown over the entire region of operation. 149

8.12 ROC Curve for the large 0 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 150

8.13 Prediction error surface for the large 1 DNN. The region of operation is approximately 350 by 280 meters. 150

8.14 Point Responses for the large 1 DNN. Each point response is shown over the entire region of operation. 152

8.15 ROC Curve for the large 1 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 152

8.16 Prediction error surface for the large 2 DNN. The region of operation is approximately 350 by 280 meters. 153

8.17 Point Responses for the large 2 DNN. Each point response is shown over the entire region of operation. 153

8.18 ROC Curve for the large 2 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 154

8.19 Prediction error surface for the N_2 DNN. The region of operation is approximately 350 by 280 meters. 156

8.20 Point Responses for the N_2 DNN. Each point response is shown over the entire region of operation. 156

8.21 ROC Curve for the N_2 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 157

8.22 Prediction error surface for the N_{50} DNN. The region of operation is approximately 350 by 280 meters. 157

8.23 Point Responses for the N_{50} DNN. Each point response is shown over the entire region of operation. 158

8.24 ROC Curve for the N_{50} DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 159

8.25 Prediction error surface for the Final DNN. The region of operation is approximately 350 by 280 meters. 160

8.26 Point Responses for the Final DNN. Each point response is shown over the entire region of operation. 160

8.27 ROC Curve for the Deep DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 162

8.28 Prediction error surface for the Alt. Geom. DNN. The region of operation is approximately 350 by 280 meters. 162

8.29 Point Responses for the Alt. Geom. DNN. Each point response is shown over the entire region of operation. 163

8.30 ROC Curve for the Alt. Geom. DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters. 163

A.1 Row-Column encoding, model 1: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. . . . 173

A.2 Row-Column encoding, model 3: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. . . . 174

A.3 Row-Column encoding, model 4: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. . . . 174

A.4 Row-Column encoding, model 5: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. . . . 174

A.5 Row-Column encoding, model 6: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. . . . 175

A.6 Row-Column encoding, model 7: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. . . . 175

A.7 Row-Column encoding, model 8: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. . . . 175

A.8 Row-Column encoding, model 9: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. . . . 176

A.9 Direct encoding, MSE: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. 176

A.10 Direct encoding, emphasis on non-overlapping region: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. 176

A.11 Direct encoding, emphasis on overlapping region: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. 177

A.12 Direct encoding, emphasis on prediction overlapping region shape, $\lambda = 5$: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search. 177

A.13 Row-Column encoding, model 1: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 177

A.14 Row-Column encoding, model 3: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 178

A.15 Row-Column encoding, model 4: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 178

A.16 Row-Column encoding, model 5: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 179

A.17 Row-Column encoding, model 6: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 179

A.18 Row-Column encoding, model 7: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 180

A.19 Row-Column encoding, model 8: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 180

A.20 Row-Column encoding, model 9: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 181

A.21 Direct encoding, MSE: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. . . 181

A.22 Direct encoding, emphasis on non-overlapping region: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 182

A.23 Direct encoding, emphasis on overlapping region: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 182

A.24 Direct encoding, emphasis on predicted overlapping region shape, $\lambda = 5$: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position. 183

ACRONYMS

CNN	Convolutional Neural Network
DNN	Deep Neural Network
FCN	Fully Connected Network
GPU	Graphics Processing Unit
MSE	Mean Squared Error
RELU	Rectified Linear Unit (commonly seen as relu)
RF	Radio Frequency
RGB	Red Green Blue

CHAPTER 1

Introduction

Deep learning's recent rise in popularity began around 2006 and is actually a renaissance of a technology dating back to a 1943 paper by McCulloch and Pitts [1]. Since then, interest in deep learning ebbed and flowed, resulting in many names such as cybernetics, connectionism, and artificial neural networks [2]. In the current resurgence, deep learning has become a buzzword fueled in part by large tech companies, such as Google, Facebook, Twitter, Amazon, and Microsoft, all devoting significant resources to deep learning research in search of solutions to challenging problems.

The emergence in the past few years is not entirely unfounded however. High performing models in computer vision [3], speech processing [4], and natural language processing [5] are all beneficiaries of recent advances in dataset availability and more importantly, hardware availability. Deep learning development is computationally expensive but recently, consumer level graphics processing units (GPU) have made deep learning more accessible to researchers and hobbyists alike. Training a single deep learning model on a GPU may still take days of computation, but training on a CPU may take weeks. Research that took months previously can now be accomplished in weeks.

Deep learning development APIs have played a critical role in the rise in popularity as well. Open source libraries, such as Theano [6], Tensorflow [7], Caffe [8], and Torch7 [9], dramatically simplify the development process by providing a convenient interface to NVIDIA GPUs. Furthermore, high-level wrappers for popular APIs, such as Keras [10], TFLearn [11], and Lasagne [12], simplify the process further, allowing users to define a deep learning model and begin training in a matter of minutes.

Even with the seemingly meteoric rise in popularity, deep learning is still relatively limited to computer vision, speech processing, and natural language processing problems for various reasons including skepticism, researchers' backgrounds, and dataset availability.

If recent results in the aforementioned problems are indicative of potential results in other research fields, then there are tremendous potential benefits to incorporating deep learning in alternative fields.

1.1 Overview of Deep Learning

This section contains a brief overview of basic principles and structures of deep learning. For a more comprehensive overview, refer to [2]. Throughout the remainder of this dissertation, deep learning will be referred to by its alternate name, deep neural networks (DNN).

The fundamental building block in a DNN is the single neuron. Named because of its similarity to biological neurons, the single neuron produces a single output from numerous inputs. This is typically accomplished by computing a nonlinear function on a weighted sum of the neuron's inputs. On a high level, a DNN is a network of layered neurons where the output of one neuron is input into another neuron. Realistically, a single layer is often composed of hundreds of individual neurons and one layer's outputs are all used as inputs for another layer, which also has hundreds of individual neurons. An example is given in figure 1.1. The first layer in a DNN is the input layer, the last layer is the output layer, and every other layer is known as a hidden layer.

Two fundamental types of DNNs are used in this dissertation: a fully connected network and a convolution neural network (CNN). A fully connected network is the quintessential artificial neural network. In every layer, as seen in figure 1.1, the input for every individual neuron is a weighted combination of the previous layer's neuron's outputs. For CNNs, the input of a single neuron is not a weighted combination of every neuron's output from the previous layer; instead, the input resembles traditional convolution and is a weighted combination of a localized region of the outputs of the previous layer. An example of a convolutional layer is given in figure 1.2. A more detailed discussion of convolution layers is left until later in this section.

The nonlinear function used with the individual neurons is also referred to as an activation function, and a variety of activations functions have been used in practice. In this

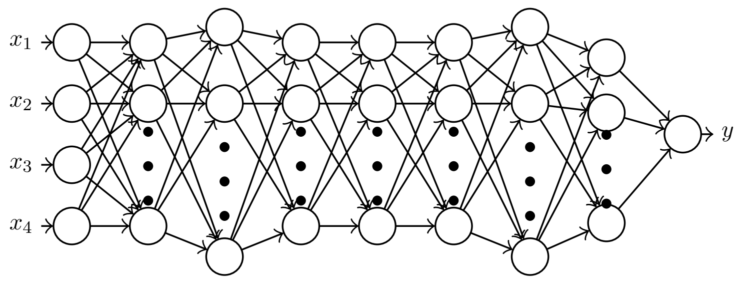


Fig. 1.1: Example of a fully connected network having four input neurons and a single output neuron.

research, three activation functions are used: the sigmoid,

$$\sigma(\mathbf{z})_n = \frac{1}{1 + e^{-z_n}}, \text{ for } n = 1, \ldots, N, \tag{1.1}$$

the softmax,

$$\sigma(\mathbf{z})_n = \frac{e^{z_n}}{\sum_{m=1}^{N} e^{z_m}}, \text{ for } n = 1, \ldots, N, \tag{1.2}$$

and the rectified linear unit (relu),

$$r(\mathbf{z})_n = \max(0, z_n), \text{ for } n = 1, \ldots, N. \tag{1.3}$$

In every function, the input vector $\mathbf{z}$ is a linear combination of the previous layer's outputs, $\mathbf{x}$ and a bias term $\mathbf{b}$ giving

$$\mathbf{z} = W^T \mathbf{x} + \mathbf{b},$$

where W is a matrix of a layer's weights and N is the number of elements in $\mathbf{z}$.

The process of training a DNN having L layers is nearly always a gradient based optimization algorithm implemented to find a matrix W_l for every layer, $l = 1, \ldots, L$ which minimizes some desired function. Most deep learning APIs perform the optimization process using the back-propagation algorithm, which is a recursive implementation of the differentiation chain rule, to provide efficient computation of the necessary gradients. When a DNN is being trained from scratch, it is common practice to randomly initialize the weights in an effort to prevent the weights from converging on identical values; however, it is occasionally desirable for different layers to have identical weights and in these instances the identical weights are referred to as being tied. Additionally, when weights are referred to as being frozen, they are considered constants and non-trainable.

Similar to nearly every machine learning technique, DNNs are trained using either a supervised or unsupervised approach using a dataset of examples. During the training procedure, a DNN computes an output for an individual example and evaluates a loss function using the computed output and the example output. The optimization procedure then adjusts each layer's weights in an attempt to minimize the loss function. Processing each example and finding the best weights requires a lot of computations. To decrease the training time, mini-batches of K examples from the training dataset are often jointly processed instead of individual examples. After the entire mini-batch is processed using the DNN, the weights are adjusted to minimize the average loss function over every example.

Without implementing any sort of protections, over fitting to the training dataset during training is extremely likely. Over fitting occurs when the DNN becomes specialized for the examples in the training dataset and results in a low training loss but poor performance on previously unseen examples. To reduce over fitting, different techniques known as regularizations are implemented.

In this research, three types of regularization are implemented: early stopping, dropout, and L2 kernel regularization. In early stopping, the training dataset is partitioned into training and validation samples. Following every training iteration on the training partition, the loss is computed for the validation dataset. Increasing validation loss and decreasing training loss across a series of training iterations is indicative of over fitting and when this is detected, the training process is terminated. Dropout involves removing different neurons at random during different training iterations and has been shown to be simple and effective at limiting over fitting during training [2]. The L2 kernel regularization places an additional L2-norm constraint on the weights during the updating procedure and has also been shown to be an effective form of regularization [2].

The final concepts in this section are specific to convolution layers. Similar to traditional two dimensional convolution used in image processing, the weights are referred to as kernels. The kernel stride is how many elements are skipped between convolution evaluations. It is the kernels that are learned during the training process. If two dimensional inputs having dimensionality $X \times Y$ are processed, then a convolution layer having K kernels produces a set of $K \times X \times Y$ (assuming zero-padding is implemented during the convolutions) neurons referred to as both activations and feature maps. Convolutional layers are often followed by a pooling layer to condense the outputs. The condensing of the outputs not only down samples the layer's output, but it also provides regularization. How the condensing is accomplished is dependent on the type of pooling layer used. For example, the max pooling approach only returns the maximum value within a neighborhood of outputs. As a result of the neighborhood processing used in pooling, downsampling via pooling is different from downsampling via increased convolution strides. The behavior of a convolutional layer is depicted in figure 1.2.

The deep learning libraries mentioned previously provide an easy API to define complex DNNs and implement the distributed training on GPUs for a variety of standard and custom optimization algorithms to minimize both standard and custom loss functions. Additionally, Tensorflow and Theano both implement automatic differentiation, removing the need for a researcher to hand-calculate the gradients of custom loss functions.

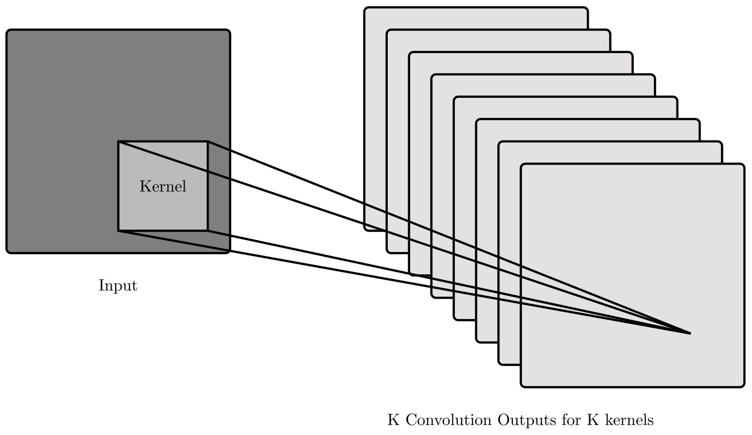


Fig. 1.2: Example CNN with K convolution outputs for K kernels in the layer.

1.2 Dissertation Outline

The research presented in this dissertation addresses the applicability to and implementation of DNNs in both unimodal and multimodal data processing areas. Deep learning in multimodal processing is not a novel idea [13] [14] [15], instead, this research contributes to:

1. Further validate the use of DNNs for multiple input processing outside of speech processing and visual object detection
2. Demonstrate a DNN's output format's effect on learning an association between two inputs
3. Expand existing DNN interpretation techniques to fit different problems

The unimodal problem is introduced in chapter 2, including a discussion of related work and the training and testing datasets. Chapter 3 details three different DNN output encodings and their accompanying DNNs. The training results and DNN comparisons are given in chapter 4, and a discussion of the DNN's ability to learn an association between the inputs is found in chapter 5.

The multimodal problem is developed in chapter 6. Proposed solutions are introduced in chapter 7 and the training results and evaluations are presented in chapter 8.

Chapter 9 is a summary of the contributions of the unimodal and multimodal problems.

CHAPTER 2

Image Overlap Detection Introduction

2.1 Problem Description

The portion of this research emphasizing learning associations between equivalent data modalities focuses on the case when both inputs are images. Specifically, this problem is concerned with the detection of an overlapping region between two input images using deep neural networks (DNN). The overlap detection problem is fundamentally different from the object detection or content classification problems commonly seen in DNN literature and is better framed as an association problem.

To understand the difference between overlap detection and content classification, first consider the object detection or content classification problem using DNNs. An image is input into a DNN, and a class identifier is the output. The class identifier indicates to which class, from a set of N possible classes, the object belongs. In the basic case where object localization is not required, the DNN is used to simply indicate that an object of a specific class exists within the input image. During training, the loss function and target outputs directly encourage the DNN to interpret the contents of a picture. Consequentially, it learns to discriminate between dogs, cats, cars, people, etc.

Identifying an overlapping region between two images increases the degree of difficulty. To illustrate this, let one image be known as the reference image and let another image be known as the test image. Detecting the overlap requires the DNN to not only interpret the content in the reference and test images, but it further requires that the DNN identifies where the content is similar. Furthermore, contrary to object detection where objects of the same class have a similar appearance, the appearance of the overlapping regions is almost guaranteed to be different in every pair of images. For simplicity, consider only having regions of complete overlap so that the reference and test images are always identical.

Throughout an entire dataset, each reference and test image pair will likely be different from the others. So even though the overlapping region is constant, the appearance of the overlapping region is entirely different.

Because the overlap detection problem emphasizes locating where two images are similar, it is a problem better described as an association problem. Given two images, can a DNN associate the content of one with the content of the other? The formulation of the problem this way distances it from the classification and object detection problems; however, the content dependency cannot be entirely decoupled from the problem because the content in the overlapping images needs to match.

The solution to the overlap detection problem developed in the following sections and demonstrated in subsequent chapters uses siamese neural networks using convolutional neural networks (CNN) to learn image representations and fully connected layers to determine the association. Related work is discussed in section 2.2. A new dataset is justified, described, and presented in section 2.3.1. Three different DNN output encodings implemented by 15 individual DNNs are presented as possible solutions to the overlap detection problem in chapter 3. The training results for the 15 networks are discussed in chapter 4 and a discussion on the ability of the DNNs to learn the association between two images is found in chapter 5.

2.2 Related Work

2.2.1 Image Overlap

Analytical solutions to the image overlap problem exist, and one of the most popular approaches is the algorithm presented by Brown and Lowe in [16].The author's analytical approach is extremely successful at finding matching image pairs using extracted SIFT features. Even in the presence of a small amount of rotations, scaling, and perspective shifts, their approach is able to correctly identify overlap between two images.

Instead of using neural networks to learn content descriptors to replace the SIFT features used in the analytical approach of Brown and Lowe, the main focus of this research is concerned with the training of DNNs to learning the location association between the images, not the content.

2.2.2 Convolutional Neural Networks

The recent increase in the popularity of using convolutional neural networks (CNN) for object detection and image classification tasks is also seen in other image processing applications. The trainable convolution kernels of CNNs preserve spatial information in a way that a fully connected layer is unable to match. Additionally, each output neuron of a CNN layer can be traced back to a localized region of neurons in the input layer giving each output neuron a localized reception field. As a result, it is easier to understand what the network is learning than it is when using fully connected layers. A discussion of the approaches to understanding CNNs is left until section 2.2.3. The focus of this section is not on the current state-of-the-art object detection networks. Instead, the focus of this section is on modified CNN architectures and other applications.

Training CNNs can result in repeated convolution kernels. Le et al. propose a reconstruction cost for independent component analysis (ICA) to make learning an over complete feature set using ICA easier [17]. After drawing some connections between ICA and neural networks, they show that their cost function can reduce the repetition of convolution kernels and can also obtain competitive performance in object recognition tasks.

As an alternative to the single column networks which have set multiple benchmarks for various datasets, Cireşan et al. introduce a multiple column DNN [18]. In the multiple column approach, each individual column is its own DNN and the predictions of each column are averaged together. Their reported success isn't entirely unexpected as the authors of [2] point out in section 7.11, "Model averaging is an extremely powerful and reliable method for reducing generalization errors." However, in the same section, the authors caution against using a model averaging technique in benchmark comparisons because "any machine learning algorithm can benefit substantially from model averaging."

Recent developments using CNNs have not been limited to traditional object detection and image classification tasks, but instead address problems further complicated by individual goals and requirements. As an example of further complications, consider the 2011 CNN based image segmentation model proposed by Schulz and Behnke [19]. Image segmentation requires the image to be partitioned into various groups and, in the authors' case, these groups are object classes. This type of segmentation problem not only needs to successfully identify objects in the image, but also their position within the image. Consequently, the target outputs are unable to be the one-hot vectors used by many classification networks because position information must be communicated to the network during training. Instead, an image including every pixel's class ID is used.

More recently, Long et al. proposed the use of a fully convolutional network to perform the class segmentation task [20]. Fully convolutional networks differ from standard CNNs in that the former do not include any fully connected layers, whereas the latter typically have a few on the output which make the final classification decisions. Fully convolutional networks are advantageous because they accept images of any size, making them notably more useful than networks requiring fixed sizes. The authors adapt existing pre-trained models with fully connected layers into fully convolutional layers capable of producing classification maps which can be used to perform the segmentation. To do this, they observe that by viewing a fully connected layer as a convolution kernel covering the entire input, the model can be treated as fully convolutional network accepting any size input. The authors also identify their inclusion of outputs at different stages to incorporate coarse and fine features as a key reason their model simplifies training and speeds up learning while also improving the state-of-the-art performance.

Instead of combining outputs from multiple convolutional layers, Noh et al. propose learning a deconvolution network to perform image segmentation [21]. The approach the authors describe begins like other object detection networks. An image is processed through a series of convolutional layers and max pooling layers resulting in a single vector. Instead of using the single vector as an output like many other networks, the authors proceed to do a series of unpooling and deconvolution layers to reverse the process. The unpooling, an idea similar to upsampling, results in sparse layers which are made dense through the deconvolution operation. The different levels of deconvolution each contribute to the final segmentation decision by providing varying levels of detail.

By modifying previously trained CNNs, Szegedy et al. presented an approach to the object detection problem using regression instead of one-hot vectors [22]. The problem is formulated as a bounding box problem and by replacing the final classification layers of a previously trained CNN with regression layers, the model is able to compute the location of a bounding box. To increase robustness, the authors combine the output of multiple models, each estimating a bounding box, to finalize a bounding box decision. Additionally, they include a discussion of constraints needed on the loss function to prevent the model from learning to only output zeros.

2.2.3 Visualizing CNN Layers

A common complaint against neural networks is the abstraction of what they are learning. They are commonly referred to as black boxes and are often cited as lacking any intuitive understanding. However, many researchers have found success in visualizing and understanding the CNN layers. This field of research attempts to provide insights into a qualitative understanding of the CNN's performance.

Regarding the convolution kernels, Yosinski et al. address what they refer to as a curious phenomenon for CNNs trained on images. They observed that the CNNs all seem to learn similar Gabor like filters on the first convolutional layer [23]. That observation introduces their discussion about the transferability, or the ability of a network trained on one task to work well on another task of CNNs.

Erhan et al. argue that although merely looking at the convolution kernels in the first convolutional layer of a CNN does work, it is insufficient for any arbitrary layer [24]. The authors alternatively demonstrate two related techniques they argue to be better at providing qualitative understanding. Both approaches operate on individual neurons. One approach performs a gradient ascent on the input to find the input which maximizes the output of specific unit. The other method assumes that the traversal of a path from the model's input, through the convolutional layers, to the unit of interest is able to provide some understanding about the unit. They also suggest that taking a linear combination of the previous layers' filters from a specific neuron yields a better understanding of what input will activate that unit rather than simply looking at the layers individually.

Another approach taken to visualize the deeper convolutional layers of CNNs is the proposed use of deconvolution networks by Zeiler and Fergus [25]. After a CNN model is trained, the authors suggest attaching a deconvolution network to each convolution layer. The deconvolution network attempts to provide a path from an individual activation back to the input layer by reversing the pooling and convolution operations. One drawback is that examining a specific activation requires all other activations in the layer to be set to zero prior to passing the values through the deconvolution layers. Like other visualization approaches, this is also able to provide an understanding of what a network is learning in its convolution stages using an input space interpretation.

Instead of learning the deconvolution layers, Simonyan et al. present two deconvolution related gradient-based methods [26]. The first approach generates an input image which maximizes a regularized score function for a desired class by performing a gradient ascent through the input space. Like other visualizations mentioned previously, the interpretation occurs in the input space. The other approach creates class saliency maps for a specific class and a given image by once again implementing a gradient ascent. This approach is different from others in that it does not provide the interpretation in the input space. The fundamental difference between the numerically generated image and the image-specific class saliency map is the former's general representation of a class and the latter's limitation to a specific image. The difference makes both approaches beneficial depending on the desired visualization.

In contrast to much of the CNN visualization work where it is implemented almost purely to understand and evaluate a network, the visualization approach used by Gatys et al. is intended to be the main output [27]. The authors suggest that because subsequent convolutional layers learn more and more detailed representations of the content, the outputs of individual layers are able to represent the higher level content of the input image. The authors also recognize that the overall style, as opposed to the content, of the input image can be visualized by computing correlations between different convolution kernels across multiple layers. In a demonstration of the separable nature of the content and style learned by the CNN, the authors generate a new image containing the style from one image and the content for another. The generated image contains the structure of a photograph but the style of painting.

2.2.4 Siamese Neural Networks

In addition to the object detection and classification networks, CNNs have found a home in image comparison problems. The foundation for many recent solutions to comparison problems using DNNs was laid in 1994 when Bromley et al. introduced the siamese neural network as a solution to a signature verification problem [28]. The siamese neural network is named after its architecture which is composed of two identical branches with joined outputs. For an example of a siamese network, refer to figure 3.1.

A common research problem in computer vision known as patch matching sees significant use of siamese networks. The underlying question in patch matching is if two pictures are pictures of the same subject. What begins as a moderately simple and direct problem is tremendously more complicated when variations in subject illuminance, rotation, viewing angle, and scale are included.

Han et al. implement a CNN based siamese network which they refer to as MatchNet to solve the patch matching problem [29]. Like other CNN based siamese networks, MatchNet has two distinct sections, each performing different tasks, but trained together. The fully connected layers decide if the input images match by computing a similarity score using the image representations learned by the convolutional layers. The authors also discuss the use of MatchNet in a pipeline to increase computational efficiency when two sets of image patches are all extracted from two larger images. This is a regulary occurring scenario arising whenever the images in question have dimensions different from the training images. The proposed pipeline approach recognizes that many of the patch features are used in many comparisons. To avoid recomputing the same features, MatchNet is broken apart into its sections which allows the reuse of the computed patch features.

Another proposed solution to the patch matching problem does not include any fully connected layers. Melekhov et al. train a patch matching network comprised entirely of convolutional layers to specifically learn matching descriptors [30]. The two identical CNN branches produce two feature representation vectors. Instead of feeding the vectors into a fully connected layer, the Euclidean distance between them is calculated and used as a measure of similarity. The Euclidean distance is also used as the cost function during training to encourage the learning of a representation where similar patches have a small distance while dissimilar patches have a large distance.

Another take on on the patch matching problem where the Euclidean distance between the output vectors is used as a cost function is demonstrated by Simo-Serra et al. as part of their effort to learn convolutional feature point descriptors [31]. The goal of their work was not to produce a binary output indicating whether the patches matched or not; instead they wanted to learn features that could be used as a direct replacement for SIFT features used in other algorithms.

Bailer et al. add further variations to the siamese structure in a proposed approach for using patch matching for optical flow [32]. In developing their approach, the authors claim that using the Euclidean distance as a loss function during training has a fundamental flaw. They suggest that training to a unnecessarily small distance for patches that don't match exactly comes at the cost of future pairings. Alternatively, they propose including a threshold value in addition to the Euclidean distance to prevent the training process from trying too hard on some samples. The authors also diverge from the basic siamese models by including multiple CNNs which were trained separately on differently scaled versions of the input image. Prior to making a yes or no decision, the multiple scale features are combined to provide a better decision.

Zagoruyko and Komodakis also implement a siamese network to compare image patches and additionally include two derivatives of the Siamese network [33]. The siamese network presented follows the typical architecture including the two identical CNN branches followed by an output layer which decides if the two input patches are similar or not. The authors also propose a network they call a pseudo-siamese network. Like the typical siamese network, it is composed of two branches having identical architectures. Unlike the siamese network, the weights are not identical. The output of the pseudo-Siamese portion is fed into a fully connected network which handles the decision making process. The authors call their last architecture a two-stream network because it has two parallel streams which are each individual siamese networks. One of the streams processes a central crop from the image patch and the other stream processes a downsampled version of the image patch.

Zbontar and LeCun address the problem of matching images from stereo cameras using siamese networks [34]. The authors identify the matching problem as the first and most expensive step in stereo image processing and propose two different models to use when determining matches. One model emphasizes speed and the other accuracy. Instead of binary decision, the siamese networks compute a similarity metric for individual patches and not the entire image. Like other siamese networks, which only process small selections of an entire image, the authors needed to address some implementation simplifications to avoid redundant computations.

Finally, as a variation to the patch matching problem, the work presented by Chung et al. is concerned with the re-identification of people in various video recordings [35]. The authors present their siamese network as a method for tracking people through multiple videos recorded by multiple cameras with non-overlapping fields of view. Their approach to the problem relies on using both a spatial, and a temporal siamese network. The spatial network operates on raw images and the temporal operates on optical flow stacks. The final output is a weighted combination of both models' outputs

There are some similarities between the image overlap problem presented in section 2.1 and the patch matching solutions discussed in this section. However, there is also a significant difference. The obvious similarity is the need to compare two distinct inputs. The difference is the scale on which the comparisons are calculated. As will be demonstrated later, the solution to the overlap detection problem provides an estimate over the entire input image, and not small patches.

2.2.5 Additional Related Work

The patch matching problem is applicable to various image editing problems. This includes denoising, structural completion (to remove occlusions), and texture synthesis. As it is commonly defined, the patch matching problem requires the finding of the nearest patch from one image that matches another. Nearest can mean many things, so it usually specified using some metric relevant to the problem.

The PatchMatch algorithm introduced by Barnes et al. is intended to be used as an editing tool in image editing software [36], so a focus on speed is necessary. This algorithm attempts to find an approximate nearest neighbor between two image patches and the authors identify the neighbor search as the bottleneck. Their proposed use of random sampling instead of a naive brute force search yields significant speed improvements.

The fast exact nearest patch matching algorithm proposed by Xiao et al. is another attempt to improve the performance during the search process [37]. The authors agree with the observations of other research and cite the search process as the bottleneck of the entire process. Their algorithm is an exact search and not random like the algorithm used in [36].

Both of the algorithms introduced by Barnes et al. in [36] and Xiao et al. in [37] rely on individual yes or no decisions for individual patches. Similar to the approaches in 2.2.4, no effort is made to process entire regions.

Moving on from the patch matching problem to address other examples of multiple input processing, additional work is considered. An example of equivalent modality processing is the proposed approach introduced by Ma et al. to classify solar radio bursts using neural works [38]. Various radio frequency bands are measured and each is considered an individual modality. Deep neural networks are used to learn a joint representation of the modalities to be used when making the classifications. The use of autoencoders by Ma et al. to learn the joint representation on modalities of the same data type demonstrates a potential approach to learning the association.

The analysis performed by Wang et al. compares several proposed techniques for learning features from multiple unlabeled views of data [39]. The challenge of the multi-view problem is that only one view is available during testing. This fundamental goal is to learn as much as possible about a single view by leveraging other views during training. The image overlap detection problem does not have the same test time constraint but must still learn features using multiple input views.

The successful segmentation of remote sensing images using a fully convolutional neural network by Audebert et al. is notable for a couple of reasons [40]. The authors were able to transfer a fully convolutional neural network trained using basic everyday images to use remote sensing images. Additionally the authors joined an additional segmentation network trained on other types of data to the optical image segmentation network to improve the overall segmentation accuracy. In the end, the authors successfully identified the content in various remote sensing datatypes, but they were not focused on learning the association between those datatypes.

Instead of using multiple inputs, the concept of multitask learning introduced in 1997 by Rich Caruana [41] addresses the use of multiple outputs with a single input. Caruana observed that different neural networks sharing the same input domain can be used together to improve overall performance and improve the generalization of neural networks for each task. The idea of multitask learning is applicable to the output encodings introduced in chapter 3.

2.3 Methodology

2.3.1 Data format

Input and Output Target Formats

Because one of the goals of this work is to teach a DNN to predict the overlapping region between two images, using standard red-green-blue (RGB) images as the input format is a natural choice. Choosing the input image dimensions is discussed in section 2.3.1.

Determining a format for the output of the DNN is not as direct as choosing the input format. The output needs to communicate the DNN's identified overlapping region to humans in an intuitive way. To satisfy this requirement, a mapping from output space to input space should be easily determined. Additionally, the output format is also dependent on the dataset used (this is described further in section 2.3.1).

The use of a masking matrix as the output format satisfies the requirements. The use of a mask is motivated by the bounding boxes described in [22] to localize objects in the image. The interpretation of the mask is straight forward. The overlapping region is represented by ones in the mask and non overlapping regions are represented by zeros. Overlaying the masking matrix on one of the images gives a clear indication of the region in the image also found in the other. How to choose the dimensions of the mask is presented in section 2.3.1.

The overlap region is relative to which of the two inputs is used as a reference. For clarity and consistency throughout the remainder of this work, the overlap mask is always overlaid on the same input and that input's image is defined to be the reference image. The image being compared to the reference image is always referred to as the test image.

In summary, the overlap region is represented as a mask of ones and zeros. Overlaying the mask on the reference image will indicate where the reference image overlaps with the test image.

Source Image Dataset

The training and testing datasets were generated using images included in Microsoft's Common Objects in Context (COCO) dataset [42]. The dataset, created by Brown et al. from multiview stereo data [43], used in many of the patch matching approaches discussed in section 2.2.4 cannot be easily adapted to this problem. The dataset contains pairs of images labeled as either matching or not. There isn't any partial matching indicator to be leveraged in adapting this dataset to the image overlap problem. Furthermore, the nature of the images in the dataset are such that they are taken from different angles, scales, and illuminations. The images are not translational images from the same scene, and a prohibitive amount of work would be necessary to even attempt to make them appear as such.

The COCO dataset was chosen for two main reasons. The content of the images is from a broad set of categories and the images have relatively large dimensions. The various categories are beneficial because they provide an additional level of variation in the training and testing datasets. More importantly, however, are the larger image sizes. The reference and test images are cropped from the same source image and the larger images provide more variation when creating the image overlap dataset.

The new image overlap dataset consists of two input images and an output mask. The first image is always the reference image and the second is always the test image. The dimensions of the input images are desired to be comparable to those used in the image classification datasets and depending on the dataset, may be anywhere between 64×64 pixels to 256×256 pixels. Again, The larger images provided by the COCO dataset allow for larger dataset image dimensions while maintaining the flexibility with the two input images' overlap size and orientation.

Choosing the Output Mask Dimensions

The target output of the DNNs is a matrix with $n \times n$ elements. The overlapping region is always rectangular and the size of the overlap is easily defined as the area of the overlapping region. In terms of the overlap mask, this is simply the number of rows containing a one multiplied by the number of columns containing a one. When n is less than the input image dimension, each element in the overlap mask simply represents more than a single pixel in the input space.

The choice of n directly determines the number of possible outputs. If V is defined as the set of valid overlap sizes (in terms of the overlap area), V can be expressed as

$$V = \{a \mid a = j \times k,\ \forall\ j \in [0, n] \text{ and } k \in [0, n]\}, \tag{2.1}$$

with the cardinality of V given by

$$|V| = \binom{n+1}{2} + 1. \tag{2.2}$$

Equation 2.2 is obtained by sampling from j and k without order and with replacement. The upper limit on both j and k is n so the sampling from j and k can be interpreted as choosing two values from the set $[0, n]$. Furthermore, because the product of the two samples is the value of interest, any sample which returns zero is viewed as one. After removing zero from the set, there are n elements left. Therefore, the cardinality is the number of ways to choose two samples from a set of n elements with the addition of one to account for the zero.

When only considering the possible overlap areas as potential outputs, equation 2.2 clearly demonstrates how the number of potential outputs increases as n increases. The relationship between n and the potential overlap areas is also shown graphically in figure 2.1.

The significance of this relationship between n and $|V|$ is best described within the context of the training and testing dataset sizes. If the number samples in each dataset is fixed, then there must be fewer samples to represent each possible overlap area. If the number of samples representing each overlap area is fixed, then the size of the training and testing datasets must increase. This presents a trade off between performance, decision resolution, and speed.

The relationship between n and $|V|$ is beneficial in understanding the requirements on the training and testing datasets, but it is incomplete. The number of ways for two images to overlap (and be represented using the overlap mask) cannot be described entirely only using the overlap area.

Let all possible overlaps be the set of possible outputs, O_{total}. Like $|V|$, an increase in n causes a larger increase in $|O_{total}|$. To demonstrate the relationship between n and the cardinality of O_{total}, five cases and their contributions to O_{total} need to be considered. The five cases encompass the possible overlap orientations that can be represented with the

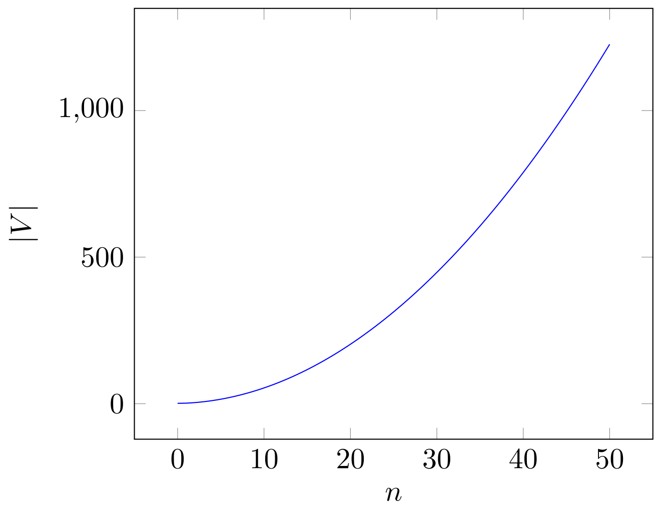


Fig. 2.1: The cardinality of the set of valid overlap sizes, $|V|$, increases as the size of the overlap mask increases.

mask. Unlike the analysis of $|V|$, order matters and the position of the test image relative to the reference image must be considered. These additional considerations are referred to throughout the analysis as overlap orientations. For simplicity, it is assumed the input images are square and that the mask is also square.

Let the mask $M \in \{0,1\}^{n \times n}$ have its overlapping region be defined in terms of its overlap area $a = j \times k$, with j and k being the number of rows and columns in the overlapping region respectively.

For the first case, let $a = 0$. This can only occur when there is no overlap between the two input regions. See figure 2.2 for an example of this overlap type. Though there are infinite possibilities for two images that have no overlap to be arranged, they can all be accounted for when $j = 0$ and/or $k = 0$. Even though there are $2n + 1$ ways $a = 0$, the entire set is treated as a single case. Therefore, for the case $a = 0$, only one possible unique overlap orientation is contributed to O_{total}.

For the second case, let $a = n^2$. This only occurs when $j = k = n$ or alternatively said, when the two input images overlap completely. An illustration of this overlap type is given in figure 2.3. If one of the images is given, only one possible orientation exists for the second image if it is going to be an exact match. Therefore, for the case $a = n^2$, only one

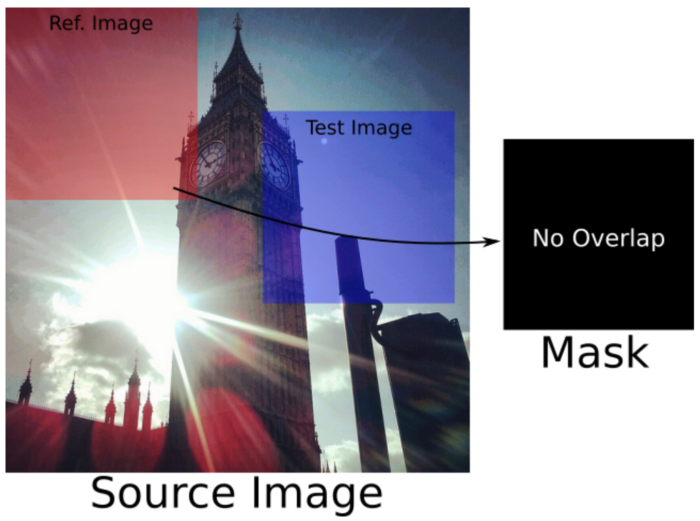


Fig. 2.2: Illustration of the reference image and test image with no overlapping areas. The overlap is defined relative to the reference image. The source image is from the COCO dataset.

possible orientation is contributed.

For the third case, let $a \in (0, n^2)$ but also impose one of the two constraints: 1. $j = n$ and $0 < k < n$ or 2. $k = n$ and $0 < j < n$. This scenario occurs when one of the images is translated horizontally (constraint 1) or vertically (constraint 2), but not both. This scenario is shown in figure 2.4. When each constraint is examined individually, ambiguity in the direction of the shift is present. For constraint 1, the shift may be either left or right and for constraint 2, the shift may be either up or down. Therefore, for the case with $a \in (0, n^2)$ forcing either $j = n$ OR $k = n$, four possible orientations exist; one for each corner of the reference image. And because this case doesn't let $j = n$ AND $k = n$ (case 2), there are $(n - 1)$ occurrences of this case, giving a total contribution of $(n - 1) \times 4$ overlap orientations.

For the fourth case, again let $a \in (0, n^2)$ but now constrain $j = k$ with $0 < j = k < n$. With this constraint, the overlapping region is forced to be a square. This is demonstrated in figure 2.5. Again, the constraints don't provide any information as to which of the four corners of the reference image the overlap occurs in. Therefore, for the case $a \in (0, n^2)$ with $j = k$, four possible orientations exist. And similar to the previous case, $j = k \neq n$.

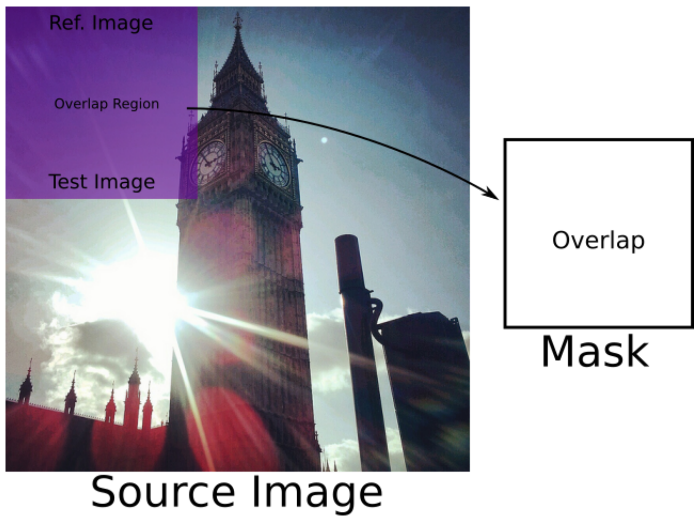


Fig. 2.3: Illustration of complete overlap between the reference image and test image. The overlap is defined relative to the reference image. The source image is from the COCO dataset.

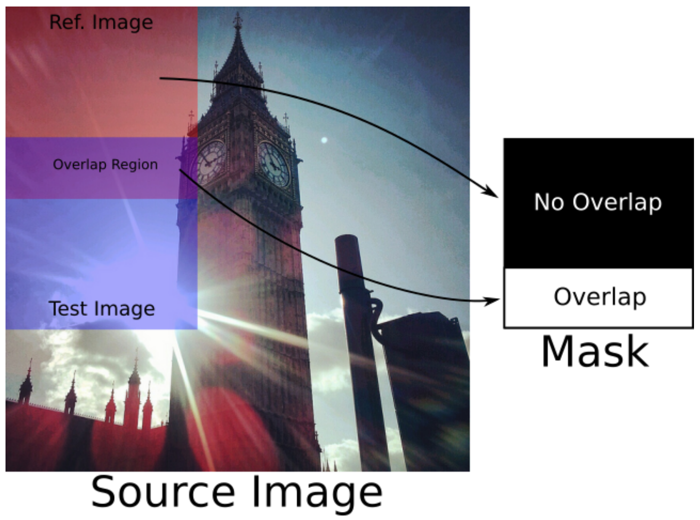


Fig. 2.4: Illustration of complete column overlap between the reference image and test image. The overlap is defined relative to the reference image. The source image is from the COCO dataset.

There are $(n-1)$ ways for this constraint to be met giving this case a total contribution of $(n-1) \times 4$ overlap orientations.

And finally for the fifth case, again let $a \in (0, n^2)$ but now require $0 < j < n$ and $0 < k < n$ with $j \neq k$. Just as in the fourth case, the constraints don't specify which of the four corners of the reference image the overlap occurs in, but now the overlap region is rectangular instead of square. See figure 2.6 for an example of this scenario. When $j > k$, the rectangle is vertical and when $j < k$, the rectangle is horizontal. This difference means that in every corner of the image, two possible orientations exist. Therefore, for the case with $a \in (0, n^2)$ where $0 < j < n$, $0 < k < n$, and $j \neq k$, eight possible orientations exist. The total number of ways for this case to occur is given by $(n-2)(n-1)/2$ which means this case contributes $(n-2)(n-1)/2 \times 8$ possible overlap orientations.

The total number of possible overlaps in O_{total} can be found by adding up the total contributions from all five cases. Adding and simplifying gives

$$|O_{total}| = 4n^2 - 4n + 2. \tag{2.3}$$

The five cases and the contribution to O_{total} are summarized in table 2.1.

As mentioned previously, if n increases then either the size of the dataset must increase, or the number of samples representing each possible output must decrease. The relationship between $|O_{total}|$ and n is shown graphically in figure 2.7.

Table 2.1: Types of overlap and their contributions to the possible orientations.

Overlap Type	# of Orientations	Total For Mask Size $n \times n$
Case 1: None	1 possible orientation	1
Case 2: Complete	1 possible orientation	1
Case 3: Full Row/Column	4 possible orientations	$(n-1) \times 4$
Case 4: Square	4 possible orientations	$(n-1) \times 4$
Case 5: Rectangle	8 possible orientations	$(n-2)(n-1)/2 \times 8$
Total Orientations: $\lvert O_{total}\rvert = 4n^2 - 4n + 2$		

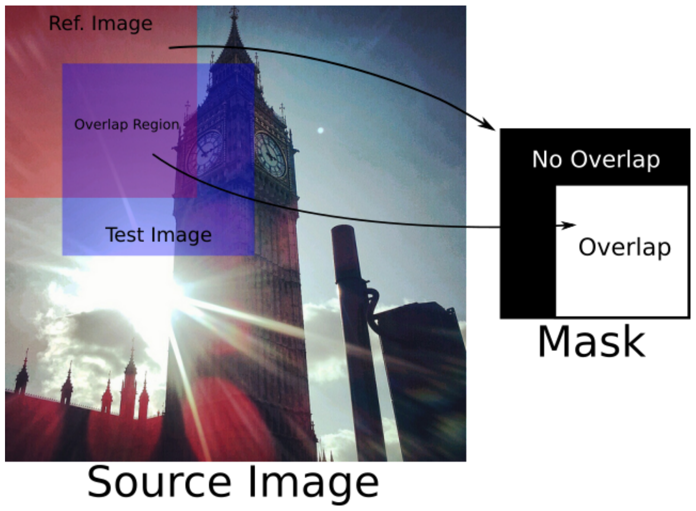


Fig. 2.5: Illustration of the square overlap between the reference image and test image. The overlap is defined relative to the reference image. The source image is from the COCO dataset.

