Advancing Radar Doppler Techniques for Activity Monitoring using Machine Learning

Qingchao Chen

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Security and Crime Science

University College London

April 4, 2019

I, Qingchao Chen, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

This thesis presents research on the design and development of novel machine learning methods for activity monitoring using micro-Doppler signatures (µ-DS) data collected from passive Wi-Fi radar (PWR) and multi-static pulse systems.

For PWR, *first* we propose a phase-sensitive signs-of-life detection method using PWR. *Second*, we propose a pipeline for µ-DS classification based on the Sparse Representation Classifier (SRC). *Third*, we adopt and modify the deep transfer network (DTN) to address the limited volume of the dataset. The phase-sensitive method is proved effective in detecting signs-of-life signatures and the modified DTN outperforms shallow methods and the conventional DTN by by 10% and 3% in PWR µ-DS dataset.

For active multi-static radar, *first* we propose Single-Channel (SC-) and Multi-Channel (MC-) DopNet for classifying personnel walking with or without a rifle. Based on the Deep Convolution Neural Network, SC-DopNet and MC-DopNet are designed for mono-static and multi-static radar respectively. They have been verified to improve the state-of-the-art results by around 8% in this task. *Second*, as µ-DS classification depends on two factors of variation: aspect angle and the target personnel, we design two unsupervised deep adaptation networks: i) re-weighted adversarial adaptation network and ii) joint adversarial adaptation network so that they can generalize well to unseen variation factors. The two networks are proved effective to learn useful features invariant to the factors and achieve the state-of-the-art results in both computer vision and µ-DS datasets. *Third*, we propose cooperative and adversarial relationship between the main activity and the two auxiliary classification tasks, including aspect angle and target personnel classification. To improve the performance of mono-static µ-DS classification, we propose ENet to integrate auxiliary tasks in activity classification network by selecting either the cooperative or adversarial learning strategy. ENet has been verified to outperform the SC-DopNet by 6% in average.

Impact Statement

The work carried out in this thesis develop several radar Doppler techniques for activity monitoring. The proposed methods focus on the signs-of-life detection using a PWR system and the activity monitoring using μ-DS classification of both PWR and multi-static active radar. Specifically, this thesis makes the following contributions to the field of radar and remote sensing.

- We propose the phase-sensitive real-time signal processing using PWR systems. This method can detect the breathing in both Line-of-sight and through-wall scenarios, which has been accepted in IEEE RadarConf 2016 and may be potentially patented and integrated in current PWR systems.

- We design the dictionary learning layer (DLL) and make modifications of existing DTN for μ-DS classification. This method successfully transfers vision knowledge to the μ-DS classification and outperforms existing DTN results by 3% in PWR μ-DS classifications. The design of DLL has been published in the Advanced Conference on Artificial Intelligence in Association for the Advancement of Artificial Intelligence (AAAI) 2018.

- We propose the SC-DopNet and MC-DopNet based on DCNN for mono-static and multi-static μ-DS classification respectively. In addition, we design two novel schemes to integrate multi-static μ-DS for improving the final decision making. This has been submitted to the IEEE Sensor Journal.

- We propose a new problem in μ-DS classification to eliminate the two factors of variation in an unsupervised manner, including the aspect angle and the target personnel. To address this practical and challenging problem, we propose two deep adaptation networks to eliminate the variations and they have been verified to achieve the state-of-the-art in both radar and computer vision benchmark datasets. One of the network has been published in premier annual computer vision conference, Computer Vision and Pattern Recognition (CVPR) 2018 and the other is ready to submit to AAAI 2019.

- We may be the first to consider different relationships between the main activity classification task and the auxiliary ones (aspect angle and target personnel classification) in radar μ-DS classification. We also propose the ENet to investigate the optimal strategies to ensemble these two auxiliary tasks. ENet has been proved to outperform the methods without ensembling schemes. We potentially wish to publish ENet in high impact journals.

Acknowledgements

I would like to give my sincere thank for the very useful advice from my supervisors Dr. Kevin Chetty and Prof. Karl Woodbridge. I also would like to thank the supports from my parents, Hong Chen, Danjuan Zhu and my girlfriend, Yang Liu. Finally, I would like to thank my wonderful collaborators in the group, Dr. Bo Tan, Dr. Matthiew Ritchie, Dr. Matthiew Ash, Dr. Riccardo, Dr. Francesco Fionelli and Prof.Paul Brennan.

Contents

List of Figures **23**

List of Tables **28**

List of Glossary **30**

1 Introduction **31**

1.1 Overview . . . 31

1.2 Contributions . . . 33

1.3 Thesis Outline . . . 36

2 Background **39**

2.1 Radar Theory and Systems . . . 39

2.1.1 Range Resolution and Maximum Unambiguous Range . . . 40

2.1.2 Radar Range Equation . . . 40

2.1.3 Doppler Processing . . . 42

2.1.4 Radar Systems and the Doppler Processing . . . 43

2.2 μ-DS Analysis for Activity Monitoring . . . 47

2.2.1 Human μ-DS . . . 49

2.2.2 Matched Filter . . . 50

2.2.3 Time-Frequency Analysis . . . 52

2.2.4 μ-DS Classification . . . 53

2.3 Sparse Coding and Dictionary Learning . . . 54

2.3.1 Sparse Coding . . . 55

2.3.2 Dictionary Learning 60
2.4 Deep Convolutional Neural Network 62
2.4.1 Convolutional Neural Network 62
2.4.2 Network Training Strategies in DCNN 68
2.4.3 DCNN Architectures 69
2.5 Transfer Learning in Deep Neural Networks 71
2.5.1 Multi-task Learning in Deep Neural Networks 73
2.5.2 UDA in DCNN 76
2.6 Activity Recognition in Diverse Modalities 79
2.6.1 Camera Video based Methods 79
2.6.2 Wearable Sensor based Methods 80
2.6.3 Radar Sensor based Methods 81

3 Passive Wi-Fi Radar for Activity Classification and Signs-of-life Detection 83
3.1 Introduction 83
3.2 Related Works 85
3.2.1 Signs-of-life detection using PWR 85
3.2.2 Activity recognition using PWR μ-DS 86
3.3 Signal Model and Processing of PWR 86
3.4 Breathing Detection using PWR 87
3.4.1 Limitations of CAF and Phase-Sensitive Detection 87
3.4.2 Hampel Filter and the Phase Stability Analysis 89
3.5 Activity Recognition using PWR 90
3.5.1 Problem Formulation 90
3.5.2 Capability of PWR for Human Activity Recognition 91
3.5.3 μ-DS Pre-processing Methods 91
3.5.4 SRC for Activity Recognition 95
3.5.5 DTN for Activity Recognition 96
3.5.6 Integrate Sparsity Prior in DTN 98
3.6 Results 102
3.6.1 PWR System Design and Implementation 102