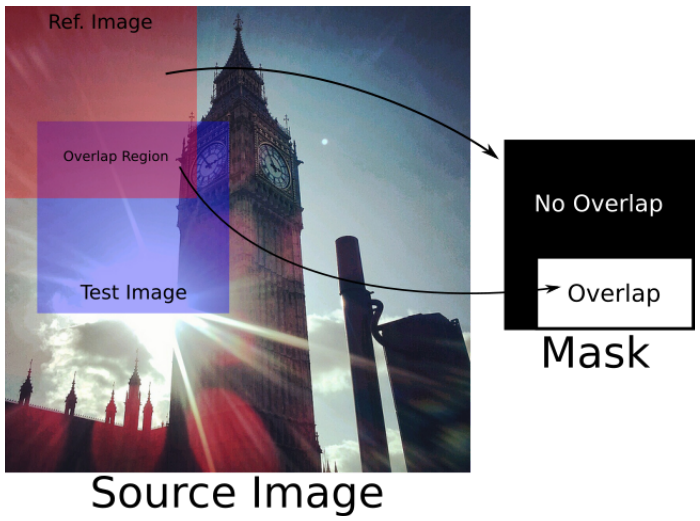


Fig. 2.6: Illustration of the general case of overlap between the reference image and test image. The overlap is defined relative to the reference image. The source image is from the COCO dataset.

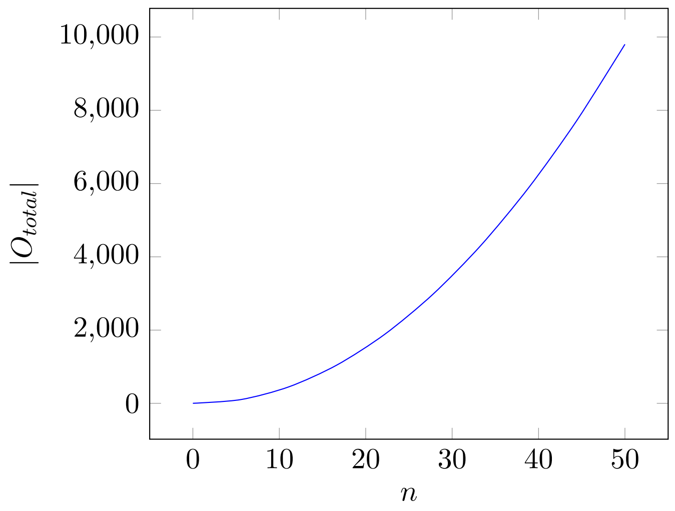


Fig. 2.7: The cardinality of the set of all possible overlap orientations, $|0_{total}|$, increases as the size of the overlap mask increases.

The generated dataset is ideally uniformly distributed over all possible overlaps. However, if care is taken to include all possible orientations, a uniform distribution over V is sufficient when n is small enough. For example, if $a = 20$ then either $j = 4$ and $k = 5$, or $j = 2$ and $k = 10$. Each of these cases will not occur as often as $j = 1$ and $k = 1$ if $a = 1$, but they will still occur regularly enough.

Actual Generated Image Overlap Datasets

The actual dataset generated and used in this research was generated to have an approximately uniform distribution over the overlap area a with enough samples each to provide a sufficient level of coverage across the entire set of all possible overlap types. The dataset is normalized by performing channel normalization across the RGB channels. The means and standard deviations of each channel are:

$$\mu = [\mu_r \mu_g \mu_b] = [126.114311,\ 115.1353372,\ 102.325304128], \tag{2.4}$$

$$\sigma = [\sigma_r \sigma_g \sigma_b] = [76.736191,\ 74.135101,\ 74.71067]. \tag{2.5}$$

The parameters of the generated dataset are summarized in table 2.2.

Table 2.2: Training and testing dataset parameters

Parameter	Parameter Value
Image Size	120×120 pixels
Mask Size	10×10 elements
$\lvert V \rvert$	56
$\lvert O_{total} \rvert$	362
# Training Samples	224,400
# Testing Samples	28,050

The algorithm used to generate the datasets is given in algorithm 1.

Algorithm 1 Algorithm used to produce the image overlap datasets.

```
for Each image in subset of COCO do
    Uniformly sample an overlap type (NE, NW, SE, SW)
    Uniformly sample the position of a reference image
    Uniformly sample from overlap sizes
    Uniformly sample from overlap size factors
    Using overlap type and overlap dimensions, select the test image
    Determine the target output mask from the parameters
    Normalize the reference and test images with $\mu$ and $\sigma$
    Write the reference image, test image, and mask to file
end for
```

Histograms showing the uniformity over the overlap area for the generated training and testing datasets are shown in figures 2.8 and 2.9 respectively.

2.4 Summary

The image overlap problem posed in this chapter, though similar to the patch matching problem has a key difference. Because the solution is intended to process an input image instead of patches, the common datasets cannot be used. The theory outlining the selection of the output mask size was presented and the details of the actual dataset were included. The implementation and analysis of the model training is left for the following chapters.

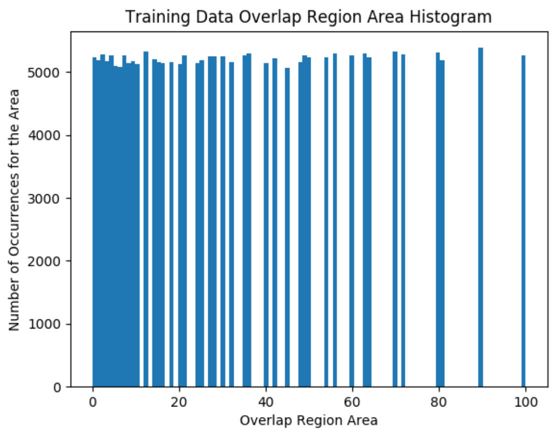


Fig. 2.8: Histogram of generated training dataset. The dataset is uniformly distributed over the set V.

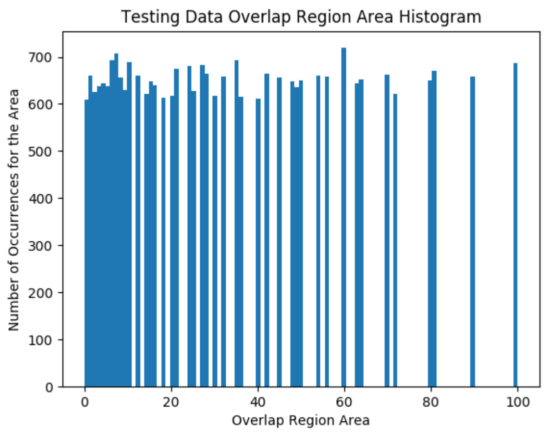


Fig. 2.9: Histogram of generated testing dataset. The dataset is uniformly distributed over the set V.

CHAPTER 3

Image Overlap Detection Experiments

The overlap detection problem was introduced in chapter 2 along with a discussion of the training and testing datasets. In this chapter, 15 different siamese deep neural networks (DNN) are presented. The different networks vary the number of layers, size of the layers, cost functions, and encoding of the overlap mask.

The networks are grouped into three different classes according to the overlap mask encoding used. The three overlap mask encodings are referred to as direct, corners, and row-column. The direct encoding and the the networks using it are presented in section 3.1. The corners encoding the single network using it are found in section 3.2. The details of the row-column encoding, along with its associated models are given in section 3.3. The training details and the model evaluations are presented in chapter 4.

All models presented in this chapter are defined and trained using the Keras neural network API [10] with the Tensorflow [7] backend. All models were trained using the Adam optimizer [44]. To limit overfitting during the training process, early stopping was used as a form of regularization. The use of mini-batches during training also requires the loss functions used to be averaged over the mini-batch.

3.1 Direct Overlap Mask Encoding

In the direct overlap mask encoding, the output mask is simply reshaped into a vector. The direct overlap mask encoding is found by writing the mask M in terms of its column vectors $\mathbf{c}_l$,

$$M = \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \dots & \mathbf{c}_{10} \end{bmatrix} \in \{0,1\}^{10\times 10}, \tag{3.1}$$

and then by stacking the columns of M to form a vector $\mathbf{m}$,

$$\text{vect}(M) = \mathbf{m} = \begin{bmatrix} \mathbf{c}_1^T & \mathbf{c}_2^T & \dots & \mathbf{c}_{10}^T \end{bmatrix}^T \in \{0,1\}^{100\times 1}. \tag{3.2}$$

The vector $\mathbf{m}$ is used directly as the target output of the DNN during training. A high level overview of this approach is shown in figure 3.1.

The models in section 3.1 are all trained using variations of the mean squared error as loss functions. The vector $\mathbf{m}$ is not one-hot encoded (a vector where all elements are zero except for one) so using the categorical cross-entropy as a loss function is not applicable. Categorical cross-entropy is used during the training for many classification tasks and its formulation doesn't make much sense in the overlap vector case. Additionally, the Keras API requires the use of one-hot encoding when using the categorical cross-entropy.

Every model in section 3.1 has the same architecture and nonlinear activation functions. The sigmoid function is used as the nonlinear activation on the output layer, and the rectified linear unit (relu) is used everywhere else. L2 regularization is used on some of the fully connected layers and a dropout layer is included on the second to last layer to further discourage over fitting during training. The detailed architecture is shown in figure 3.2. The numbers indicate how many neurons are present in every layer.

For clarity, the true output vector is the target vector used during supervised training. The predicted output vector is the vector produced by a DNN for a specific input. The predicted outputs are differentiated from the targets through the use of a hat. For example, $\mathbf{x}$ and $\hat{\mathbf{x}}$.

3.1.1 Mean Squared Error

The first loss function discussed is the standard mean squared error. Let the true mask vector be $\mathbf{m}$ and let the predicted mask vector be $\hat{\mathbf{m}}$. The mean squared error loss for an individual sample equals

$$L = \frac{1}{100}||\mathbf{m} - \hat{\mathbf{m}}||_2^2 \tag{3.3}$$

and the average loss over K mini-batch samples equals

$$L_{avg} = \frac{1}{K}\sum_{i=1}^{K}\frac{1}{100}||\mathbf{m}_i - \hat{\mathbf{m}}_i||_2^2. \tag{3.4}$$

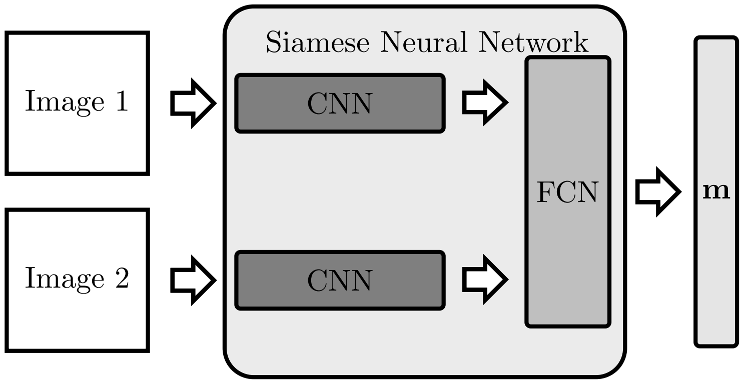


Fig. 3.1: Overview of the DNN structure using the direct overlap mask encoding. Two images are input into the siamese neural network and a vector $\mathbf{m}$ representing the overlap is output.

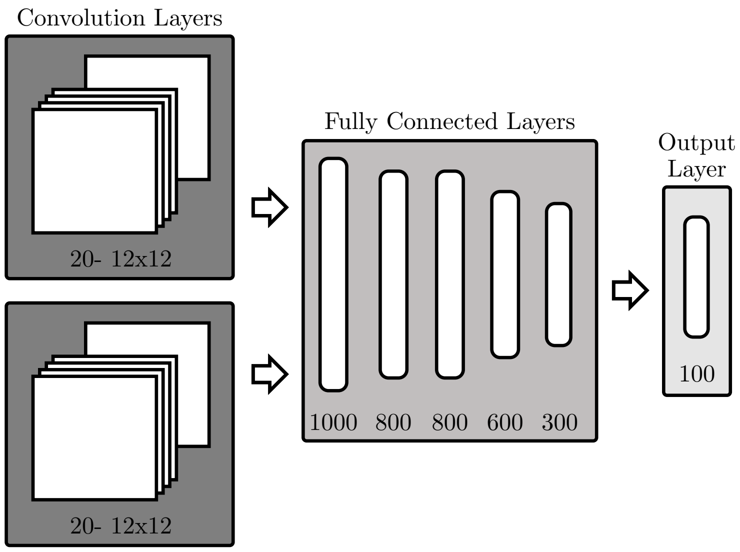


Fig. 3.2: The direct encoding DNN's specific siamese neural network details. The number of neurons in each layer is indicated below the layer.

3.1.2 Penalizing Incorrect Predictions

The true output mask $\mathbf{m} \in \{0,1\}^{100\times 1}$ can be used to penalize the model for incorrect predictions. This can be used to place a higher emphasis on predicting all of the overlapping region correctly or predicting all of the non-overlapping region correctly. Letting the predicted mask be $\hat{\mathbf{m}}$, the prediction error is written

$$\mathbf{e} = (\mathbf{m} - \hat{\mathbf{m}})^{\odot 2}$$

where $\odot 2$ indicates the power is applied element wise.

When the prediction is correct, $m_i \approx \hat{m}_i$ and $e_i \approx 0$ is small and when the prediction is incorrect, $e_i \approx 1$. To penalize the model for predicting $\hat{m}_i \approx 0$ when $m_i = 1$ the elements where $m_i = 0$ can be ignored. Mathematically this is written

$$[(\mathbf{m} - \hat{\mathbf{m}})^{\odot 2}]^T \mathbf{m}.$$

Similarly, to penalize the model for predicting $\hat{m}_i \approx 1$ when $m_i = 0$, the elements where $m_i = 1$ can be ignored. Again, mathematically this is

$$[(\mathbf{m} - \hat{\mathbf{m}})^{\odot 2}]^T (\mathbf{1} - \mathbf{m})$$

where $\mathbf{1}$ is a vector of all ones.

Combining both penalties with weighting factors α and β, the loss function for a single sample equals

$$L = \alpha[(\mathbf{m} - \hat{\mathbf{m}})^{\odot 2}]^T \mathbf{m} + \beta[(\mathbf{m} - \hat{\mathbf{m}})^{\odot 2}]^T [\mathbf{1} - \mathbf{m}] \tag{3.5}$$

and the average loss over K mini-batch samples equals

$$L_{avg} = \frac{1}{K}\sum_{i=1}^{K} \alpha[(\mathbf{m}_i - \hat{\mathbf{m}}_i)^{\odot 2}]^T \mathbf{m}_i + \beta[(\mathbf{m}_i - \hat{\mathbf{m}}_i)^{\odot 2}]^T [\mathbf{1} - \mathbf{m}_i]. \tag{3.6}$$

Higher Emphasis on Non Overlap Regions

Using equation 3.6, it is possible to place a higher emphasis on correctly identifying the non-overlapping region. Setting $\beta > \alpha$ puts more weight on the second term, thus placing the emphasis on correctly predicting the non overlap region at the expense of missing the overlapping region. Letting $\alpha = 1$ and $\beta = 2.5$, the loss function used during training equals

$$L_{avg} = \frac{1}{K}\sum_{i=1}^{K} 1[(\mathbf{m}_i - \hat{\mathbf{m}}_i)^{\odot 2}]^T \mathbf{m}_i + 2.5[(\mathbf{m}_i - \hat{\mathbf{m}}_i)^{\odot 2}]^T [\mathbf{1} - \mathbf{m}_i]. \tag{3.7}$$

Again, a detailed architecture for this model is given in figure 3.2.

Higher Emphasis on Overlap Regions

To place a higher emphasis on correctly identifying the overlapping region at the expense of incorrectly identifying portions of the non-overlapping region, using equation 3.6 let $\alpha > \beta$. With $\alpha = 2.5$ and $\beta = 1$, the loss function used during training equals

$$L_{avg} = \frac{1}{K}\sum_{i=1}^{K} 2.5[(\mathbf{m}_i - \hat{\mathbf{m}}_i)^{\odot 2}]^T \mathbf{m}_i + 1[(\mathbf{m}_i - \hat{\mathbf{m}}_i)^{\odot 2}]^T [\mathbf{1} - \mathbf{m}_i]. \tag{3.8}$$

Again, a detailed architecture for this model is given in figure 3.2.

3.1.3 Forcing the Shape of The Predicted Overlapping Region

The shape of the overlapping area in the overlap mask is always rectangular within the scope of this research. The loss functions presented thus far ignore this constraint but it can be included with a modification. To impose the prior belief that the predicted region should be rectangularly shaped, let the loss function for a single sample be

$$L = \lambda||\mathbf{m} - \hat{\mathbf{m}}||_1 + |\ ||G\hat{\mathbf{m}}||_1 - 1|, \tag{3.9}$$

where $\hat{\mathbf{m}}$ is the predicted overlap mask, and $\mathbf{m}$ is the target overlap mask. The parameter λ is a weighting factor used to adjust the balance between making correct individual predictions and predicting the correct shape. The matrix G is explained below during a discussion of the loss function. The average loss function over the K mini-batch samples equals

$$L_{avg} = \frac{1}{K}\sum_{i=1}^{K} \lambda||\mathbf{m}_i - \hat{\mathbf{m}}_i||_1 + |\ ||G\hat{\mathbf{m}}_i||_1 - 1|. \tag{3.10}$$

Explanation of the Loss Function

The first term in the sum in equation 3.9 encourages the model to learn where the overlap actually occurs by penalizing incorrect individual predictions. The second term encourages the model to predict rectangularly shaped overlap regions. The parameter λ is used to weight the individual element predictions either more or less than the rectangular shaped prediction. The rectangular region constraint, including the matrix G, is developed in the the following paragraphs.

To find G, first define a matrix D as

$$D = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & 0 & \dots & -1 & 1 \end{bmatrix} \in \{0,1\}^{9\times 10}.$$

Left multiplying a matrix by D finds value changes in the columns along the rows of that matrix. Right multiplying by D^T finds value changes in the rows along the columns.

The columns and the rows of the target mask M only have value changes on the edges of the overlapping region. Left and right multiplying M by D and D^T gives an all zero matrix except for a single element equal to one. For example, consider the 3×3 matrix A and the 2×3 matrix D,

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix},\ D = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix},$$

Left multiplying A by D,

$$DA = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix},$$

and then right multiplying DA by D^T,

$$DAD^T = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix}^T = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}.$$

The matrix product DAD^T is nearly all zero except for the single element equal to one. The nonzero element is equal to one because the nonzero region in A is rectangular. If the nonzero region was not rectangular, either the absolute value of the nonzero element would not be one or there would be more than one nonzero element.

Before the matrix product $D\hat{M}D^T$ can be used to enforce the square constraint on the predicted mask $\hat{M}$, it needs to be reformatted for implementation in the Keras API (Tensorflow computes gradients using automatic differentiation so all operations need to be implemented using the Keras functions).

The output of the DNN, $\hat{\mathbf{m}}$, is a vector and the product $D\hat{M}D^T$ uses the matrix version of the output. Therefore, the matrix product needs to be reshaped to be of the form $G\hat{\mathbf{m}}$. Using the Kronecker product,

$$\text{vect}(D\hat{M}D^T) = (D \otimes D)\hat{\mathbf{m}} = G\hat{\mathbf{m}} \tag{3.11}$$

thus giving a value for the matrix G in equation 3.10,

$$G = (D \otimes D).$$

With the matrix G defined, the remainder of the shaping constraint can be discussed. The L1-norm returns the number of nonzero elements in $G^T\hat{\mathbf{m}}$. Without further constraints, minimizing this directly returns the trivial solution where $\mathbf{m}$ is either all zeros or all ones. Taking the absolute value after subtracting one from the L1-norm term avoids the trivial solution and encourages the DNN to predict rectangular regions.

Forcing the Overlap Shape with $\lambda = 5$

The first model using equation 3.10 as the loss function uses $\lambda = 5$. The detailed architecture for this model is the same as the DNNs in this section and is given in figure 3.2. The complete loss function is

$$L_{avg} = \frac{1}{K}\sum_{i=1}^{K} 5||\mathbf{m}_i - \hat{\mathbf{m}}_i||_1 + |\; ||G\hat{\mathbf{m}}_i||_1 - 1|. \tag{3.12}$$

Forcing the Overlap Shape with $\lambda = 10$

The second model using equation 3.10 as the loss function uses $\lambda = 10$. Again, the detailed architecture for this model is given in figure 3.2. The complete loss function is

$$L_{avg} = \frac{1}{K}\sum_{i=1}^{K} 10||\mathbf{m}_i - \hat{\mathbf{m}}_i||_1 + |\; ||G\hat{\mathbf{m}}_i||_1 - 1|. \tag{3.13}$$

3.2 Corners Overlap Mask Encoding

The second encoding considered in this chapter is the corner overlap mask encoding (also referred to simply as the corner encoding). Instead of predicting every element in the overlap mask, this encoding uses two output vectors to uniquely encode the overlap region in the overlap mask via two corner predictions. The corner encoding is illustrated in figure 3.3.

The two output vectors used in the corner encoding are $\mathbf{a} \in \{0,1\}^{4\times1}$ and $\mathbf{b} \in \{0,1\}^{100\times1}$. Each element in $\mathbf{a}$ represents one of the four corners in the overlap mask, and each element in $\mathbf{b}$ represents one of the other 100 elements in the overlap mask. Using both $\mathbf{a}$ and $\mathbf{b}$, all possible image overlap types can be represented. The overlapping region, if it exists, will always include one of the four corners represented by $\mathbf{a}$. The opposite corner of the overlapping region is represented by $\mathbf{b}$. The overlapping region is created by setting the elements the overlap mask that are between the two predicted corners to one.

$$\begin{array}{cc}
\mathbf{a} & \begin{bmatrix}0\\0\\0\\1\end{bmatrix} \\
\mathbf{b} & \begin{bmatrix}0\\ \vdots \\ 1 \\ \vdots \\ 0\end{bmatrix}
\end{array}
\begin{array}{c}
M \\
\begin{bmatrix}
a_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & b_k & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
a_2 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & a_3
\end{bmatrix}
\end{array}$$

Fig. 3.3: Illustration of the corner overlap encoding. One corner is provided by $\mathbf{a} = [a_0,\ a_1,\ a_2,\ ,a_3]$ and the other by $\mathbf{b} = [b_0,\ b_1,\ \dots\ ,b_{99}]$. In this example, $a_3 = 1$ and $b_k = 1$. The region between them is filled in to get the overlap mask.

With two output vectors, the general DNN architecture used in this section is different than that of section 3.1 and is shown for reference in figure 3.4.

The output vectors of this model are **a** and **b** and are one-hot encoded. Because they use one-hot encoding, the loss function used is a weighted sum of the categorical cross-entropy of both **a** and **b**. The single sample loss equals

$$L = L_a + 2.75 L_b \tag{3.14}$$

where

$$L_a = -\frac{1}{4}\sum_{n=1}^{4}[a_n log(\hat{a}_n) + (1 - a_n) log(1 - \hat{a}_n)], \tag{3.15}$$

$$L_b = -\frac{1}{100}\sum_{n=1}^{100}[b_n log(\hat{b}_n) + (1 - b_n) log(1 - \hat{b}_n)], \tag{3.16}$$

and $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$ are defined as the predicted corner vectors. The average loss over the K mini-batch samples equals

$$L_{avg} = \frac{1}{K}\sum_{i=1}^{K} L_a^{(i)} + 2.75 L_b^{(i)}. \tag{3.17}$$

Corner encoding is only implemented in one DNN (see section 4.3.2 for an explanation

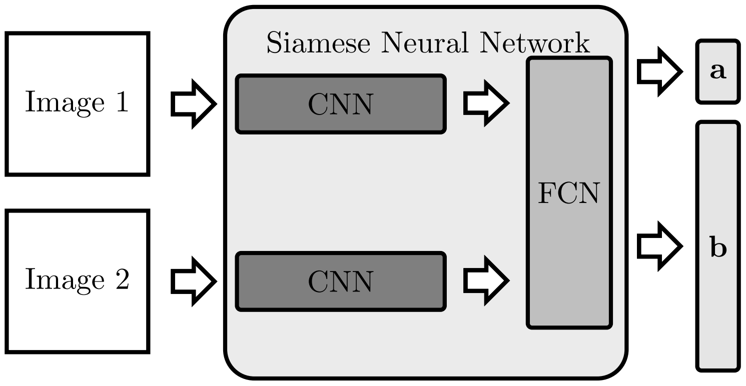


Fig. 3.4: Overview of the DNN structure using the corner overlap mask encoding. Two images are input into the siamese neural network and the vectors **a** and **b** indicating which corners of the overlapping region are output.

of this decision). The architecture details for single DNN are given in figure 3.5. The softmax function is used as the nonlinear activation function on the output layers and the rectified linear unit (relu) is used with every other neuron in the network.

3.3 Row-Column Overlap

The final encoding considered in this research is the row-column encoding. Like the encoding described in section 3.2, this encoding uses two vectors to represent the entire overlap mask. This time, however, the two vectors represent the rows and columns of the mask. An example of the row-column encoding is shown in figure 3.6. The nonzero elements in **r** indicate which rows of the overlap mask the overlapping region is in. Similarly, the nonzero elements in **c** indicate which columns of the overlap mask the overlapping region is in. Using this encoding, all possible overlap types can be represented using only two 10-element vectors.

The specific details of every DNN in this section vary but they all share the same general architecture shown in figure 3.7. Neither **r** nor **c** are one-hot encoded vectors. Consequently, the models presented in this section all use the mean squared error as their loss function instead of categorical cross-entropy. The individual sample loss, accounting

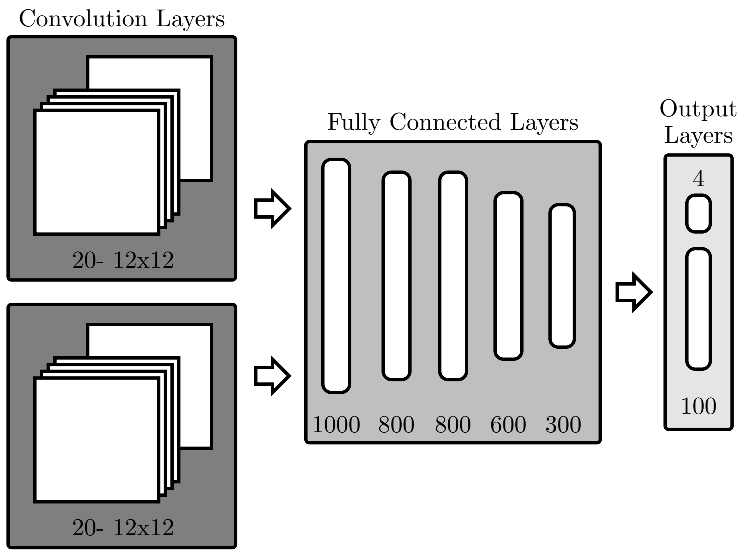


Fig. 3.5: The architecture of the siamese network using the corner encoding. The number of output neurons in each layer is indicated by the number below the layer.

$$\mathbf{r}\begin{bmatrix}0\\0\\1\\1\\1\\1\\1\\1\\1\\1\end{bmatrix}\quad \overset{M}{\begin{bmatrix}0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0\\0&1&1&1&1&1&1&1&1&1\\0&1&1&1&1&1&1&1&1&1\\0&1&1&1&1&1&1&1&1&1\\0&1&1&1&1&1&1&1&1&1\\0&1&1&1&1&1&1&1&1&1\\0&1&1&1&1&1&1&1&1&1\\0&1&1&1&1&1&1&1&1&1\\0&1&1&1&1&1&1&1&1&1\end{bmatrix}}$$

$$\underset{\mathbf{c}}{\begin{bmatrix}0&1&1&1&1&1&1&1&1&1\end{bmatrix}}$$

Fig. 3.6: Example of the row-column overlap mask encoding. The elements of **r** are used to indicate whether or not overlap occurs in each specific row. Similarly for the elements of **c**, each element indicates if overlap exists in its represented column.

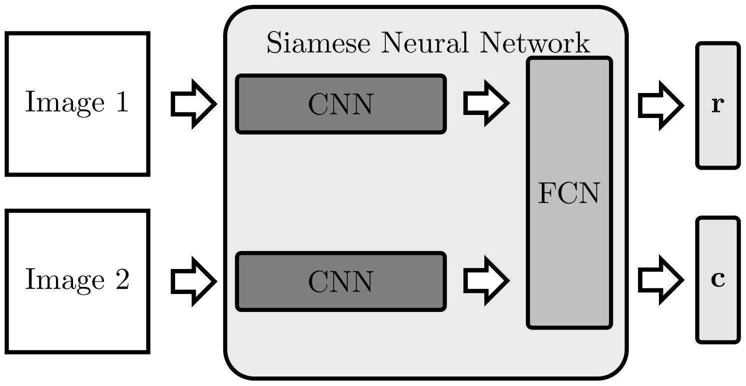


Fig. 3.7: The overview of the DNN structure using the row-column overlap mask encoding. Two images are input into the siamese neural network and the vectors **r** and **c** indicating which rows and columns contain the overlapping region are output.

for both output vectors, equals

$$\text{Loss} = \frac{1}{10}(||\mathbf{r} - \hat{\mathbf{r}}||_2^2 + ||\mathbf{c} - \hat{\mathbf{c}}||_2^2) \tag{3.18}$$

where $\hat{\mathbf{r}}$ and $\hat{\mathbf{c}}$ are the predicted row and column vectors respectively. The average loss over the K mini-batch samples equals

$$L_{avg} = \frac{1}{K}\sum_{i=1}^{K}\frac{1}{10}(||\mathbf{r}_i - \hat{\mathbf{r}}_i||_2^2 + ||\mathbf{c}_i - \hat{\mathbf{c}}_i||_2^2). \tag{3.19}$$

3.3.1 Model 1

Model 1 uses the sigmoid function as the nonlinear activation function on the final output layer of both the **r** and **c** branches. The rectified linear unit (relu) is used in every other layer. The loss function is given by equation 3.19, and the detailed architecture is shown in figure 3.8.

3.3.2 Model 2

Model 2 uses the sigmoid function as the nonlinear activation function on the final output layer of both the **r** and **c** branches. The rectified linear unit (relu) is used in every other layer. The loss function is given by equation 3.19. The detailed architecture is shown

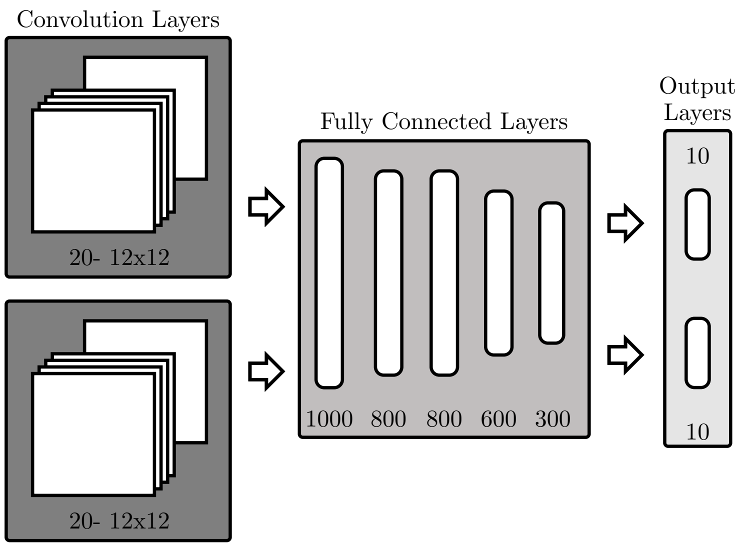


Fig. 3.8: The architecture of the first siamese network (Model 1) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer.

in figure 3.9. Model 2 is similar to model 1 with the addition of another output layer.

3.3.3 Model 3

Model 3 uses the sigmoid function as the nonlinear activation function on the final output layer of both the **r** and **c** branches. The rectified linear unit (relu) is used in every other layer. The loss function is given by equation 3.19. The detailed architecture is shown in figure 3.10. Model 3 includes an additional convolution layer and smaller kernels. The convolutional layers output neurons have the same field of view as those in Models 1 and 2.

3.3.4 Model 4

Model 4 uses the sigmoid function as the nonlinear activation function on the final output layer of both the **r** and **c** branches. The rectified linear unit (relu) is used in every other layer. The loss function is given by equation 3.19. The detailed architecture is shown in figure 3.11. The capacity of this network is increased over previous models by adding more fully connected layers, output layers, and convolution kernels.

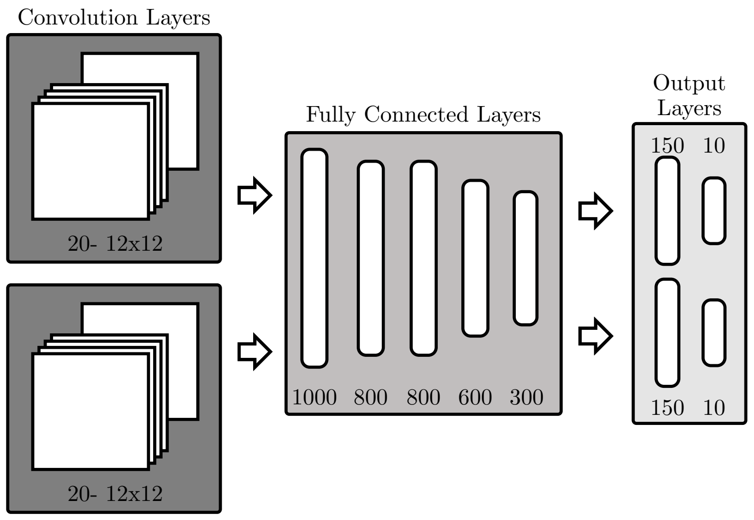


Fig. 3.9: The architecture of the second siamese network (Model 2) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer.

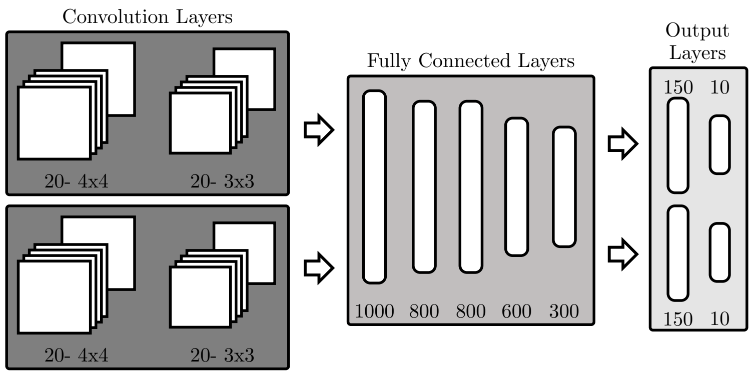


Fig. 3.10: The architecture of the third siamese network using (Model 3) the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer.

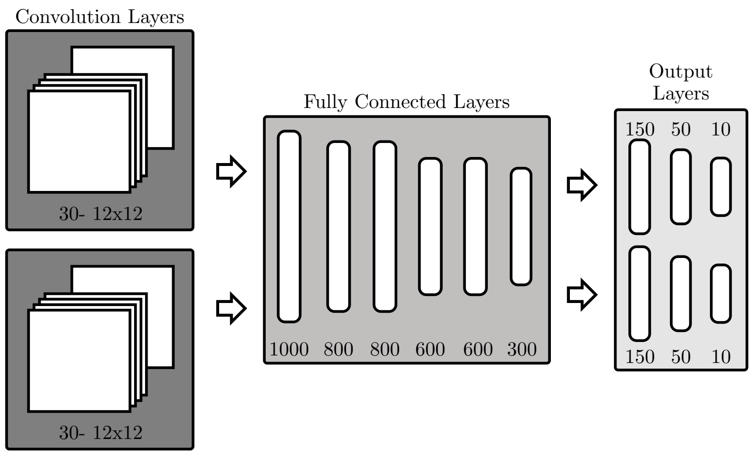


Fig. 3.11: The architecture of the fourth siamese network (Model 4) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer.

3.3.5 Model 5

Model 5 uses the sigmoid function as the nonlinear activation function on the final output layer of both the **r** and **c** branches. The rectified linear unit (relu) is used in every other layer. The loss function is given by equation 3.19. The detailed architecture is shown in figure 3.12. Model 5 maintains the higher capacity introduced in model 4 and includes an additional convolution layer while reducing the size of the kernels.

3.3.6 Model 6

Model 6 uses the sigmoid function as the nonlinear activation function on the final

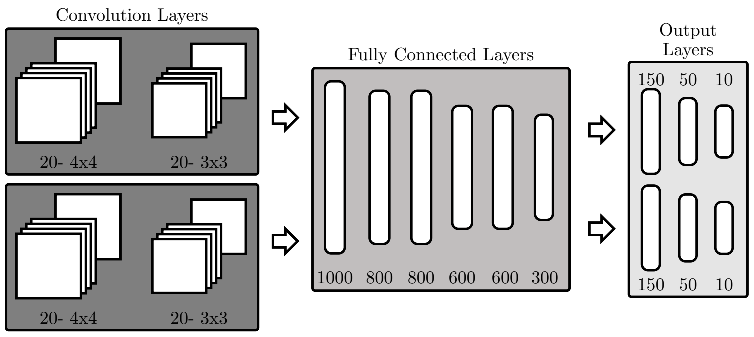


Fig. 3.12: The architecture of the fifth siamese network (Model 5) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer.

output layer of both the **r** and **c** branches. The rectified linear unit (relu) is used in every other layer. The loss function is given by equation 3.19. The detailed architecture is shown in figure 3.13. Model 6 reduces the network capacity by reducing the number of convolution kernels.

3.3.7 Model 7

Model 7 uses the sigmoid function as the nonlinear activation function on the final output layer of both the **r** and **c** branches. The rectified linear unit (relu) is used in every other layer. The loss function is given by equation 3.19. The detailed architecture is shown in figure 3.14. Model 7 further reduces the capacity of Model 6 by removing an output layer.

3.3.8 Model 8

Model 8 has the same architecture as Model 2 but the convolution kernel weights were frozen. In other words, the weights were initialized randomly but not changed during training. Model 8 still uses the sigmoid function as the nonlinear activation function on the final output layer of both the **r** and **c** branches. The rectified linear unit (relu) is still used in every other layer. The loss function is given by equation 3.19. The detailed architecture is shown in figure 3.15.

3.3.9 Model 9

Model 9 uses the sigmoid function as the nonlinear activation function on the final output layer of both the **r** and **c** branches. The rectified linear unit (relu) is used in every other layer. The loss function is given by equation 3.19. The detailed architecture is shown in figure 3.15. Model 9 reduces the network capacity of model 7 by removing layers and removing neurons from some of the layers.

3.4 Models Summary

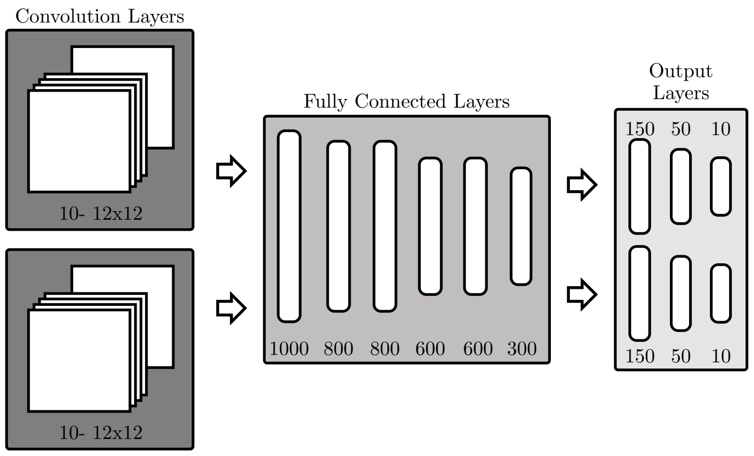


Fig. 3.13: The architecture of the sixth siamese network (Model 6) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer.

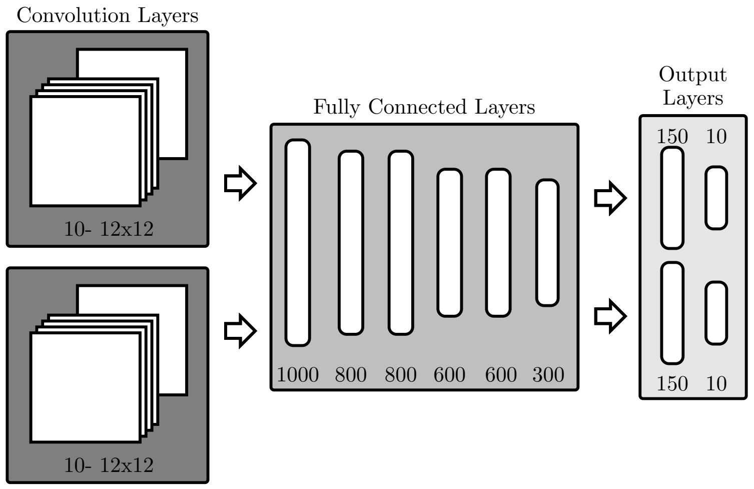


Fig. 3.14: The architecture of the seventh siamese network (Model 7) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer.

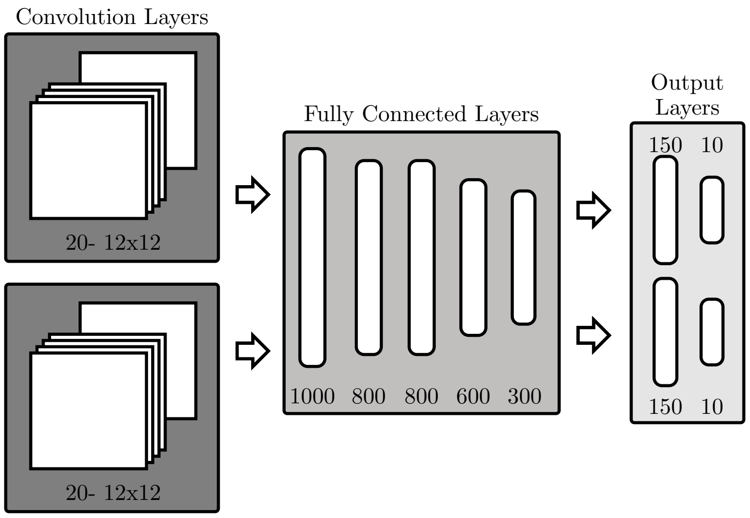


Fig. 3.15: The architecture of the eighth siamese network (Model 8) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer.

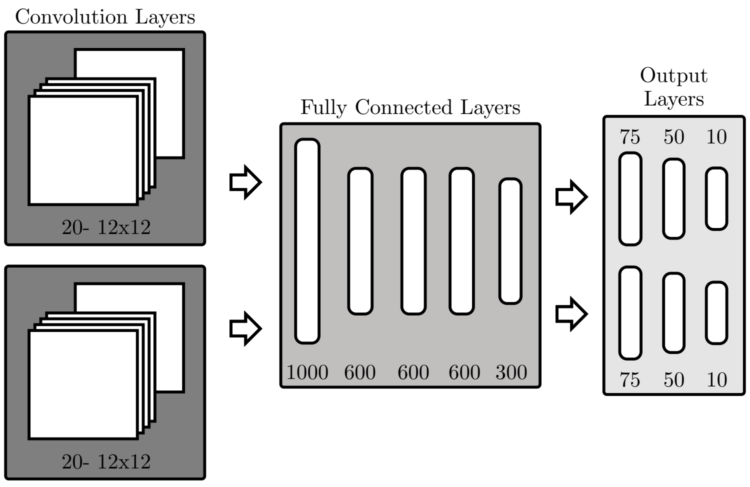


Fig. 3.16: The architecture of the ninth siamese network (Model 9) using the row-column encoding. The number of output neurons in each layer is indicated by the number below the layer.

In this chapter, three different overlap mask encodings were presented: direct, corner, and row-column. Details for a total of 15 different DNN models, each using one of the three encodings, were given. The details of all 15 models are summarized in table 3.1. The training results of all 15 models are the subject of chapter 4. A discussion of the 15 models' ability to learn the association between the two input images is in chapter 5.

Table 3.1: Summary of image overlap detection DNNs, including the number of neurons in each layer.

Name	CNN Layers	FCN Body	Output Layers
MSE	20- 12x12	[1000, 800, 800, 600, 300]	100
$\alpha = 1$, $\beta = 2.5$	20- 12x12	[1000, 800, 800, 600, 300]	100
$\alpha = 2.5$, $\beta = 1$	20- 12x12	[1000, 800, 800, 600, 300]	100
Force, $\lambda = 5$	20- 12x12	[1000, 800, 800, 600, 300]	100
Force, $\lambda = 10$	20- 12x12	[1000, 800, 800, 600, 300]	100
2 corners	20- 12x12	[1000, 800, 800, 600, 300]	[100], [4]
RC 1	20- 12x12	[1000, 800, 800, 600, 300]	2 x [10]
RC 2	20- 12x12	[1000, 800, 800, 600, 300]	2 x [150, 10]
RC 3	[20- 4x4, 20- 3x3]	[1000, 800, 800, 600, 300]	2 x [150, 10]
RC 4	[30- 12x12]	[1000, 800, 800, 600, 600, 300]	2 x [150, 50, 10]
RC 5	[20- 4x4, 20- 3x3]	[1000, 800, 800, 600, 600, 300]	2 x [150, 50, 10]
RC 6	[10- 12x12]	[1000, 800, 800, 600, 600, 300]	2 x [150, 50, 10]
RC 7	[10- 12x12]	[1000, 800, 800, 600, 600, 300]	2 x [150, 10]
RC 8	20- 12x12 (FROZEN)	[1000, 800, 800, 600, 300]	2 x [150, 10]
RC 9	20- 12x12	[1000, 600, 600, 600, 300]	2 x [75, 50, 10]

CHAPTER 4

Image Overlap Detection Results

The 15 models presented in chapter 3 were all trained using the same training dataset, and they were all tested using the same testing dataset. The training dataset and testing datasets are different and were generated using algorithm 2. The training dataset consists of 224,400 reference/test image pairs, and their true overlap mask. The testing dataset consists of 28,050 reference/test image pairs, and their true overlap mask.

Before a direct comparison between the 15 models can happen, an additional metric needs to be specified. Nearly every model was training using a different loss function and no simple, meaningful relationship between the various loss functions exists. Therefore, the additional comparison metric used in this chapter is the prediction accuracy.

The prediction accuracy is defined as the percentage of correctly predicted elements in the overlap mask. Every model uses the same size mask, so the comparison between models is simple and intuitive. The overlap mask is 10×10 elements and, as a result, the prediction accuracy is also just a count of the number of elements correctly predicted. A prediction is considered correct when a zero in the mask is predicted to be zero or when a one in the mask is predicted to be a one.

Two additional models not presented previously are included as references for the performance of the trained models. The new models demonstrate two training extremes and provide insight into the performance of the other models. One of the models only predicts an all ones overlap mask, and the other only predicts the all zeros overlap mask.

The rest of the chapter is as follows. Graphics used for comparisons, in addition to the prediction accuracy, are included in section 4.1. Example graphics using the all ones and all zeros prediction models are given in section 4.2. The results for the 15 trained models are presented and discussed in section 4.3.

4.1 Figures for Evaluation

The image overlap problem is complex and condensing the prediction performance of a model down to a single number is an altogether inadequate means for comparison and results interpretation. The average prediction accuracy is capable of providing a high-level comparison between models, but it lacks the detail necessary to understand performance over a range of variables. The graphics presented in this section all use the prediction accuracy but include additional variables to provide a more complete representation of the DNN's performance.

Prediction Accuracy Histogram

Once a model processes the entire testing dataset, a histogram is constructed using the prediction accuracies. The histogram shows an approximate distribution for the model's prediction accuracy across the entire dataset.

Overlap Area vs. Prediction Accuracy

The overlap area vs. prediction accuracy plot shows the average prediction accuracies for all testing samples with the same amount of overlap. To fully evaluate a model, an understanding of how it performs over all different amounts of overlap is required. The histogram alone cannot decouple the prediction accuracy from the size of the overlapping area. To illustrate this, consider the scenario where only a single corner of the overlap mask is one. If the predicted mask is all zeros, except for a one in the corner opposite of the true corner, the prediction accuracy will be 98%. Even though the prediction accuracy is nearly perfect, the identified overlapping region is far from correct as the opposing corner was selected.

Detailed Histogram

The detailed histogram shows how well the model performs on all sizes of overlaps. This is accomplished by including the prediction accuracy histogram and the overlap area vs. prediction accuracy plots in a single graphic. This graphic shows how well the model predicts the overlap mask for overlaps of all sizes. Instead of averaging the prediction accuracies for every overlap size, a histogram of the prediction accuracies is created. The number of occurrences for each prediction accuracy and overlap size are represented via the pixel intensities. A color bar provides the relationship between the pixel intensities and number of occurrences.

4.2 All Ones and All Zeros Predicting Models

Prior to presenting and discussing the results of the 15 trained DNNs, a brief examination of models which only predict the all ones overlap mask or the all zeros overlap mask is beneficial. Using models with known behavior to demonstrate the figures from section 4.1 helps provide intuition necessary for understanding the behavior of the trained models.

First, consider the prediction accuracy histogram for the all ones predicting model in figure 4.1 and also the histogram for the all zeros predicting model in figure 4.2. At first glance it may seem incorrect for the all ones model to have an average accuracy of 34% while the all zeros model has an average accuracy of 66%. However, the results make sense when the actual distribution of the testing dataset is considered. Taking a closer look at the testing dataset histogram introduced in figure 2.9 reveals there is a much higher density of low area overlaps than high area overlaps. As a reminder, the samples in the dataset were chosen to be approximately uniform over the elements of the set of valid overlaps V (see equation 2.1), not the area itself. In the set V, there are simply more elements having smaller areas; ergo, a DNN always predicting ones will be incorrect more often than a DNN always predicting zeros.

Second, consider the accuracy vs. overlapping area plots for both the all ones predicting DNN in figure 4.3 and the all zeros predicting DNN in figure 4.4. While the histograms discussed previously show how the models perform across the entire dataset, these figures convey the performance of the models for different amounts of overlap. As expected, the prediction accuracy of the all ones predicting DNN increases linearly with the size of the overlap and conversely, the all zeros predicting DNN decreases linearly with the size of the overlap.

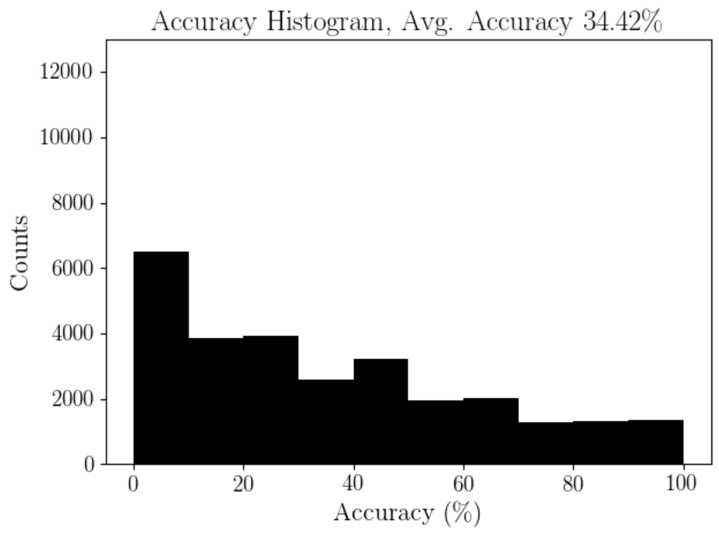


Fig. 4.1: Histogram of prediction accuracies for the DNN predicting all ones.

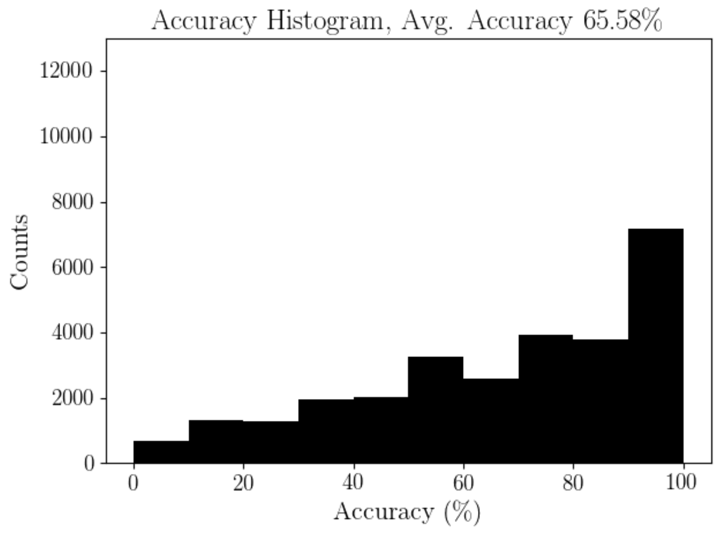


Fig. 4.2: Histogram of prediction accuracies for the DNN predicting all zeros.

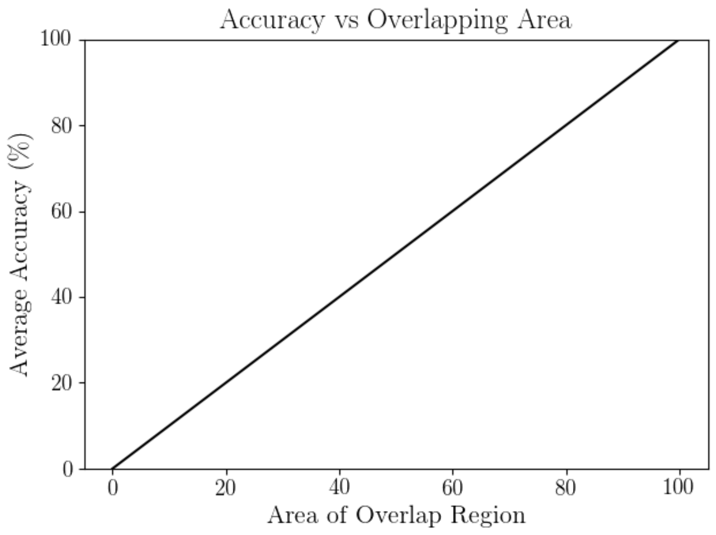


Fig. 4.3: The average prediction accuracy for the possible sizes of overlapping regions for the DNN predicting all ones.

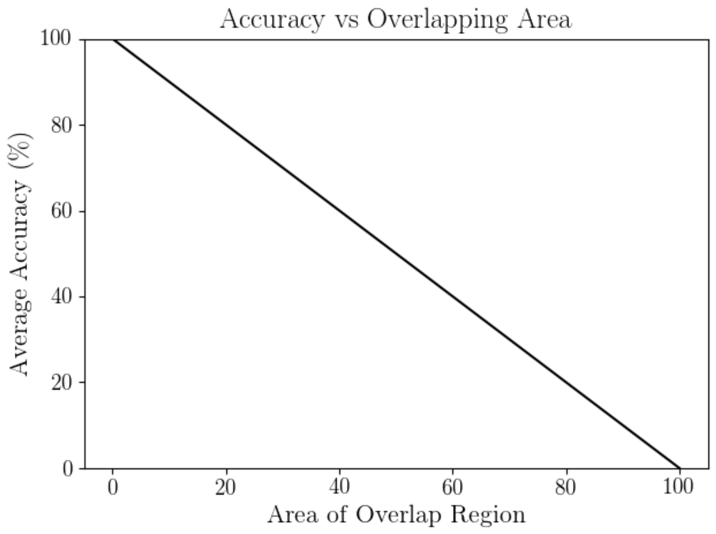


Fig. 4.4: The average prediction accuracy for the possible sizes of overlapping regions for the DNN predicting all zeros.

And third, examine the detailed histograms for the all ones predicting model and the all zeros predicting model shown in figures 4.5 and 4.6 respectively. The results in these figures once again line up well with the other figures. In both detailed histograms there is a linear relationship between the accuracy and the area of the overlapping region. Additionally, the imbalance of low and high overlap area occurrences is clearly visible.

In summary, a good performing model exhibits these characteristics in the three different figures:

1. The histogram is heavily skewed to the right (high prediction accuracies)
2. The prediction accuracy is similar for all amounts of overlap (i.e. independent of overlap area size)

The detailed histogram contains the same information as the other two plots, but they are still included because they both present a quick and simple overall picture of the performance.

4.3 Training Results

The training results for the 15 trained models are presented according the overlap mask encoding used. Initially, performance comparisons are limited to models using the same encoding but a general comparison is provided in section 4.4.

4.3.1 Direct Encoding

MSE

The DNN trained by minimizing only the mean squared error (the architecture is shown in figure 3.2) is able to predict the overlap mask with an average prediction accuracy of 75.77% across the testing dataset.

The histogram in figure 4.7 clearly indicates something is being learned other than the trivial cases introduced in section 4.2. The histogram is skewed towards the high accuracy

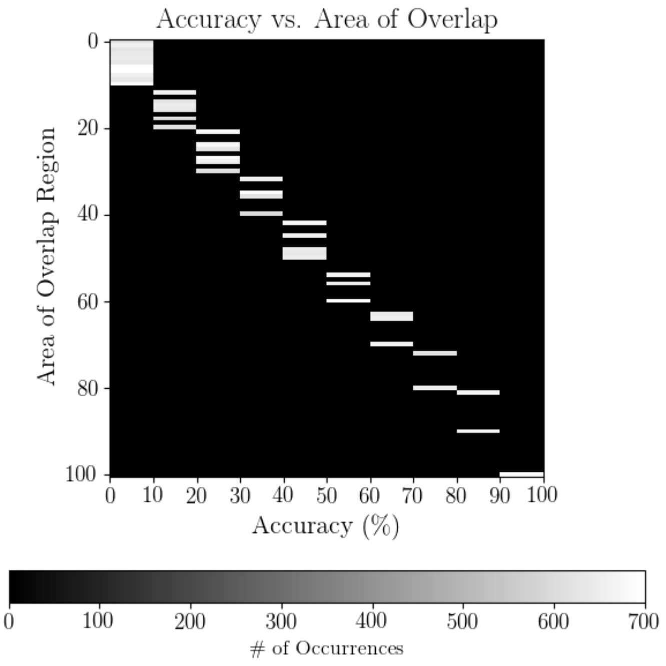


Fig. 4.5: Detailed histogram of prediction accuracies and the area of the overlapping regions for the all ones prediction models. The number of occurrences of each is indicated by the pixel intensity.

side and correctly predicts over 90% of the overlap mask in approximately 20% of the samples.

In figure 4.8, the average prediction accuracy is over 80% for samples with both small and large amounts of overlap. The poorest performance occurs when the amount of overlap and non-overlap are roughly equal. These results imply the average prediction accuracy of 75.77% is not inflated by a model performing superbly on a small subset of overlap sizes and poorly otherwise.

The results seen previously are confirmed using the detailed histogram in figure 4.9. Furthermore, it adds some clarity the other figures lacked. The highest performance occurs when the reference image and test images are either the same image (100% overlap) or immensely different (<10% overlap). When the reference and test images are just as similar as dissimilar the DNN manages anywhere between 50-80% prediction accuracy. The model is performing better than guessing, but it many cases, not by much.

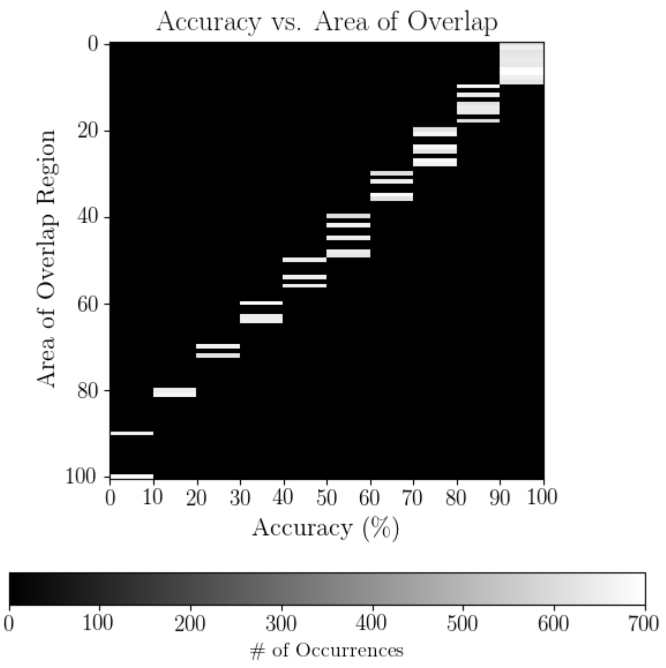


Fig. 4.6: Detailed histogram of prediction accuracies and the area of the overlapping regions for the all zeros prediction models. The number of occurrences of each is indicated by the pixel intensity.

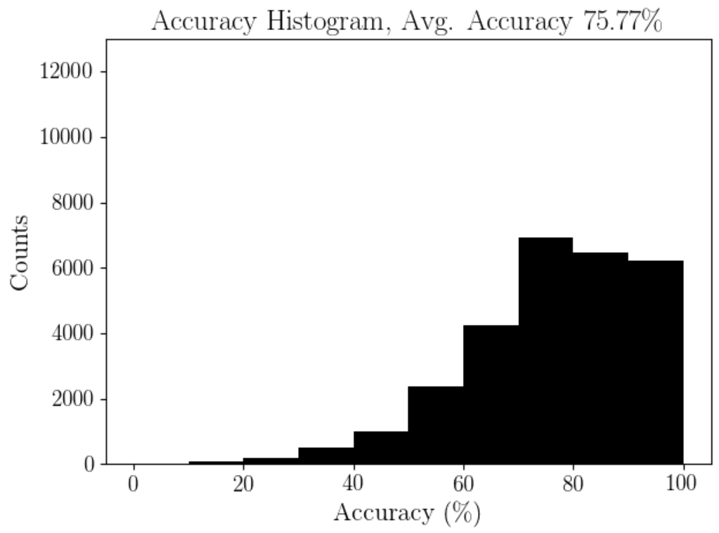


Fig. 4.7: Histogram of the prediction accuracies for the DNN with the mean squared error loss function.

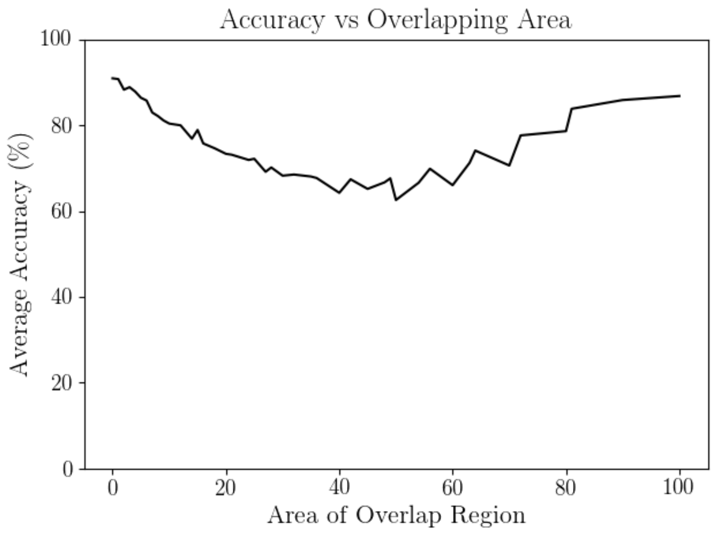


Fig. 4.8: The average prediction accuracy for the possible sizes of overlapping regions for the DNN with the mean squared error loss function.

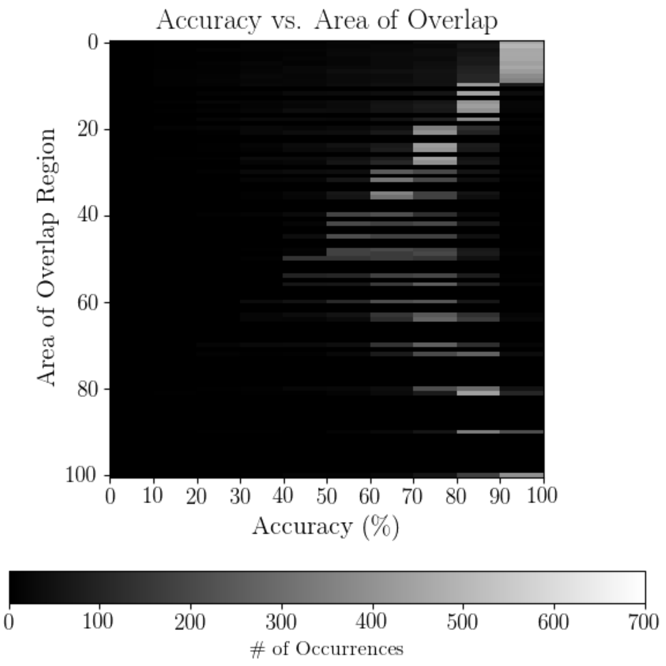


Fig. 4.9: Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN with the mean squared error loss function. The number of occurrences of each is indicated by the pixel intensity.

Finally, visualizing the outputs in figure 4.10 for a testing sample yields another interesting observation. The general region for the overlap was correctly predicted, but the shape of the region is not. There are gaps in the predicted mask and there are individual parts of the mask branching off the main body. The model still managed to achieve a fairly high prediction accuracy at 82%, further demonstrating the ineptitude of the average prediction accuracy's at capturing the model's performance. The example shown here was chosen because of the predicted overlap shape, but it is not an anomaly. The predicted overlaps throughout the entire dataset possess similar shapes and only a small fraction are rectangular.

Emphasizing Non-Overlap Regions

The DNN trained with an emphasis on correctly predicting non-overlapping regions was done so by minimizing equation 3.6 with $\alpha = 1$ and $\beta = 2.5$ (the model's architecture is once again shown in figure 3.2). The trained DNN saw a negligible increase in the average prediction accuracy over the model trained using the mean squared error only. Over the entire testing dataset, the average prediction accuracy is 76.28%.

The loss function penalized the model more for incorrectly predicting a non-overlapping region as overlapping than it did for incorrectly predicting an overlapping region as non-overlapping. The effectiveness of the penalty isn't immediately obvious when inspecting the histogram in figure 4.11, but it can be observed by comparing this histogram to the all zero prediction model's histogram (figure 4.2). The number of samples with a nearly perfect prediction is almost identical to the number of nearly perfect predictions made by the all zero predicting model. Furthermore, the rise in samples with prediction accuracy ¡10% suggests the model has a tendency to predict zeros over ones.

Inspecting the accuracy of the model's predictions as a function of the overlapping area size provides additional evidence of the model's tendency to predict zeros instead of ones. For samples with overlap size between 0 and 40, there is a near linear decline in accuracy (much like the decline over the same region for the all zero predicting model in figure 4.4). The average accuracy increases slightly as the overlapping area begins to occupy

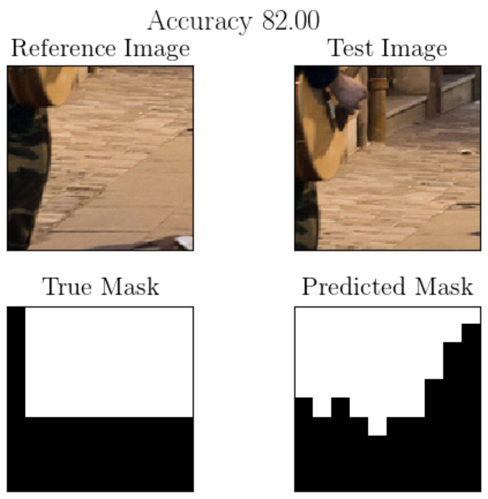


Fig. 4.10: Sample prediction of the DNN with the mean squared error loss function. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

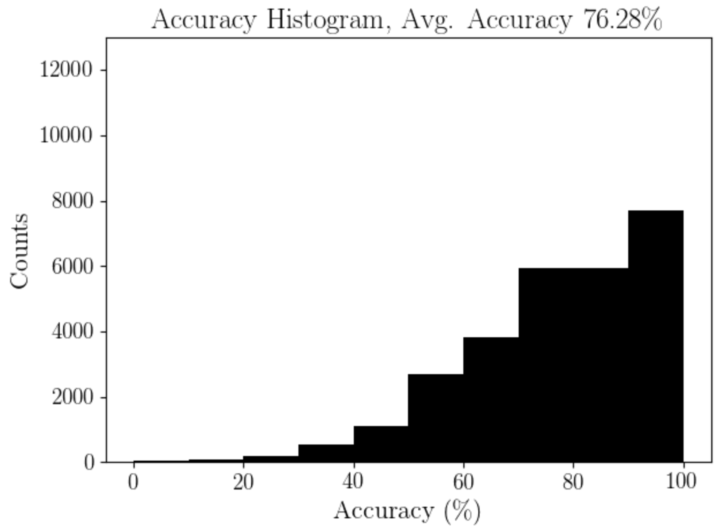


Fig. 4.11: Histogram of the prediction accuracies for the DNN trained to emphasize the correct prediction of the non-overlapping regions.

the majority of the overlap mask. This indicates that while the model does lean towards predicting zeros, the model will predict ones given the appropriate circumstances.

Further insights into the model's behavior is seen in the detailed histogram in figure 4.13. There is a prominent diagonal pattern similar to the results for the all zero model seen in figure 4.6. However, an important observation can be made by looking at the prediction accuracies for overlap masks between two nearly identical images (i.e. entire overlap between the two images). While there are instances where the model predicted the overlap masks almost entirely incorrect, there are more instances where the masks were predicted almost entirely correct. This again clearly demonstrates that while the model has a tendency to predict zeros in the mask, it is able to predict overlapping regions.

Once again, by visualizing an instance of the the model's output in figure 4.14, the same observation can be made. The general region for the overlap was correctly located, but gaps and branches in the predicted region are very prevalent. Once again, this is representative of many samples from the dataset, but not the dataset as a whole. There are a few samples where the predicted overlap masks have a rectangular, or nearly rectangular, shape but there are an unacceptable number of samples more similar to the sample displayed.

Emphasizing Overlap Regions

Training a DNN with an emphasis on correctly predicting overlapping regions involves minimizing equation 3.6 with $\alpha = 2.5$ and $\beta = 1$. For reference the DNN architecture is again pictured in figure 3.2. With an average prediction accuracy of 77.43% over the testing dataset, this trained DNN saw a negligible improvement in performance over the results seen in the previous sections.

The histogram of the predicted accuracies over the entire dataset is given in figure 4.15. The histogram is heavily skewed towards higher accuracies and this is important. The loss function is intended to encourage the DNN to predict ones instead of zeros, but the exhibited behavior is not anything like the behavior displayed by the all ones predicting model in figure 4.1. Additionally, when compared to the model emphasizing the correct prediction of the non-overlapping region presented in figure 4.11, this model appears to

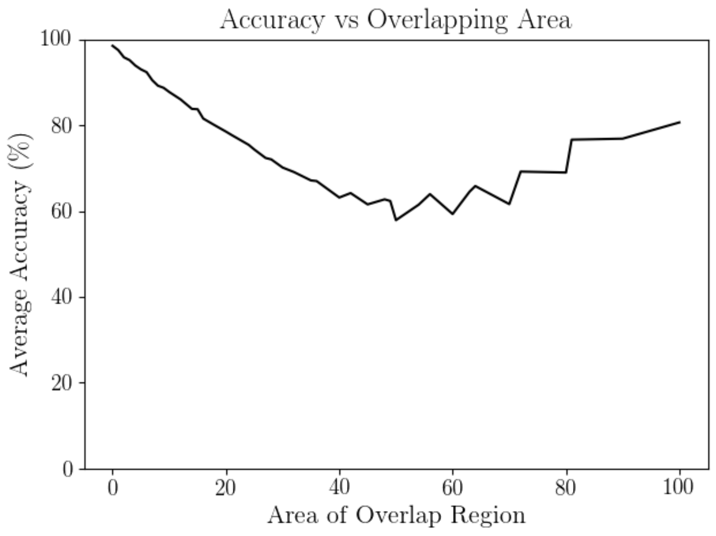


Fig. 4.12: The average prediction accuracy for the possible sizes of overlapping regions for the DNN trained to emphasize the correct prediction of the non-overlapping regions.

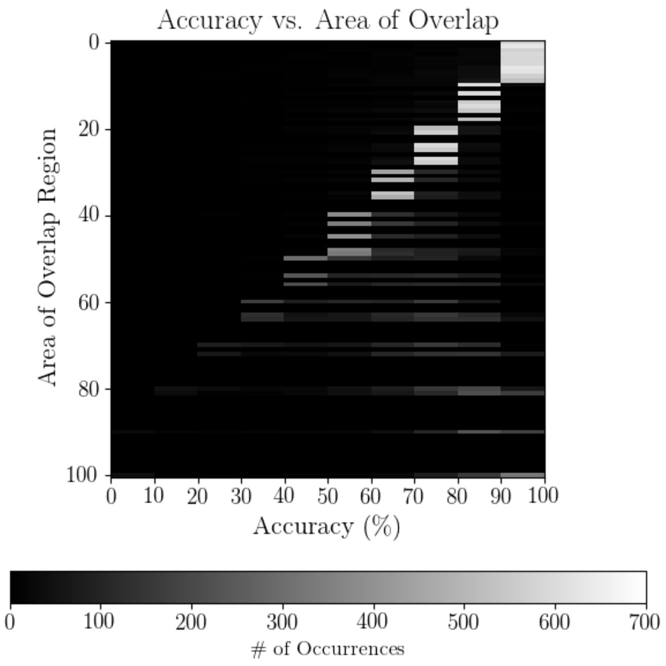


Fig. 4.13: Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN trained to emphasize the correct prediction of the non-overlapping regions. The number of occurrences of each is indicated by the pixel intensity.

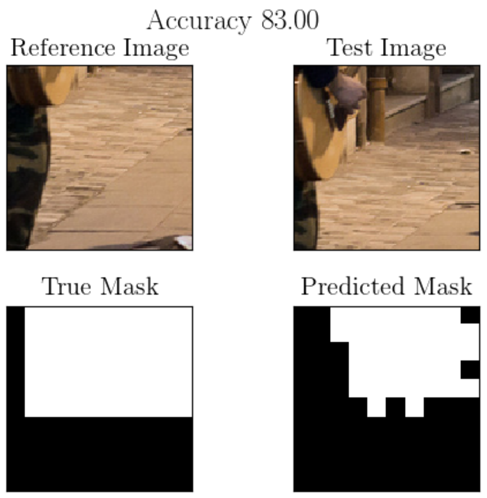


Fig. 4.14: Sample prediction of the DNN trained to emphasize the correct prediction of the non-overlapping regions. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

predict overlaps with higher accuracy more often.

The prediction accuracy vs. overlap area size displayed in figure 4.16 further suggests the model isn't only predicting ones. Notwithstanding, the lowest average prediction accuracy occurs when the overlap is roughly 20% of the reference image which does suggest the model is more likely to predict ones. Additionally, this is the first model without a significant drop off in prediction accuracy for the samples with half overlap.

When comparing the detailed histogram of figure 4.17 to that of figure 4.5, there are again noticeable differences demonstrating that the model is not only predicting ones. There is more of an upper triangular structure in the intensities than a pronounced diagonal band. This model, as expected, is able to correctly predict the high area overlaps, but the absence of the performance drop off is a little unexpected.

Even though this model appears to be performing better than the models previously considered, it still is unable to regularly predict rectangular overlap regions. This is shown in figure 4.18.

In summary, the loss function minimized when training this DNN allowed the model to

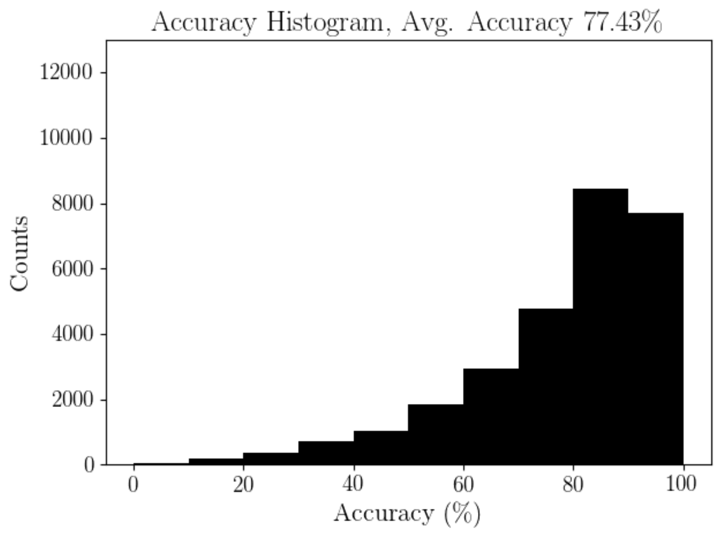


Fig. 4.15: Histogram of the prediction accuracies for the DNN trained to emphasize the correct prediction of the overlapping regions.

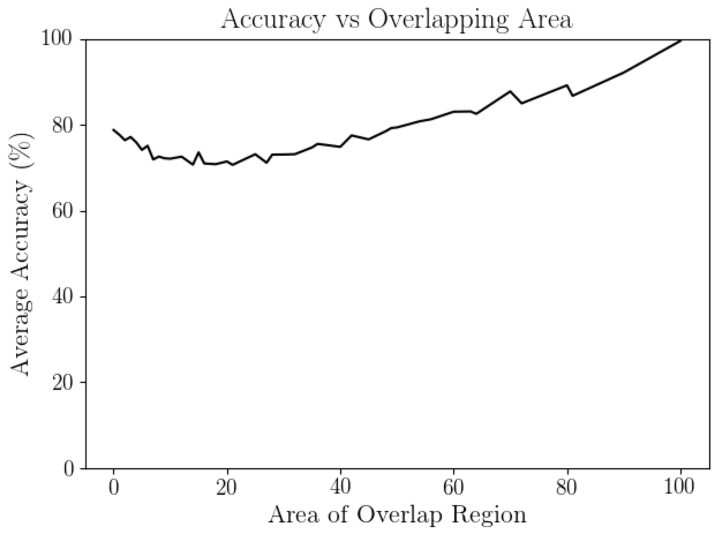


Fig. 4.16: The average prediction accuracy for the possible sizes of overlapping regions for the DNN trained to emphasize the correct prediction of the overlapping regions.

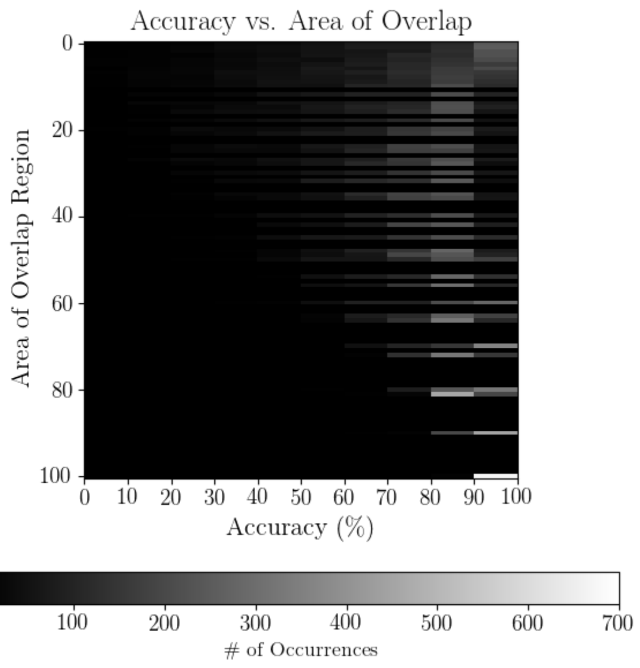


Fig. 4.17: Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN trained to emphasize the correct prediction of the overlapping regions. The number of occurrences of each is indicated by the pixel intensity.

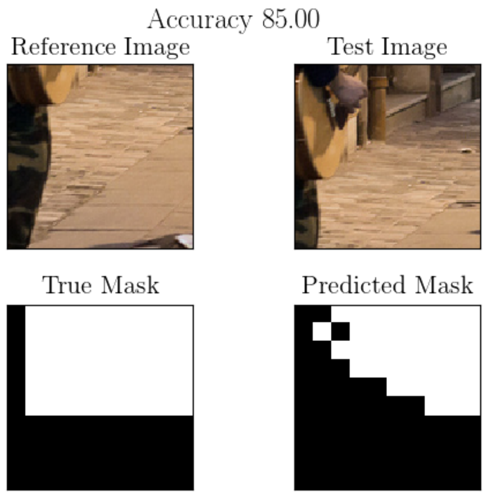


Fig. 4.18: Sample prediction of the DNN trained to emphasize the correct prediction of the overlapping regions. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

place more emphasis on the samples with a higher overlap area than the samples with lower overlap area. These samples don't occur as regularly in the training dataset, and the extra emphasis appears to help balance the influence of the samples throughout the dataset.

Forcing the Shape with $\lambda = 5$

Up until this point, none of the models have been able to regularly predict rectangular overlapping regions. To encourage the model to predict the rectangular regions, the loss function in equation 3.10 with $\lambda = 5$ is used. Minimizing this loss during training places an emphasis not only on the correct individual predictions, but also on the shape of the region. Using $\lambda = 5$ places a higher priority on the individual predictions than on the shape. Like other DNNs in this section, the architecture is pictured in figure 3.2. The average accuracy throughout the entire testing dataset using this loss function is 78.75%.

The histogram in figure 4.19 is heavily skewed towards the higher accuracies as desired and roughly 50% of the predictions using the testing dataset are over 80% accurate. There are still samples where the model's prediction accuracy is less than 30% correct. This, however, cannot provide any insight into the shape of the predicted region. It simply shows that even if the regions are not of the desired shape, the model still provides a comparable, general overall performance.

Analyzing the predicted accuracy as a function of the overlapping area can only provide insight into the general performance of the model and not the shape of the predicted overlap regions. Once again, there is a dip in average prediction accuracy when samples are half overlapping. However, the accuracy at the low point in the dip is still nearly 70% and the accuracy only increases as the size of the overlap area heads towards both 100 and zero. This alone hints that this loss function helps learning the association between images better than the other loss functions used with the direct overlap mask encoding.

In can be seen in the detailed histogram shown in figure 4.21 that this model has good overall performance but again, the details of the predicted overlap region's shape remain unclear. There are, again, a few instances where the DNN's predictions are less than 50% accurate; however, the vast majority of samples are predicted with accuracy over 70%.

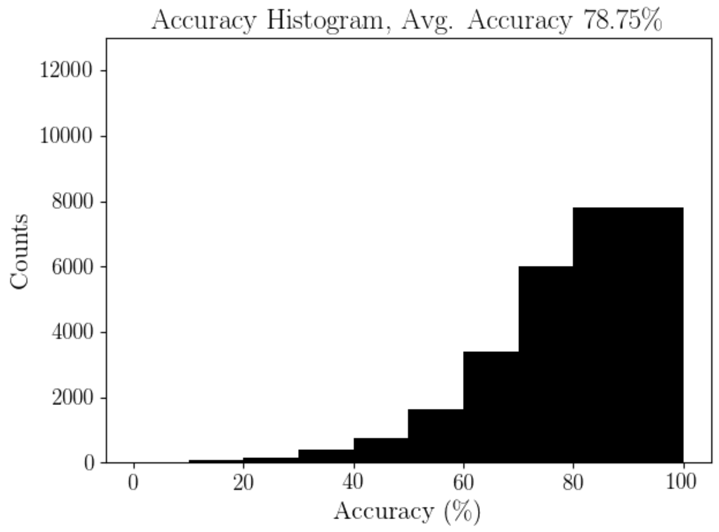


Fig. 4.19: Histogram of the prediction accuracies for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 5$

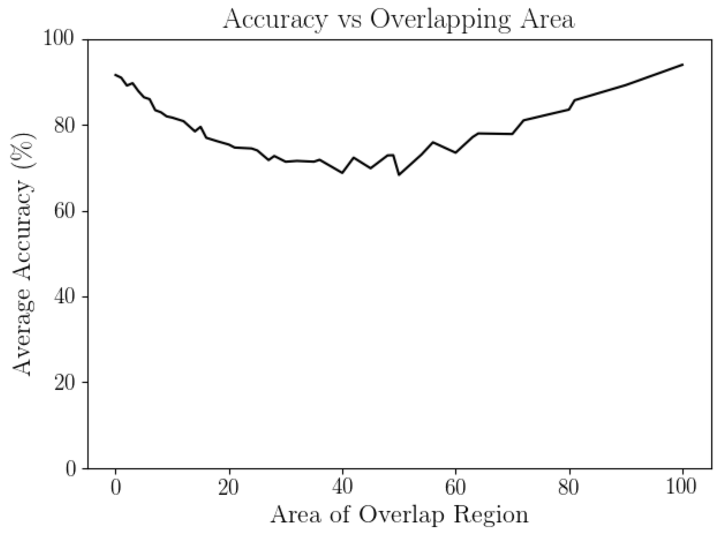


Fig. 4.20: The average prediction accuracy for the possible sizes of overlapping regions for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 5$.

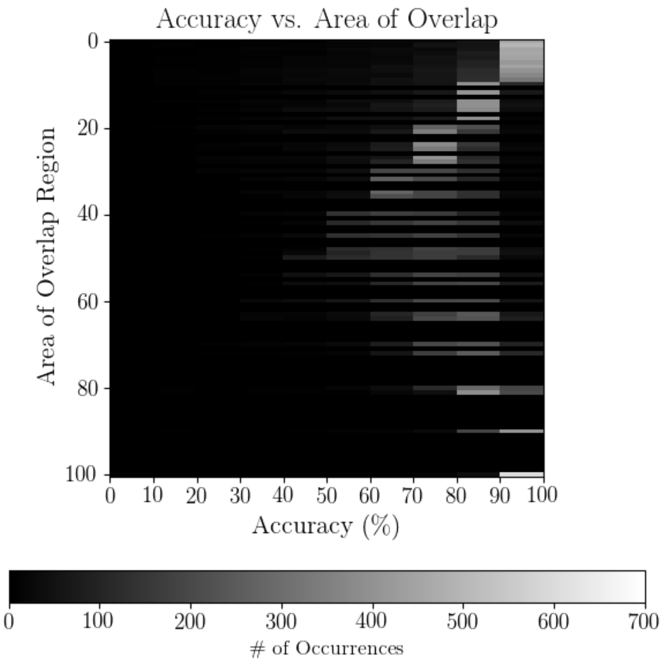


Fig. 4.21: Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 5$. The number of occurrences of each is indicated by the pixel intensity.

The only way to determine if the regions learned are rectangularly shaped is to visualize the predictions, and a single example is shown in 4.22. To maintain consistency, the testing sample used previously is shown. The predicted overlap region is not a rectangle, and the accuracy for this particular sample decreased. However, the predicted mask in this instance adequately represents a large populate of the results because, while not a perfect rectangle, it is closer. The region is more contiguous and has more continuity along the border between overlap and non-overlap but only manages to do so in some instances with a drop in prediction accuracy.

The loss function imposing a constraint on the shape of the predicted overlap region was the first loss function to include any prior knowledge that neighbors in the mask are going to be similar more often than not. As such, the overall performance increased throughout the dataset even though the shapes were not perfect.

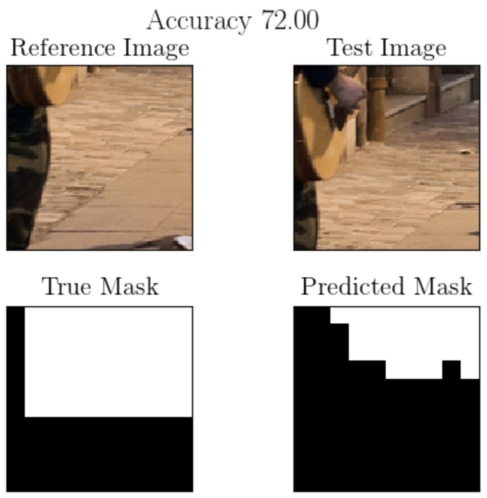


Fig. 4.22: Sample prediction of the DNN trained to predict the correct shape of the overlapping region. $\lambda = 5$. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

Forcing the Shape with $\lambda = 10$

Even though the results in section 4.3.1 demonstrated that the loss function of equation 3.10 does not force the regions to be rectangular, it did incorporate the prior belief that every element in the mask is likely similar to its neighbors. Following this observation, the parameter λ is now adjusted to $\lambda = 10$. This adjustment places more emphasis on correctly predicting individual elements of the overlap mask even more than the training in section 4.3.1 did. This adjustment resulted in a DNN with an average prediction accuracy of 78.88% over the entire testing dataset.

The prediction accuracy histogram in figure 4.23 isn't significantly different from that of the model with $\lambda = 5$ shown in figure 4.19. The histogram is again heavily skewed towards higher accuracies with peak accuracy occurrences near 8000. There are no obvious indications in the histograms that using $\lambda = 10$ or $\lambda = 5$ performed better than the other.

Performing a similar comparison of the average prediction accuracy vs. size of the overlapping area between this model (given in figure 4.24) and the previous model (given in figure 4.20) cannot confirm that either DNN is superior. They are almost identical with

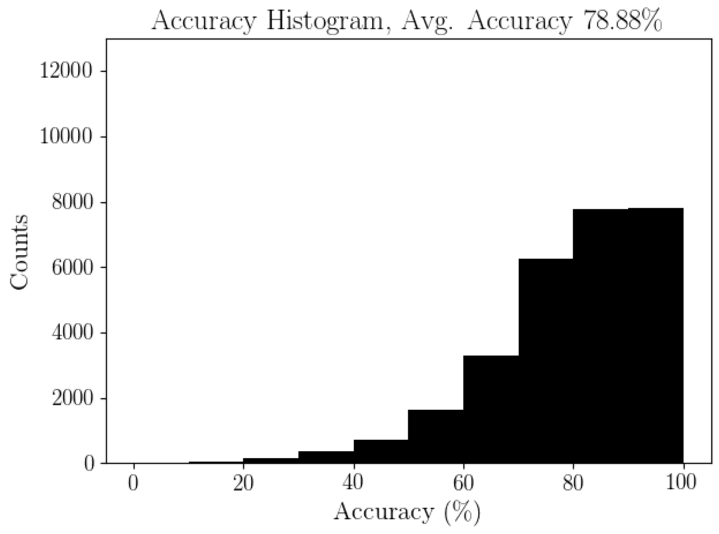


Fig. 4.23: Histogram of the prediction accuracies for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 10$

a few, minuscule exceptions. There is a dip in the performance when the overlap area is near 50 but even then, the average accuracy is around 70%.