3.6.2 Results of Breathing Detection using PWR 104
3.6.3 Results of Activity Recognition using PWR 108
3.7 Analysis and Discussion 114
3.7.1 Limitations of the Phase Extraction Method 116
3.7.2 Sparsity of Neural Activations and the Effect of Number of Recurrent Units in DLL 119
3.8 Summary 120

4 DopNet: A DCNN to Recognize Armed and Unarmed Human Targets 123
4.1 Introduction 123
4.2 Related Work 125
4.3 DopNet Architecture 125
4.3.1 DopNet Component and Architecture 126
4.3.2 DopNet Operations for Overfitting Preventions 128
4.4 SC-DopNet 131
4.5 MC-DopNet 131
4.6 Implementation 135
4.6.1 Radar System 135
4.6.2 Experiment 136
4.6.3 Processing Method 136
4.7 Results and Ablation Study of SC-DopNet 137
4.7.1 Raw and Augmented μ-DS 138
4.7.2 Evaluating Radar Operational Parameters 139
4.7.3 Evaluating the Drop-out rate 143
4.7.4 Evaluating E-dist and M-dist 144
4.7.5 Evaluating the SNRs and Memory Usage 145
4.7.6 Comparison with Empirical Features and Classifiers 145
4.8 Results and Ablation Study of MC-DopNet 149
4.8.1 Analysis of Node-Angle Combinations in MC-DopNet 149
4.8.2 Comparison of Fusion Methods in MC-DopNet 151
4.8.3 Comparison with Empirical Features and Classifiers 152

4.9 Summary 154

5 Deep Adaptation Network using Optimal Transport 157

5.1 Introduction 157

5.2 Related Works 159

5.2.1 Distribution Matching using Adversarial Training 160

5.2.2 Matching Feature Distribution using OT 161

5.3 Problem Formulation 161

5.4 Re-weighted Adversarial Adaptation Network 162

5.4.1 Optimal Transport in Adversarial Training 164

5.4.2 Adapting the Classifier by Label Distribution Matching 166

5.4.3 Optimization in RAAN 167

5.5 Joint Adversarial Adaptation Network(JAAN) 168

5.5.1 Distribution Matching using Entropy-Regularized OT 171

5.5.2 Adaptive Distance Function Learning in OT 172

5.5.3 Optimization 173

5.6 Implementation 173

5.7 Experiments and Results 174

5.7.1 Adaptation between Hand-Written Digit Datasets 174

5.7.2 Adaptation between Cross-Modality Datasets 177

5.7.3 Adaptation between Radar μ-DS Datasets 178

5.8 Analysis 187

5.8.1 Evaluate the Re-weighting Scheme in RAAN 187

5.8.2 Ablation Study of Modules in JAAN 188

5.8.3 Evaluate Distribution Divergence of Feature Embeddings 190

5.9 Summary 191

6 ENet: Ensembling Cooperative and Adversarial Learning for μ-DS Classification 195

6.1 Introduction 195

6.2 Related Works 196

6.3 Cooperative Learning and Adversarial Learning 197
6.3.1 Problem Formulation and ENet Architecture 197
6.3.2 Cooperative Learning and Adversarial Learning 198
6.3.3 ENet: Ensemble Cooperative and Adversarial Learning 200
6.4 Experiments and Result Analysis 200
6.4.1 Results of Ensembling Two Tasks (Exp1-6) 201
6.4.2 Results of Ensembling Three Tasks (Exp7-9) 205
6.5 Summary 207

7 Conclusions 209
7.1 Contribution Summary 209
7.2 Future Work 213

Appendices 215

A Materials of sparse coding and dictionary learning 215
A.1 Restricted Isometry Properties 215
A.2 Pseudo-Codes of MP, OMP and Subspace Pursuit 215

B Materials of DCNN and Deep Adaptation Network 219
B.1 Optimization of DCNN 219
B.2 Batch Normalization in DCNN 219
B.3 Relationship between JS divergence and Adversarial Training Optimization 221
B.4 Pseudo-code of Sinkhorn Algorithm 222

C Results of Ensembling Two Tasks in Exp7-9 in Chapter 6 225

Bibliography 229

List of Figures

2.1 Architecture of the CW radar. 44

2.2 Architecture of the pulse radar. 44

2.3 Architecture of the PWR. 46

2.4 AF results using a simulated single rectangular pulse waveform of 2 second from [1]. 48

2.5 AF result using experiment data of a bi-static PWR: (a) zero Doppler cut, (b) zero range cut. 48

2.6 Human target point scatter model. Radar sends the waveform at the frequency of f_c; When a target walks approaching to the radar antenna, the Doppler modulated frequencies $f_c + f_{arm1}$, $f_c + f_{arm2}$, $f_c + f_{leg1}$, $f_c + f_{leg2}$ and $f_c + f_{torso}$ represent the velocities of different body parts. 49

2.7 Simulated μ-DS of a walking target, from [26, 90] 50

2.8 (a) Fast-time four pulses modulated by Doppler frequency. (b) The phases of Doppler frequency within the four pulses. 51

2.9 (a) Matched filter response matrix MF, the column index $j \in [1,2,...,N_{delay}]$ indicates the fast time index; the row index $i \in [1,2,...,N_{pulse}]$ indicates the slow time index. (b) Time-Frequency analysis, taking the range-time history from (a) to obtain the frequency-time history responses; green vector indicates the averaged range-time history at target's range bins. 51

2.10 (a) ISTA framework to calculate sparse coding s of the input y using dictionary D and linear operator H. (b) LISTA framework to calculate the sparse code s by unfolding the iterative methods in ISTA to recurrent units. 58

2.11 Convolutions in Conv layer: input feature map with the size of $H_{in} \times W_{in} \times C_{in}$ is convoluted by C_{out} filters with the size of $h_i \times w_i \times C_{in}$, where $i \in [1, 2, ..., C_{out}]$. The output is with the size of $H_{out} \times W_{out} \times C_{out}$. 64

2.12 Max-pooling operations. 64

2.13 Activation functions in DCNN: input is $x \in \mathbb{R}$ and output is $f(x)$ 65

2.14 DCNN architectures: 1000 FC indicates the output dimension of the FC layer is 1000; 3×3 Conv, 256 Chs indicates the convolutional filter size is 3×3 and we use 256 filters. Softmax indicates the softmax function. (a)Architecture of AlexNet; (b)Architecture of VggNet-16. 70

2.15 An “unpopular” animal called Okapi, with Giraffe’s head, Horse’s body and Zebra’s leg. 72

2.16 An example of DTN: transferring knowledge from object recognition in computer vision to µ-DS classification in PWR. Source domain (data, label) pair x_s, y_s from ImageNet is fed to the DCNN T and network training is based on the CE loss L_{CE}. We transfer network and domain knowledge T to the µ-DS classification task. Fine-tuning the network T using the target domain (data, label) pair x_t, y_t and CE loss. 74

2.17 An example of DEN: ensembling multiple factors (aspect angle and target personnel) recognition in µ-DS based activity recognition. Given the µ-DS data x_{input}, feature extractor T extracts useful feature $T(x_{input})$ based on three tasks, namely the motion, target personnel and the aspect angle recognition. 75

2.18 Adversarial training in UDA networks. Given the source (data, label) pair x_s, y_s and target data x_t, DCNN network T aims to extract useful features $T(x_s)$ and $T(x_t)$ so that the domain discriminator D cannot differentiate which domain they are from. The classifier is represented as *CLS* and learned by CE loss using x_s, y_s. 76

3.1 Pre-processing steps for activity recognition using PWR. 91

3.2 (a) Start and end point illustration; (b)The useful part of the µ-DS is adjusted into the same size, with the DSI and the ambiguity peak examples. 92

3.3 DTN for PWR based µ-DS classification using AlexNet. 97

3.4 (a) DTN for PWR using AlexNet and DLL; (b) Recurrent network design of the highlighted DLL7 layer; (c) Detailed architecture of the highlighted unit in DLL7 layer. 99

3.5 (a) Two-channel PWR system; (b) RF front-end daughter board; (c) FPGA of USRP device. 103

3.6 (a) One-thread sequential data processing framework; (b) Multi-thread parallel data processing. 103

3.7 (a) Indoor experiment scenario; (b) Through-wall experiment scenario. . 104

3.8 Bi-static triangle and the bi-static aspect angles in the four test positions. 105

3.9 Breathing detection using beacon signals at four positions; (a),(b),(c) and (d) correspond to results obtained from P1 to P4 respectively; red curves in (a) and (c) indicate the phase variations caused by irregular chest-wall movements. 106

3.10 Breathing detection using data signals at four positions; red curve in (c) indicates the phase variations caused by irregular chest-wall movements. 107

3.11 Breathing detection results in through-wall scenario (a) at P1 (b) at P2. . 108

3.12 Overview of PWR µ-DS in dB scale. (a) to (f) are the µ-DS of motion 1 (M1) to 6 (M6) in Table 3.3. 110

3.13 Detection results of the proposed start and end point detection method. Blue line indicates the time history of normalized spectrum energy of the µ-DS in Figure 3.12. The red star represents the detected time bins with active µ-DS. 112

3.14 Overview of false detection results: red, blue and gree line and markers indicate the missed, false alarm and the total false detection numbers respectively. 113

3.15 (a) Phase changes caused by breathing and the static clutters; (b)Phase changes caused by breathing and large motions. 117

3.16 (a) Phase output using 1000 batches in the processing; (b) Phase output using 500 batches in the processing. 118

3.17 Signal amplitude of the three batches that generate discontinuous phase outputs in Figure 3.16(a). 118

3.18 ℓ_1-norm of the neural activations of the T_{DLL7} and T_{FC7} using different number of recurrent units. 120

4.1 Architecture of SC-DopNet. 126

4.2 Raw µ-DS of a target walking unarmed from angle 1, using receiver node 1 in Figure 4.7; two black boxes indicate the augmented µ-DS, with window size of 1s and window height of 100Hz. 129

4.3 Augmentation results: (a), (b), (c) and (d) are four augmented data examples generated from the 5-second µ-DS in Figure 4.2. The augmentation method and parameters are illustrated in Section 4.3.2. 130

4.4 Architecture of MC-DopNet 131

4.5 µ-DS of two activities from three aspect angles using the mono-static radar. All x-axis represents time with unit of second while all y-axis represents frequency with unit of Hz. 132

4.6 Picture of NetRad System [2]: (a) RF chain and FPGA; (b) Data collection module and mini-PC 135

4.7 Experiment scenario in the field using NetRad. 137

4.8 Raw μ-DS of walking among three angles and three nodes. All x-axis represents time with unit of second while all y-axis represents frequency with unit of Hz. 138

4.9 Raw Doppler signature of walking with rifle among three angles and three nodes. All x-axis represents time with unit of second while all y-axis represents frequency with unit of Hz. 139

4.10 Recognition rates of three training data percentages and three aspect angles, using data from node1, dwell time of 1s and SC-DopNet. 140

4.11 Recognition rates of three training data percentages and three aspect angles, using data from node1, dwell time of 1.5s and SC-DopNet. 140

4.12 Recognition rates of three training data percentages and three aspect angles, using data from node1, dwell time of 2s and SC-DopNet. 141

4.13 Recognition rates of three training data percentages and three aspect angles, using data from node1, dwell time of 2.5s and SC-DopNet. 141

4.14 Averaged recognition rate among all dwell time with respect to different angles and training ratios. 142

4.15 the averaged recognition rate among all training data percentages with respect to different aspect angles and dwell times. 142

4.16 Recognition rates using different drop-out rates and training data percentages. All data come from angle 1, node 1 using dwell time of 1s. . . 144

4.17 Recognition rates among different regularization weights of the E-M distance. All results use data from node 1, angle 1, dwell time of 1s with 20% the training percentage. 145

4.18 Average recognition rates among three aspect angles with respect to dwell time and classifiers. We report the best feature combination results for the non-deep classifiers. 146

4.19 Average recognition rates among different dwell time but ranging from different aspect angles. We report the best feature combination result for the non-deep classifiers. 146

4.20 Averaged recognition results of four classifiers on aspect angles, ranging different nodes. We report the best feature combination results. 147

4.21 Recognition results of five classifiers and different aspect angles. We use all multi-static data with dwell time of 1s. We report the best feature combination results from [3]. 153

5.1 Architecture of RAAN. 162

5.2 Architecture of JAAN. 169

5.3 Adversarial training in JAAN: (a) Distance between joint distributions described by a fixed distance function (b) Distance between joint distributions using the adaptive distance function, defined by the cosine distance of presenter network output. Presenter network P pushes the two distributions to increase the distance between them. (c) Although P pushes distributions, T_t, C_t,T_s and C_s pull the distribution closer even than before, so that joint distributions under the adaptive distance function are matched closer. (d) Distance between joint distributions under the fixed distance function after adaptation is closer so that the distributions are matched between domains. 171

5.4 UDA datasets: (a) Three hand-written digit datasets; (b) cross-modality dataset including RGB and RGB-depth images. 174

5.5 A-distance and recognition results in Exp1-6 182

5.6 Recognition results of Exp1-6 using RAAN's base network;AVG-M refers to average results of the 6 experiments using a specific method. 183

5.7 Recognition results of Exp1-6 using JAAN's base network; AVG-M refers to average results of the 6 experiments using a specific method. 184

5.8 A-distance and recognition results in Exp7-12. AVG represents the averaged result of the six experiments. 185

5.9 Recognition results of Exp7-12 using RAAN's base network. 186

5.10 Recognition results of Exp7-12 using JAAN's base network. 186

5.11 Ratio of label distribution between SVHN and MNIST; red line indicates the ground truth ratio, while blue one indicates the estimated ratio. . . . 188