Analyzing the detailed histogram in figure 4.25 again fails to provide any contribution to the performance comparison between this model and the model in section 4.3.1.

The first noticeable difference is seen by inspecting the model's actual predictions in figure 4.26. For this specific sample, the predicted mask is much closer to the true mask than the predicted mask shown in 4.22.

When the observations from all of the figures are considered, the best conclusion is that different DNNs trained using $\lambda = 5$ and $\lambda = 10$ are nearly comparable, and each will perform better on some testing samples than others. However, the results indicate that the including the prior knowledge of the overlapping regions shape is the best tested approach when using the direct overlap mask encoding.

4.3.2 Corner Encoding

Motivated by the inability of previous models to predict rectangular regions, a new architecture (see figure 3.4 for the details) having the shape constraint built in was proposed.

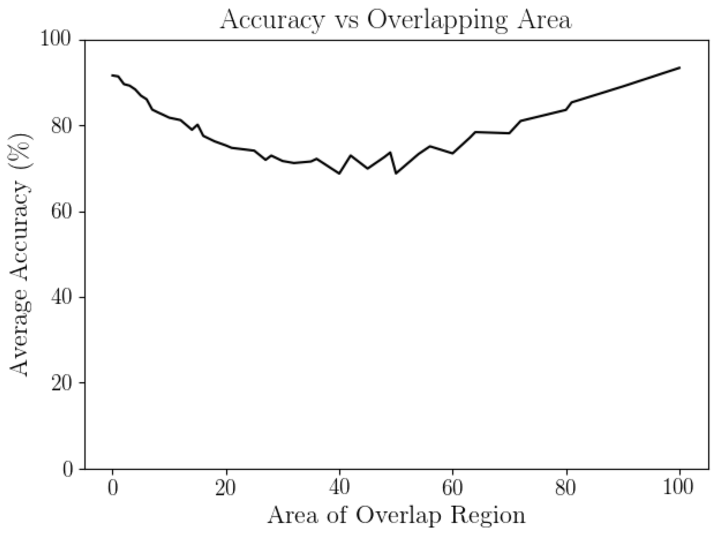


Fig. 4.24: The average prediction accuracy for the possible sizes of overlapping regions for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 10$

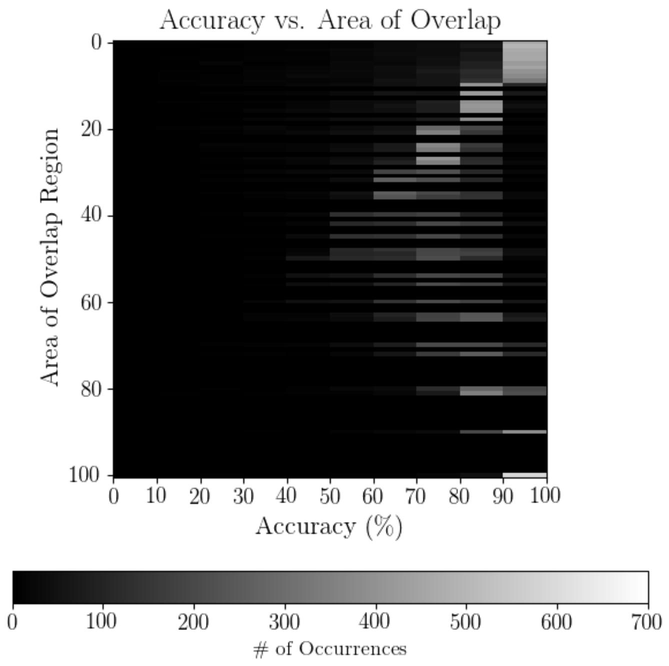


Fig. 4.25: Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN trained to predict the correct shape of the overlapping region. $\lambda = 10$. The number of occurrences of each is indicated by the pixel intensity.

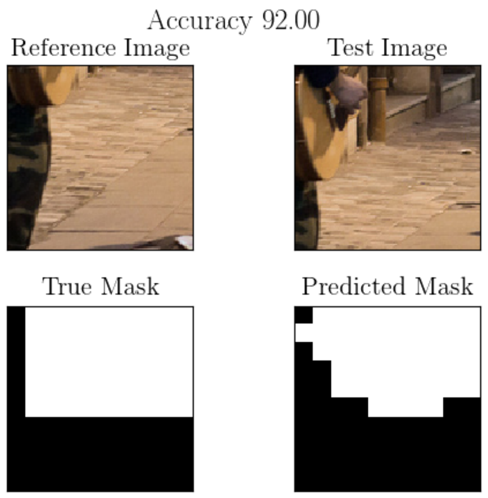


Fig. 4.26: Sample prediction of the DNN trained to predict the correct shape of the overlapping region. $\lambda = 10$. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

Modifying the architecture instead of using a constraint on the loss function guarantees the prediction of rectangularly shaped overlap regions. This model was trained my minimizing the loss function in equation 3.17. The solitary model implementing the corner encoding achieved an average prediction accuracy of 77.8% over the testing dataset.

Results initially seem positive upon inspection of the prediction accuracy histogram. There are nearly 10,000 samples when the DNN predicted more than 90% of the overlap mask correctly. This is almost 2,000 samples more than previously seen for any of the DNNs. However, there is also a noticeable increase in the occurrence of samples when the predicted mask was nearly entirely incorrect.

The prediction accuracy vs. size of the overlapping area shown in figure 4.28 demonstrates the model is able to avoid a dip in prediction accuracy performance as significant as seen in the previous DNNs. Notwithstanding, the average accuracy generally does trend upwards with the size of the area, *trend* being the key word. The average accuracy has more dramatic ups and downs throughout the upwards trend that is simply absent in other models.

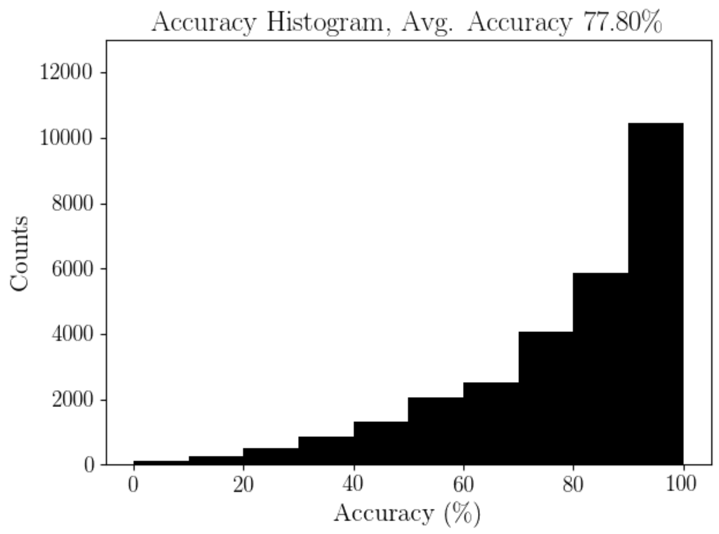


Fig. 4.27: Histogram of the prediction accuracies for the DNN implementing the corner overlap mask encoding.

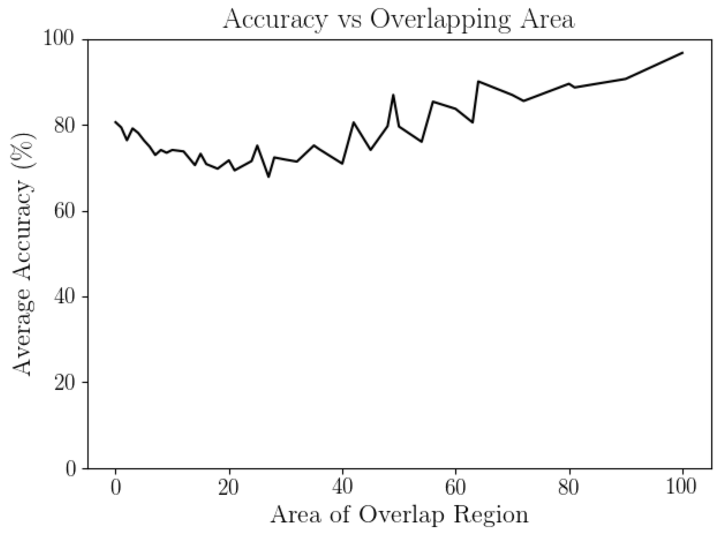


Fig. 4.28: The average prediction accuracy for the possible sizes of overlapping regions for the DNN implementing the corner overlap mask encoding.

Of all results thus far, the detailed histogram shown in figure 4.29 is perhaps the most interesting. For the samples with a high amount of overlap, there is no middle ground. The DNN either performs great or abysmally. When the area of the overlapping region is small, the accuracies are distributed more uniformly than seen from previous DNNs. A reasonable explanation for this is introduced using the visualization of a predicted mask.

Visualizing the predicted mask shown in figure 4.30 for an individual sample demonstrates one of fundamental flaws of this overlap mask encoding. The predicted overlapping area and true overlapping area do not intersect but the DNN did manage to nearly predict one of the two corners correctly. However, it failed to correctly predict the other corner. As a result, it missed the entire overlapping area. With this encoding, a single error with one of the four main corners changes the outcome tremendously. The only reason the prediction accuracy is not lower than 60% is due to the shape of the overlapping region.

Once again, the distribution of small area overlaps in the dataset appear to inflate the interpreted performance. When the true overlap area is between zero and ten, the model only has to predict a small overlapping region and its location does not matter. The accuracy for any sample with little to no actual overlap can be greater than 90% as long as the DNN doesn't predict a region with a high amount of overlap. This, in addition to the higher likelihood of entire overlaps being missed completely, is enough to remove this encoding from further consideration entirely.

4.3.3 Row-Column Encoding

Another approach to incorporating the rectangular region via architecture rather than loss function constraint is the row-column encoding. The loss function minimized during training for every model using this encoding is given in equation 3.19. The general architecture was introduced in figure 3.1 but every model presented in this section is different. However, the loss function used with every model is the same and is equation 3.19.

Row-Column Model 1

The first DNN trained using the row-column encoding is similar in size to the DNN

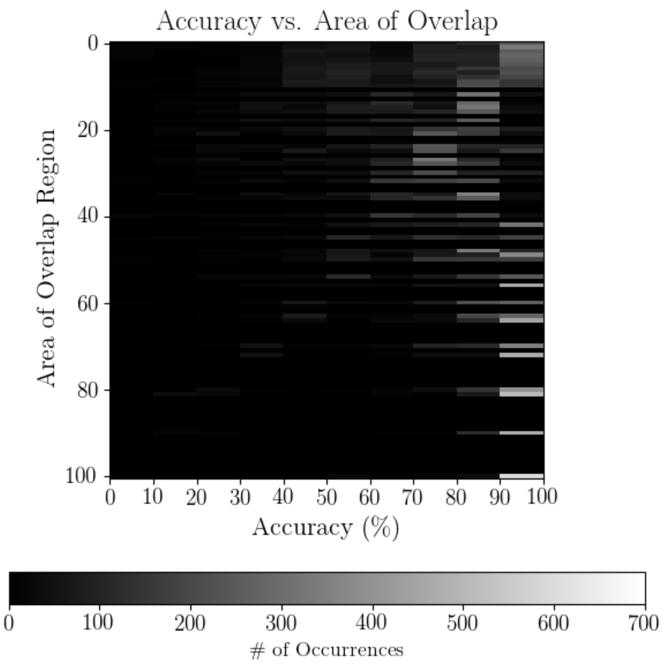


Fig. 4.29: Detailed histogram of prediction accuracies and the area of the overlapping regions for the DNN implementing the corner overlap mask encoding. The number of occurrences of each is indicated by the pixel intensity.

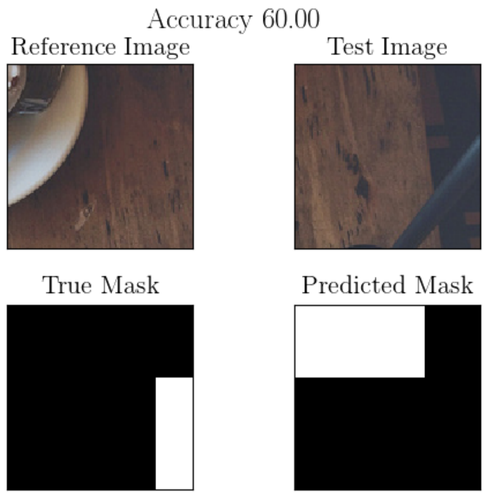


Fig. 4.30: Sample prediction of the DNN implementing the corner overlap mask encoding. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

discussed in previous sections. The details of the architecture are found in figure 3.8. Using this encoding, the first DNN's average prediction accuracy for the testing dataset is 81.98%.

The histogram of this DNN is shown in figure 4.31 and is again very promising at first glance. It is heavily skewed towards the higher prediction accuracies with over 11,000 samples greater than 90% accurate. The histogram also clearly shows the DNN continues to make severely incorrect predictions. Additionally there are no indications of a predisposition to predicting ones or zeros more than the other.

Looking at the predicted accuracy as a function of the overlapping area further validates the potential of this encoding. The slight dip in the performance seen in the other encodings still exists but the lowest prediction accuracy is just below 80%. Furthermore, as the area increases, the average prediction accuracy also increases and does so without much volatility.

An inspection of the detailed histogram in figure 4.33 further corroborates the observations made thus far. When the overlap area is small, there are some instances of poor performance but there are abundantly more instances having high prediction accuracy. Overall, the size of the overlapping region appears to have a very small effect on the average prediction accuracy.

Lastly, the visualization of the predicted mask shown in figure 4.34 further demonstrates the viability of this approach. Not only does the region have the desired rectangular shape, it is almost perfect. Once again, this is only one sample from the testing dataset. Not every prediction will be as close and accurate as this sample, but they all have a rectangular shape.

The results seen in this first attempt are overwhelmingly positive. The predicted regions are rectangular, there is a high level of performance, and most importantly, the fatal flaw of the corner encoding in section 4.3.2 is not present.

Row-Column Model 2

Due to the success seen by the first DNN to use the row-column encoding, instead of modifying the loss function, changes to the size of the DNN are investigated. The

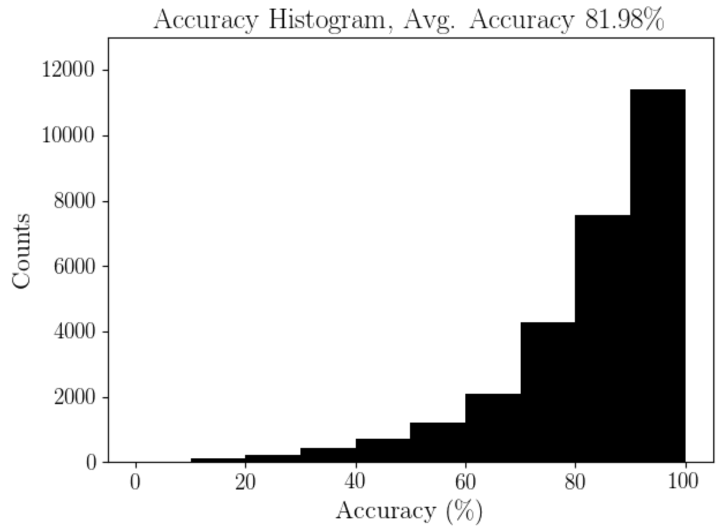


Fig. 4.31: Histogram of the prediction accuracies for row-column overlap encoding model 1.

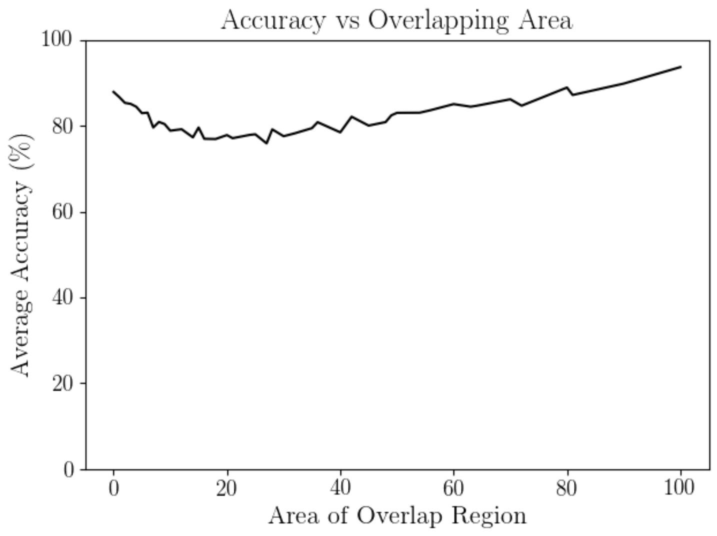


Fig. 4.32: The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 1.

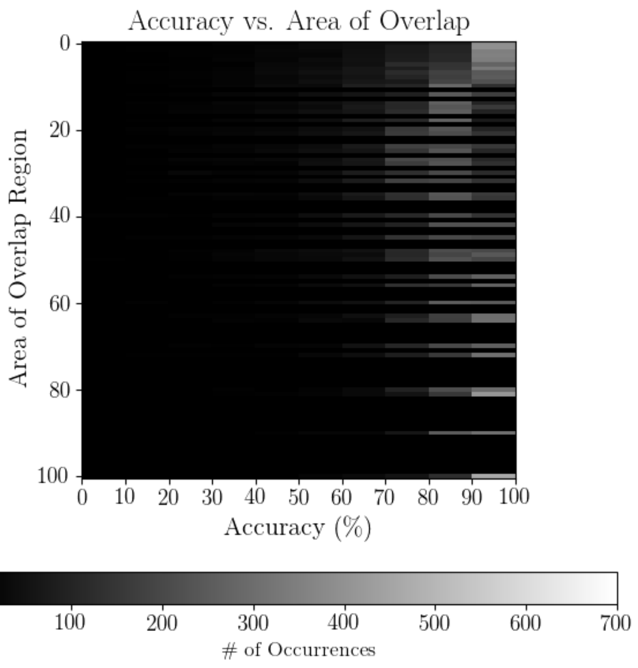


Fig. 4.33: Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 1. The number of occurrences of each is indicated by the pixel intensity.

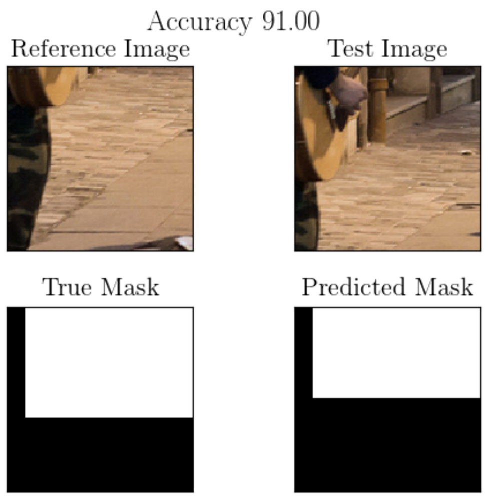


Fig. 4.34: Sample prediction for row-column overlap encoding model 1. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

modifications were introduced in figure 3.9. The changes made for this DNN are limited to adding a layer in the row and column branches. With this modification, the average prediction accuracy for the testing dataset improved to 83.93%.

The histogram for the prediction accuracies using this DNN is shown in figure 4.35. The histogram is again skewed towards higher accuracies and now more than 12,000 samples' masks were predicted with over 90% accuracy. This implies that there are fewer samples with mediocre prediction accuracy than produced by model 1.

Comparing the prediction accuracy vs size of the overlapping area for this model (figure 4.36) and that of the previous model (figure 4.32) doesn't provide any immediate indication of a superior performance level. They both follow the same general behavior with the prediction accuracy trending up as the size of the overlapping area increases. However, across all sizes of overlapping regions, the average prediction accuracy is almost always better for this DNN. 4.3.3.

The detailed histogram shown in figure 4.37 shows that overall, this model is performing desirably. The model is able to accurately predict the mask for all sizes of overlapping areas, though the lower overlap sizes tend to provide more trouble than the higher areas.

The predicted mask visualization shown in figure 4.38 happens to be one of samples where this model predicted the mask perfectly. Once again, this is only one of samples from the testing dataset and as seen in the other figures, there are instances where the predicted mask is nearly entirely incorrect.

When all of the results for this DNN are considered together, it can concluded that the extra layer before the final output layers did increase the general performance from the previous model.

Row-Column Model 3

The change in this model from the model in section 4.3.3 is in the convolution layers. The kernel sizes and strides were chosen to provide the same downsampled output dimensionality used in every other DNN. The fully connected layers' sizes are identical to those

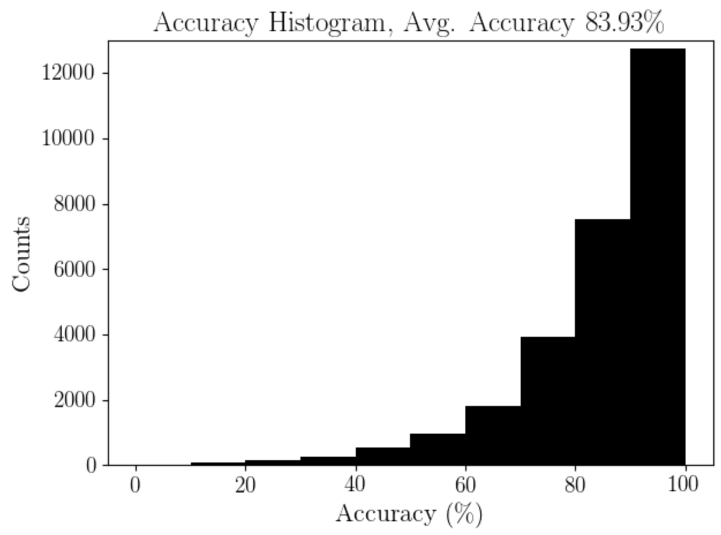


Fig. 4.35: Histogram of the prediction accuracies for row-column overlap encoding model 2.

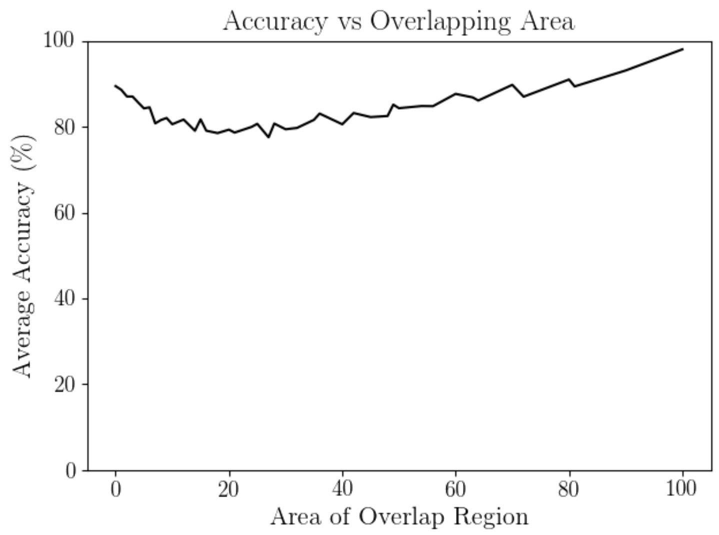


Fig. 4.36: The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 2.

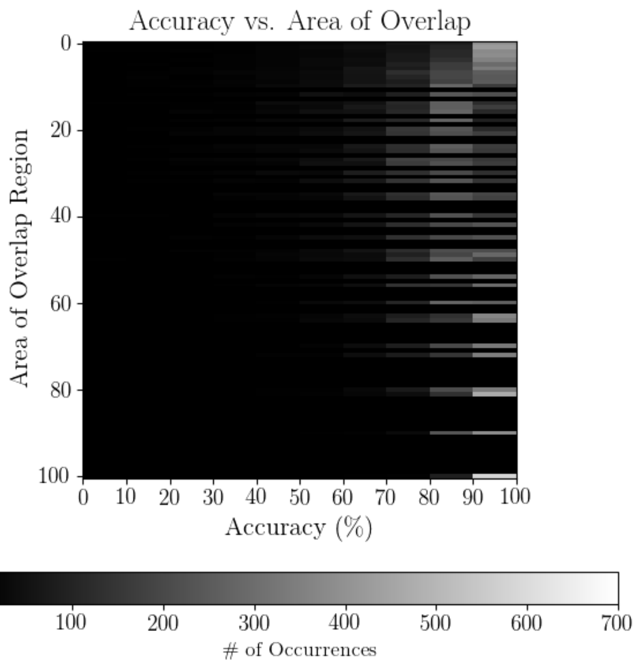


Fig. 4.37: Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 2. The number of occurrences of each is indicated by the pixel intensity.

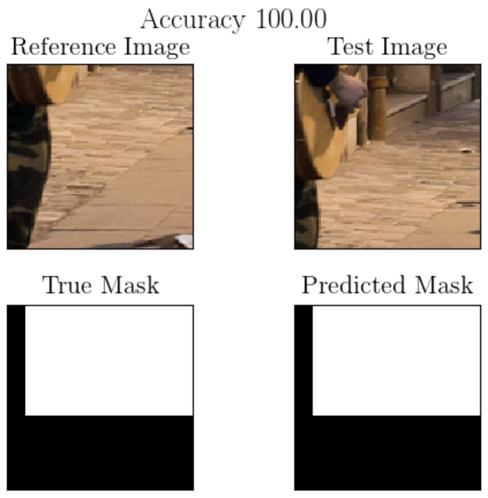


Fig. 4.38: Sample prediction for row-column overlap encoding model 2. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

used in model 2. Relative to model 2, this DNN managed a negligible increase with an average prediction accuracy of 84.33%.

The histogram of the predicted accuracies for this model shown in figure 4.39 is very similar to the histogram shown in figure 4.35. The heavy skew towards the high accuracies is present again but this model manages to predict even more masks with accuracy greater than 90%. Neither histogram shows enough to indicate better performance one way or the other.

Just as before, comparing the predicted accuracies vs. size of the overlapping area for this model (figure 4.40) and model 2 (figure 4.36) fails to yield any indication of superior performance for either model. They both follow the same curve and have approximately equal prediction accuracies for all sizes of overlapping regions.

The detailed histogram for this model shown in figure 4.41 cannot provide any obvious conclusion either way as well. One difference between this model and model 2 can be seen by inspecting instances where the prediction accuracy is less than 10%. They occur less often for this model but behavior cannot be attributed solely to the additional convolution layers as network initializations may also play a part.

Once again, for the testing sample under comparison, this model was able to predict the mask with perfect accuracy.

The results fail to provide sufficient evidence that this model is conclusively better than model 2. Even though the average prediction accuracy is slightly higher, it is not enough to confidently claim that the additional convolution layers with smaller kernels provides better performance. Model 3 is also both more computationally and memory expensive than model 2.

Row-Column Model 4

The variation applied for this DNN is intended to increase the capacity of the DNN. The convolution kernels are back to 12×12 kernels but now there are 30 kernels instead of the 20 seen previously. Additionally, more layers were added to the fully connected layers.

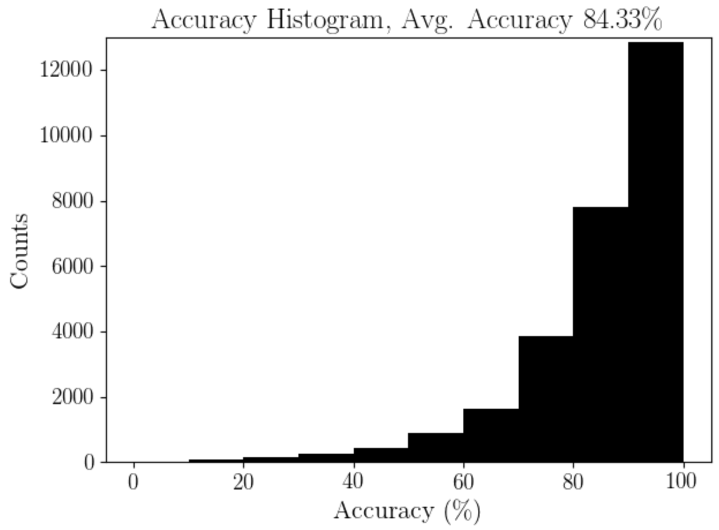


Fig. 4.39: Histogram of the prediction accuracies for row-column overlap encoding model 3.

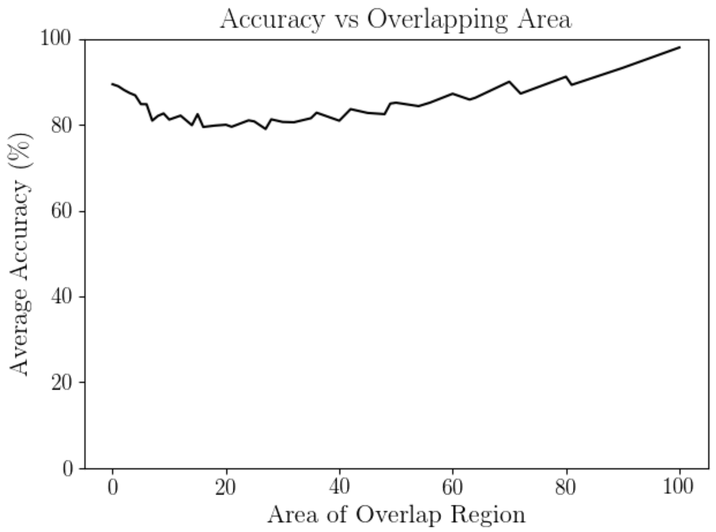


Fig. 4.40: The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 3.

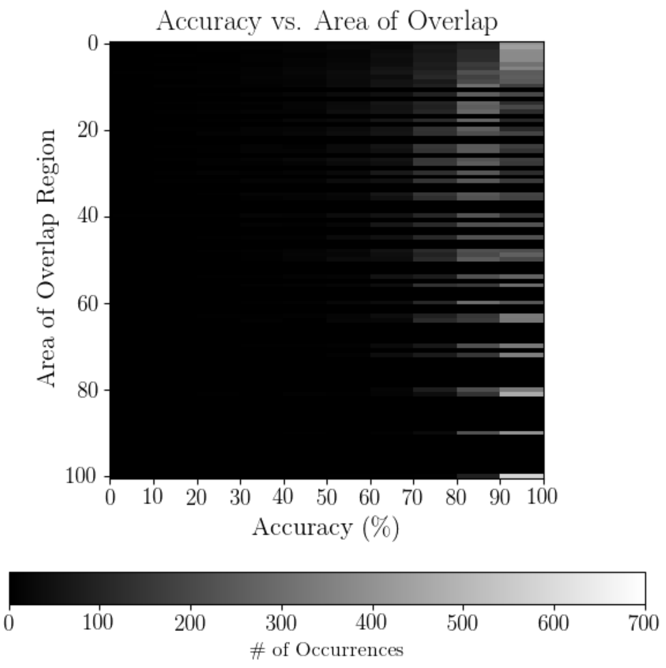


Fig. 4.41: Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 3. The number of occurrences of each is indicated by the pixel intensity.

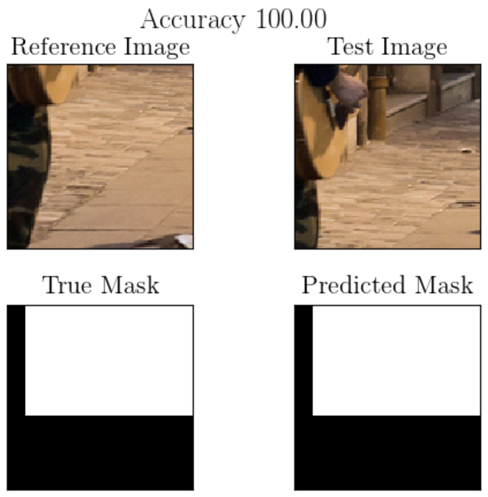


Fig. 4.42: Sample prediction for row-column overlap encoding model 3. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

With these modifications, there was a noticeable drop off in this model's average prediction accuracy over the testing dataset to 79.52%.

Upon inspection of the histogram in figure 4.43, the decrease in performance is almost immediately visible. The number of testing samples where the model predicted the mask with over 90% accuracy dropped below 10,000 and consequently, other accuracy ranges saw an increase in occurrences.

The drop in the performance is again visible in the accuracy vs. size of overlapping area shown in figure 4.44. The performance dip common to the other models is again present but more concerning is the average prediction accuracy at the lower overlap sizes. Among the lower areas, the average prediction accuracy is visibly lower than previously seen in by models 2 and 3.

The detailed histogram given in figure 4.45 further demonstrates the drop in performance when the area of the overlapping region is small. In those regions, the occurrence of samples which saw a prediction accuracy less than 40% has increased.

The visual inspection of the predicted mask for the same image used previously shows the model still performs decently in this case. The model was only off on its prediction by a single element in the row vector. It's worth repeating that this lone sample cannot adequately represent the entire testing dataset.

The conclusion to draw from this is the increased capacity was not directly tied to an increased level of performance. With 30 convolution kernels, the fully connected layers need to make the associations using more features. The addition likely made the association process more difficult because of the added dimensionality. This isn't an issue of over fitting the DNN due to the use of early stopping during training. The average predicted accuracy over the training dataset is still lower than the average prediction accuracy from both models 2 and 3.

Row-Column Model 5

This model keeps the the fully connected structure used in section 4.3.3 but uses the two convolution layers introduced with model 3. The use of the two convolutional layers changes

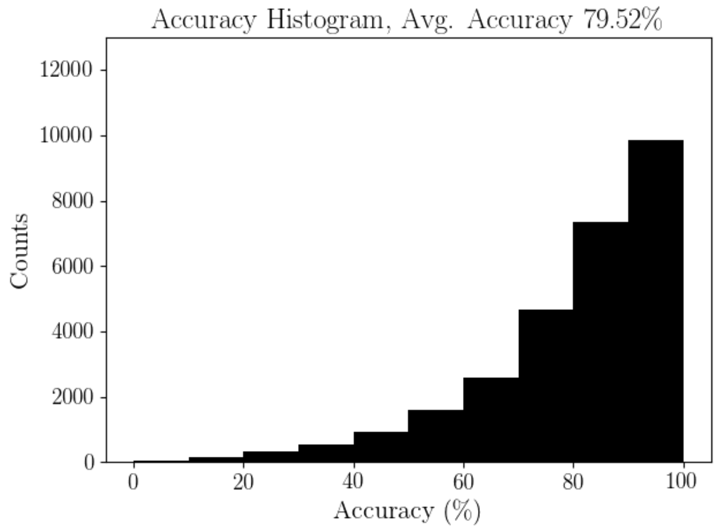


Fig. 4.43: Histogram of the prediction accuracies for row-column overlap encoding model 4.

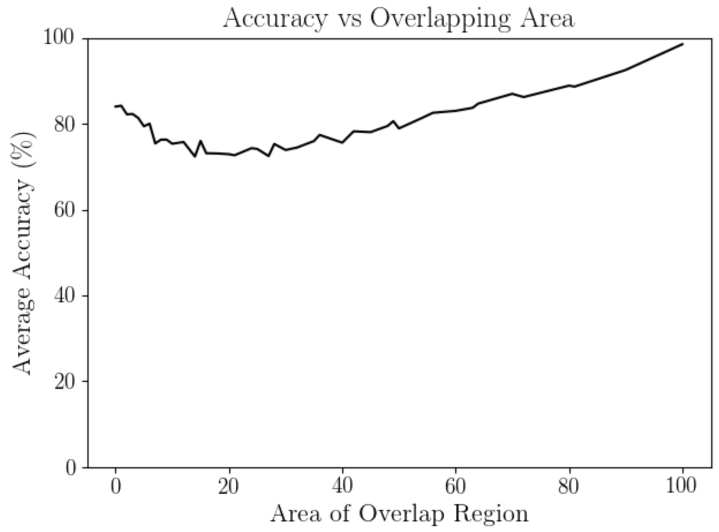


Fig. 4.44: The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 4.

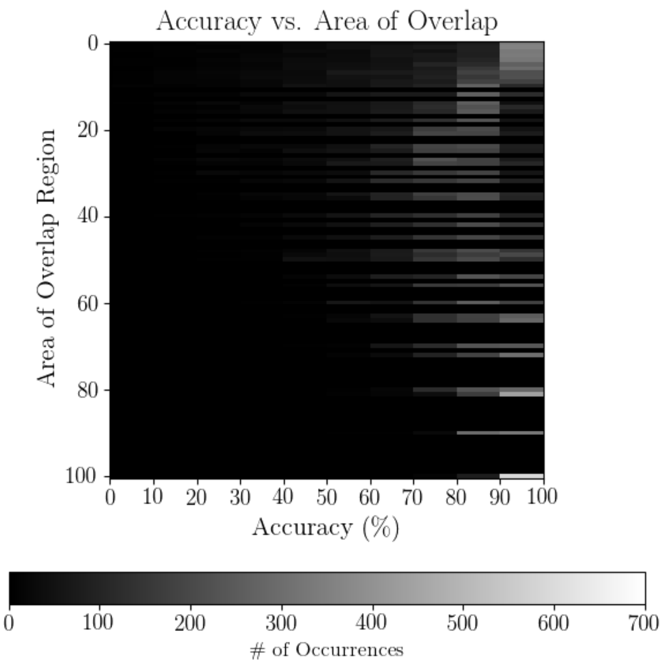


Fig. 4.45: Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 4. The number of occurrences of each is indicated by the pixel intensity.

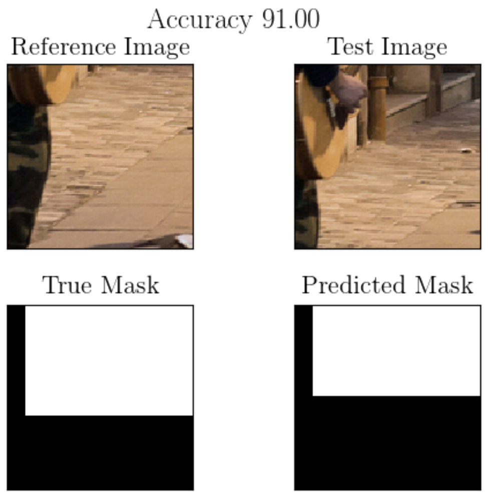


Fig. 4.46: Sample prediction for row-column overlap encoding model 4. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

the types of features learned and as a result, the features available during the association process. This architecture was introduced in figure 3.12. The 20 features used by this model to describe each image is consistent with those used in models 2 and 3. This change again resulted in a noticeable drop in the average prediction accuracy over the testing dataset to 76.27%.

The prediction accuracy histogram for this model is shown in figure 4.47. There is still a skew towards the higher accuracies but not as much as seen in the previous models. This observation confirms the performance level drop indicated by the decrease in the average prediction accuracy.

The predicted accuracy vs. the size of the overlapping area shown in figure 4.48 further confirms the observations of decreased performance. Like figure 4.44, the dip in performance occurs when then the size of the overlapping region is lower but the dip in figure 4.48 is more pronounced.

The detailed histogram in figure 4.49 provides the final confirmation needed to conclude this model's performance is down. There is no doubting that the model performs well when there is a lot of overlap but as the size of the overlap decreases, the prediction accuracies are spread out all over from 20% to 80% accurate.

The prediction visualization for this model are again included, though no significant observations are made.

The addition of more kernels with the deeper fully connected layers didn't provide any increase in performance. The training of the model was halted using early stopping to prevent over fitting just as it was done in the other models. In the end, the DNN does not appear to benefit at all from the availability of potentially more complex features.

Row-Column Model 6

The goal of this model is to evaluate the performance when the large fully connected body of layers is fed only using 10 convolution kernels instead of the 20 or 30 used previously. Under investigation is whether a deeper model can learn the association using fewer features.

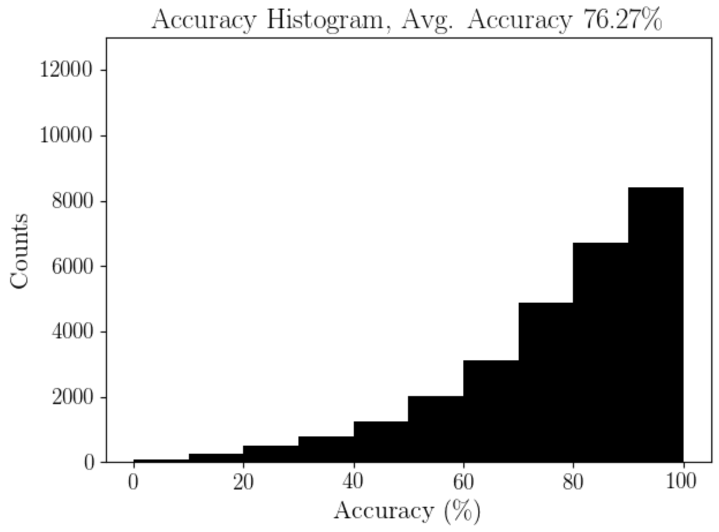


Fig. 4.47: Histogram of the prediction accuracies for row-column overlap encoding model 5.

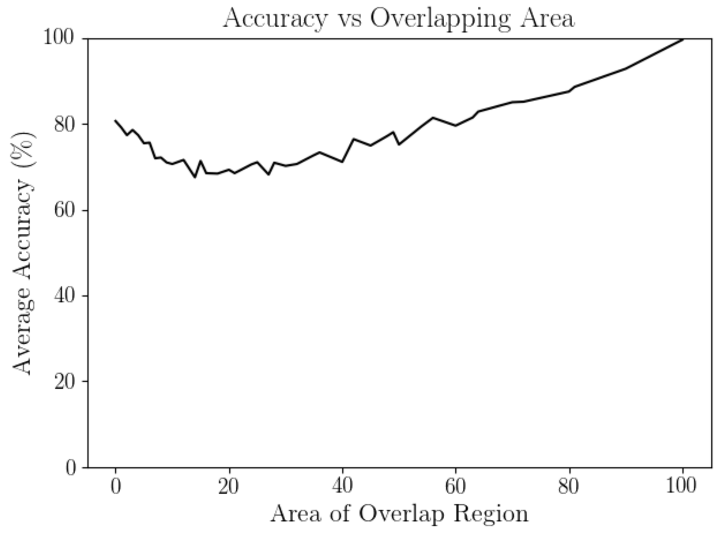


Fig. 4.48: The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 5.

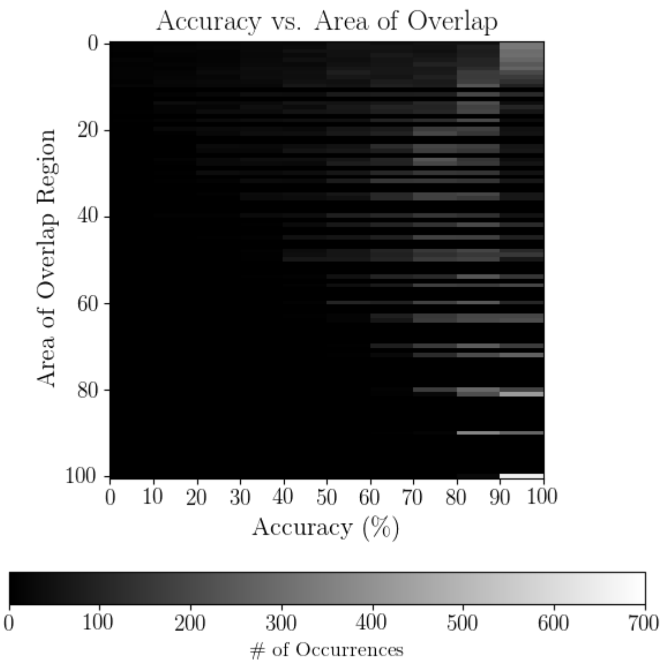


Fig. 4.49: Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 5. The number of occurrences of each is indicated by the pixel intensity.

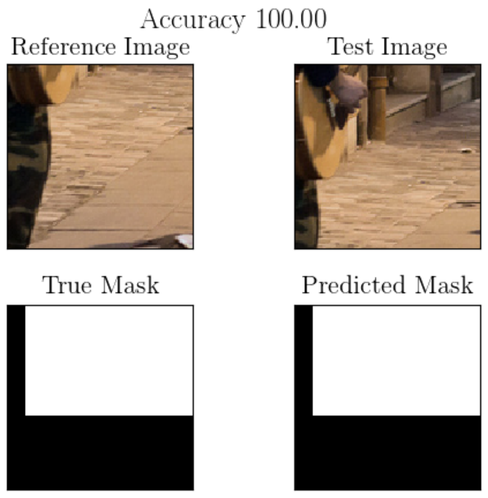


Fig. 4.50: Sample prediction for row-column overlap encoding model 5. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

The DNN architecture details were presented in figure 3.13. This model achieved an average prediction accuracy over the entire testing dataset of 75.74%.