5.12 T-SNE plot of features when adapting from SVHN to MNIST; (a) No adaptation (b) Adaptation after ADDA (c) Adaptation after RAAN (d) Adaptation after JAAN. We randomly select 1000 features samples from 10 classes, with 100 samples per class. 191

6.1 Architecture of ENet; M, H and A refer to motion, human target personnel and aspect angle respectively. Conv and FC and CLS refer to convolutional, fully connected and the classifier respectively. 198

6.2 Results of CL and AL strategies in Exp1-6, summarizing the results of Table 6.4, 6.5 and 6.6; motion recognition rates (a) in Exp1 and Exp2 using CL; (b) in Exp1 and Exp2 using AL; (c) in Exp3 and Exp4 using CL; (d) in Exp3 and Exp4 using AL; (e) in Exp5 and Exp6 using CL; (f) in Exp5 and Exp6 using AL. 203

6.3 Result comparisons of Exp1-6 with the baseline; Ens-Angle-Best: ensemble angle recognition and show best result; Ens-Human-Best: ensemble target personnel and show best results. 204

6.4 Result comparison of Exp7-9 with the baseline; Ens-Multiple-Avg: average result when ensembling both angle and target personnel recognition; Ens-Angle-Best: best result when ensembling both angle and target personnel recognition. 206

B.1 Back-propagation in Neural Networks. 220

List of Tables

3.1 Phase variations of the four test positions from P1 to P4 based on Figure 3.10 with the corresponding aspect angle and cosine value of the aspect angles. 104

3.2 Methods under evaluation for μ-DS classification and their descriptions. 109

3.3 Dataset description of the six activities in PWR experiments. M1(20) indicates the first motion including 20 μ-DS samples. 109

3.4 Activity recognition results for different features and classifiers, percentage in (%). 113

3.5 Confusion matrix of SVM method with PCA features using 20% data for training, percentage in (%). 114

3.6 Confusion matrix of SRC with PCA features,using 20% data for training, percentage in (%). 115

3.7 Confusion matrix of AlexNet+FC+FT using 20% data for training, percentage in (%). 115

3.8 Confusion matrix of AlexNet+WholeNet+FT, using 20% data for training, percentage in (%). 115

3.9 Confusion matrix of AlexNet+DLL7 using 20% data for training, percentage in (%). 115

3.10 Recognition rates using different number of recurrent units in DLL, evaluated using 20% data for training. 119

4.1 Processing parameters including the ones to collect and generate the μ-DS, train the DCNN and augmentation. 136

4.2 Radar operating parameter set used for evaluating in the μ-DS classification experiments. 139

4.3 Recognition rates with different input SNRs. All data come from node 1, dwell time of 1s using SC-DopNet. 144

4.4 Recognition results of three non-deep classifiers with the best empirical feature combinations and SC-DopNet results. We use data from node 1 with training percentage of 20%, different dwell time and aspect angles, percentage in (%). 147

4.5 Recognition results of three non-deep classifiers with the best empirical feature combinations and SC-DopNet results. We use data from all aspect angles, each radar node, different dwell times, with training percentage of 20%, percentage in (%). 148

4.6 MC-DopNet recognition results of three aspect angles, using dwell Time of 1s, 20% training percentage but using the BV method for fusion. . . . 150

4.7 MC-DopNet recognition results of three aspect angles, using dwell Time of 1s, 20% training percentage but using the GIR method for fusion. . . 150

4.8 MC-DopNet recognition results of three aspect angles, using dwell Time of 1s, 20% training percentage but using the ℓ_{21}-Norm method for fusion. 151

4.9 Mean and standard deviation of the results in Table 4.6 fused by BV method. 152

4.10 Comparison of MC-DopNet results with other feature extraction methods and classifiers. For fair comparison, all voting schemes are the best thresholding voting scheme and features are selected the best combinations reported in the paper [4]; percentage in (%). 153

5.1 A-distance between hand-written digit datasets; M, U and S refer to MNIST, USPS and SVHN respectively. 175

5.2 Recognition rates (%) of adapting hand-written digit dataset. 176

5.3 Recognition rates (%) of adapting hand-written digit dataset. 176

5.4 Recognition rates (%) of adapting cross-modality dataset. 179

5.5 Experiment details for adapting two factors of variations including aspect angle and target personnel. 179
5.6 Experiments to evaluate the factors of variation; the main interest factor is the motion recognition while two distraction factors include aspect angle and target personnel. We conduct these experiments in both “Factor Control” and “No Control” scenario. 180
5.7 Deconvolution network for JAAN with SC-DopNet. 181
5.8 A-distance for adaptation tasks Exp1-6. AVG represents the averaged result of the six experiments. 181
5.9 Recognition rates (%) in evaluation 1, adapting between the factors of target personnel; AVG-M refers to average results of the 6 experiments using a specific method. Base-Net refers to results using base network of RAAN and JAAN. 182
5.10 Averaged recognition rates (%) of evaluation 1 in “No Control” scenarios. Base-Net refers to results using base network of RAAN and JAAN. 183
5.11 A-distance for adaptation tasks Exp7-12. 184
5.12 Recognition rates (%) in evaluation 2, adapting the factors of aspect angle; AVG-M refers to average results of the 6 experiments using a specific method. Base-Net refers to result of base network of RAAN and JAAN. 185
5.13 Average recognition rates (%) of evaluation 2 in “No Control” scenarios. Base-Net refers to results using base network of RAAN and JAAN. . . 187
5.14 *A*-distance of adversarial training method 188
5.15 Recognition rates (%) using different batch sizes 190
5.16 Recognition rates (%) using different weights of entropy regularization. 190
5.17 Recognition rates (%) using different iterations of updating presenter network per update of the feature transformer and classifier. 191

6.1 Experiment summary details for evaluating different task relationships and the optimal learning strategies. 201

6.2 Experiment summary for evaluating different task relationships and the optimal learning strategies. 202
6.3 Experiment summary of the learning strategies under evaluation and the parameter set under selection. 202
6.4 Evaluated Task Relationships using Exp 1 and 2. Recognition rates $A(B)$ indicate: $A\%$ for motion recognition rate and $B\%$ for auxiliary task recognition rate. 204
6.5 Evaluated task relationships using Exp3 and 4, recognition rates $A(B)$ indicate: $A\%$ for motion recognition rate and $B\%$ for auxiliary task recognition rate. 205
6.6 Evaluated task relationships using Exp5 and 6, recognition rates $A(B)$ indicate: $A\%$ for motion recognition rate and $B\%$ for auxiliary task recognition rate. 205
6.7 Summarized optimal relationships when ensembling single auxiliary task. 206
6.8 Evaluated Relationships among Three Tasks. 206

C.1 Result of ensembling two tasks, node 1, percentage in (%). 226
C.2 Result of ensembling two tasks, node 2, percentage in (%). 227
C.3 Result of ensembling two tasks, node 3, percentage in (%). 228