The histogram for this model shown in figure 4.51 still maintains the skew towards the high accuracies, but it clearly demonstrates the DNN is not nearly as proficient at predicting the overlap masks as models 1, 2, and 3.

The accuracy vs size of the overlapping area graph in figure 4.52 again shows the familiar dip in performance for the lower area overlaps.

The detailed histogram in figure 4.53 further demonstrates the limits of this model's performance. Again, nearly every mask for the instances of complete overlap were predicted with over 90% accuracy. And like before, as the size of the overlapping region decreases, the reliability of the predicted overlap mask is diminished.

The predicted overlap mask for the same sample used throughout this section is included in figure 4.54.

This DNN's performance is not anywhere near the level of performance seen from models 2 and 3. The increased capacity of the network and fewer features to make decisions with did not help the model learn the image associations any better than the less complex models.

Row-Column Model 7

Model 7 is created by decreasing the capacity of model 6. The decrease is accomplished by removing a layer from the output branch layers. There are still only 10 convolution kernels in the network. After this change, this DNN's performance jumped up to an average prediction accuracy on the testing dataset of 82.23%.

The decrease in capacity appears to have helped with this model's performance. In the prediction accuracy histogram shown in figure 4.55, the skew is much further right than is seen in figure 4.51. Additionally the number of samples where the mask was predicted to higher than 90% accuracy also increased and even managed to approach the highest levels established by models 2 and 3.

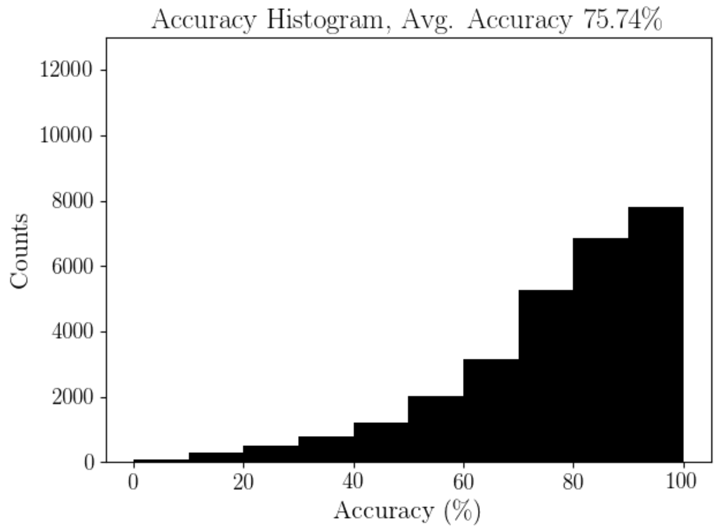


Fig. 4.51: Histogram of the prediction accuracies for row-column overlap encoding model 6.

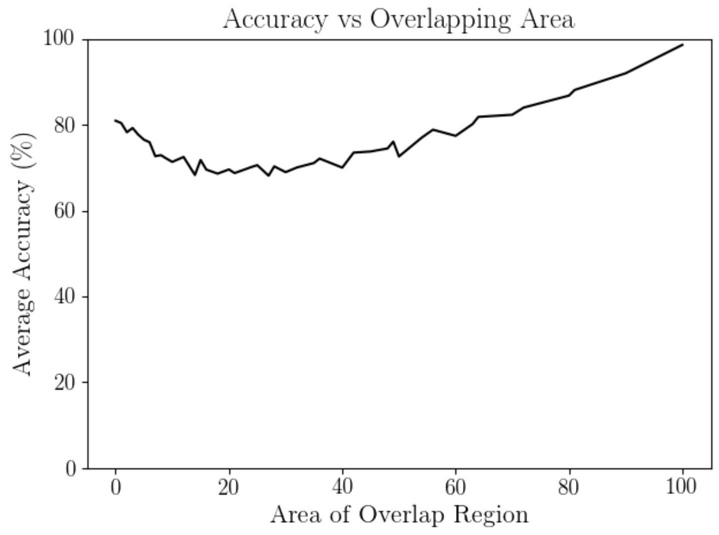


Fig. 4.52: The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 6.

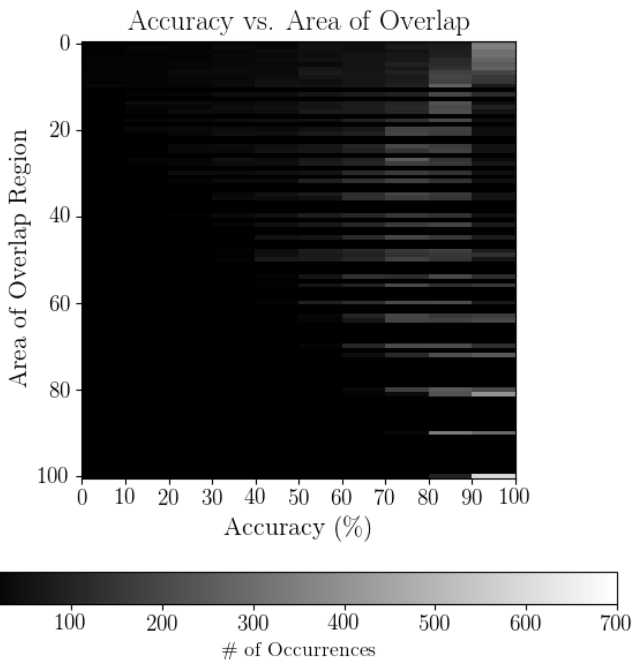


Fig. 4.53: Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 6. The number of occurrences of each is indicated by the pixel intensity.

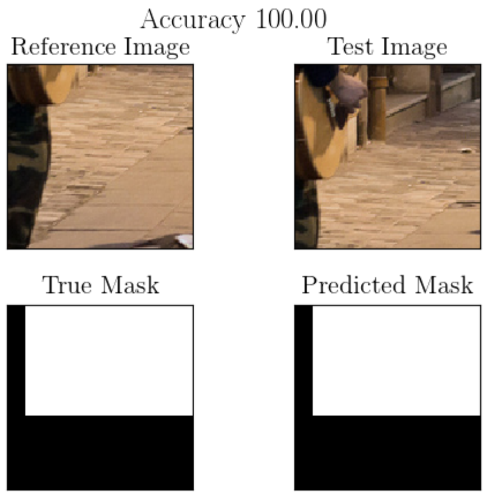


Fig. 4.54: Sample prediction for row-column overlap encoding model 6. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

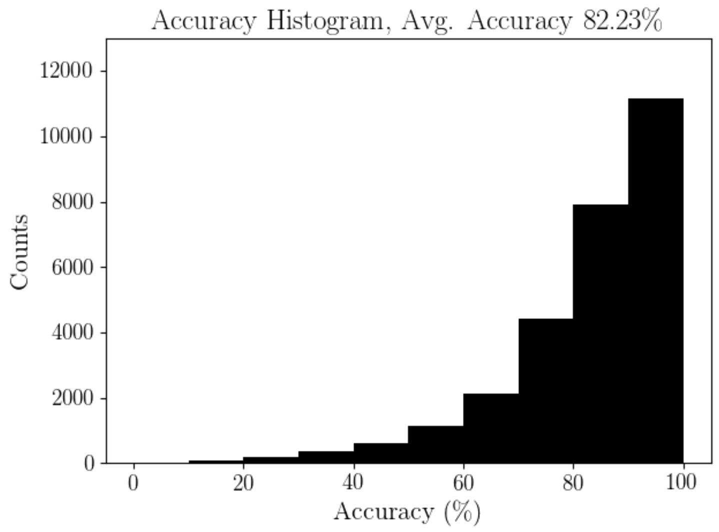


Fig. 4.55: Histogram of the prediction accuracies for row-column overlap encoding model 7.

A comparison of the prediction accuracies vs. the size of the overlap area for this DNN shown in figure 4.56 with those in previous models further acknowledges the improved performance. The dip in performance associated with the smaller overlapping regions bottoms out at roughly 80%; furthermore, the dip is no longer as pronounced as the dips in figures 4.48 and 4.52.

The detailed histogram in figure 4.57 also shows the increased performance of the model on samples where the overlap area is small. While there are still occasions where the predicted mask was more incorrect than correct, the predictions are clearly more correct for all sizes of overlapping regions more often than not.

Though not perfect, the predicted overlap for the same test sample seen throughout this chapter is nearly exact. The prediction is shown in figure 4.58.

The improved performance of this DNN over models 5 and 6 is evident in the results. This suggests that the increased capacity provided added difficulty to the training process. However, due to the nature of neural networks this cannot be made confidently.

Row-Column Model 8

The architecture used in this model is identical to the model 2 discussed in section 4.3.3

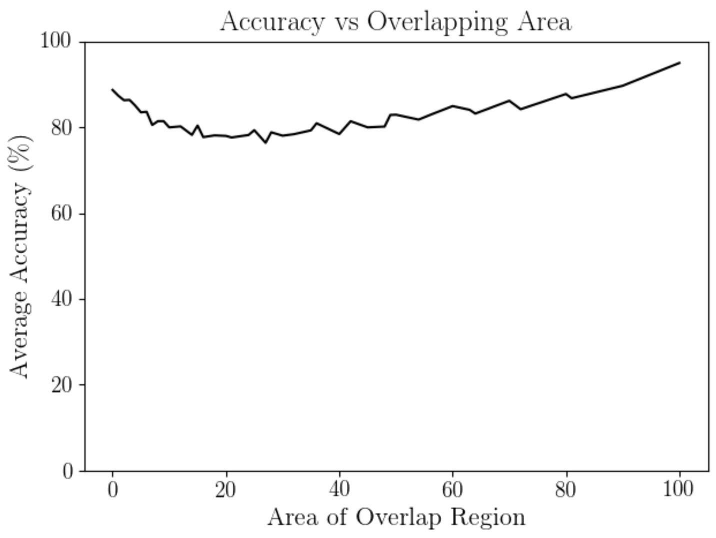


Fig. 4.56: The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 7.

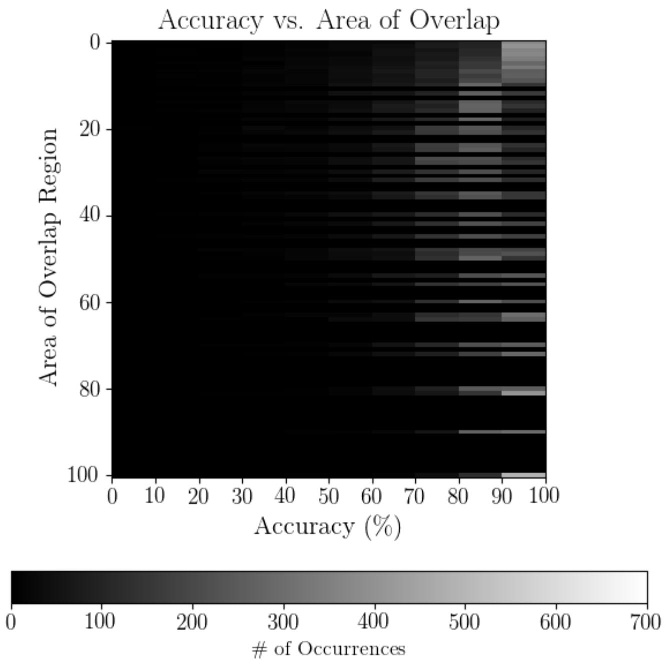


Fig. 4.57: Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 7. The number of occurrences of each is indicated by the pixel intensity.

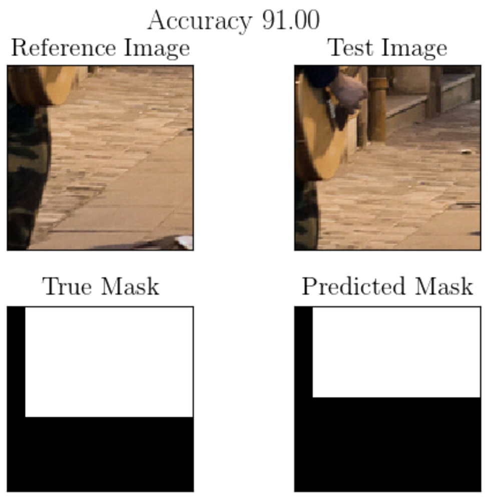


Fig. 4.58: Sample prediction for row-column overlap encoding model 7. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

but with the convolutional kernels frozen at their initialized weights. They are randomly assigned and then not adjusted during the training process. This test is to determine if the kernels are learning anything useful or if any random choice will work. The nature of this test necessitates that the comparison be made between this DNN and model 2. Surprisingly, this model had a higher than anticipated average prediction accuracy of 77.12% over the testing dataset.

The prediction accuracy histogram for this model in figure 4.59 shows there is a decent level of performance. The skew towards higher prediction accuracies is present again but the number of samples predicted with higher than 90% is nowhere near those seen in figure 4.35.

A similar comparison between this DNN and the model 2 can be made using the accuracy vs. size of overlap area figures. The shape of the curves in figures 4.60 and 4.36 is similar but it's obvious this model cannot perform as well as model 2 throughout the entire range of input area overlap sizes.

The detailed histogram in figure 4.61 provides the last evidence necessary to conclude

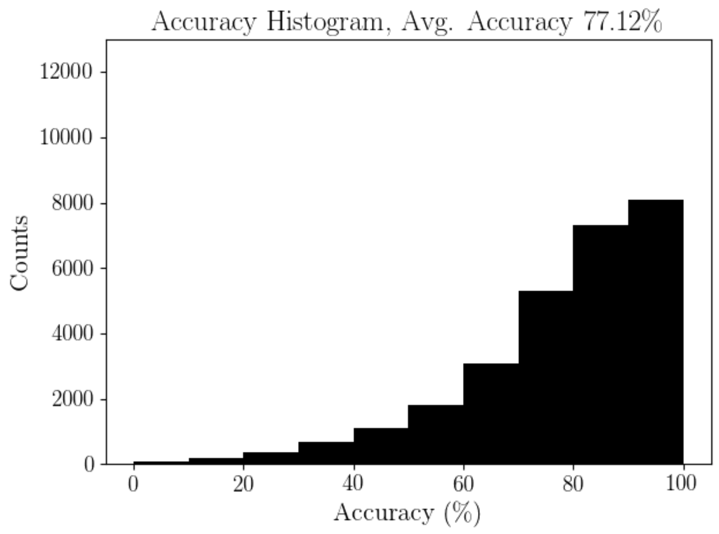


Fig. 4.59: Histogram of the prediction accuracies for row-column overlap encoding model 8.

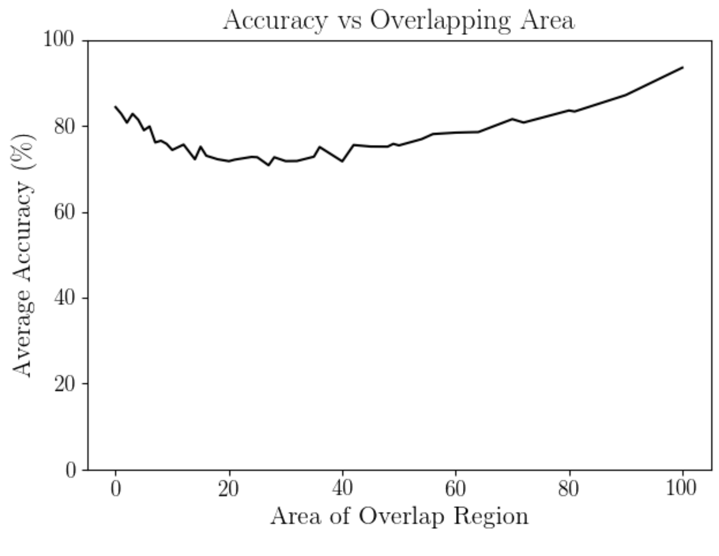


Fig. 4.60: The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 8.

that this model is simply not as good at predicting the masks as model 2. There are simply more instances for all overlapping region sizes where this model cannot predict as accurately.

Again, for consistency, the overlap mask predictions are included in figure 4.62.

This test provides enough evidence supporting the use of the trainable convolution kernels in the CNNs. Every result indicates the DNN is better off, on average, when the kernels are adjusted rather than simply using the random initialization.

Row-Column Model 9

In this model, the network's capacity is further reduced by removing some fully connected layers and by also removing neurons from many of the layers. The detailed architecture for this DNN is shown in figure 3.16. The average prediction accuracy over the entire testing dataset is 80.26%.

The histogram of prediction accuracies in figure 4.63 appears to indicate this model has better performance than the higher capacity DNNs, models 5 and 6. The number of samples for which its prediction is over 90% accurate is again over 10,000.

Similarly, the accuracy vs the size of overlap area shown in figure 4.64 appears to be performing better. The smaller overlapping regions performance dip still exists but is still nearly 75% accurate on average. The upwards trend in prediction accuracy as area overlap size increases is also present and again without the volatility seen others.

The detailed histogram in figure 4.65 shows the limitations of this model. There are still many instances when the overlapping area is less than 40 and the prediction accuracy is less than 40%. Nevertheless, there are still more instances where the prediction accuracy is greater than 80% for all overlap sizes.

The predicted overlap mask is again included for reference in figure 4.66.

This model seems to perform at the same level as model 4 but not quite as good as model 7. It clearly outperforms the more complicated DNNs, model 5 and 6. This final test provides more evidence against the use of the higher capacity models when no pretraining is included.

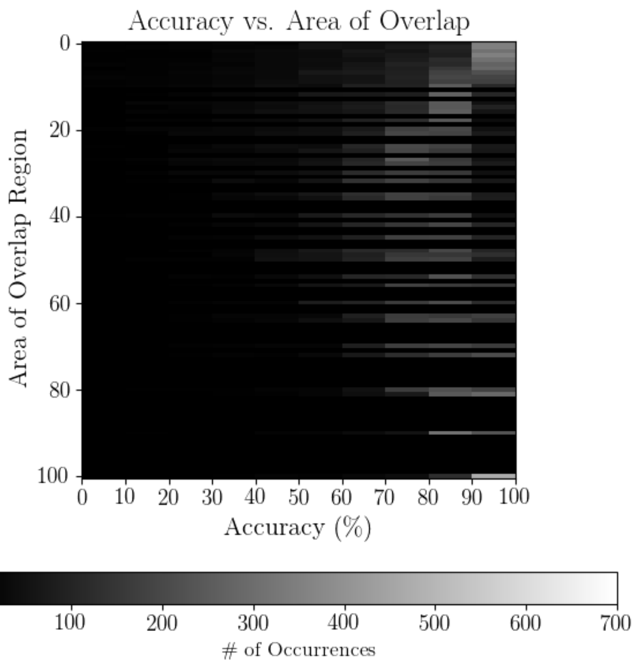


Fig. 4.61: Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 8. The number of occurrences of each is indicated by the pixel intensity.

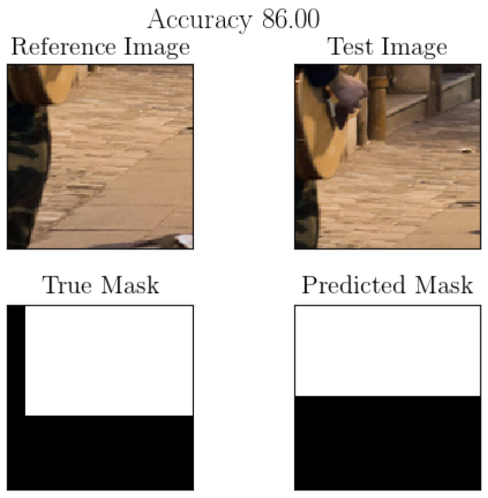


Fig. 4.62: Sample prediction for row-column overlap encoding model 8. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

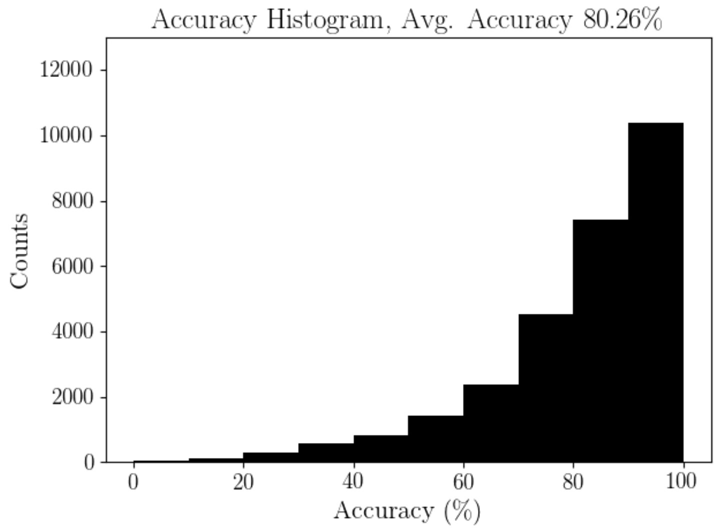


Fig. 4.63: Histogram of the prediction accuracies for row-column overlap encoding model 9.

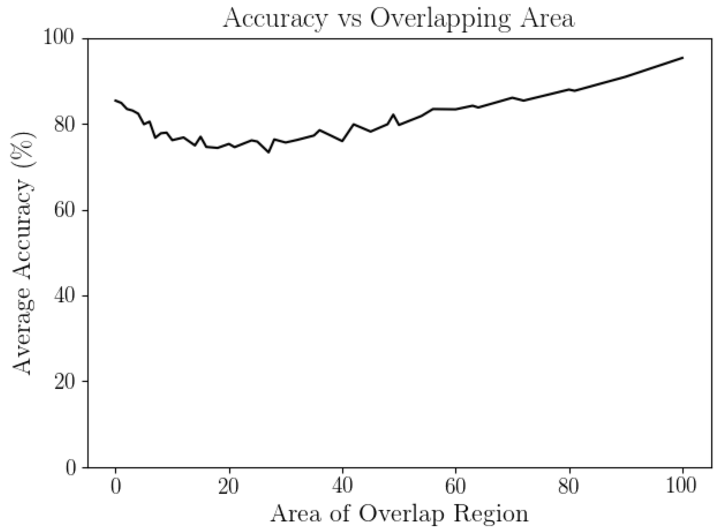


Fig. 4.64: The average prediction accuracy for the possible sizes of overlapping regions for row-column overlap encoding model 9.

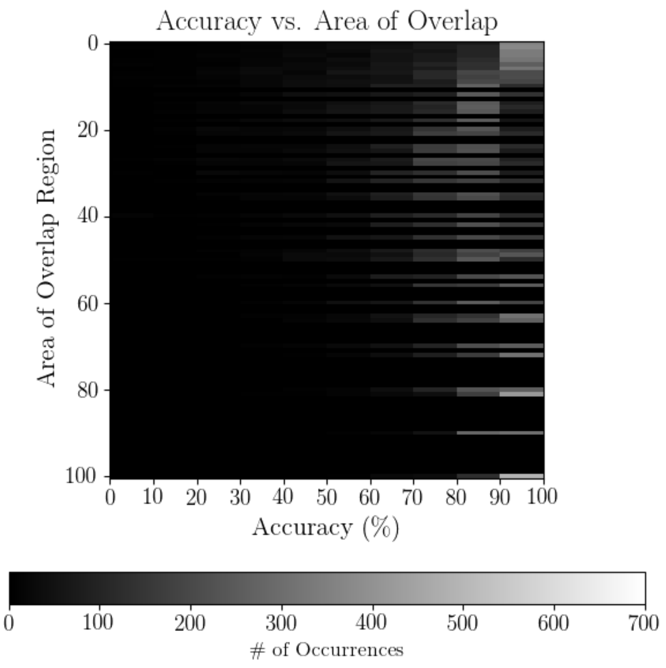


Fig. 4.65: Detailed histogram of prediction accuracies and the area of the overlapping regions for row-column overlap encoding model 9. The number of occurrences of each is indicated by the pixel intensity.

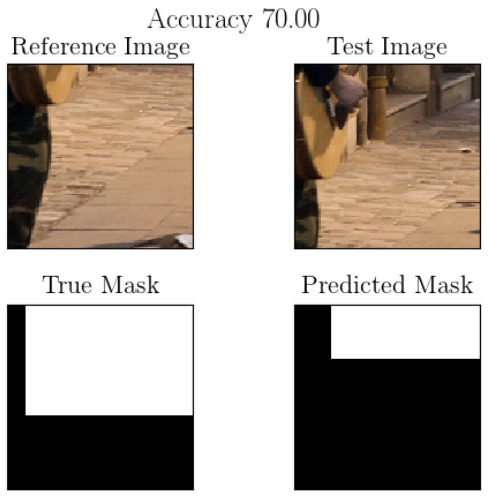


Fig. 4.66: Sample prediction for row-column overlap encoding model 9. The reference image (top-left) and test image (top-right) are shown with the true mask (bottom-left) and the predicted mask (bottom-right). The overlapping area is in white and the non-overlapping region is in black.

4.4 Summary

Overall, the row-column encoding is the best overlap mask encoding to use. Not only did the DNNs using this encoding easily outperform the DNNs using the other encodings, the loss function is tremendously more simple. Furthermore, no additional constraints were necessary for the models to learn the rectangular shape. They managed to learn contiguous regions directly. The results of the higher capacity networks may perform better with additional, more complicated training, but that is left for future work.

The results presented in this chapter only indicate the models can accurately predict the overlap between the images. However, this cannot sufficiently demonstrate that the models were able to learn the association between the reference image and test image. That problem is the subject of chapter 5.

CHAPTER 5

Image Overlap Detection Network Visualizations

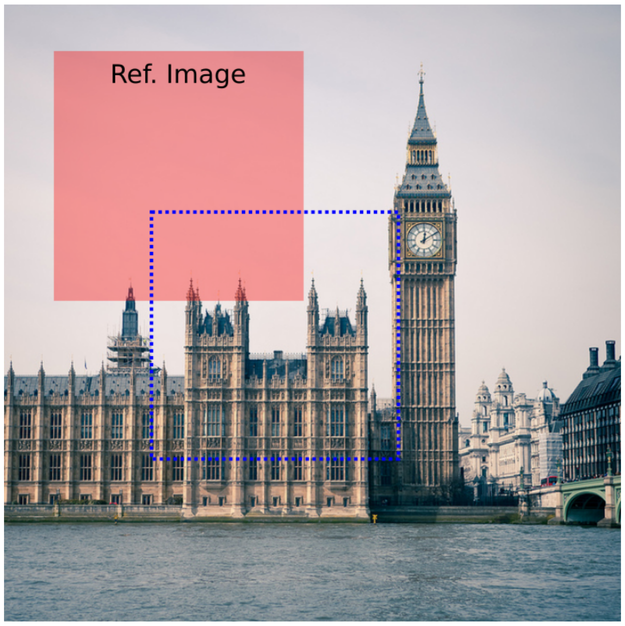
The question regarding whether or not the trained deep neural networks (DNN) have learned the association between the reference and test images is the focus of this chapter. The results and comparisons presented in chapter 4 only address how well the models predict the overlap mask, not if they learned the association.

Two methods to demonstrate how well the DNNs have learned the association are included in this chapter. A search through the test image space to find an input producing a desired output is first developed and presented in sections 5.1 and 5.2. The second method is the examination of the kernel weights and activations presented in section 5.3. The test image space search demonstrates a behavior indicative of learning the reference and test image associations and the kernel/activation visualizations shows what the convolutional layers look for. Both techniques are related to the research on visualizing CNNs discussed in section 2.2.3.

5.1 Test Image Space Search

To evaluate how well (if at all) the trained models learned the association between the images, the prediction problem is posed another way. Given only the reference image, and the mask M, can the test image with overlapping area which satisfies the overlap mask M be found? This is illustrated in figure 5.1. The reference image is the shaded red box and the desired overlap is the matrix M. The solution to the problem posed is the image bounded by the blue outline.

If the DNNs are learning to relate what they see in the reference image to what they see in the test image, then a test image matching the reference image's overlapping region should be able to be found. The image exists (it is already present in reference image) so if it cannot be found using the DNN, then it must be that the DNN has not learned the



$$\begin{bmatrix} 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&1&1&1&1&1\\ 0&0&0&0&1&1&1&1&1&1\\ 0&0&0&0&1&1&1&1&1&1\\ 0&0&0&0&1&1&1&1&1&1 \end{bmatrix}$$

M

Fig. 5.1: Given the reference image and mask M, the test image satisfying the specified overlap is outlined in blue. The DNN is used to find the test image.

association between the input images.

Numerically generating an image was attempted but failed to converge to anything meaningful. The reference image was fixed, and a desired overlap was specified. A uniform random test image served as the initialization estimate. Gradient descent was used to find an image in the test image space to minimize the loss between predicted overlap mask and the true overlap mask. The test image which the gradient descent converged upon appeared noisy and lacked any noticeable structure, much less the anything in the desired overlapping region resembling the reference image.

Instead of creating the image numerically, a manual search through the test image space is used. The trained DNNs can be used to accomplish this search but a minor adjustment to the usual DNN use case needs to be made. Typically, the predictions made by the trained DNN on previously unseen data are the primary outputs. Instead, the primary output for this experiment is an input image. The concept is presented in figure 5.2. The reference image and the desired overlap mask are given, along with a set of test images. The predicted overlap masks for every test image in the set are computed using the DNN. The predicted mask most closely resembling the desired overlap mask is identified and the responsible test image is the output from the entire process. An example of this search is given in section 5.1.2. The creation of the test image set, which the described search assumes is available, is the subject of section 5.1.1.

5.1.1 Test Image Set Creation

The set of test images needs to be created prior to implementing the test image search. If there are no test images having any overlap (i.e. similar regions) with the reference image, then the results aren't particularly useful. Selecting the test images from the same source image as the reference image guarantees some of the test images will have some overlap. Taking this a step further, assuming the position of the reference image in the source image is known, the set of test images can be formed using one reference image to represent every possible overlap orientation. Using equation 2.3 with $n = 10$, the set of test images covering all possible overlaps contains 362 images.

5.1.2 Example Search

As an example of the test image space search, consider the selection and processing of four possible test images, represented in figures 5.3, 5.4, 5.5, and 5.6. This set isn't exhaustive of all possible overlap types, but it is sufficient for this example's purpose. Each test image and the reference image are processed using the DNN to create a predicted overlap mask. The four masks belonging to the four test images are M_{TRUE}, M_{SE}, M_{NE},

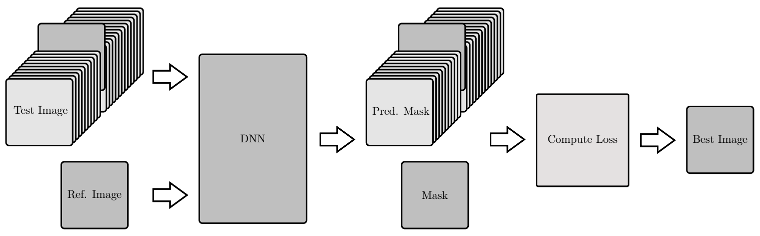


Fig. 5.2: Illustration of the test image space search. The reference image and mask are given. All possible test images are processed using a DNN. The loss is computed between the predicted masks and the true mask, and the test image yielding the lowest loss is chosen.

and M_{NW}. This example assumes the DNN makes perfect predictions.

Using the L2-norm as the loss function, the loss between the desired overlap mask M (shown in figure 5.1) and the predicted overlap masks yields:

$$||vect(M) - vect(M_{TRUE})||_2^2 = 0,$$

$$||vect(M) - vect(M_{SE})||_2^2 = 16,$$

$$||vect(M) - vect(M_{NE})||_2^2 = 25,$$

$$||vect(M) - vect(M_{NW})||_2^2 = 48.$$

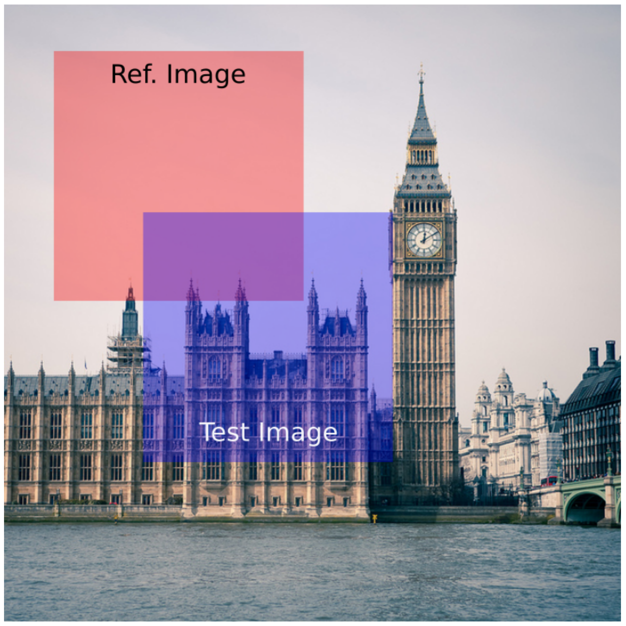
It is clear from the computed losses that the test image corresponding to M_{TRUE} is the image which provides the best desired overlap with the reference image because it resulted in the lowest loss.

Even though the model in this example makes perfect predictions of the overlap mask, its results help to shape the expectations for an imperfect DNN's results if it has successfully learned the association between the input images. A key observation from this simple example is the increase in the loss function as the distance between the test image position and the true test image position increases. This behavior is analyzed in more detail in section 5.2.1.

5.1.3 Implemented Test Image Space Search

The results demonstrated in chapter 4 make it abundantly clear that the trained DNNs are not perfect. It's also apparent from the results that the DNNs performance is dependent on the image and the overlap orientation. Consequently, the test image space search needs to be repeated using many different reference images and overlap types. Naturally this implies an averaging of the results.

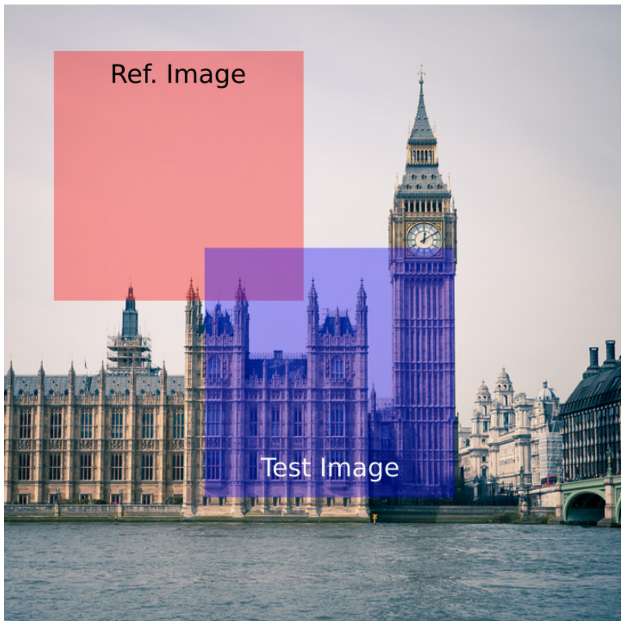
To make averaging between runs meaningful and possible, define the test images' positions relative to the location of the desired test image. The relative positions are specified



$$\begin{bmatrix} 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&1&1&1&1&1\\ 0&0&0&0&1&1&1&1&1&1\\ 0&0&0&0&1&1&1&1&1&1\\ 0&0&0&0&1&1&1&1&1&1 \end{bmatrix}$$

M_{TRUE}

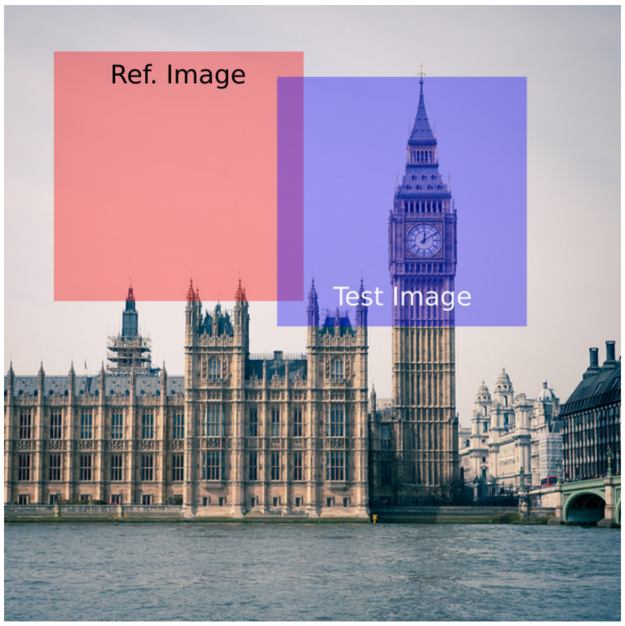
Fig. 5.3: The true test image. The mask corresponds to the overlap between the red reference image and the blue test image. The source image is from the COCO dataset.



$$\begin{bmatrix} 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&1&1&1\\ 0&0&0&0&0&0&1&1&1&1 \end{bmatrix}$$

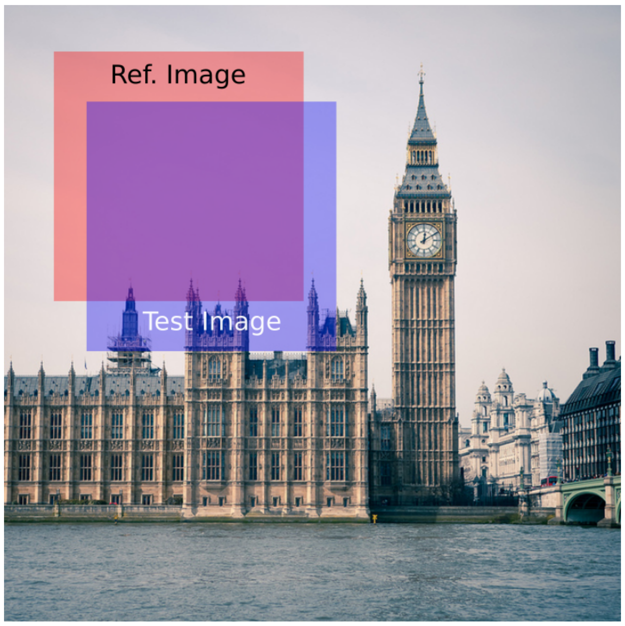
M_{SE}

Fig. 5.4: New test image is shifted down and right from the true test image. The mask corresponds to the overlap between the red reference image and the blue test image. The source image is from the COCO dataset.



$$M_{NE} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Fig. 5.5: New test image is shifted up and right from the true test image. The mask corresponds to the overlap between the red reference image and the blue test image. The source image is from the COCO dataset.



$$M_{NW} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Fig. 5.6: New test image is shifted up and left from the true test image. The mask corresponds to the overlap between the red reference image and the blue test image. The source image is from the COCO dataset.

using an offset in both horizontal and vertical directions. Using the test images from the simple example previously used, the offsets are easily given using the (x, y) offset coordinates as follows:

- The test image in figure 5.3 has offset $(0, 0)$
- The test image in figure 5.4 has offset $(2, -2)$
- The test image in figure 5.5 has offset $(5, 5)$
- The test image in figure 5.6 has offset $(-3, -4)$

Note that the offset values are given in terms of the overlap mask's elements, not the test or reference image pixels.

The implemented image search was not done using a brute force search over all source images, reference images, and overlap orientations. Instead, a dataset is populated with reference images and masks chosen at random. The reference images were selected from the source images and the overlap masks were selected from the set of possible overlap orientations. The set of test images used with every reference image and overlap mask pair was created by selecting images with actual overlaps satisfying a random sampling from the possible overlap types. Care was taken to include the true test image in the set of test images. Additionally, the offsets for every test image in the test image set were recorded for averaging.

Once the DNN makes predictions over the entire dataset, the prediction accuracy and loss are evaluated and averaged over every offset. The results are not exhaustive, but they are representative. For every image in the set of 1122 source images, 18 reference images are randomly selected with a randomly chosen desired overlap for each and for every reference image, 400 test images are randomly chosen and processed. In total, 8,078,400 sample-pairs are processed using every DNN.

5.2 Offset Image Search Results

The results of the test image space stochastic search are presented in this section. Three perfect prediction models, one for each overlap mask encoding type, are presented first in section 5.2.1. The results of the stochastic search for only three of the trained DNNs, one for each overlap mask encoding, are found in section 5.2.2. Results for the remaining models are included in appendix A.

5.2.1 Perfect Model Results

Implementing the full stochastic search using a perfect model (unlike the over simplified example in section 5.1.2) establishes the desired results and necessary performance characteristics for an imperfect DNN trained to learn the input image associations. The prediction accuracy is consistent among the three encodings but the loss functions are not. As a result, three perfect models each using one of the three overlap mask encodings are included.

Perfect Direct Encoding

The average loss at various offsets found using the stochastic search with a perfect model using the direct overlap mask encoding is shown in figure 5.7. The loss function defined in equation 3.3,

$$L = \frac{1}{100}||\mathbf{m} - \hat{\mathbf{m}}||_2^2,$$

is used. The average loss for every offset location is represented by its intensity. The relationship between the intensities and the average loss is given by the color bar. Dark pixels indicate an offset averages a small loss and bright pixels indicate an offset averages a high loss.

The average prediction accuracy found using the stochastic search is shown in figure 5.8. As a reminder, the prediction accuracy is the percentage of elements in the mask predicted correctly. Similar to the average loss previously, the relationship between the average prediction accuracy at an offset and the pixel intensity is provided by the color bar. And again, dark pixels indicate higher values and bright pixels indicate lower values.

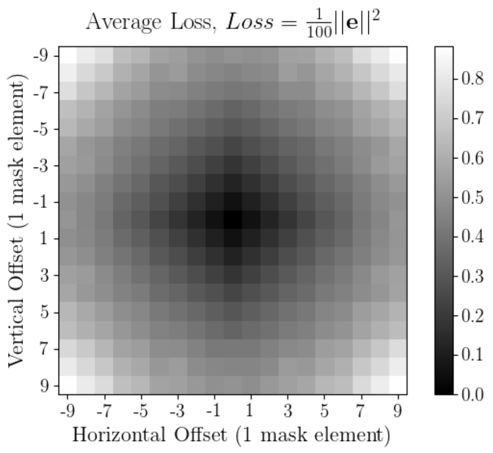


Fig. 5.7: Average loss results of the stochastic image search for the perfect prediction DNN implementing the direct encoding.

A high value is desired for the prediction accuracy so in this case, bright pixels indicate regions of good performance.

The minimum average loss and maximum average prediction accuracy both occur at the $(0, 0)$ offset location, but the shape of the average loss and prediction accuracy surfaces provides more insight into how well the DNN learned the input images association. To understand this, a fundamental realization needs to be made. A small offset between a test image and the desired test image implies both images overlap the reference image in a similar way. Consequently, their individual overlap masks won't be significantly different. This results in the loss remaining low and the prediction accuracy remaining high. This idea can be expanded to larger offsets following a similar line of thought. An increase in the offset means a greater dissimilarity between the overlap masks for the test image and the desired test image. This results in higher losses and lower prediction accuracies.

As a result of this test, the level of success any imperfect model using the direct overlap mask encoding has at learning the association between the images will be apparent in the shape of the average loss and average prediction accuracy plots. The relationship between offset distance and the loss and prediction accuracy exists because the model knows the association between reference image and test image. Without this understanding, the parabolic shape apparent in both figures 5.7 and 5.8 simply can't exist.

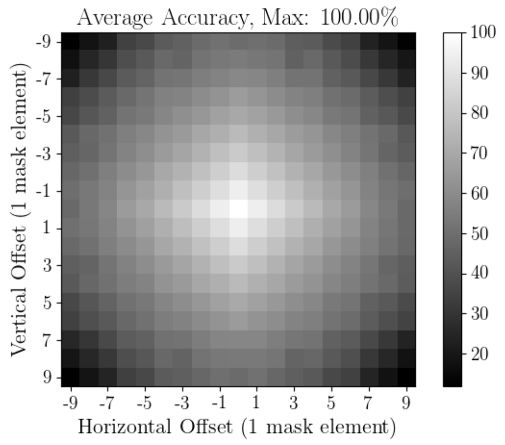


Fig. 5.8: Average prediction accuracy results of the stochastic image search for the perfect prediction DNN implementing the direct encoding.

Perfect Row-Column Encoding

The implementation of the stochastic search for the row-column overlap mask encoding is nearly identical to that implemented in section 5.2.1. The only difference is the loss function used. For this encoding, equation 3.18,

$$L = \frac{1}{10}(||\mathbf{r} - \hat{\mathbf{r}}||_2^2 + ||\mathbf{c} - \hat{\mathbf{c}}||_2^2),$$

is the loss function. The average loss for the various offsets is shown in figure 5.9 and the average prediction accuracy for the offsets is shown in figure 5.10.

The average prediction accuracy for this encoding is nearly identical to that from the direct encoding. However, the change of loss function results in an expected change in the average loss surface.

Comparing the average loss and average prediction accuracy plots in figures 5.9 and 5.10 to any imperfect model which uses the row-column overlap mask encoding provides insight into how well the imperfect model learned the association between the input images.