Glossary of Terms

AF	Ambiguity Function
AL	Adversarial learning
BN	Batch normalization
CAF	Cross Ambiguity Function
CE	Cross-Entropy
CFAR	Constant False Alarm Rate
CL	Cooperative learning
CNN	Convolutional neural network
CW	Continuous wave
DAN	Deep adaptation network
DCNN	Deep convolutional neural network
DEN	Deep ensemble network
DL	Dictionary learning
DLL	Dictionary learning layer
DNN	Deep Neural Networks
DSI	Direct signal interference
DTN	Deep transfer network
EM	Earth Mover
FC	Fully connected
FFT	Fast Fourier Transform
GAN	Generative adversarial networks
GIR	Greedy Importance Reweighting
GPU	Graphics processing unit

IF	Intermediate frequency
ISTA	Iterative Shrinkage-Thresholding Algorithm
JAAN	Joint adversarial adaptation network
JS	Jenson-Shannon
LISTA	Learned Iterative Shrinkage-Thresholding Algorithm
LoS	Line of Sight
LNA	Low Noise Amplifier
MP	Matching pursuit
MMD	Maximum Mean Discrepancy
OMP	Orthogonal matching pursuit
OT	Optimal transport
PCA	Principle component analysis
PRF	Pulse Repetition Freuqency
PWR	Passive Wi-Fi radar
RAAN	Re-weighted adversarial adaptation network
ReLU	Rectified linear unit
RFID	Radio-frequency identification
RCS	Radar cross section
SGD	Stochastic gradient descent
SNR	Signal to noise ratio
SRC	Sparse representation based classification
STFT	Short time Fourier transform
SVD	Singular value decomposition
SVM	Support vector machine
UDA	Unsupervised domain adaptation
Wi-Fi	Wireless Fidelity
µ-DS	Micro-Doppler signature

Chapter 1

Introduction

With the aim of RAdio Detection And Ranging, fundamental radar function is to detect the targets and estimate the ranges. From the 1880s when Henrich Hertz first investigated the usage of electromagnetic waves to detect metal objects, to the modern 77GHz radar systems for automotive cars, radar technologies have been applied to a much wider area and not limited to military applications. This thesis investigates the advanced radar Doppler techniques for activity monitoring using machine learning methods.

1.1 Overview

To monitor the human activities, activity recognition and signs-of-life detection have been important research topics to facilitate enhanced situational awareness. Compared to other alternative sensors, radars are advantageous because they are *device-free to targets under detection, robust to weather conditions such as foggy weather, more widely coverage compared to cameras, easily deployable, with high frame rates and privacy preserved.*

Besides fundamental radar information such as range and bearing, the micro-Doppler signature (µ-DS) (see more details in Section 2.2 and 2.2.1) is a mixture of the modulated Doppler frequencies caused by the backscattered signals from different scatters of the target object [5, 6]. Recently µ-DS has been an essential radar modality for identifying and monitoring the human activities. The radar community have applied various machine learning methods to classify the radar µ-DS for activity monitoring since it contains detailed information of target's movements and activities.

In general, machine learning methods aim to provide system the ability to automatically learn and improve from the experience (the large amount of data). The last decade witnessed the success of the machine learning community especially the deep neural network (DNN) based methods, mainly due to the large amount of data and the powerful computation resources. Recent DNN based developments have surpassed the human object recognition performances, outperformed the human's capability to mimic and generate other people's voices and even beaten the best human player in the Chinese board game Go. With such great successes, we investigate machine learning methods, especially DNN based methods in μ-DS classifications in this thesis.

For monitoring the general human activities such as walking, falling, sitting etc., we need to firstly extract features that are discriminative and meanwhile beneficial for generalization of the classification method. Large amount of research have focused on designing hand-crafted features for μ-DS classifications, such as maximum Doppler frequency and time width of the activity, however, these may largely depend on experts' knowledge and have been proved to be sensitive to the experimental environments. This motivates the research in automatically learning the features and classifiers in an end-to-end training manner in the recent success of DNN. The learned features are ideal and useful if they can represent the intrinsic structure and valuable information about the comprising factors of the original μ-DS. Until now, how to efficiently learn these features is still an open question.

With the exponentially increasing amount of data, the current mainstream feature learning method requires large amount of well-annotated data. These annotation labels then serve as the direct prior knowledge to guide the feature learning for the classification. However, label annotations are expensive and time consuming, not to say that we are sometimes restricted to access the large amount of data in particular applications. More specifically, in radar community, it is difficult to collect a μ-DS dataset as large as the ImageNet [7], considering the different experimental environments, the time-consuming and the highly repetitive human target movements and different radar sensor types. Since μ-DS dataset is relatively small, we need more *generalized prior knowledge* for learning discriminative features that may also contribute to classifier generalizations.

Compared with the principles of radar µ-DS hand-crafted feature design illustrated in previous paragraph, these *generalized prior knowledge* are relatively high-level and also scalable and applicable to other scenarios. We illustrate the *generalized prior knowledge* in Section 1.2 in more details.

1.2 Contributions

This thesis proposes advanced radar Doppler techniques for activity recognition and signs-of-life detection. The newly proposed signs-of-life detection using passive Wi-Fi radar (PWR) is more focused on real-time signal processing using the phase information. In addition, the thesis is mainly focused on designing and integrating *generalized prior knowledge* in feature learning for the µ-DS classification. More specifically, contributions of the thesis and the relevant *generalized prior knowledge* are illustrated in the following:

1. **Signs-of-life detection for PWR:** we utilize phase information for signs-of-life detection in real-time PWR processing and verify the effectiveness using experimental results.

2. **Apply sparsity prior for µ-DS classification:** we apply sparse coding procedure for sparse representation classifier (SRC) design.

3. **Integrate sparsity prior and hierarchical prior in µ-DS classification:** to extract useful features within a small-scale µ-DS dataset, we leverage the deep transfer network (DTN), use the pre-trained weights on ImageNet and fine-tune the whole networks using the PWR µ-DS dataset. More importantly, we incorporate sparsity prior in DTN design (based on hierarchical prior) by replacing the fully connected (FC) layer with the newly designed dictionary learning layer (DLL).

4. **Integrate the prior of shareable representations and common factors of multi-static µ-DS in the classification:** with the multi-static µ-DS data, we propose to first learn deep convolution neural network (DCNN) representations for each channel and then design two schemes to ensemble these features for the final classification. These ensemble methods are based on the prior knowledge that common

factors exist in multi-static (multi-channel) µ-DS which can robustly constrain the feature learning and improve the final classification results.

5. **Integrate the prior of common factors among different data distributions in µ-DS classification:** we consider the same task for example activity classification but the µ-DS distributions are different. For example, we train network using the µ-DS sampled from target A and B but test on target C. The prior we are incorporating is that the features learned from different data distributions shall only represent the common factor directly indicating the activity, rather than the target personnel.

6. **Investigate independent and correlated factors in µ-DS classification:** µ-DS are composed of multiple factors and we investigate the prior knowledge about the relationships among multiple factors to improve the main activity classification. Two relationships are proposed including independent and correlated ones, which correspond to adopt the newly proposed adversarial and cooperative learning strategies in network training. For example, if we assume the target personnel is independent to the activity, we may apply adversarial training to learn the shareable features, which tend to discriminate the activities but confuse the target personnel recognition.

These contributions have generated the following publications:

- **Q.Chen**, K.Chetty, K.Woodbridge, B.Tan (2016, May). Signs of life detection using wireless passive radar. In 2016 IEEE Radar Conference (RadarConf), 2016 IEEE (pp. 1-5). ***[Contribution 1]***

- **Q.Chen** , B.Tan, K.Chetty, K.Woodbridge. (2016). Activity Recognition Based on Micro-Doppler Signature with In-Home Wi-Fi. In 2016 IEEE International Conference of HealthCom. ***[Contribution 2]***

- **Q.Chen**, Y.Liu, W.Chen, I.Wassell. "Dictionary Learning Inspired Deep Networks for Scene Recognition" In Advanced Association of Artificial Intelligence Conference (AAAI) 2018. ***[Contribution 3]***

- **Q.Chen**, Y.Liu, F.Fionalli, M.Ritchie, K.Chetty, DopNet: a DCNN Trained from Scratch for Classification of Armed/Unarmed Human Targets using Multiple-Channel Micro-Doppler Signatures. Submitted to IEEE Sensor Journal. ***[Contribution 4]***

- **Q.Chen**, Y.Liu, Z.Wang, I.Wassell, K.Chetty. "Re-weighted Adversarial Adaptation Network for Unsupervised Domain Adaptation" In IEEE Computer Vision and Pattern Recognition (CVPR) 2018. ***[Contribution 5]***

- **Q.Chen**, Y.Liu, I.Wassell, K.Chetty. "Joint Adversarial Adaptation Network using Optimal Transport for Unsupervised Domain Adaptation" Ready for Advanced Association of Artificial Intelligence Conference (AAAI) 2019 ***[Contribution 5]***

In addition to the above publications, a number of papers from the early stages of the Ph.D. (pre-MPhil upgrade) have been published in conference proceedings. These are not directly related to the work presented in this thesis, but for completeness are listed below.

- **Q.Chen**, B.Tan , K.Chetty, K.Woodbridge. "Doppler Based Detection of Multiple Targets in Passive Wi-Fi Radar using Underdetermined Blind Source Separation" In IEEE International Radar Conference, 2018 (Best Student Paper Nomination Award)

- **Q.Chen**, M.Ritchie, Y.Liu, K.Chetty, and K.Woodbridge, Joint fall and aspect angle recognition using fine-grained micro-Doppler classification. In Radar Conference (RadarConf), 2017 IEEE (pp. 0912-0916). IEEE..

- **Q.Chen**, K.Chetty, P.Brennan, " A coherent through-the-wall MIMO phased array imaging radar based on time-duplexed switching", in SPIE Defense, Security, and Sensing. International Society for Optics and Photonics 2017.

- Y.Liu, **Q.Chen**, I.Wassell, " Deep Network for Image Super-resolution with a dictionary learning layer ", to appear in ICIP 2017.

- K.Chetty, **Q.Chen**, M.Ritchie, K.Woodbridge. A low-cost through-the-wall FMCW radar for stand-off operation and activity detection. InSPIE Defense+ Security 2017 May 1 (pp. 1018808-1018808). International Society for Optics and Photonics.

- Y.Liu, W.Chen, **Q.Chen**, I.Wassell (2016, October). Support Discrimination Dictionary Learning for Image Classification. In European Conference on Computer Vision (pp. 375-390). Springer International Publishing.

- K.Chetty, **Q.Chen** and K.Woodbridge, 2016, May. Train monitoring using GSM-R based passive radar. In Radar Conference (RadarConf), 2016 IEEE (pp. 1-4). IEEE.

- Tan, B., Burrows, A., Piechocki, R., Craddock, I., **Q. Chen**, Woodbridge, K. and Chetty, K., 2015, December. Wi-Fi based passive human motion sensing for in-home healthcare applications. In Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on (pp. 609-614). IEEE.

- **Q.Chen**, B.Tan, K.Woodbridge, K.Chetty (2015, April). Indoor target tracking using high doppler resolution passive Wi-Fi radar. In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 5565-5569).

- B.Tan, **Q.Chen**, W.Li, K.Chetty, K.Woodbridge, WiFi CSI signal Processing and Behavior Recognition Methods. In IEEE Communications Magazine.

- M.Ritchie, M.Ash, **Q.Chen** and K.Chetty, 2016. Through Wall Radar Classification of Human Micro-Doppler Using Singular Value Decomposition Analysis. Sensors, 16(9), p.1401.

1.3 Thesis Outline

The thesis is outlined as follows:

Chapter 2 first reviews the background in radar theory, systems, signal processing and classification methods for μ-DS. Since we mainly adopt the sparse coding, dictionary learning and DCNN for µ-DS classification, their fundamentals are also illustrated. Finally, we review the transfer learning framework and introduce its applications in activity monitoring.

Chapter 3 presents the research contribution of activity classification and signs-of-life detection using PWR. First the phase information extraction methods based on cross ambiguity function (CAF) is proposed for detecting the sensitive breathing patterns. Second, facing the small-scale µ-DS dataset including six daily motions, we adopt to use SRC and DTN for classifying the six activities. The proposed DTN is aimed to integrate the sparsity prior and hierarchical prior in DCNN by replacing the newly proposed DLL with the FC layer in AlexNet [8].

Chapter 4 presents the newly proposed single-channel DopNet (SC-DopNet) and multi-channel DopNet (MC-DopNet) to recognise armed and unarmed human targets using multi-static µ-DS. We propose novel data augmentation schemes, new regularization term to prevent overfitting and two fusion schemes to improve the final decision making.

Chapter 5 focuses on increasing the generalization capability of DCNN to the unseen factors of variations in an unsupervised manner. We propose two deep adaptation networks to eliminate the variations caused by the target personnel and the aspect angle.

Chapter 6 aims to investigate two relationships among multiple factors, including independent and correlated ones. We investigate relationships between the auxiliary factors (target personnel and aspect angle) and the main one (activity) using the deep ensemble network (DEN) but adopting two learning strategies: cooperative or adversarial learning. Through extensive experiments in µ-DS classification, we identify the relationships among them and adopt the optimal learning strategy for activity recognition.

Chapter 7 concludes the whole thesis and proposes the future work.

Chapter 2

Background

Before discussing the advanced radar Doppler techniques for activity monitoring using machine learning, it is necessary to introduce the basic radar theory, Doppler processing and relevant machine learning methods. In this chapter, we first introduce the radar theory basics, the pulse radar and the PWR systems to collect data in Section 2.1. Next, we review μ-DS theory, the associated signal processing and classification methods for activity monitoring in Section 2.2. Sparse coding and dictionary learning (DL) fundamentals relevant to μ-DS classification are then covered in Section 2.3. Finally we introduce the basics of DCNN in Section 2.4, review the transfer learning theory and the mainstream methods (DTN, DEN and deep adaptation network (DAN)) in Section 2.5.