Perfect Corner Encoding

The implementation of the stochastic search for the corner overlap mask encoding is

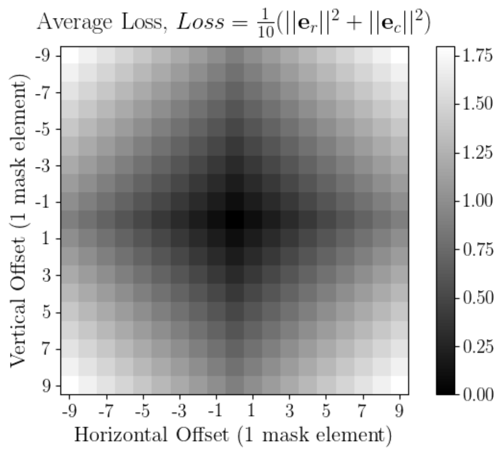


Fig. 5.9: Average loss results of the stochastic image search for the perfect prediction DNN implementing the corner encoding.

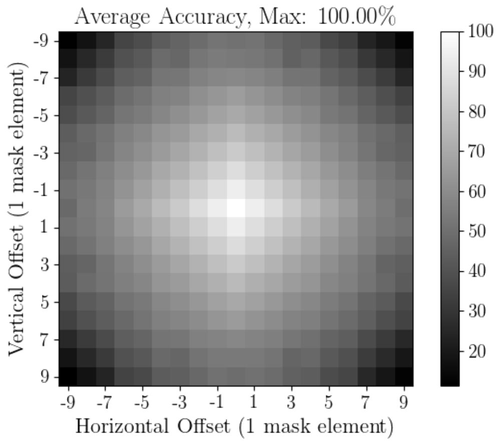


Fig. 5.10: Average prediction accuracy results of the stochastic image search for the perfect prediction DNN implementing the corner encoding.

nearly identical to that implemented in sections 5.2.1 and 5.2.1. Again, the only difference is in the loss function used. For this encoding, equation 3.14 is used as the loss function. The loss equals

$$L = L_a + 2.75L_b$$

with

$$L_a = -\frac{1}{4}\sum_{n=1}^{4}[a_n log(\hat{a}_n) + (1 - a_n)log(1 - \hat{a}_n)],$$

$$L_b = -\frac{1}{100}\sum_{n=1}^{100}[b_n log(\hat{b}_n) + (1 - b_n)log(1 - \hat{b}_n)],$$

and with **a** and **b** being the two corner vectors. The average loss and average prediction accuracy results are shown in figures 5.11 and 5.12 respectively.

The average loss when using this encoding is interesting. The loss is still lowest at the $(0, 0)$ offset but there isn't an intuitive interpretation of the remaining offsets. This is the first loss considered that isn't naturally a distance metric comparing the masks. However, the average prediction accuracy retained the correct shape which indicates the average prediction accuracy is dependent on the offset distance. Therefore, this visualization approach for this encoding is still valid but not as informative as seen in the other encodings because of the loss function used.

5.2.2 Real Model Results

The results of the stochastic test image input space search are presented in this section. If the trained models are able to learn the association between the reference and test images, then their average losses and average prediction accuracies should be similar in appearance to those shown in section 5.2.1. Results from one model for each of the three encodings are presented in this section and the results from the remaining 12 models are included in appendix A.

Row-Column Encoding, Model 2

The average prediction accuracy and loss as a function of the offset found using the

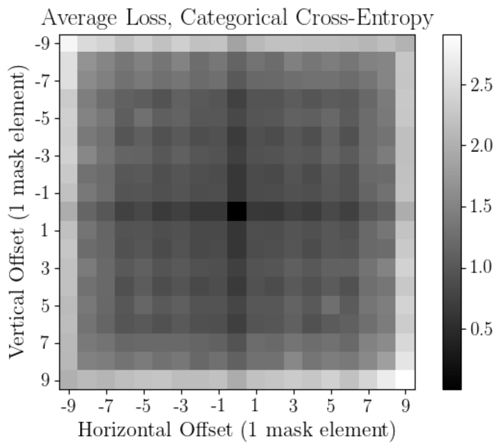


Fig. 5.11: Average loss results of the stochastic image search for the perfect prediction DNN implementing the corner encoding.

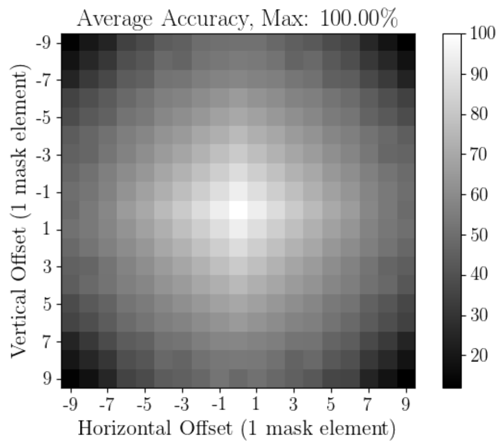


Fig. 5.12: Average prediction accuracy results of the stochastic image search for the perfect prediction DNN implementing the corner encoding.

stochastic search with the model in section 4.3.3 are shown in figures 5.14 and 5.13 respectively.

This imperfect model's average prediction accuracy seen in figure 5.14 has the same overall shape as the perfect model's prediction accuracy demonstrated in figure 5.10. The average prediction accuracy is highest at the $(0, 0)$ offset and decreases as the offset increases. However, the peak average prediction accuracy of 84.84% is a noticeable drop from the 100% from the perfect model.

The average loss in figure 5.13 also has a similar shape to its perfect model counterpart seen in figure 5.7. The imperfect model's average loss doesn't increase with an increasing offset distance as quickly as the perfect model's loss does, but the expected shape is present.

The results of the stochastic search demonstrate that model 2 from the row-column encoding has, on average, learned the association between the reference image and test images. Though not perfect, the expected characteristics observed in section 5.2.1 are present here.

Corner Encoding Model

The average prediction accuracy and loss as a function of the offset found using the stochastic search with the model in section 4.3.2 are shown in figures 5.16 and 5.15.

This imperfect model's average prediction accuracy seen in figure 5.16 has the same overall shape as the perfect model's accuracy in figure 5.12. The average prediction accuracy is highest at the $(0, 0)$ offset and decreases as the offset increases. With a peak average prediction accuracy of 79.61%, the results suggest this DNN managed to learn, on average, the association between test and reference images.

The average loss in figure 5.15 is similar to the perfect model's average loss seen in figure 5.15. The lowest loss occurs in when there is no offset at $(0, 0)$. The semi-regular structure in the perfect case isn't as noticeable in figure 5.15 but there are hints of some structure. While not as conclusive, this is still evidence demonstrating the learned associations.

The results of the stochastic search demonstrate that the trained model using the corner encoding has, on average, learned a noticeable ability to correctly perform the association

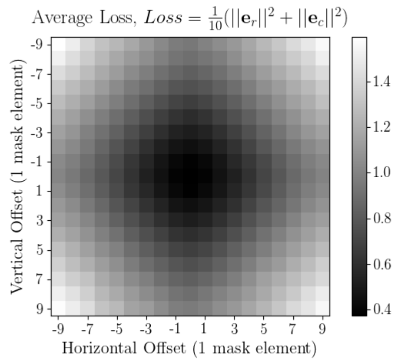


Fig. 5.13: Row-Column encoding, model 2: The average loss results of the stochastic test image space search.

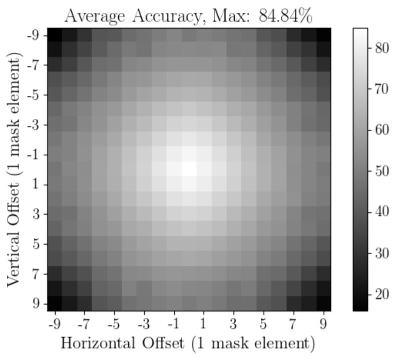


Fig. 5.14: Row-Column encoding, model 2: The average prediction accuracy results of the stochastic test image space search.

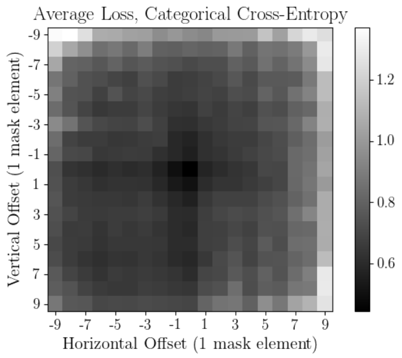


Fig. 5.15: Corner encoding: The average loss results of the stochastic test image space search.

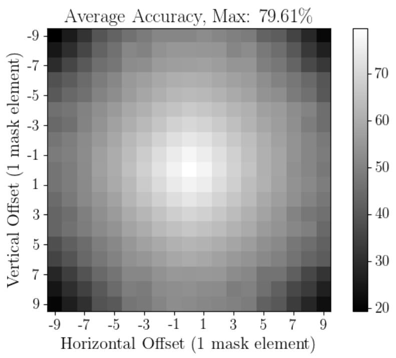


Fig. 5.16: Corner encoding: The average prediction accuracy results of the stochastic test image space search.

between the reference image and test images. There is enough resemblance between the figures in this section and those in section 5.2.1 to arrive at this conclusion.

Direct Encoding Shape Forcing Model

The average prediction accuracy and loss as a function of the offset found using the stochastic search with the model in section 4.3.1 are shown in figures 5.18 and 5.17.

This model uses the direct encoding so a comparison is made to the results in section 5.2.1. Comparing the average prediction accuracies yields results similar to those seen in the previous sections. The surface shapes are similar but this model's average prediction accuracy only reaches a peak average prediction accuracy of 79.60%.

The loss function used to train this model is different than the mean squared error so the values in the comparison will be different. As a reminder, the loss function used is equation 3.9. The most important characteristic is the shape of the surface, specifically, the loss is lowest at $(0, 0)$ and increases as the offset distance increases. This is clearly observable; therefore, this suggests the model has learned the associations between the images.

The results of the stochastic search demonstrate that model using the direct encoding with a shape forcing constraint has, on average, learned the association between the reference image and test images. The expected characteristics observed in section 5.2.1 also observed in this section support this conclusion.

5.3 Weights and Activations Visualizations

The evidence of the DNN's ability to learn the association between the reference image and test image presented in section 5.2 provides no indication of how the DNNs are making the associations. Additional inspection of the DNNs is necessary but understanding the models fully connected layers isn't yet feasible and is regularly a focus of the criticism directed towards DNNs. Instead, the approach used in this section is limited to an examination of the convolutional layers.

The approach to understanding and interpreting the convolution layer used in this section is simple. The convolution kernels are displayed as RGB images and the activations

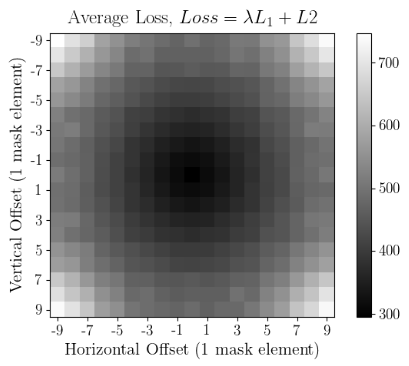


Fig. 5.17: Direct encoding, emphasis on prediction overlap region shape, $\lambda = 10$: The average loss results of the stochastic test image space search.

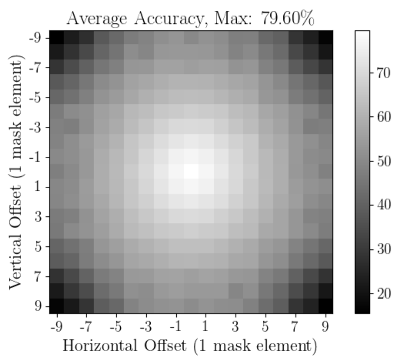


Fig. 5.18: Direct encoding, emphasis on prediction overlap region shape, $\lambda = 10$: The average prediction accuracy results of the stochastic test image space search.

(the outputs of convolution layer) are displayed as grayscale images. This follows work summarized briefly in [24]. Only the kernels and activations for three models are presented in this section and the visualizations for the remaining models can be found in appendix A. The image used in the generation of the activations is shown for reference in figure 5.19.

5.3.1 Row-Column Encoding, Model 2

The 20 convolution kernels for model 2 are displayed in figure 5.20. Each kernel is $12 \times 12 \times 3$ and is intended to operate on an RGB image. Each kernel is displayed using the same scale with a small amount of interpolation to provide smoothness. Unlike the kernels discussed in [24] which exhibit structure and Gabor-like features, these individual kernels are seemingly random. Prior to the training process, the kernels are similar to those seen in figure A.19 and obviously, the kernels were unable to the fully move past the randomness entirely. However, the differences between the kernels when viewed as a group indicate adjustments were made to the kernels during the training process. A more thorough discussion on the structure of the kernels is delayed until section 5.3.4.

The 20 activations (the outputs of the convolution layer) shown in figure 5.21 are each 10×10 pixels. The activations shown use some interpolation between pixels to provide a smoother visualization and easier interpretation. Each activation is the output of filtering

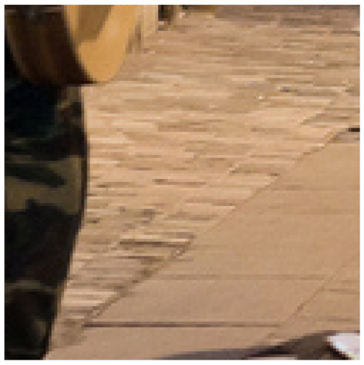


Fig. 5.19: Image used to generate the activations.

and downsampling the image in figure 5.19 using the kernel of the same position in 5.20. The fully connected layers ultimately make the decision about where the overlap exists and do so using the activations. Visually inspecting the activations provides no indication of the model's preference for shapes or patterns. Structural details of the original image are visible and in many instances appear to be repeated.

5.3.2 Corner Encoding Model

The 20 convolution kernels for the corner model are displayed in figure 5.22. Like the kernels in section 5.3.1, these kernels are all $12 \times 12 \times 3$ and again operate on an RGB image. In general, the kernels appear similar to those in figure 5.20 even though the order may vary. Some kernels are visibly lighter than others and some are darker. But perhaps more importantly, the individual kernels still show a clear affiliation to randomness rather than structure.

The 20 activations shown in figure 5.23 are also similar to those seen in figure 5.21. This observation is not entirely unexpected because of similarities seen in their kernels. As before, the structure of the input image is still noticeable in many of the activations.

5.3.3 Direct Encoding, Shape Forcing Model

The 20 convolution kernels for the predicted overlap shape forcing model are shown in

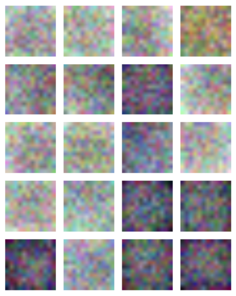


Fig. 5.20: Row-Column encoding, model 2: Convolution kernels.

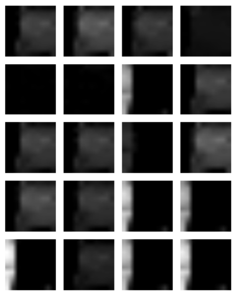


Fig. 5.21: Row-Column encoding, model 2: Activations. Each activation is produced by the kernel in the same position.

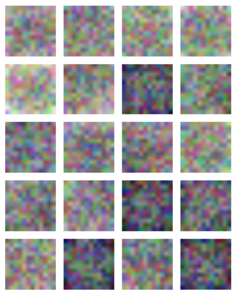


Fig. 5.22: Corner encoding: Convolution kernels.

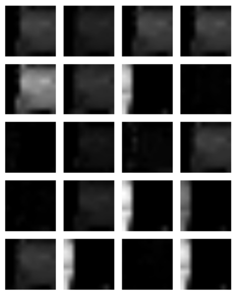


Fig. 5.23: Corner encoding: Activations. Each activation is produced by the kernel in the same position.

figure 5.24. The kernels are again seemingly random just as in the previous two sections, but even more so now. The visibly light and dark kernels are not as pronounced and the kernels are now much closer to those seen in figure A.19. The kernels were randomly initialized and there is no indication the training process was able to move them in any meaningful way.

The activations shown in figure 5.25 are again similar to those seen previously. Just as before, the structure of the test image is present in many of the kernels and the activations appear to be repeated.

5.3.4 Kernel Discussion

If the claim made by [23] about kernels all being similar is assumed true, then something is amiss. However, there are fundamental differences between this problem and the object detection problems. In object detection problems, the main concern is understanding the contents of the image. All cats are similar, all dogs are similar, all cars are similar, etc. and the DNNs are trained to minimize the classification error using the categorical cross-entropy. In the image overlap detection problem, the main concern is not so much what is in the image, but if something similar is found anywhere in the other image.

Object detection networks are presented with positive examples of a class (and also negative examples as a result) during training and it adjusts its weights accordingly. In the overlap detection problem, the concept of a positive or negative example is not as defined. Each element in the mask indicates that its represented region of the reference image either does or does not exist somewhere in the test image. It is unable to provide any information about where it exists, only that it exists. Therefore, a positive example for any individual element in the overlap mask simply means that its represented region in the reference image can be found somewhere in the test image.

Furthermore, the kernels are setup with horizontal and vertical strides equal to the size of the kernel. The result of this setup is that each element in the activation image is calculated using different perceptive fields without any overlap between them. This downsamples the input image and in some cases, the downsampling may be too aggressive resulting in some spatial aliasing.

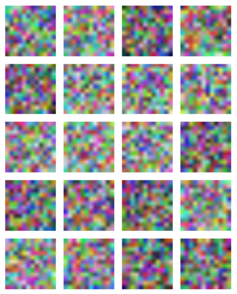


Fig. 5.24: Direct encoding, emphasis on prediction overlap region shape, $\lambda = 10$: Convolution kernels.

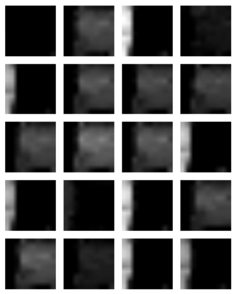


Fig. 5.25: Direct encoding, emphasis on prediction overlap region shape, $\lambda = 10$: Activations. Each activation is produced by the kernel in the same position.

It is apparent from the kernels and activations presented that these models are not learning to understand the content of the images. The kernels they are learning are simply representations of the input image that provide the fully connected layers enough detail to perform the comparison.

The authors in [23] suggest that the seemingly randomness is also indicative of poorly chosen parameters for the convolutional layer. It may be that the parameters of the convolution layers are poorly chosen for an object detection task, but as discussed above, this problem is fundamentally different.

5.4 Conclusion

The fundamental concern regarding the models of this research is whether or not they are actually learning the association between the reference image and the test image. The image search experiment revealed the models do indeed learn the association between the images and visualizing the weights and activations further suggests the models are not learning the content of the images. Given the observations presented in this chapter, it is reasonable to conclude that the models did learn the association between the reference and test images and not their content.

The problem addressed in these chapters is restricted to purely translational overlaps.

Future work with this problem may remove this constraint and address rotational overlap, scaling differences, and perspective changes. At the moment, the general applicability of this work is limited, but including these other types of overlap will make this work more generalized to a broader range of applications.

CHAPTER 6

Emitter and RF Emission Association Introduction

The portion of this research emphasizing learning associations between multiple modalities focuses on the specific case when one input is radio frequency (RF) measurements and the other is RGB video. Specifically, the RF measurements are observed RF emissions and the location of the emitter is unknown but potentially observable in the video data.

The emitter and emission association problem is identifying which observed RF emissions originated from which potential emitter. The problem is illustrated in figure 6.1. The region of operation (depicted by the black square) is defined as the region within the composite field of view of a set of video cameras. Two potential emitters are in the region of operation and one is outside the region. The example RF spectrum shown indicates two separate emissions, each observable and measured by the three antennas. The emitter and emission problem is correctly identifying which, if any, of the two potential emitters are the source of the two observed RF emissions.

A simplified overview of existing RF localization techniques is presented in section 6.1. Work related to this research is discussed in section 6.2.

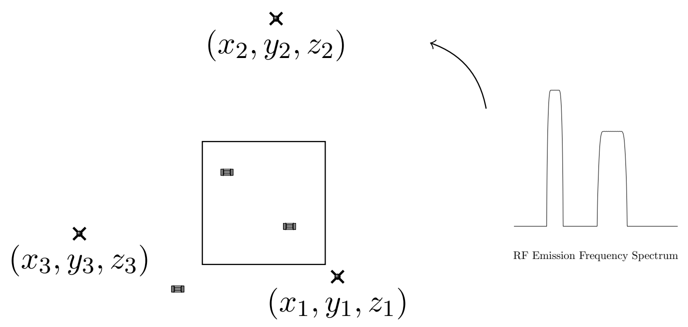


Fig. 6.1: The emitter and RF emission problem involves identifying if any of the emitters detected in the region of operation (black square) are the sources of the observed RF emissions. The RF emissions are observed using three antennas with known locations.

6.1 Locating an RF Emitter

Localizing an emitter using an array of antennas is accomplished in various ways. Each array processing approach exploits the physical properties of the array geometry and the signal itself. A recent survey by Tahat et al. performs a comparison of common range-based array processing approaches, namely: received signal strength (RSS), time of arrival (TOA), time difference of arrival (TDOA), frequency difference of arrival (FDOA), and angle of arrival (AOA) [45]. This research makes use of TDOA measurements so a brief overview of TDOA is included in section 6.1.1. For a description of this research's incorporation of TDOA measurements, refer to chapter 7.

6.1.1 Time Difference of Arrival Calculation

Consider the scenario presented in Figure 6.2 where an object is transmitting some arbitrary signal s, at time t, $s(t)$. Antennas 1 and 2 are d_1 and d_2 meters away from the transmitter respectively. After a finite amount of time, τ_1, the signal $s(t-\tau_1)$ is observed and at antenna 1, and similarly for a delay of τ_2, the signal $s(t-\tau_2)$ is observed at antenna 2. TDOA isn't concerned with the initial transmission time t but instead uses the difference in the two arrival times, $\tau = \tau_1 - \tau_2$. Time difference τ can be used to rewrite the observed signals as $s(t')$ and $s(t'-\tau)$ and is the key component in the TDOA based localization techniques.

In a simplified case, the time difference can be determined using cross correlation to compare the received signals. If real signals are assumed, then for two arbitrary signals $g(t)$ and $h(t)$ the correlation is defined as the convolution of one signal with the time reversal of other,

$$R_{gh}(t) = g(t) * h(-t) = \int_{-\infty}^{\infty} g(t')h(t'+t)dt'. \tag{6.1}$$

Using the Fourier transform property of convolution and $H(-f) = H^*(f)$ for real signals, equation 6.1 in the frequency domain equals

$$\mathcal{F}(R_{gh}(t)) = G(f)H(-f) = G(f)H^*(f). \tag{6.2}$$

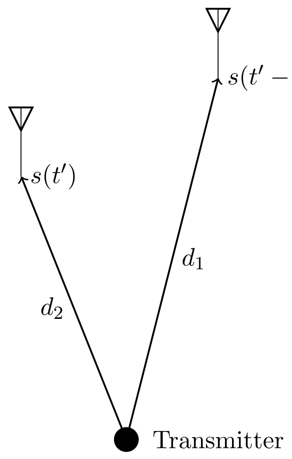


Fig. 6.2: Example TDOA calculation where two antennas observe the same transmitted signal τ seconds apart.

Now letting $g(t) = s(t)$ and $h(t) = s(t-\tau)$, the correlation of the two received signals equals (substituting into equation (6.2))

$$\mathcal{F}(R_s(t)) = \mathcal{F}(s(t))\mathcal{F}^*(s(t-\tau)) = S(f)S^*(f)e^{-j2\pi f\tau}, \tag{6.3}$$

and simplifying gives

$$\mathcal{F}(R_s(t)) = |S(f)|^2 e^{-j2\pi f\tau}. \tag{6.4}$$

The Fourier transform of the correlation is complex, and the time difference of interest τ is present only in the phase term. Furthermore, the phase is a linear function of frequency and the slope of the phase is determined by τ. Therefore, to determine the time difference, one only needs to compute the slope of the phase resulting from the multiplication of the received signals' Fourier transforms.

6.1.2 Localization Using TDOAs

In an isotropic propagation environment, the TDOA between two antennas corresponds to two physical paths of constant time difference called isochrones. The two isochrones are symmetric about the midpoint between the two antennas and any point along either path has the same TDOA. Using more antennas to compute additional independent TDOAs gives additional isochrones and the intersection of the isochrones is the source of the emitter. This principle is illustrated in figure 6.3.

6.2 Related Work

The approach described thus far assumes an ideal setting, i.e. the time delays of the systems are known and accounted for leaving only the propagation delay in the TDOA calculation, and an isotropic propagation environment including line-of-sight (LOS) between emitters and antennas. Adjustments need to be made when these assumptions don't hold. To improve the accuracy of TDOA techniques in non-line-of-sight (NLOS) environments, Thomas et al. used a biased Kalman filter on the TOA data prior to computing the TDOAs [46]. An alternative approach using pilot signals is demonstrated by Jiang et al. to improve accuracy in NLOS environments [47].

The research of Bao et al. uses an analytical approach to fusing both RF and video observations [48]. In their problem, the authors focus on tracking a wireless endoscopic capsule inside a small intestine. The position of the capsule is inferred from the video data and is then refined using the RF signal emitted from the capsule. Two fundamental differences between the capsule tracking and the approach used in this research exist. First, in this research the modalities are weighted equally during processing and one set of observations is not used to refine the other. Second, the approach used in this research uses deep neural networks (DNN) instead of an analytical solution.

Successful implementation of disjoint multimodal processing using deep neural networks is demonstrated by Ngiam et al. [49] and by Mroueh et al. [50] for a lip reading application. The lip reading problem focuses on understanding speech using only the video frames of recorded lips during speech. The authors of both papers explore a couple approaches for incorporating audio data during the DNN training phase to improve the visual recognition performance. Because their goal was to improve the performance of one modality by using another modality, like the capsule tracking above, this work is still different from the emitter and emission association problem.

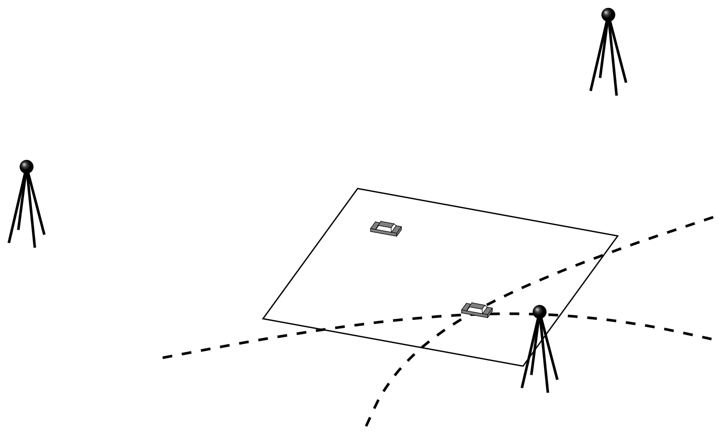


Fig. 6.3: TDOA localization involves finding the intersection of multiple isochrones. Three antennas provide two sets of independent TDOAs and each set is symmetric between its antennas. Only the pair which intersect in the operating region are depicted.

6.3 Summary

The emitter and RF emission association problem focuses on identifying which observed RF emissions are from which RF emitters. Specifically, this is the association of emitters identified in video frames with the RF measurements collected using multiple antennas. The DNN approach is introduced in chapter 7 along with a discussion about the training and testing datasets. The training results and an evaluation of the ability of the DNNs to learn the association between the two modalities is the subject of chapter 8.

CHAPTER 7

Emitter and RF Emission Association Experiments

As was introduced in chapter 6, the emitter and RF emission association problem asks a DNN to learn a relationship between emitter and RF emission. The emitters and observed RF emissions are inherently related because of the physical properties of the RF emissions. It is because of this inherent relationship that the RSS, AOA, TOA, TDOA, and FDOA techniques summarized in [45] are even viable solutions to the localization problem.

Instead of learning to localize a source using the RF measurements, the DNN is trained to learn a relationship between the observable potential emitters in the video frames and the RF measurements from the antenna array. The proposed approach along with some logistical challenges are discussed in section 7.1. Multiple datasets having various parameters are presented in section 7.3 followed by an overview of the DNNs in section 7.4.

7.1 Solution Approach

The purpose of using multiple modalities is to utilize observations of separate physical properties from a single process. However, a fundamental challenge in multimodal processing is the individual sampling of the modalities. Maintaining time-aligned samples is difficult and is further complicated by sample rate disparities. For example, with a video frame rate of 30 frames per second (fps) and a low RF sample rate of 1 kHz, there are approximately 33 RF samples for every video frame. Ensuring the roughly 33 RF samples match up with the video frame is difficult, but doable. However, if the RF sample rate is 1 MHz then there are at least 33,000 RF samples for every video frame and difficulties in the alignment are amplified.

This work is introductory in nature and is concerned more with the feasibility of learning the associations and less with the implementation challenges. As such, the sample rate discrepancy is not addressed and time alignment between modalities is assumed. These assumptions imply that preprocessing, both for the video data and the RF data, is available and additionally provides features representing the separate modalities in place of the raw measurements.

Optical preprocessing is assumed to be capable of identifying the positions of potential emitters within the region of operation in the video data. No restrictions are made as to how the objects are detected, only that the positions are available. The RF preprocessing is assumed to calculate two independent TDOAs from measurements made using an array composed of three antennas. The processing chain is illustrated in figure 7.1. The video data is processed to provide positions of potential emitters, and the RF data is processed to provide TDOAs. The (x, y) position and the two TDOAs are concatenated together to form a four element input vector for the DNN.

The use of features contrary to raw data modifies the association problem. Instead of learning a relationship between the raw video frames and the raw RF measurements, the DNN must learn a relationship between the features used. For this research, the DNN is trained to learn a relationship between positions in the region of operation and the TDOAs.

The DNNs implemented are basic fully connected networks due in large part to the simple nature of the inputs. Each association is made independent of past or future associations so added complexities of recurrent neural networks outweigh potential benefits. The inputs are four element vectors having no spatial structure so convolutional neural networks

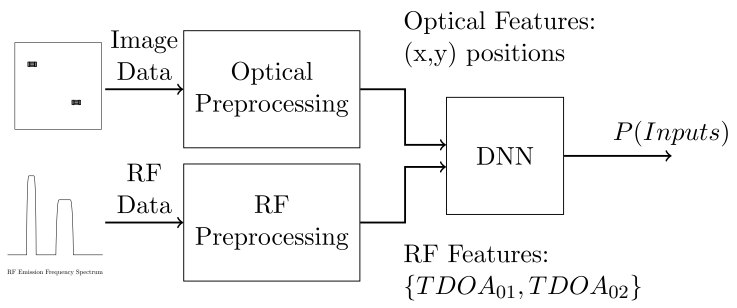


Image Data
Optical Preprocessing
Optical Features: (x,y) positions
RF Data
RF Preprocessing
RF Emission Frequency Spectrum
RF Features: $\{TDOA_{01}, TDOA_{02}\}$
DNN
$P(Inputs)$

Fig. 7.1: Overview of the processing chain. The video frames are preprocessed to obtain optical features and the RF data are preprocessed to obtain the RF features and both features are used as inputs of a DNN. The focus of this research is the DNN and the format of the output $P(Inputs)$.

don't possess an advantage over the fully connected networks. However, the size of the fully connected network is less important than the format of the network's output.

The relationship between the video features and the RF features is communicated to the DNN during the supervised training process using the target outputs. Specifying the pair of features as either a valid or invalid pairing drastically simplifies the relationship between the video and RF features and establishes the association problem as a binary classification problem. However, approaching the problem from the binary classifier perspective has a subtle flaw.

Classifiers are discriminators. In input space, the classes are not typically linearly separable resulting in decision rules which are intractable. During supervised training, the DNN uses examples from different classes to learn a mapping from the input space to a classification space where the classes are more linearly separable. Once a model is trained it does indicate how different an input vector is from the vectors of the training dataset, it simply computes the mapping and then makes its decision. Therefore, for an input vector not adequately represented during training, the perfectly reasonable output decision for a DNN may seem entirely unreasonable to a human observer. Within the context the emitter and RF emission association problem, if the set of all possible invalid pairings is not properly represented during training, unpredictable results on different datasets are possible.

Instead of formulating the relationship between the video and RF features as a binary classification problem, this research defines it as a probability estimation problem. Instead of a yes or no for every (x, y) and TDOAs pairing, a probability of the validity is implemented. Consequently, this research focuses more on representing the probability and less on finding the optimal DNN architecture.

7.2 Probability Representation

The desired validity probability of a pairing between a position and set of TDOAs is defined in terms of two separate positions. The true source of the RF emissions is denoted $\mathbf{x}_{true}$ and the position being tested is denoted $\mathbf{x}_{test}$. Using the two separate positions, the probability of a valid pairing equals

$$P(\mathbf{x}_{test} = \mathbf{x}_{true} | TDOAs) = e^{-\alpha ||\mathbf{x}_{test} - \mathbf{x}_{true}||_2}, \tag{7.1}$$

where

$$\alpha = -\frac{1}{d} \times ln(p) \tag{7.2}$$

with p being the desired probability for a separation distance d.

The desired response of the DNN is illustrated in figure 7.2. Larger values of α require the separation between the tested position and the true position to be small before the DNN will predicts a high probability. Small values of α allow the DNN to predict a higher probability when the separation between tested and true positions is higher. Essentially α allows for the specification of how confident the DNN needs to be that the two positions are the same.

The DNN only sees the computed probabilities, the TDOAs, and $\mathbf{x}_{test}$ and as such, loses any direct information regarding the true position $\mathbf{x}_{true}$. If the DNN successfully learns the relationship between the input features, the predicted probabilities should provide an indication of how close the tested position is to the true position.

Three desired responses defined by equation 7.1 are used as a foundation in this research. Using equation 7.2 with a desired probability $p = 0.001$ at distances $d = 50$, 150, and 350 produces three values for α: $\alpha_0 = 0.01973644$, $\alpha_1 = 0.0460517$, and $\alpha_2 = 0.13815511$. There are no publicly available datasets using the desired probability responses, let alone any dataset containing the two modalities measuring the RF emissions. To remove the sample rate challenges previously mentioned, the datasets are synthetic and the generation procedure is introduced in section 7.3.

7.3 Generating Datasets

Prior to generating the synthetic datasets, the simulation environment must be specified. In the simplest of cases, this involves the region of operation (the region in the field

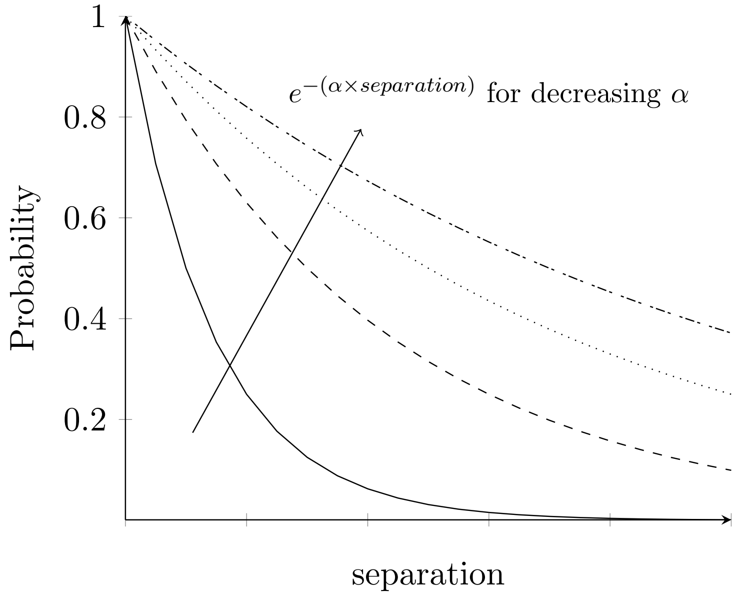


Fig. 7.2: Desired probability behavior. The probability decay rate decreases as α decreases.

of view of the camera) and the configuration of antenna array. More complex and realistic environments can be made, but simplified cases are implemented in this work.

The ratio of valid pairings and invalid pairings must also be specified. The main objection to implementing a binary classifier is the need to sufficiently represent all invalid pairings. A similar requirement exists with this approach but to a lesser extent. Enough samples of the desired probability response need to be available during training for the DNN to learn the response over the entire region of operation.

The script used to generate the datasets is based on algorithm 2. The initialization steps specify the antenna array geometry, the dimensions of the region of interest, the desired α, and the number of invalid samples for every valid sample. Generating the individual samples is a multi-step procedure. The valid pairing input vector is formed by uniformly sampling a position from the region of operation followed by the calculation of the TDOAs. The N invalid pairing input vectors are formed by uniformly sampling N positions from the region of operation and then pairing them with the calculated TDOAs used in the valid pairing. The target outputs for every sample are determined using equation 7.1 with the specified α. The three main datasets are summarized in table 7.1.

Algorithm 2 Algorithm for generating the synthetic dataset used in this research.

```
Initialize sensor array geometry
Identify region of interest
Specify $\alpha$
Set $N$ number of incorrect samples for every correct sample
for k = 1 : Number of input samples do
    Randomly select position for every correct sample
    Compute the TDOAs for every correct sample
    Set target probability for every correct sample to 1
    for Every correct sample do
        Randomly select $N$ positions for incorrect samples
        Pair incorrect sample with correct sample's TDOA
        Compute probability for position and TDOA pair
    end for
end for
Use appropriate scaling and dataset normalization
Write to HDF5 file
```

Table 7.1: Summary of the three main datasets.

Name	α	Probability	Distance	N	Training Samples	Testing Samples
α_0	0.01973644	0.001	350	25	520,000	1040
α_1	0.0460517	0.001	150	25	520,000	1040
α_2	0.13815511	0.001	50	25	520,000	1040

Additional Datasets

In addition to the three main datasets summarized in table 7.1 four additional datasets are used, with each one used to provide further understanding. In two of the added datasets, the number of invalid pairings for every valid pairing is adjusted to let $N = 2$ or $N = 50$. Another dataset increases the number of samples used to 1,560,000 training samples. The final dataset uses an altered antenna array geometry. These additional datasets are summarized in table 7.2.

7.4 Models Summary

In this research, 10 DNNs were trained using one of the seven datasets introduced in section 7.3. Each dataset in table 7.1 was used to train two DNNs resulting in six trained, primary DNNS. Each of the four remaining DNNs were trained using one of the datasets in table 7.2.

With the exception of one, all 10 DNNs use one of two possible architectures. The first architecture has three hidden layers with 50, 60, and 50 neurons in the respective layers. The second architecture adds two additional hidden layers of 50 neurons each to the first architecture. The final model appends two more hidden layers on top of the layers included in the second architecture. The single output neuron uses the sigmoid function while every other neuron uses the rectified linear unit (relu) for the nonlinear activation functions. The DNNs are summarized in table 7.3.

An evaluation of the trained models and a discussion of their ability to learn the video and RF feature relationship is presented in chapter 8.

Table 7.2: Summary of four additional datasets.

Name	α	Probability	Distance	N	Training Samples	Testing Samples
N_2	0.0460517	0.001	150	2	750,000	15000
N_{50}	0.0460517	0.001	150	50	1,530,000	10200
Max Data	0.13815511	0.001	50	25	1,560,000	10400
Alt. Geom.	0.0460517	0.001	150	25	1,560,000	10400

Table 7.3: Summary of the 10 emitter and RF emission association DNNs.

Name	Dataset	Layers
Base 0	α_0	50-60-50
Base 1	α_1	50-60-50
Base 2	α_2	50-60-50
Large 0	α_0	50-60-50-50-50
Large 1	α_1	50-60-50-50-50
Large 2	α_2	50-60-50-50-50
N_2	N_2	50-60-50
N_{50}	N_{50}	50-60-50
Alt. Geom.	Alt. Geom.	50-60-50
Final	Max Data	50-60-50-50-50-60-30

CHAPTER 8

Emitter and RF Emission Association Results and Discussion

The 10 DNNs outlined in table 7.3 were all implemented using the TFlearn API [11] and trained using the Adam optimizer [44] to minimize the mini-batch mean squared error between the predicted and target probabilities.

The positions of the antennas and the region of operation are all specified in the same coordinate system and the region of operation is not required to be centered on the origin. As a result, the individual components of the input vector need to be scaled and shifted to be closer to the input range $[-1, 1]$. The following adjustments to the input vector are used:

$$\begin{aligned} x &= \frac{x - c_x}{w}, \\ y &= \frac{y - c_y}{h}, \\ TDOA_{01} &= TDOA_{01}\frac{c}{s_{01}}, \\ TDOA_{02} &= TDOA_{02}\frac{c}{s_{02}}. \end{aligned} \tag{8.1}$$

The region of operation is centered at (c_x, c_y) and has width w and height h. TDOAs are scaled by the speed of light c because the they are on the order of micro seconds resulting in a small variation between samples. The scale factor s_{jk} adjusts the TDOA between antennas j and k to be closer to the range $[-1, 1]$ and is chosen empirically. Specifically in the datasets and tests presented in this chapter, the region of operation has dimensions $w \approx 280\ m$ and $h \approx 350\ m$ and the TDOAs are scaled using $s_{01} = 700$ and $s_{02} = 850$.

Early stopping was used to reduce over fitting and no other forms of regularization were implemented.

The mean squared error is capable of proving a high-level, direct performance comparison between trained DNNs but it contributes little to the discussion about how well the DNNs learned the association introduced in chapter 7. Instead, two visualizations, a prediction error surface and point responses, are implemented to understand the level of association learning. Additionally, to provide a better means of performance evaluation for comparison amongst models, receiver operating characteristic (ROC) curves are included. After a discussion of the visualization techniques and an overview of the ROC curves, the results for the trained models are given in section 8.1.

Error Surface

The trained DNNs aren't perfect and the prediction error surface is used to understand the learned response throughout the region of operation. As a reminder, the region of operation is roughly 350 by 280 meters. For a given source emitter location in the region of operation, the error surface indicates how far away the predicted emitter location actually is. The predicted emitter location is determined finding the highest DNN probability prediction for the region surrounding the true emitter and using the true TDOA. For example, if the true emitter is located at (x_i, y_i) but the highest predicted probability using the TDOAs corresponds to location (x_j, y_j), then the color of the error surface at (x_i, y_i) represents the euclidean distance between (x_i, y_i) and (x_j, y_j).

For an imperfect DNN, the learned relationship is likely location dependent and the error surface clearly illustrates how well the RF and video features are associated throughout the entire region of operation. If the DNN is unable to learn the relationship between the features, then the DNN will incorrectly assign high probability to invalid feature pairings and thus the error surface distances will be large. If the relationship between the video and RF features is learned perfectly by any DNN, then the prediction error surface will be uniformly zero because the highest predicted probability is always one and always corresponds to the true source location. Therefore, the desired prediction error surface has relatively low and uniform prediction error throughout the entire region of operation.

Point Response

The point responses demonstrate how well the DNNs learned to represent equation 7.1

for a single emitter. The predicted probability at every location is indicated via the pixel color and the mapping between probabilities and pixel colors provided by the color bar. The images are formed by first pairing a single set of TDOAs with every position and then by making predictions using the DNN.

A DNN which has not sufficiently learned the relationship between the RF and video features is easily seen using the point responses. Four point responses for every DNN are provided. The locations of the four emitters are the same for every DNN. If a DNN has learned the relationship, then the probability decay away from the source location in all directions will match the decay seen in equation 7.1.

ROC Curves

The output of the DNN is a probability and as such, has a value in the range $[0, 1]$. Eventually a decision on the validity of an feature pair needs to be made using the predicted probability and the ROC curve is used to evaluate the decision making. Define the set V to be the set of valid input pairs (with valid meaning the position (x, y) the the TDOAs have probability one) and let the two competing hypotheses be

$$\begin{array}{c} H_0 : \mathbf{x} \notin V \\ H_1 : \mathbf{x} \in V \end{array}. \tag{8.2}$$

The null hypothesis is that the calculated TDOAs are not due to an RF emitter at location (x, y). The alternative hypothesis is the calculated TDOAs resulted from an RF emitter at location (x, y). Setting up a likelihood ratio test and simplifying (recognizing $P(\mathbf{x} \in V) + P(\mathbf{x} \notin V) = 1$), the decision rule

$$\phi(\mathbf{x}) = \begin{cases} \mathbf{x} \notin V & P(\mathbf{x}) \geq \tau' \\ \mathbf{x} \in V & P(\mathbf{x}) < \tau' \end{cases} \tag{8.3}$$

for a threshold value τ', is obtained.

Decisions made using the rule given in equation 8.3 are evaluated using traditional definitions of the probability of false alarm and the probability of detection:

$$\begin{aligned} P_{FA} &= P(\text{decide } \mathbf{x} \in V \mid \mathbf{x} \notin V \text{ is true}) \\ P_D &= P(\text{decide } \mathbf{x} \in V \mid \mathbf{x} \in V \text{ is true}) \end{aligned} \tag{8.4}$$

A false alarm is when a video and RF feature pair is not valid but they were incorrectly determined to be valid. A detection is when a video and RF feature pair is valid and was correctly determined to be valid.