2.1 Radar Theory and Systems

Radars are electronic systems that detect and estimate the range, velocity and bearing of objects via Electromagnetic waves. Most of the concepts and knowledge in this Section 2.1 are based on the books [9, 10] and we provide the reference here for convenience.

This section *first* reviews the basic concepts of range resolution and maximum unambiguous range in Section 2.1.1 and introduces the radar range equation in Section 2.1.2. *Second*, in Section 2.1.4), we introduce the concept of Doppler and the pipeline to obtain the Intermediate Frequency (IF) signal in PWR and pulse radar systems. Note that we consider only the super-hetrodyne receiver in this thesis which down-converts the radio frequency to the IF signals before sampling and digitizing.

2.1.1 Range Resolution and Maximum Unambiguous Range

Range estimation is fundamental and performed by calculating the time delays between pulse transmission and receiving the reflected echo, as shown in Eq.(2.1), where R is the range between target and radar, c is the speed of light and τ is the time delay.

$$R = \frac{c \times \tau}{2} \tag{2.1}$$

The range resolution measures the minimum distance where two targets with such distance in the bearing are still separable. It is directly related to the pulse width T_{pulse} in the Eq.(2.2) and (2.3), where two targets are at the range of R and $R + \delta R$, with the minimum range resolution δR and the speed of light c.

Due to the two-way propagation, δR is proportional to the half of the pulse width T_{pulse}, as derived in the Eq.(2.3). Conventionally, the range resolution is usually interpreted by the 3dB bandwidth B of the pulse. Considering a simple pulse radar system with rectangular pulse width T_{pulse}, the range resolution can be calculated following the latter part of Eq.(2.3).

$$T_{pulse} = \frac{2(R + \delta R)}{c} - \frac{2R}{c} = \frac{2\delta R}{c} \tag{2.2}$$

$$\delta R = \frac{cT_{pulse}}{2} = \frac{c}{2B} \tag{2.3}$$

The maximum unambiguous range measures the maximum range the radar system can detect without range ambiguities. This ambiguity happens when the target echo of the first pulse is received after the second pulse transmission. The maximum unambiguous range, denoted as R_u, depends on the pulse repetition frequency (PRF) or the pulse repetition interval of the system and can be calculated based on Eq.(2.4), given the defined PRF of the radar system.

$$R_u = \frac{c}{2PRF} \tag{2.4}$$

2.1.2 Radar Range Equation

The radar range equation is a mathematical and deterministic equation measuring the received power from the radar receiver based on the physical environment and the transmitter power. As shown in Eq.(2.5), the received signal power of the target, P_r can be represented as:

$$P_r = \frac{P_t G_t G_r \sigma_T \lambda^2 F_t F_r Q}{{R_t}^2 {R_r}^2 4\pi^3 L_t L_r} \tag{2.5}$$

- P_t is the transmit power,
- G_t, G_r are the gains of transmitter and receiver antenna respectively,
- σ_T is the radar cross section of the target,
- λ is the wavelength of the transmit signal at certain frequency,
- F_t, F_r are the pattern propagation factors of transmitter and receiver respectively,
- R_t, R_r are the transmitter to target and receiver to target ranges,
- L_t, L_r are the transmitter and receiver system losses,
- Q is the number of returns from the transmitter to the receiver.

In general, three important measurements including the Signal to Noise Ratio (SNR), signal to interference ratio and the signal to clutter ratio can be calculated based on the range equation. SNR measures the ratio between the received signal powers from the target to the noise power. Signal to interference ratio is the ratio of the received power from the target to the power of interference signals (such as the direct signal path from the transmitter to the receiver). Signal to clutter ratio measures the received power from the target to the power from other static or background clutters. These three measurements are illustrated in Eq.(2.6), (2.7) and (2.8), where P_C is the clutter power, P_J is the jammer power, R_{t-c} and R_{r-c} are the ranges from transmitter to clutter and the one from receiver to the clutter. Note that:

- σ_c is the radar cross section (RCS) of the clutter,
- K is the Boltzmanns constant, which is 1.38×10^{-23},
- T is the receiver noise temperature,
- B is the receiver noise bandwidth,

- F is the noise factor of the receiver.

$$SNR = \frac{P_r}{N} = \frac{P_t G_t G_r \sigma_T \lambda^2 F_t F_r Q}{{R_t}^2 {R_r}^2 4\pi^3 L_t L_r KTBF} \tag{2.6}$$

$$SIR = \frac{P_r}{N + P_C + P_J} \tag{2.7}$$

$$SCR = \frac{P_r}{P_c} = \frac{P_r}{\frac{P_t G_t G_r \sigma_c \lambda^2 F_t F_r Q}{{R_{t-c}}^2 {R_{r-c}}^2 4\pi^3 L_t L_r}} \tag{2.8}$$

2.1.3 Doppler Processing

The observed Doppler frequency varies based on the relative distance or velocity between the target under detection and the radar. In specific, the wavelength of the back-scattered echo in the radar receiver may decrease or increase depending on the relative velocity, which gives rise to the observed Doppler frequency.

Suppose the range between the target and the radar is R, the phase changes $\delta\phi$ between transmission and received signal depends on R and the wavelength of the carrier frequency λ_c, as shown in Eq.(2.9).

If we define R as time-dependent, the formulation can be derived in Eq.(2.10) by replacing R with a function $R(t)$. Taking the time derivative of the phase changes, the Doppler frequency is obtained in Eq.(2.10) and (2.11), with v the relative radial velocities between target and the radar.

$$\delta\phi = 2\pi\frac{2R}{\lambda_c} = \frac{4\pi R}{\lambda_c} \tag{2.9}$$

$$\omega_{Dop} = \frac{\partial \delta\phi}{\partial t} = \frac{4\pi\frac{\partial R}{\partial t}}{\lambda_c} = \frac{4\pi v}{\lambda_c} \tag{2.10}$$

$$f_{Dop} = \frac{\omega_{Dop}}{2\pi} = \frac{2v}{\lambda_c} \tag{2.11}$$

To calculate the Doppler frequency, we need to estimate the phase-time history of multiple pulses during a certain period of integration time. Suppose that the complex I-Q Doppler signal are obtained, the most direct way is to use the Discrete Fourier Transform and its faster alternative Fast Fourier Transform (FFT). Utilizing the modern digital processing tools including Analog Digital Conversion and the computer processing, Doppler processing can be performed quickly. Note that the detailed way to obtain the I-Q Doppler time domain signal will be described later in Section 2.1.4.