Four different ROC curves are generated for every DNN. Each ROC curve corresponds to different levels of difficulty for the decisions making. Every sample tested has a 50% chance of being a valid pair and a 50% chance of being an invalid pair. However, according to the probability desired response, not all invalid samples are as invalid as others. The four different levels of difficulty correspond to how invalid the sample pair is. The samples of the invalid pairs are generated by perturbing the position from the true position by some amount determined by difficulty levels The four difficulty levels and the position perturbations are

- Level 1: 7-9 meters
- Level 2: 5-7 meters
- Level 3: 3-5 meters
- Level 4: 1-3 meters

Some of the probability decays used in the datasets are quite slow and poorly shaped ROC curves are expected due to the size of the perturbations. The DNNs trained on datasets having a faster decay are expected to provide a high probability of detection and low probability of error for this test.

8.1 Training Results

The training results from the six main DNNs trained using the α_0, α_1, and α_2 datasets are presented first. Following their discussion, the performance of the four remaining DNNs is discussed.

8.1.1 Base 0 DNN

The first model discussed is the base 0 DNN trained on the α_0 dataset. As anticipated, the prediction error surface in figure 8.1 clearly shows the DNN performance is location dependent. The upper left corner is particularly poor but that area also happens to be centered between two antennas. Though not desirable, this behavior is acceptable and is observed in traditional TDOA geolocation techniques as well. Most of the poor performance areas are near the borders but there are some peaks and valleys in the middle.

The α_0 dataset has the slowest probability decay of all the datasets used and it is visible in the four point responses seen in figure 8.2. As indicated by the colorbar, the yellow pixels indicate probability one and the deep purple indicate probability zero. The desired slow decay from high to low probabilities is visible but a prominent peak is absent and instead appears spread out. And the point response spread seen in the corners of the operating region provide an explanation of the poor performance seen in the corners of the prediction error surface.

It is apparent from the prediction error surface and point responses that this DNN has learned a relationship between RF and video features but, it is limited. There are areas throughout the region of operation where the DNN has learned the response better than others.

The ROC curves in figure 8.3 reaffirm the low performance indicated by the error surface and point responses. Large regions seen in the prediction error surface had expected position errors greater than two or three meters and the effect is visible in the decisions. The decisions for levels 3 and 4 are nearly as poor as possible (the position perturbations are less than five meters). Even when the perturbations are between seven and nine meters, the valid pairings are only identified roughly 70% of the time. To achieve a higher probability of detection, more and more invalid pairings need to be incorrectly determined to be valid.

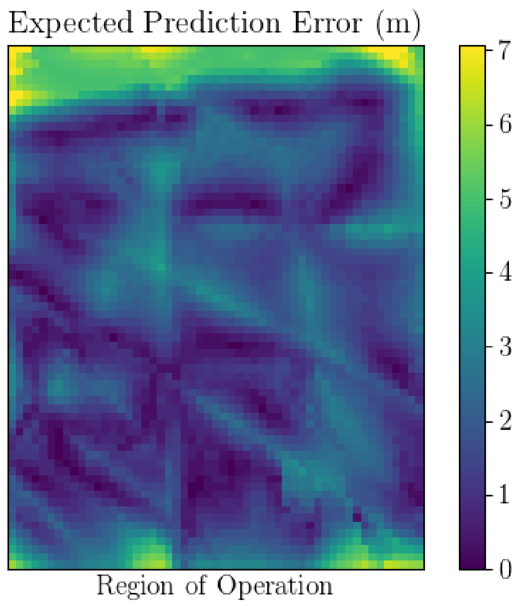


Fig. 8.1: Prediction error surface for the base 0 DNN. The region of operation is approximately 350 by 280 meters.

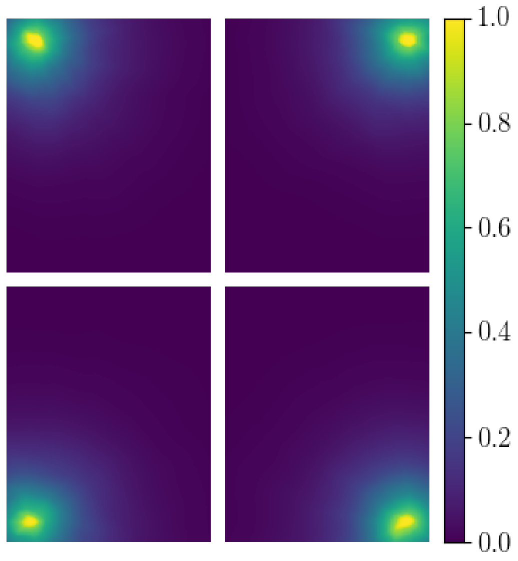


Fig. 8.2: Point Responses for the base 0 DNN. Each point response is shown over the entire region of operation.

8.1.2 Base 1 DNN

The prediction error surface in figure 8.4 is from the base 1 DNN trained on the α_1 dataset. Like the error surface for base 0, the border of the operating region has areas of low performance but now the expected prediction error is higher in many other areas. Within the interior of the operating region, there are now fewer areas having a prediction error less than two meters.

The decrease from base 0 is also seen in the point responses in figure 8.5. The probability decay is present and visible but the general shape of the responses is objectively worse. The peak is more pronounced but the rotational symmetry which should be present is missing.

Even though the desired response for this DNN is different from that of base 0, it is still apparent that this DNN did not learn its intended relationship as well as base 0 did its own. However, like the base 0 model, this model has managed to learn a very limited relationship between the RF and video features.

Even though the error surface in figure 8.4 appears worse than the error surface in figure 8.1, the decision making is demonstrably better for levels 1, 2, and 3 using the Base 1 DNN. The probability decay is faster now, making this model better suited for the decision making. The error surface only indicates how far off the highest probability occurs, not what the probability is. It is therefore possible for an incorrect location to have the highest probability while the correct location also maintains a high probability.

The results thus far suggest that, as expected, the DNN trained to learn the relationship having the faster decay is better suited for use because it is able to resolve smaller distances. Furthermore, the results also indicate that the relationship between the features having the fast decay is harder to learn.

8.1.3 Base 2 DNN

The base 2 DNN is the same as the two previous models except it was trained using

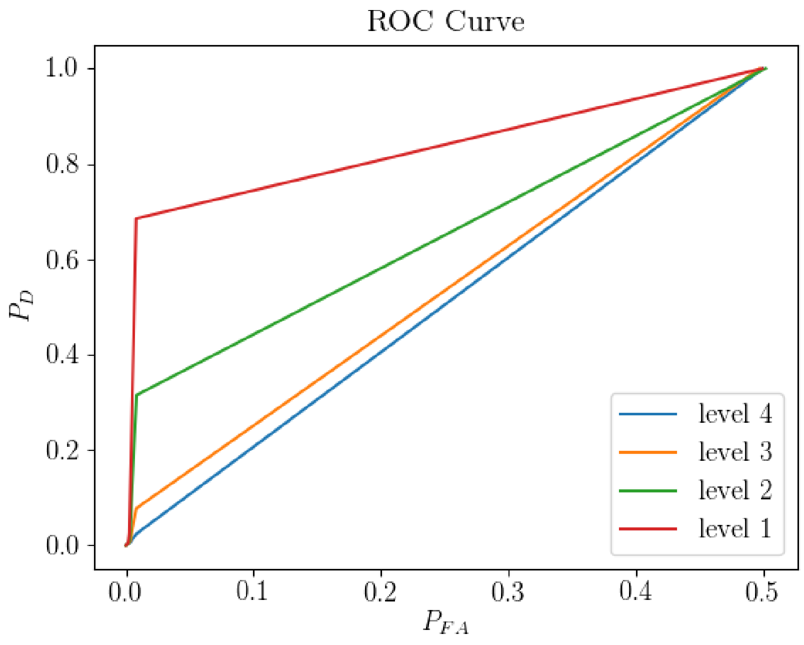


Fig. 8.3: ROC Curves for the base 0 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

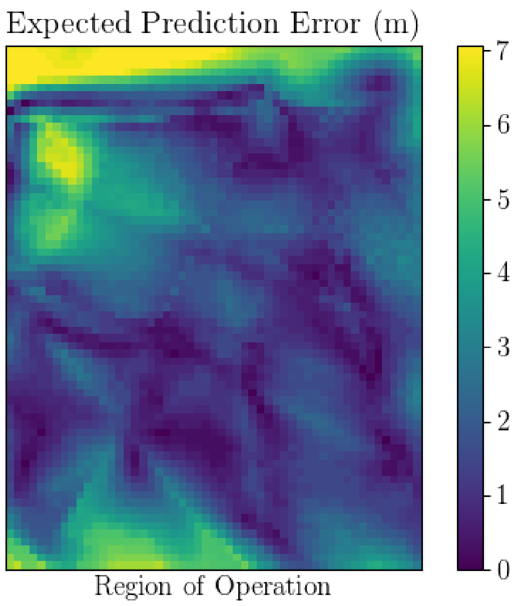


Fig. 8.4: The prediction error surface for the base 1 DNN. The region of operation is approximately 350 by 280 meters.

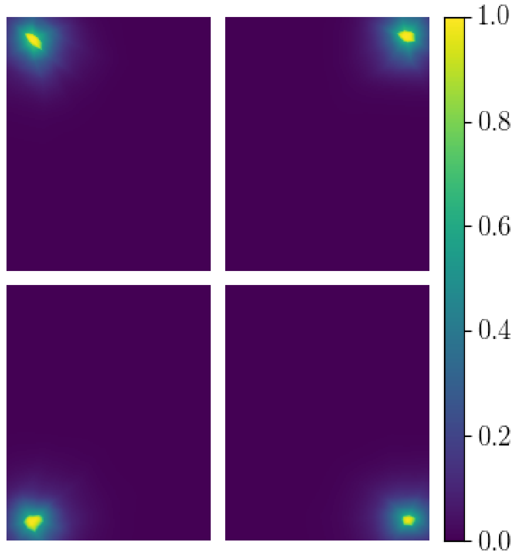


Fig. 8.5: Point Responses for the base 1 DNN. Each point response is shown over the entire region of operation.

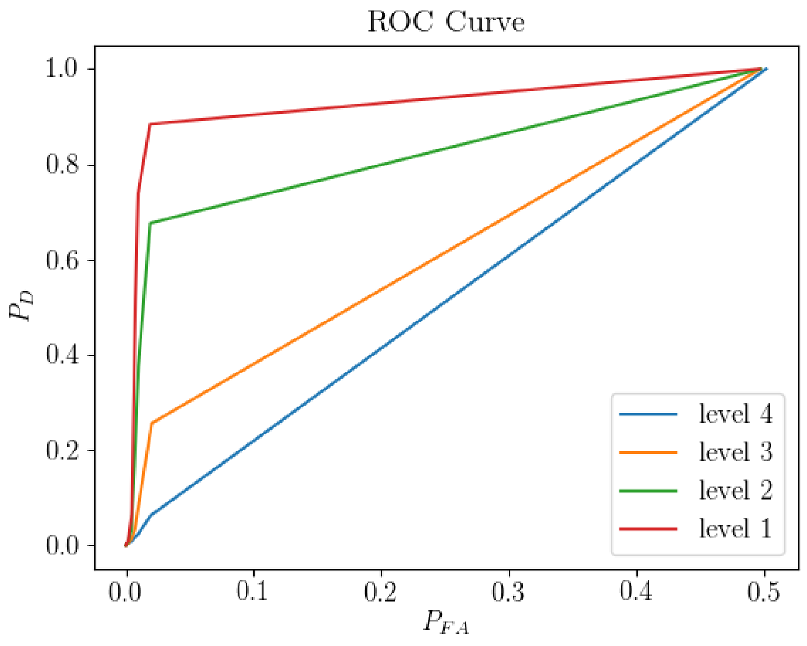


Fig. 8.6: ROC Curve for the base 1 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

the α_2 dataset. The error surface in figure 8.7 shows improvement in some areas but also decline in others. Notably, the upper left corner is improved but now the upper right corner is deteriorated.

The point responses in figure 8.8 bear some resemblance to points but they are smeared and distorted. However, they exhibit the desired decay. As desired, there is very little transition between the high and low probability areas.

The ROC curves in figure 8.9 are as expected. The curves for levels 1 and 2 are correctly shaped and show improved decisions over the previous base models. Level 3 also improved but the slight improvement for the level 4 perturbations is promising. This is the first instance where the DNN shows any ability to correctly make decisions when the perturbations are small.

These results provide more evidence supporting the use of the fast decay to make better decisions as to the validity of the input feature pairing. And like before, the prediction error surface and point responses further suggest that the relationship with the faster decay is more difficult to learn. Larger DNNs are used to further investigate this in the following sections.

8.1.4 Large 0 DNN

The error surface for the deeper model trained using the α_0 dataset is shown in figure 8.10. Much like the shallower variant, the edges are still areas of diminished performance and the upper left corner remains the area with poorest performance. There are still peaks and valleys throughout the interior and while this model may appear to have learned the relationship better, it cannot be concluded from the error surfaces alone.

The point responses in figure 8.11 maintain a similar probability decay but now appear more point like. The plateau-like peaks seen in the point responses for base 0 are not as significant. The peaks are tighter and appear more rotationally symmetric. These point responses and the prediction error surface indicate that this deeper DNN is better able to learn the slow decay relationship between the RF and video features.

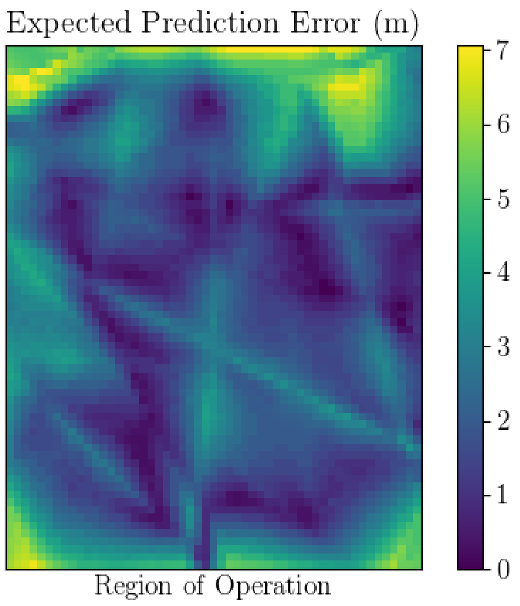


Fig. 8.7: Prediction error surface for the base 2 DNN. The region of operation is approximately 350 by 280 meters.

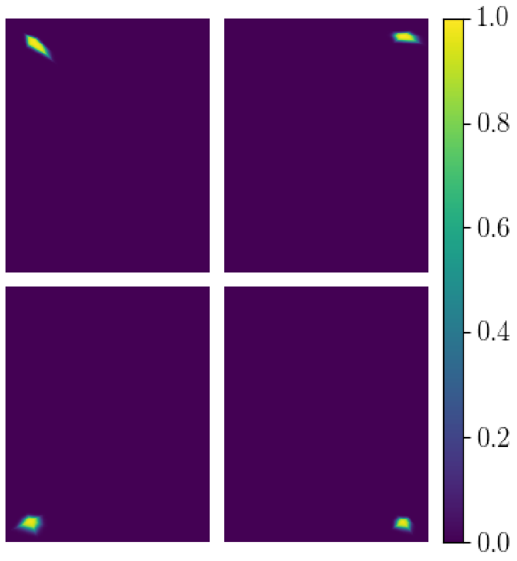


Fig. 8.8: Point Responses for the base 2 DNN. Each point response is shown over the entire region of operation.

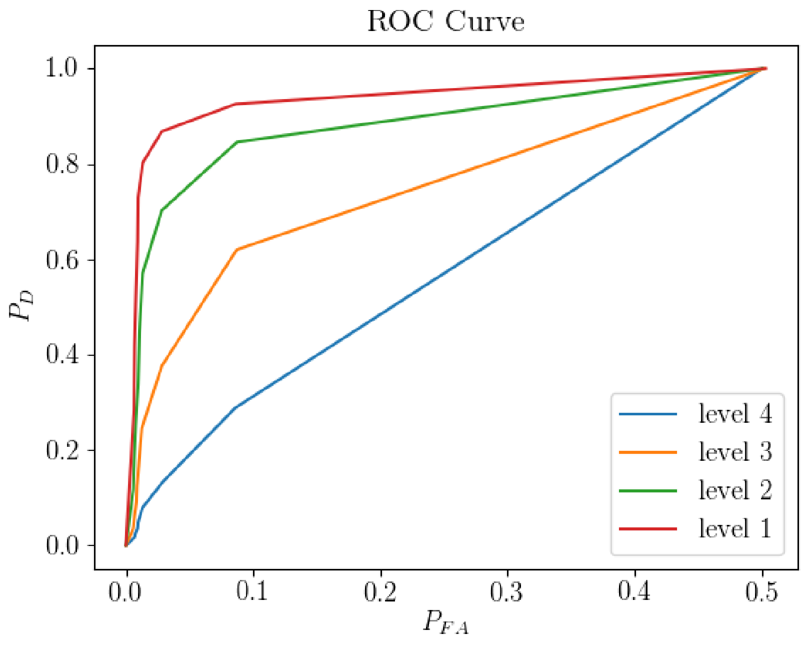


Fig. 8.9: ROC Curve for the base 2 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

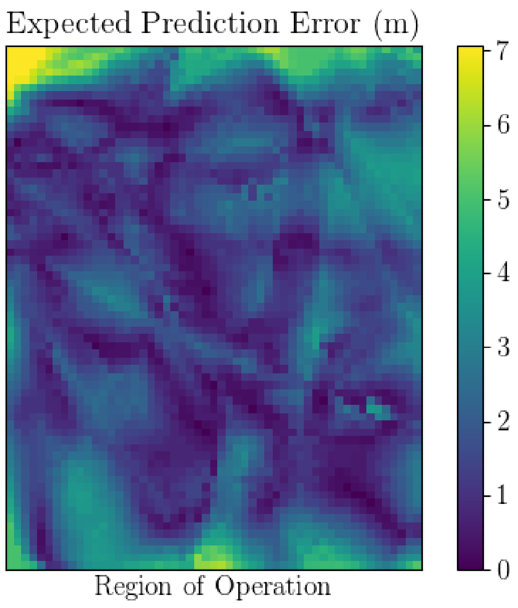


Fig. 8.10: Prediction error surface for the large 0 DNN. The region of operation is approximately 350 by 280 meters.

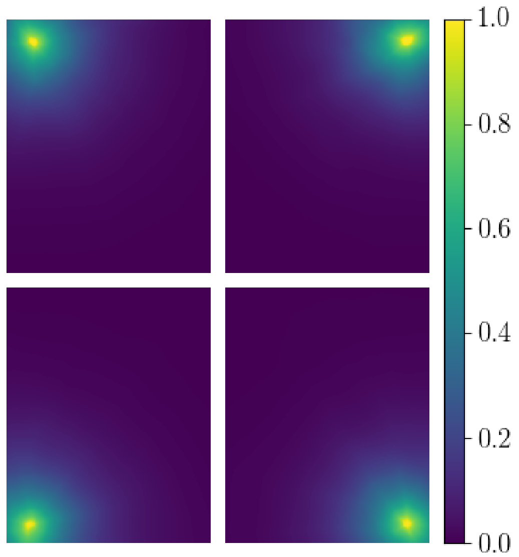


Fig. 8.11: Point Responses for the large 0 DNN. Each point response is shown over the entire region of operation.

The ROC curves in figure 8.12 show improved performance over the shallower model for levels 1, 2, and 3. However, there is no noticeable improvement for level 4. Even though the point response is a better representation of the desired response, the desired decay simply is not fast enough to handle the small position perturbations. Even if the DNN were perfect it still could not make the correct decisions.

8.1.5 Large 1 DNN

The error surface in figure 8.13 is generated using the deeper DNN trained using the α_1 dataset. The area with the worst performance is clearly along the upper border, the same as found with the base 1 DNN, but the magnitude of the expected error is now lower. A visual comparison of the prediction error surface for this DNN and the error surface from the base 1 DNN indicates the deeper model does perform better.

The point responses in figure 8.14 parallel the observations seen with the large 0 DNN. The individual responses using a deeper model are not as smeared and are generally better at

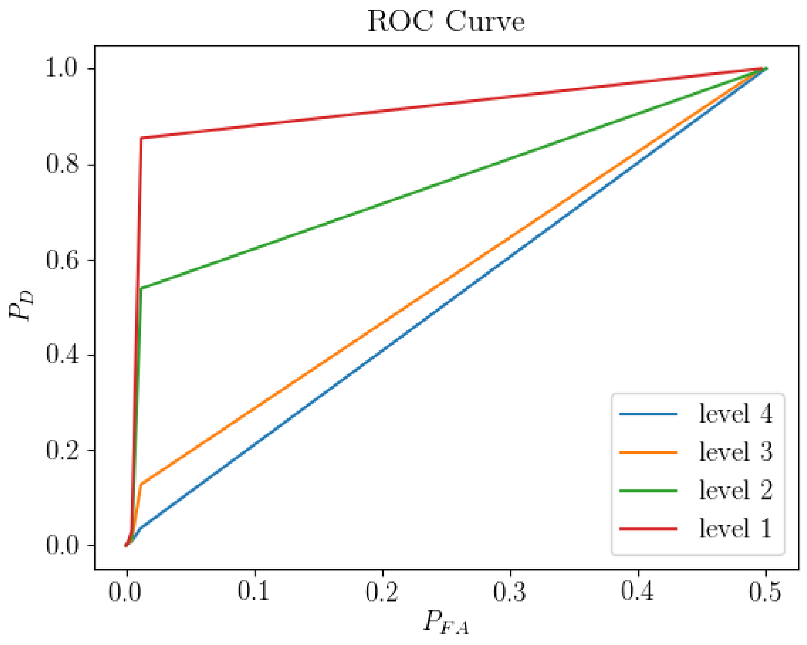


Fig. 8.12: ROC Curve for the large 0 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

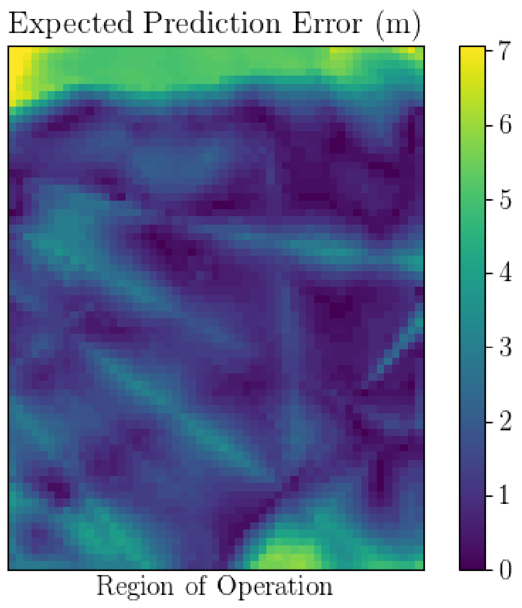


Fig. 8.13: Prediction error surface for the large 1 DNN. The region of operation is approximately 350 by 280 meters.

producing the desired point response. Additionally, the predicted probability decay occurs at the expected rate. The increased capacity of the deeper DNNs resulted in better learning of the faster decay relationship.

The ROC curves in figure 8.15 show improved decision making over the shallow model. The performance on levels 1 and 2 increased but significant improvements for levels 3 and 4 cannot be claimed. Changes for levels 3 and 4 do exist but don't suggest any real increases in performance. Again, this desired probability does not decay quickly enough to improve decision making.

8.1.6 Large 2 DNN

The error surface in figure 8.16 is for the DNN trained using the α_2 dataset. This case is the only instance where a decrease over the entire region of operation is easily visible. There are isolated pockets having an expected position error of less than one meter, however, the expected prediction error is regularly greater than five meters. This observation goes contrary to the previous two deeper models.

The point responses in figure 8.17 illustrate a clear decline in performance. The desired decay from the high to low probabilities is present in some directions but the responses are more smeared and less point shaped. This validates the results of the error surface and further suggests something more than the additional layers resulted in the increased performance seen in the two previous deeper DNNs.

Based on the observations made thus far, the ROC curves in figure 8.18 are not surprising. Even though the desired decay is quick enough to make correct decisions on higher levels, the DNN's learned probability decay is inadequate. When this DNN's ROC curves are compared to the curves in figure 8.9, it is obvious that shallower DNN is much better at making the decisions than the deeper DNN.

The DNNs discussed thus far represent two architectures and three datasets. The DNNs trained using the dataset with the fastest probability decay are generally better at making decisions when the separation between the true position and the test position is minimal. However, the fast probability decay has proven to be more difficult to learn than

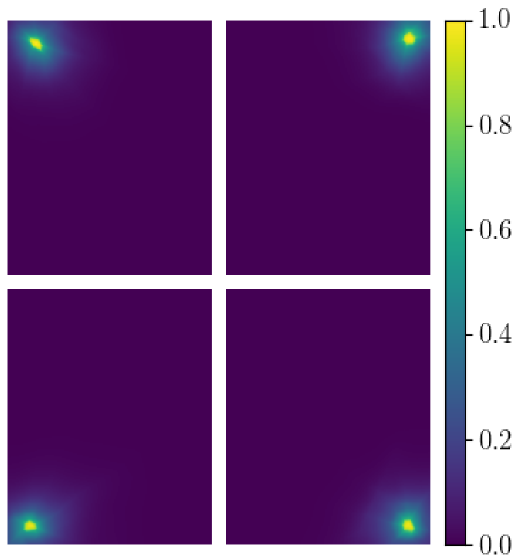


Fig. 8.14: Point Responses for the large 1 DNN. Each point response is shown over the entire region of operation.

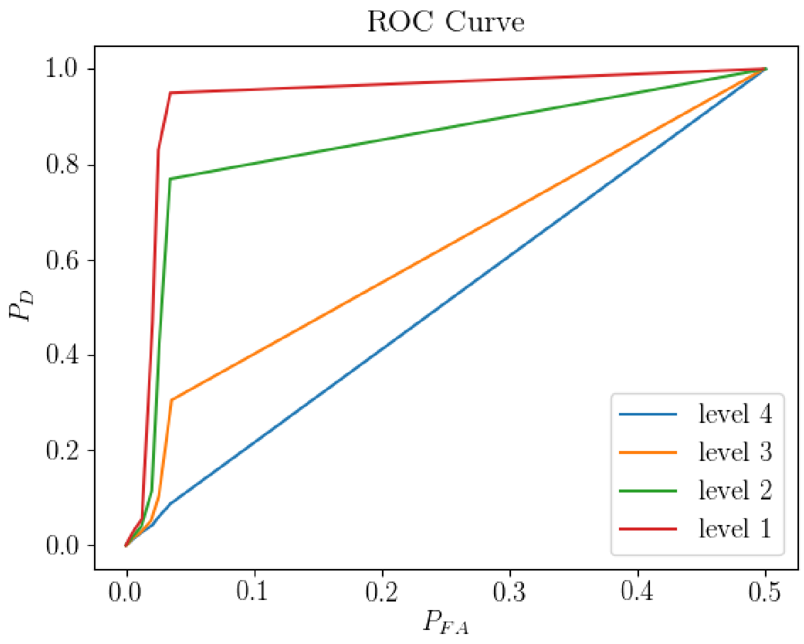


Fig. 8.15: ROC Curve for the large 1 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

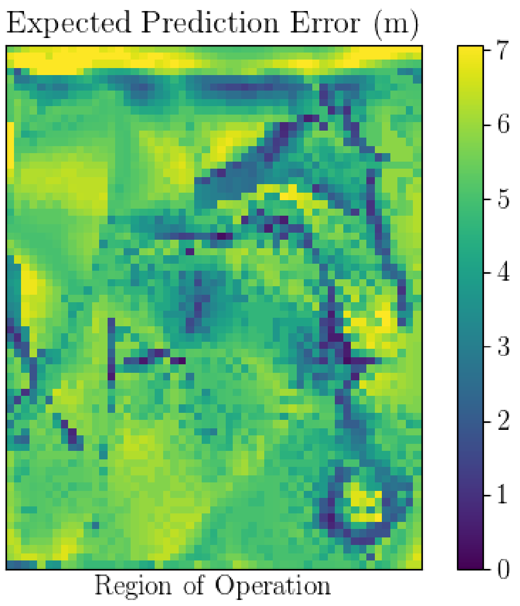


Fig. 8.16: Prediction error surface for the large 2 DNN. The region of operation is approximately 350 by 280 meters.

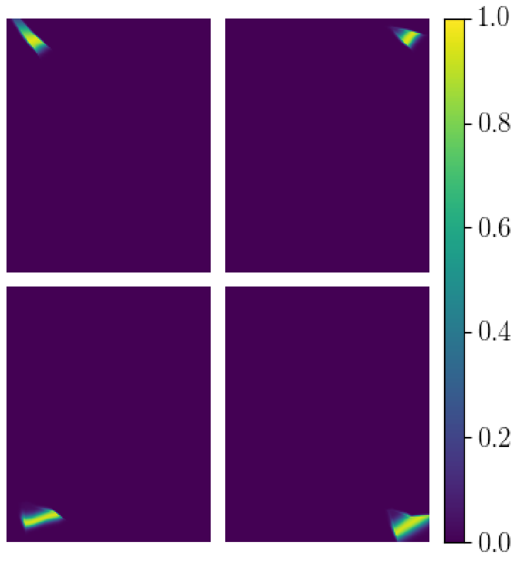


Fig. 8.17: Point Responses for the large 2 DNN. Each point response is shown over the entire region of operation.

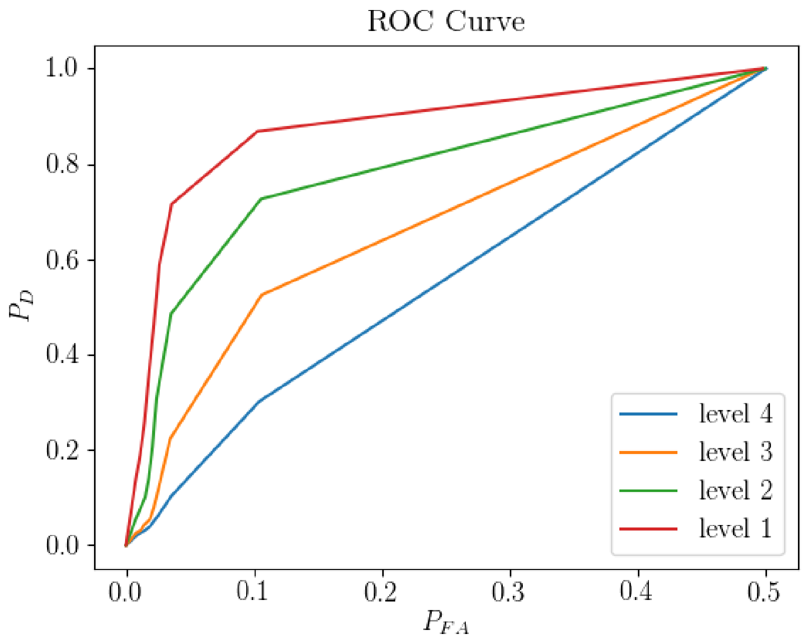


Fig. 8.18: ROC Curve for the large 2 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

the slow decays. The increased network capacity helped the deeper DNNs learn the RF and video feature relationships in datasets α_0 and α_1 but resulted in the opposite outcome using dataset α_2.

The next two DNNs are intended to provide a better understanding of how to better train the DNN to represent the fastest decay. The three layer architecture used in the base DNNs is used, along with the decay rate used in dataset α_1. In addition to the number of samples in the training dataset, the ratio of valid and invalid pairs is adjusted. As a reminder, the six models seen thus far have all had 25 invalid pairs for every valid pair.

8.1.7 N_2 DNN

The error surface in figure 8.19 is the result of a DNN trained using the N_2 dataset with two invalid pairs for every valid pair. The architecture of this DNN is identical to the base 1 DNN. The challenging areas for the base 1 DNN seen in figure 8.4 remain but this DNN's prediction error surface contains more pronounced peaks throughout the interior. It is generally believed that having more samples in the training dataset will produce better results, which isn't necessarily seen here. The decrease in performance from the base 1 DNN suggests that there are just not enough invalid pairs in the dataset to help the DNN learn the desired probability decay.

The point responses in figure 8.20 further confirms the need for more invalid pairs. Even with more samples to train on, this DNN was unable to learn the response as well as the base 1 DNN did. The decay is present, but the high probability areas are more smeared and appear less like a point.

Finally, the ROC curves in figure 8.21 are perhaps the most conclusive evidence of decreased performance. The decision making performance deteriorated for every level, except for level 4 which was already at its lowest performance.

It is evident from these results that having more invalid pairs for every valid pairs is more important than simply having more samples. Not only did the decision making performance drop, the prediction error surface and point responses also indicate a diminished ability to learn the relationship between RF and video features.

8.1.8 N_{50} DNN

The N_{50} DNN was trained using the N_{50} dataset and its error surface is given in figure 8.22. In this dataset there are 50 invalid pairs for every valid pair. Like the N_2 DNN, the architecture and desired response is the same as seen in the base 1 DNN. Comparing the error surfaces suggests the increase of invalid pairs improved performance. The upper left corner remains the area with poorest performance but the interior saw a general increase in performance overall. There are still peaks and valleys, but they are not as prominent and the expected position error is less than 3 meters in most areas.

The point responses in figure 8.23 are also indicative of improved performance. The anticipated decay is present and more significantly, the individual responses are better representations of the desired point response. While not perfect points, they tend to be less smeared than seen in figure 8.5.

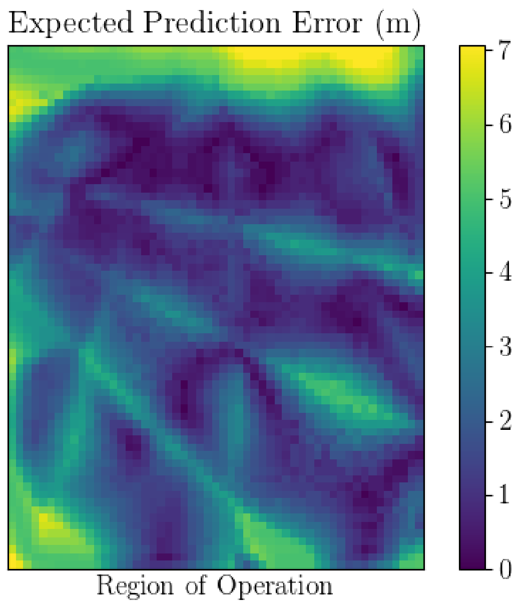


Fig. 8.19: Prediction error surface for the N_2 DNN. The region of operation is approximately 350 by 280 meters.

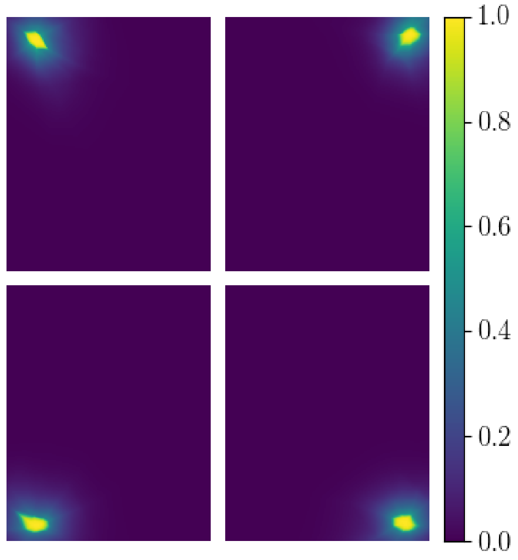


Fig. 8.20: Point Responses for the N_2 DNN. Each point response is shown over the entire region of operation.

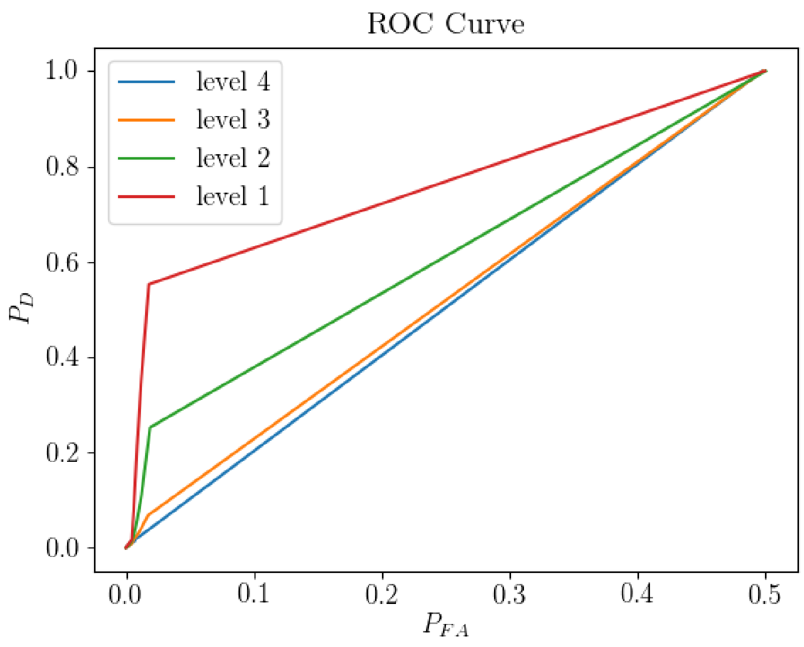


Fig. 8.21: ROC Curve for the N_2 DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

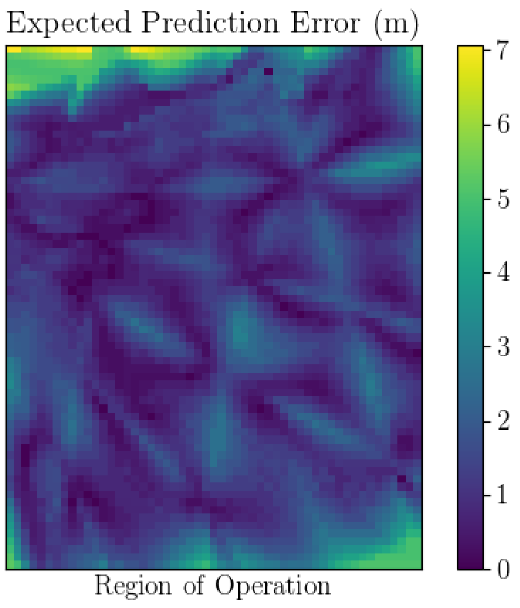


Fig. 8.22: Prediction error surface for the N_{50} DNN. The region of operation is approximately 350 by 280 meters.

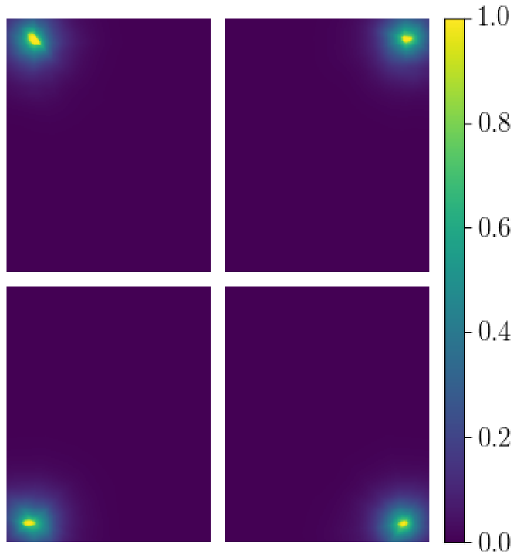


Fig. 8.23: Point Responses for the N_{50} DNN. Each point response is shown over the entire region of operation.

The ROC curves in figure 8.24 further illustrate the improved performance. The decision making for levels 1, 2, and 3 all saw noticeable improvement and even level 4 saw a small increase when compared to the curves in figure 8.6.

The results from including more invalid pairs appear to suggest increasing the ration of invalid and valid pairs will result in better performing models. However, the N_{50} dataset had over one million more training samples than the α_1 dataset and roughly 750,000 more samples than the N_2 dataset because the number of valid pairs remained unchanged. The increased dataset size is likely more responsible for the improved performance than this valid/invalid ratio.

When these results are combined with the results from the N_2 DNN, the conclusion is made that as long as there are enough invalid pairs for every valid pair, a larger dataset is in line with the generally held belief that having more samples is better. The next model presented increases the DNN's capacity further with the addition of two hidden layers but also uses a larger dataset.

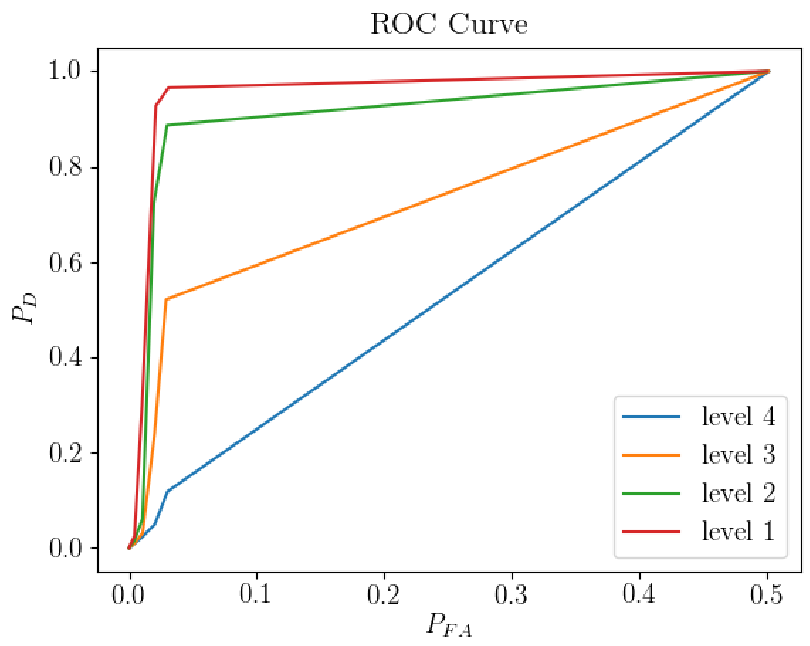


Fig. 8.24: ROC Curve for the N_{50} DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

8.1.9 Final DNN

The final DNN was inspired by the results from previous DNNs. The dataset used to train this deeper model implemented the fastest probability decay and has over 1.5 million samples and similar to the three main datasets, there are 25 invalid pairs for every valid pair. The error surface in figure 8.25 clearly shows the success of this model. The expected prediction error less than two meters nearly everywhere in the region of operation with the exception of a few trouble areas.

The point responses in figure 8.26 are expected to have the quickest probability decay and it is visible. In addition to the decay being present, each individual response is a much better approximation to the desired point responses than for any DNN in this chapter.

The ROC curves in figure 8.27 clearly demonstrate this DNN's improved performance. Nearly perfect decision making occurs for levels 1, 2, and 3. More impressively, the level 4 decisions are clearly being made better than any of the other DNNs presented. In fact, this DNN is able to make correct decisions with level 4 perturbations better than many other

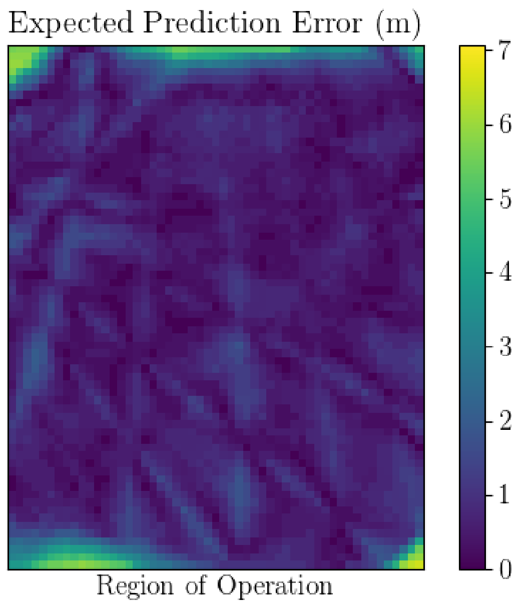


Fig. 8.25: Prediction error surface for the Final DNN. The region of operation is approximately 350 by 280 meters.

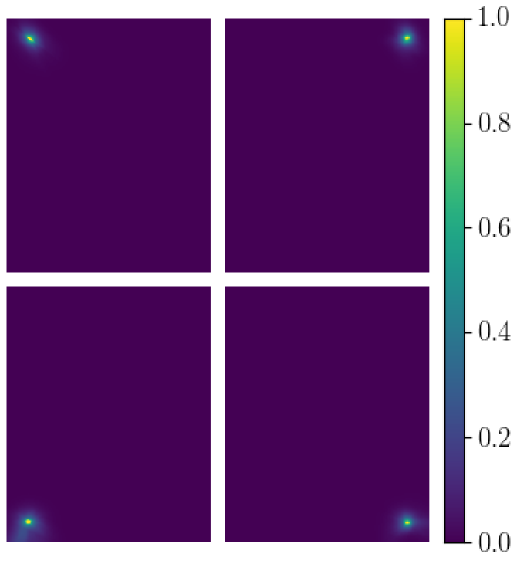


Fig. 8.26: Point Responses for the Final DNN. Each point response is shown over the entire region of operation.

DNNs having perturbations with levels 1, 2, and 3.

The results from this final DNN clearly demonstrate that a DNN is capable of learning the most difficult relationship between the RF and video features. As observed previously a deeper model worked better, provided it had enough data to properly train. One last observation cannot be made using any of the DNNs presented and is instead made using one additional DNN in the next section.

8.1.10 Alternate Geometry DNN

The alternate geometry DNN was trained used a different antenna array configuration but the tests in this section were performed assuming the original configuration. The error surface in figure 8.28 shows abysmal results. The expected error is clipped at seven meters, though the actual error is likely greater.

Two of the point responses in figure 8.29 appear to have the right shape while the other two are virtually nonexistent. The upper and lower right point responses should be in the same location seen in the point responses for other models but this just isn't the case.

The ROC curves in figure 8.30 do not come as a surprise. This clearly demonstrates the DNN's predicted probabilities cannot be trusted or used when making decisions.

This last test demonstrates the DNN's critical dependence upon maintaining a consistent array geometry between throughout deployment and training. Even though the DNNs are learning the response defined by equation 7.1 during training, they are not learning a general association. The DNNs can only learn the association for the given configuration.

8.2 Conclusion

The emitter and RF emission association problem was introduced in chapter 6. During the discussion of the proposed solution in chapter 7, the association problem was modified to focus on learning the the relationship between video and RF features prior to the presentation of the datasets and different DNNs.

In this chapter, prediction error surfaces and point responses for the trained models were used to evaluate how well the DNN learned the RF and video features relationship.

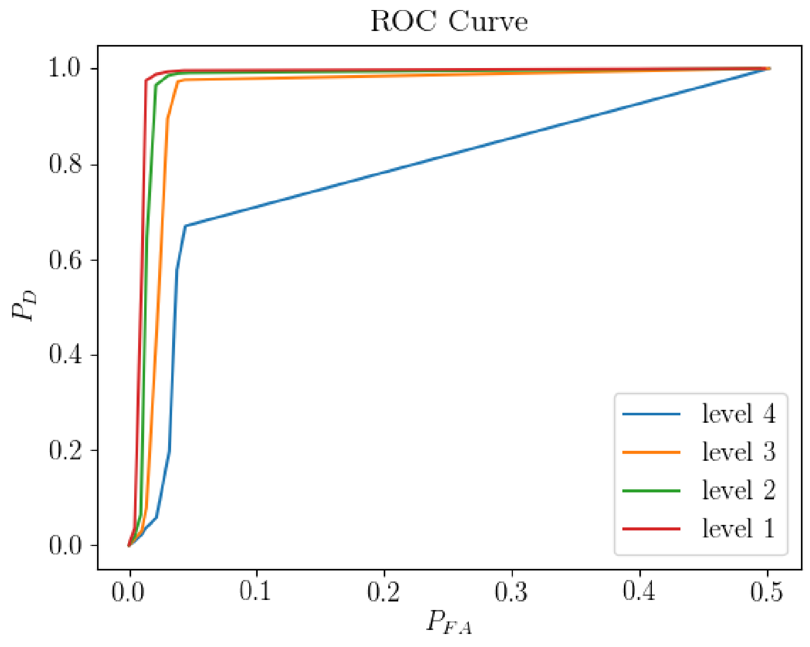


Fig. 8.27: ROC Curve for the Deep DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

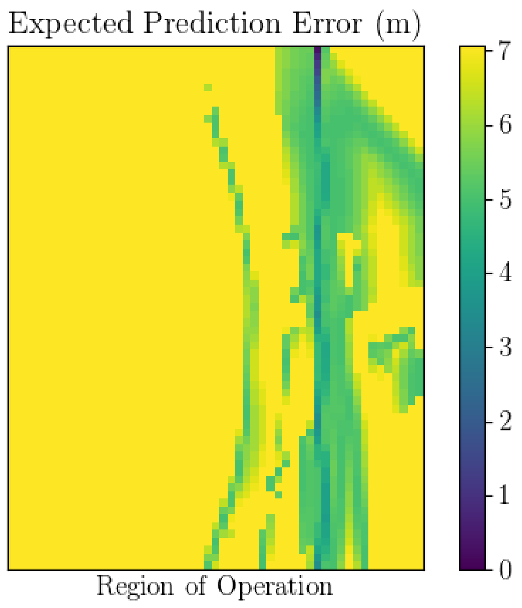


Fig. 8.28: Prediction error surface for the Alt. Geom. DNN. The region of operation is approximately 350 by 280 meters.

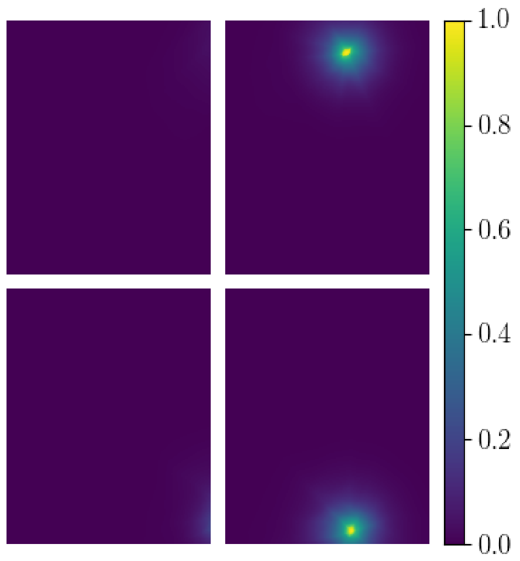


Fig. 8.29: Point Responses for the Alt. Geom. DNN. Each point response is shown over the entire region of operation.

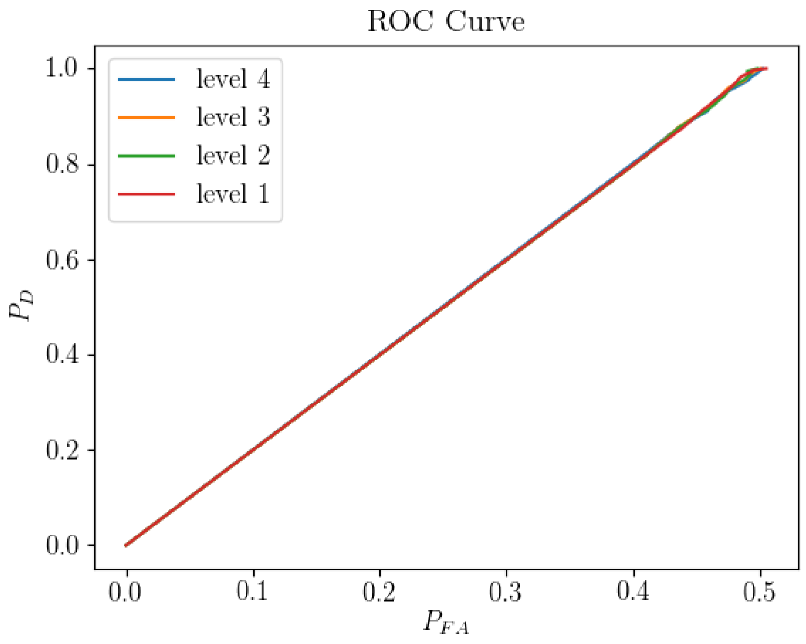


Fig. 8.30: ROC Curve for the Alt. Geom. DNN. The level indicates the amount of perturbation used. Level 1: 7-9 meters, Level 2: 5-7 meters, Level 3: 3-5 meters, and Level 4: 1-3 meters.

ROC curves were used to evaluate the invalid/valid decision making using the trained DNNs. The final DNN demonstrated that the DNNs can learn the complex relationship if enough samples are available.

Future work includes incorporating the antenna positions into the input vector in an attempt to remove the critical antenna geometry dependency and allow for a more generalized DNN. Additionally, work can be done to remove the optical and RF preprocessing stages, or at least the level of preprocessing performed, to achieve a more purely DNN based technique.

CHAPTER 9

General Conclusion

In this dissertation, two specific instances of an association learning problem were addressed using deep neural networks (DNN). The image overlap detection problem was used as a specific instance of a unimodal problem and the emitter and RF emission problem was used as an instance of a multimodal problem.

The main contributions of this research are to:

1. Further validate the use of DNNs for multiple input processing outside of speech processing and visual object detection

2. Demonstrate a DNN's output format's effect on learning an association between two inputs

3. Expand existing DNN interpretation techniques to fit different problems

In the evaluation of the various DNNs using both problems, it was shown that using DNNs is a valid approach to multiple input processing problems apart from many of the current trending problems. For both the image overlap detection problem and the emitter and RF emission association problem, evaluation of the trained DNNs supported the claim that the DNN was learning an association between the input samples. Furthermore, the visualizations implemented for the image overlap detection demonstrated a successful adaptation of CNN visualizations.

In general, the future work for both of these problems involves removing some simplifying assumptions. The emitter and RF emission problem addressed in this dissertation assumed line of sight of propagation in an isotropic propagation environment and a stationary, fixed antenna array geometry. Future work may consider propagation environments where multi-path is present and also allow for nonstationary antenna configurations. The image overlap detection problem was limited to translational overlap between the reference and test images. Future work may incorporate rotational overlap, scaling differences, and perspective changes between the reference and test images.

REFERENCES

[1] W. S. McCulloch and W. Pitts, "Neurocomputing: Foundations of research," J. A. Anderson and E. Rosenfeld, Eds. Cambridge, MA, USA: MIT Press, 1988, ch. A Logical Calculus of the Ideas Immanent in Nervous Activity, pp. 15–27. [Online]. Available: http://dl.acm.org/citation.cfm?id=65669.104377

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[3] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[4] A. Graves, A. Mohamed, and G. E. Hinton, "Speech recognition with deep recurrent neural networks," *CoRR*, vol. abs/1303.5778, 2013. [Online]. Available: http://arxiv.org/abs/1303.5778

[5] Y. Goldberg, "A primer on neural network models for natural language processing," *CoRR*, vol. abs/1510.00726, 2015. [Online]. Available: http://arxiv.org/abs/1510.00726

[6] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016. [Online]. Available: http://arxiv.org/abs/1605.02688

[7] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22Nd ACM International Conference on Multimedia*, ser. MM '14. New York, NY, USA: ACM, 2014, pp. 675–678. [Online]. Available: http://doi.acm.org/10.1145/2647868.2654889

[9] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning."

[10] F. Chollet *et al.*, "Keras," https://github.com/keras-team/keras, 2015.

[11] A. Damien *et al.*, "Tflearn," https://github.com/tflearn/tflearn, 2016.

[12] S. Dieleman, J. Schlter, C. Raffel, E. Olson, S. K. Snderby, D. Nouri *et al.*, "Lasagne: First release." Aug. 2015. [Online]. Available: http://dx.doi.org/10.5281/zenodo.27878

[13] K. Sohn, W. Shang, and H. Lee, "Improved multimodal deep learning with variation of information," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2141–2149. [Online]. Available: http://papers.nips.cc/paper/5279-improved-multimodal-deep-learning-with-variation-of-information.pdf

[14] Y. Zheng, "Methodologies for cross-domain data fusion: An overview," *IEEE Transactions on Big Data*, vol. 1, no. 1, pp. 16–34, March 2015.

[15] L. Gmez-Chova, D. Tuia, G. Moser, and G. Camps-Valls, "Multimodal classification of remote sensing images: A review and future directions," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1560–1584, Sept 2015.

[16] M. Brown and D. G. Lowe, "Automatic panoramic image stitching using invariant features," *Int. J. Comput. Vision*, vol. 74, no. 1, pp. 59–73, Aug. 2007. [Online]. Available: http://dx.doi.org/10.1007/s11263-006-0002-3

[17] Q. V. Le, A. Karpenko, J. Ngiam, and A. Y. Ng, "Ica with reconstruction cost for efficient overcomplete feature learning," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 1017–1025. [Online]. Available: http://papers.nips.cc/paper/4467-ica-with-reconstruction-cost-for-efficient-overcomplete-feature-learning.pdf

[18] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 3642–3649.

[19] H. Schulz and S. Behnke, "Object-class segmentation using deep convolutional neural networks," 01 2012.

[20] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *CoRR*, vol. abs/1411.4038, 2014. [Online]. Available: http://arxiv.org/abs/1411.4038

[21] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," *CoRR*, vol. abs/1505.04366, 2015. [Online]. Available: http://arxiv.org/abs/1505.04366

[22] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 2553–2561. [Online]. Available: http://papers.nips.cc/paper/5207-deep-neural-networks-for-object-detection.pdf

[23] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3320–3328. [Online]. Available: http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf

[24] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, “Visualizing higher-layer features of a deep network,” University of Montreal, Tech. Rep. 1341, Jun. 2009, also presented at the ICML 2009 Workshop on Learning Feature Hierarchies, Montréal, Canada.

[25] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” *CoRR*, vol. abs/1311.2901, 2013. [Online]. Available: http://arxiv.org/abs/1311.2901

[26] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” *CoRR*, vol. abs/1312.6034, 2013. [Online]. Available: http://arxiv.org/abs/1312.6034

[27] L. A. Gatys, A. S. Ecker, and M. Bethge, “A neural algorithm of artistic style,” *CoRR*, vol. abs/1508.06576, 2015. [Online]. Available: http://arxiv.org/abs/1508.06576

[28] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, “Signature verification using a ”siamese” time delay neural network,” in *Advances in Neural Information Processing Systems 6*, J. D. Cowan, G. Tesauro, and J. Alspector, Eds. Morgan-Kaufmann, 1994, pp. 737–744. [Online]. Available: http://papers.nips.cc/paper/769-signature-verification-using-a-siamese-time-delay-neural-network.pdf

[29] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, “Matchnet: Unifying feature and metric learning for patch-based matching,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3279–3286.

[30] I. Melekhov, J. Kannala, and E. Rahtu, “Siamese network features for image matching,” in *2016 23rd International Conference on Pattern Recognition (ICPR)*, Dec 2016, pp. 378–383.

[31] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, “Discriminative learning of deep convolutional feature point descriptors,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 118–126.

[32] C. Bailer, K. Varanasi, and D. Stricker, “Cnn-based patch matching for optical flow with thresholded hinge embedding loss,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 2710–2719.

[33] S. Zagoruyko and N. Komodakis, “Learning to compare image patches via convolutional neural networks,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 4353–4361.

[34] J. Zbontar and Y. LeCun, “Stereo matching by training a convolutional neural network to compare image patches,” *CoRR*, vol. abs/1510.05970, 2015. [Online]. Available: http://arxiv.org/abs/1510.05970

[35] D. Chung, K. Tahboub, and E. J. Delp, “A two stream siamese convolutional neural network for person re-identification,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 1992–2000.

[36] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, “PatchMatch: A randomized correspondence algorithm for structural image editing,” *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 28, no. 3, Aug. 2009.

[37] C. Xiao, M. Liu, N. Yongwei, and Z. Dong, “Fast exact nearest patch matching for patch-based image editing and processing,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 8, pp. 1122–1134, Aug 2011.

[38] L. Ma, Z. Chen, L. Xu, and Y. Yan, “Multimodal deep learning for solar radio burst classification,” *Pattern Recognition*, vol. 61, no. Supplement C, pp. 573 – 582, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320316300590

[39] W. Wang, R. Arora, K. Livescu, and J. Bilmes, “On deep multi-view representation learning,” in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 1083–1092. [Online]. Available: http://proceedings.mlr.press/v37/wangb15.html

[40] N. Audebert, B. L. Saux, and S. Lefèvre, “Semantic segmentation of earth observation data using multimodal and multi-scale deep networks,” *CoRR*, vol. abs/1609.06846, 2016. [Online]. Available: http://arxiv.org/abs/1609.06846

[41] R. Caruana, “Multitask learning,” *Machine Learning*, vol. 28, no. 1, pp. 41–75, Jul 1997. [Online]. Available: https://doi.org/10.1023/A:1007379606734

[42] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: common objects in context,” *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[43] M. Brown, G. Hua, and S. Winder, “Discriminative learning of local image descriptors,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 43–57, Jan 2011.

[44] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[45] A. Tahat, G. Kaddoum, S. Yousefi, S. Valaee, and F. Gagnon, “A look at the recent wireless positioning techniques with a focus on algorithms for moving receivers,” *IEEE Access*, vol. 4, pp. 6652–6680, 2016.

[46] N. J. Thomas, D. G. M. Cruickshank, and D. I. Laurenson, “A robust location estimator architecture with biased kalman filtering of toa data for wireless systems,” in *2000 IEEE Sixth International Symposium on Spread Spectrum Techniques and Applications. ISSTA 2000. Proceedings (Cat. No.00TH8536)*, vol. 1, Sep 2000, pp. 296–300 vol.1.

[47] H. Jiang, J. Xu, and Z. Li, “Nlos mitigation method for tdoa measurement,” in *2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Oct 2010, pp. 196–199.

[48] G. Bao, K. Pahlavan, and L. Mi, “Hybrid localization of microrobotic endoscopic capsule inside small intestine by data fusion of vision and rf sensors,” *IEEE Sensors Journal*, vol. 15, no. 5, pp. 2669–2678, May 2015.

[49] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Ng, “Multimodal deep learning,” pp. 689–696, June 2011.

[50] Y. Mroueh, E. Marcheret, and V. Goel, “Deep multimodal learning for audio-visual speech recognition,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 2130–2134.

APPENDICES

APPENDIX A

Image Overlap Detection Additional Figures

A.1 Additional Results

The results for models not addressed during the discussion in chapter 5 are presented here. For a detailed explanation of the stochastic images space search, refer to section 5.1.3. For further discussion of the kernels and activations, see section 5.3.

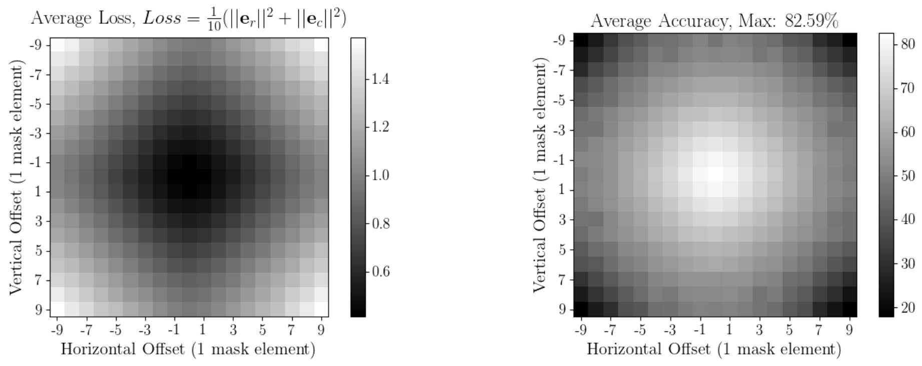


Fig. A.1: Row-Column encoding, model 1: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

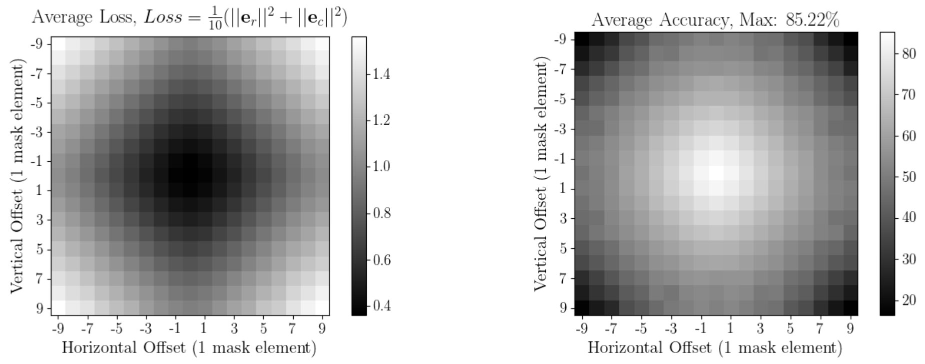


Fig. A.2: Row-Column encoding, model 3: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

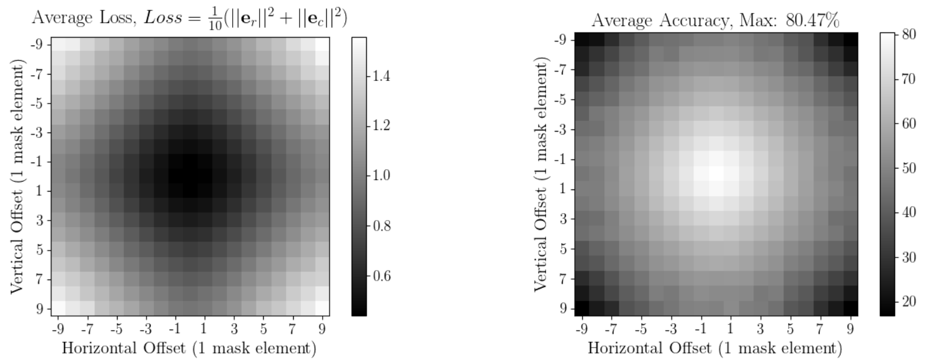


Fig. A.3: Row-Column encoding, model 4: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

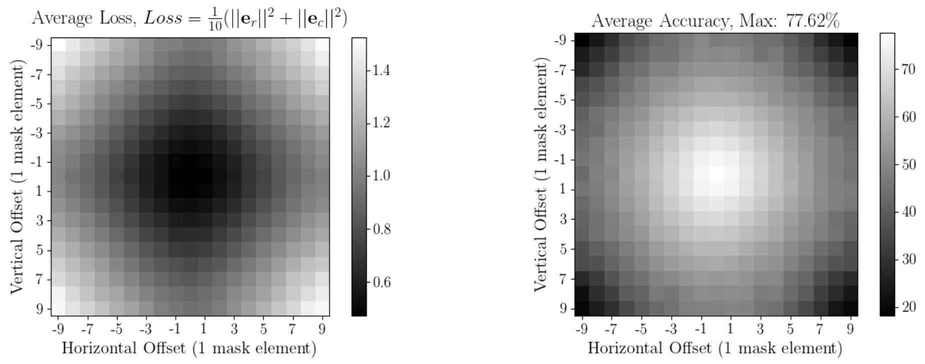


Fig. A.4: Row-Column encoding, model 5: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

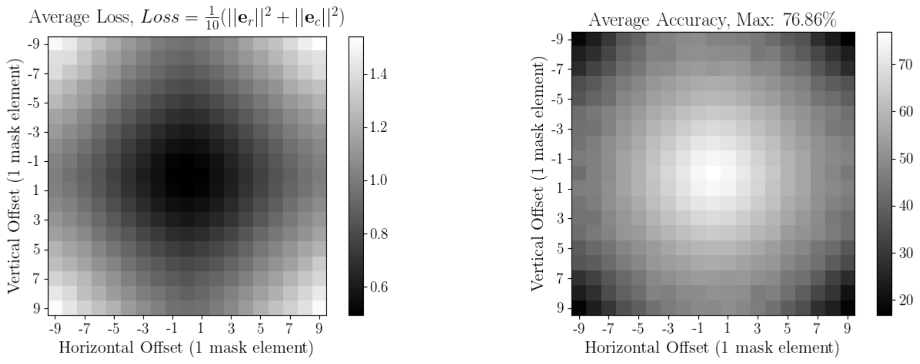


Fig. A.5: Row-Column encoding, model 6: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

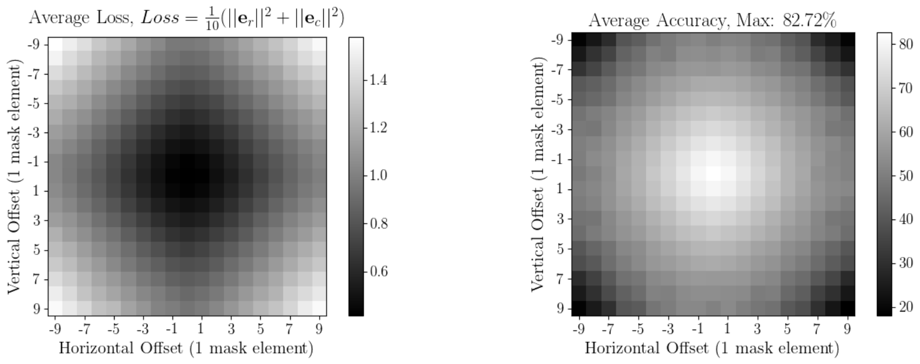


Fig. A.6: Row-Column encoding, model 7: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

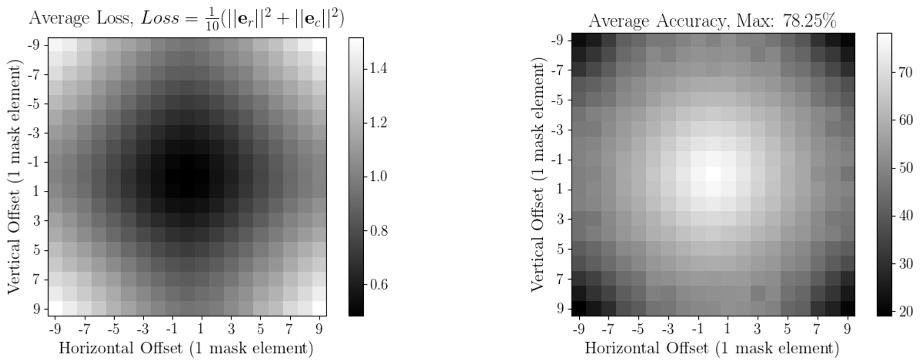


Fig. A.7: Row-Column encoding, model 8: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

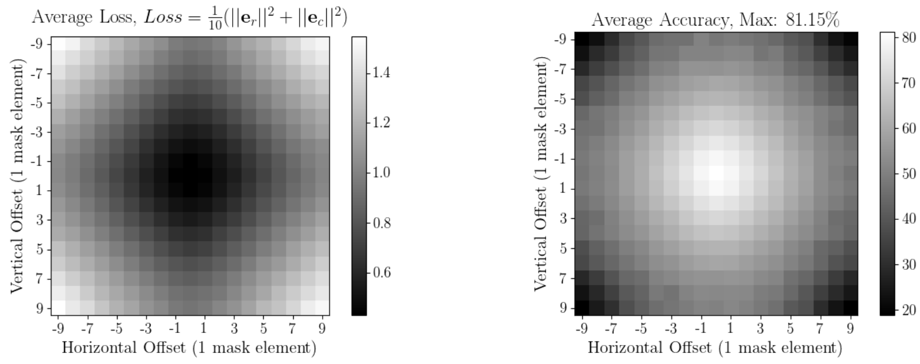


Fig. A.8: Row-Column encoding, model 9: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

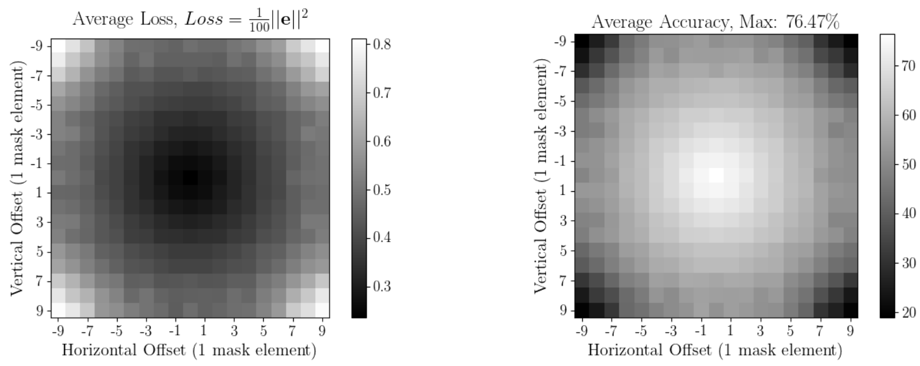


Fig. A.9: Direct encoding, MSE: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

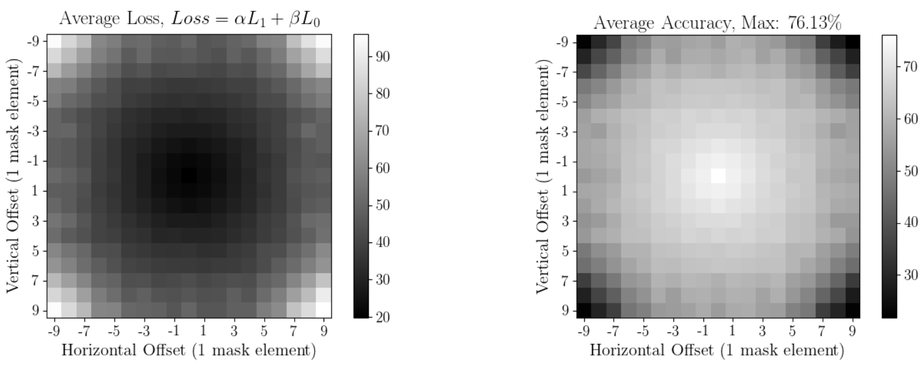


Fig. A.10: Direct encoding, emphasis on non-overlapping region: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

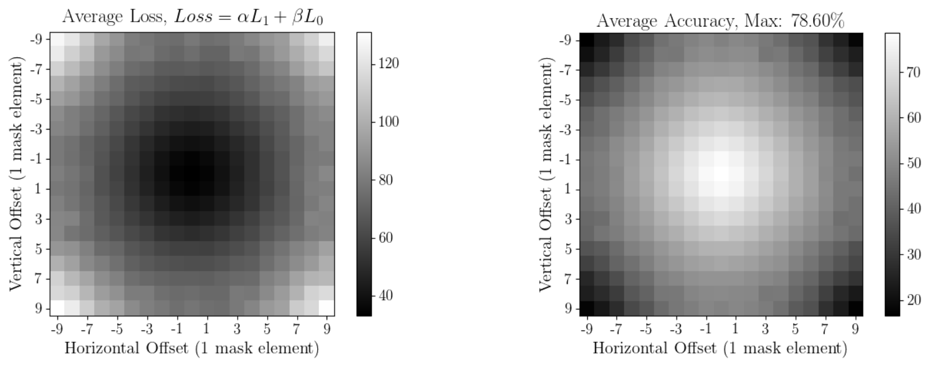


Fig. A.11: Direct encoding, emphasis on overlapping region: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

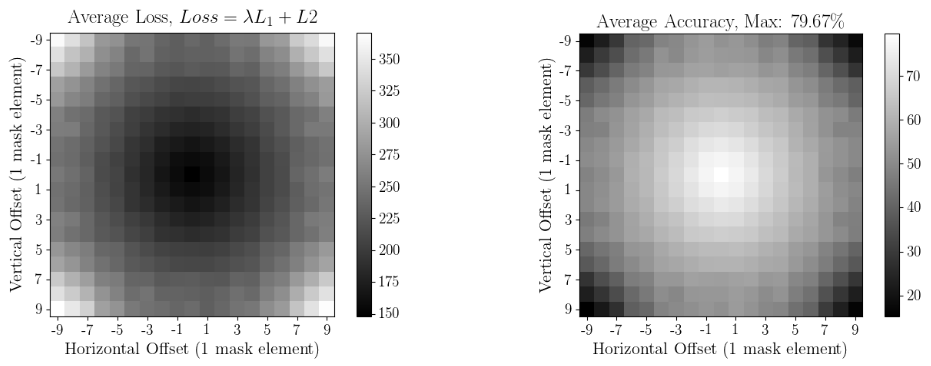


Fig. A.12: Direct encoding, emphasis on prediction overlapping region shape, $\lambda = 5$: The average loss (left) and average prediction accuracy (right) results of the stochastic test image space search.

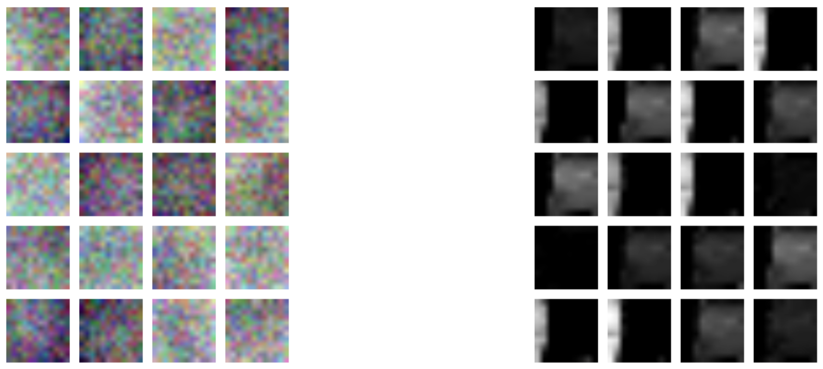


Fig. A.13: Row-Column encoding, model 1: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

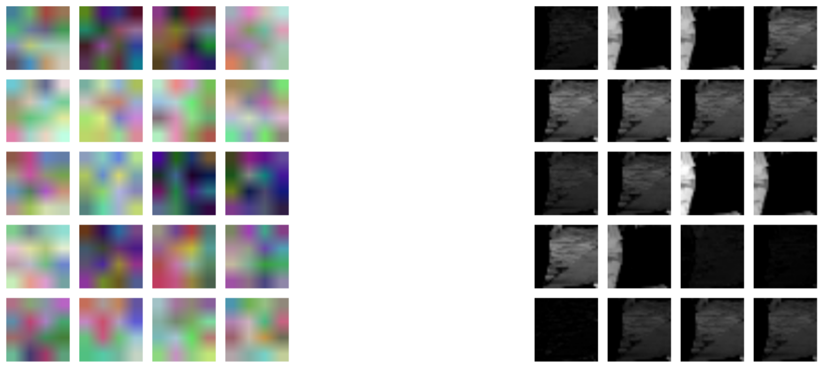


Fig. A.14: Row-Column encoding, model 3: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

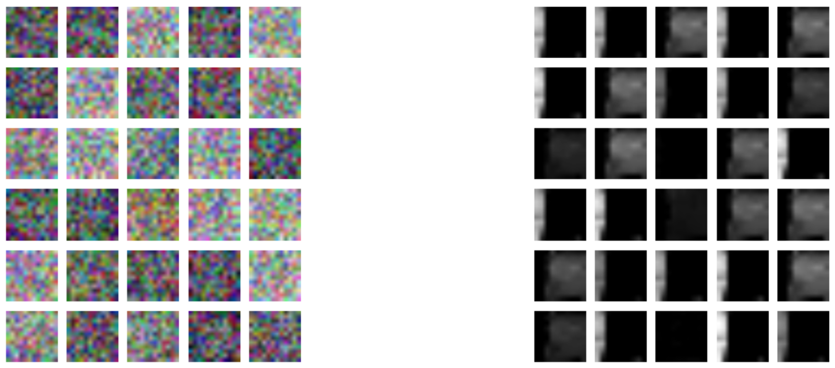


Fig. A.15: Row-Column encoding, model 4: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

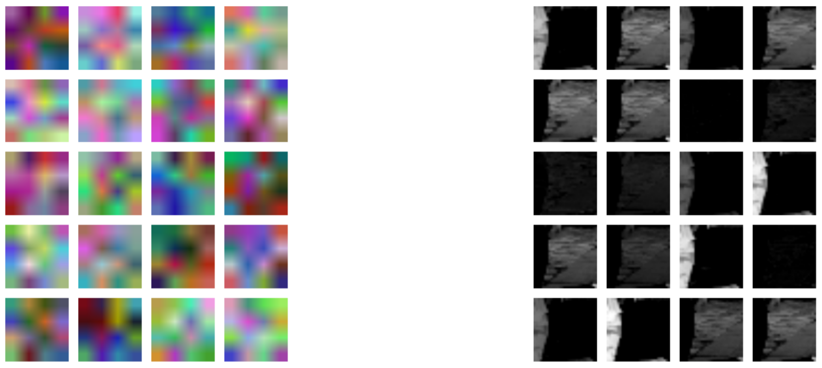


Fig. A.16: Row-Column encoding, model 5: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

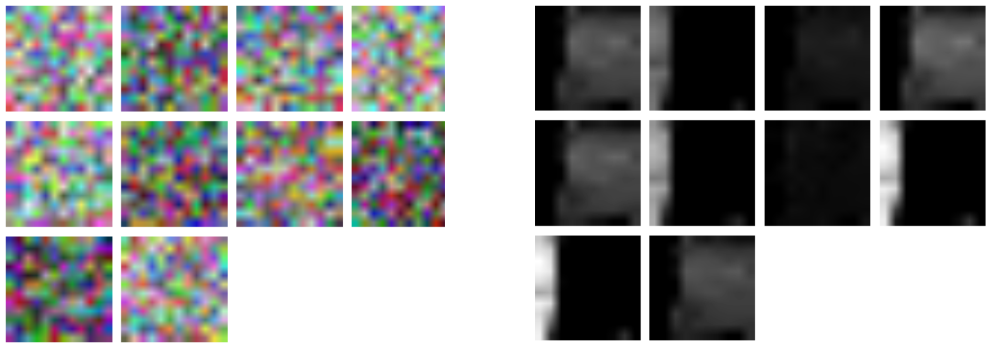


Fig. A.17: Row-Column encoding, model 6: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

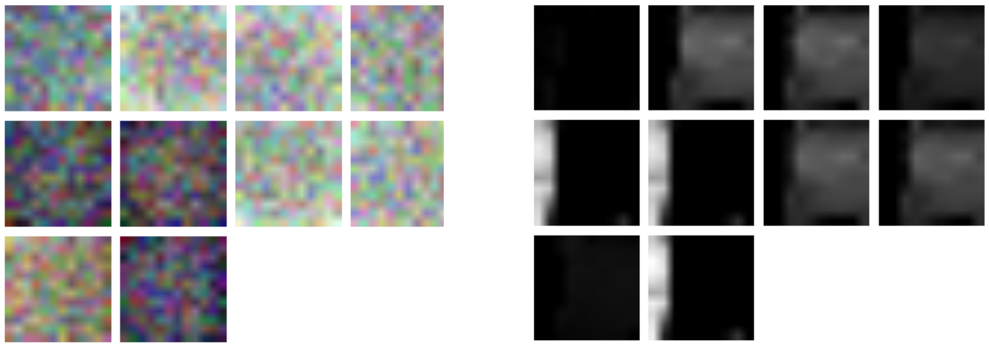


Fig. A.18: Row-Column encoding, model 7: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

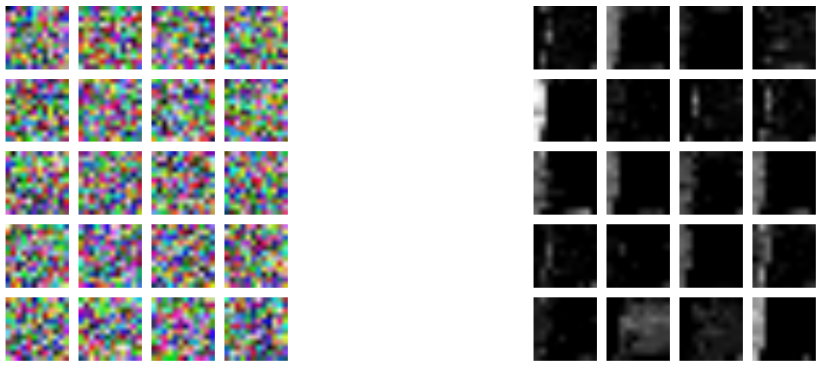


Fig. A.19: Row-Column encoding, model 8: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

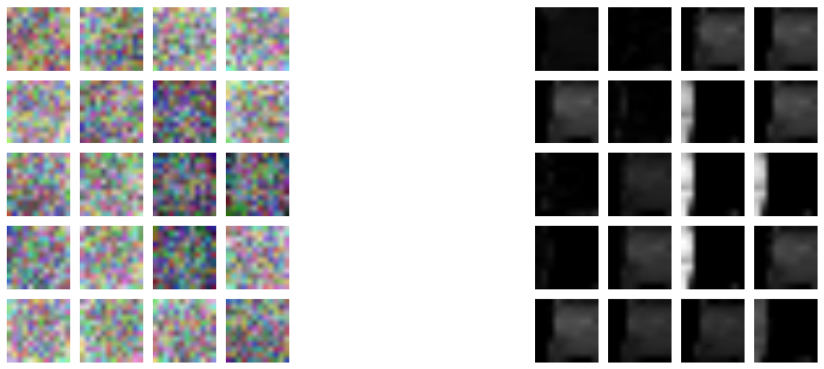


Fig. A.20: Row-Column encoding, model 9: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

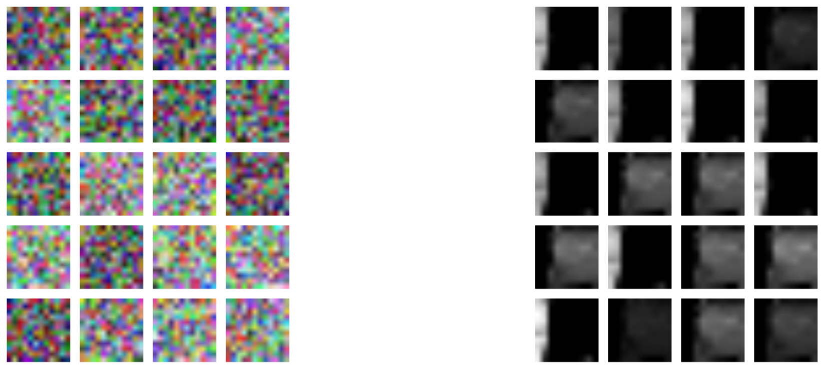


Fig. A.21: Direct encoding, MSE: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

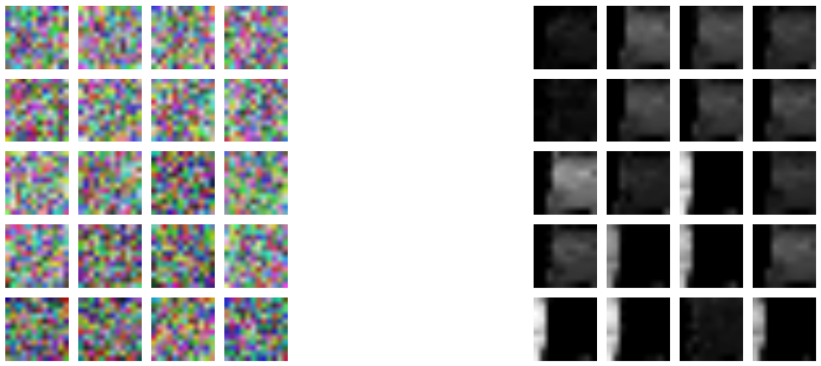


Fig. A.22: Direct encoding, emphasis on non-overlapping region: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

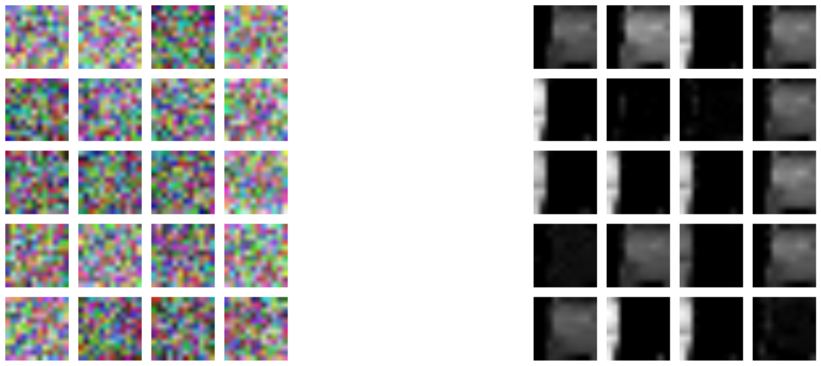


Fig. A.23: Direct encoding, emphasis on overlapping region: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

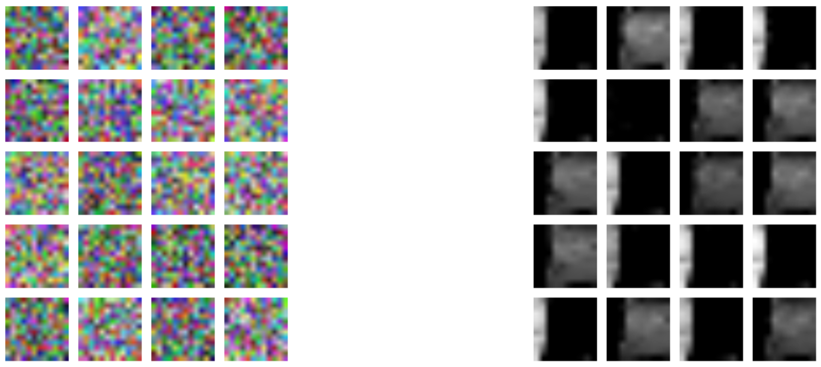


Fig. A.24: Direct encoding, emphasis on predicted overlapping region shape, $\lambda = 5$: The convolution kernels (left) and the activations (right). Each activation is produced by the kernel in the same position.

CURRICULUM VITAE

Trevor J. Landeen

Published Conference Papers

- Association of Emitter and Emission Using Deep Learning, Trevor Landeen, Jake Gunther, Todd Moon, David Ohm, and Robert North, in 2017 51st Asilomar Conference on Signals, Systems, and Computers, November 2017.

Fellowships

- Utah NASA Space Grant Consortium, 2016-2017
- Utah NASA Space Grant Consortium, 2017-2018