2.1.4 Radar Systems and the Doppler Processing

2.1.4.1 Continuous Wave (CW) Doppler Radar

CW radar is continuously transmitting the single tone waveform and receiving the echoes. The single tone waveform is not modulated, which makes it unable to detect the range but can detect the Doppler frequency described in Section 2.1.3. However, CW radar is adopted in many practical short-range scenarios as a motion detector, mainly due to its low manufacture and operating cost and continuous monitoring the environment under detection.

The architecture of the CW Doppler radar is shown in Figure 2.1. First, a signal source generates a single tone frequency waveform at f_c and a power amplifier and the transmitter antenna amplify and propagate it to the environment. In the receiver chain, first the receiver antenna picks up the returned echoes from targets and a Low Noise Amplifier (LNA) amplifies it. Then a frequency mixer compares the received echo with the clean reference signal and generates the Doppler signal with other harmonics and sideband signals. Next a properly designed Doppler filter, most commonly a bandpass filter is utilized to extract the clean Doppler signal. Conventionally, the post processing also requires the Doppler amplifier to increase the SNR. Finally, a digitiser should be utilized to generate the complex I-Q or the real number time-domain Doppler signal, ready for time-frequency analysis methods to estimate the target velocity for the radar operator.

2.1.4.2 Pulse Radar

Pulse radar is the most widely utilized system in various security control, maritime discovery and military applications. Pulse radar is able to detect both the range and Doppler information usually for medium and long-range detection therefore it requires more complex design and high power amplifier for pulse transmission.

The simplified architectures of the pulse radar is shown in Figure 2.2 and we use the superhetrodyne receiver design, which first down-converts the radio frequency to IF

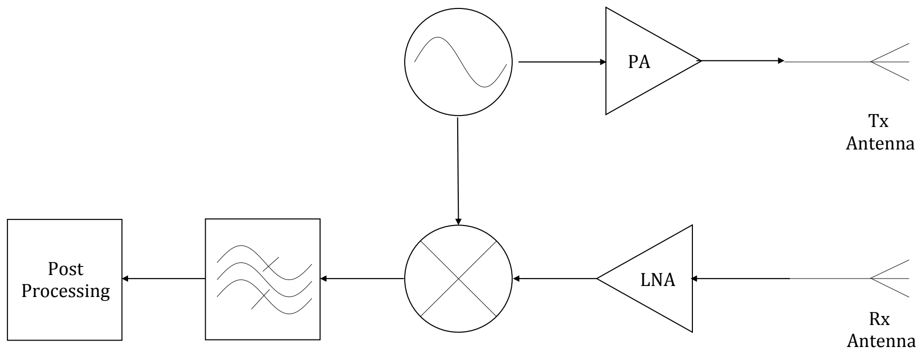


Figure 2.1: Architecture of the CW radar.

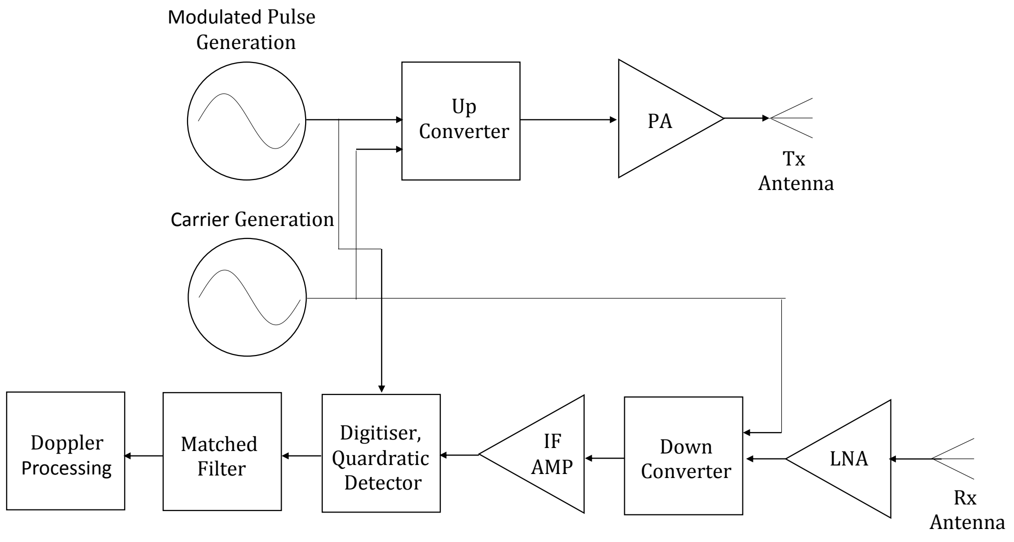


Figure 2.2: Architecture of the pulse radar.

frequency using the carrier signal. First the pulse waveform is generated and upconverted to the carrier frequency. Next, the power amplifier and the Tx antenna are utilized to amplify the pulses and propagate to the air. In the receiver chain, first the Rx antenna and the LNA amplifier are utilized to filter the backscattered echoes. Following this is the down converter with the same carrier signal and the IF amplifier is used to amplify the down-converted IF signal. Due to the powerful digital processing capability, we can directly digitise the IF signal and generates the I-Q complex number signal, ready for matched filtering processing with the pre-collected reference pulse waveforms. The output of the matched filter processing in Section 2.2.2 is the range-time intensity map and we can perform the time-frequency analysis depending on the target range under detection.

2.1.4.3 Passive Wireless Radar

Active CW and pulse radar systems are able to transmit bespoke pulse waveforms, usually with ideal properties for target detection, large bandwidths and controllable signal powers. However, in PWR system, the waveform is designed for communications which limits the performance and increases the burden of the signal processing [11]. As the waveform from the illuminator of opportunity is unknown, the most important idea in PWR is: we do not need to know the exact waveform but only to know the differences between the reflected target echo and the direct signal arrival from the transmitter to the radar receiver. Following this idea, two receiving channels are designed, termed the reference and the surveillance channels.

Ideally the reference channel should be designed to receive signals from the transmitter directly and isolate the reflected echoes from the target. On the other hand, the surveillance receiver should pick up the reflected echoes from targets, also isolating the direct arrivals from the transmitter. To achieve these criteria, high gain directional antennas with narrow beam-widths are used, pointing to the transmitter and surveillance zone respectively. In addition, the two channels are synchronized using the same reference clock so that the signals can be cross correlated to extract the range and Doppler information of the moving target. With known geolocations of the transmitter and receiver pairs, multiple detection results can be fused for final decision making. Like active radar systems, some tracking filters can be applied to the detection results for associating consecutive detections [12].

Compared with the active systems, the performance of PWR systems is limited by the following two factors: the low range resolution and the transmitter power. However, relevant solutions are also proposed in the following two points:

- **Solution to Low Range Resolution:** in general, the bandwidth of the illuminator of opportunity is limited, which gives rise to the fairly low range resolution. However, receiving antenna arrays can be utilized to provide higher angular resolution for compensation.

- **Solution to the Transmitter Power:** due to the limited power of the transmitter, the detection probability can be enhanced by increasing the integration time for CAF processing. After all, for the conventional CAF processing in PWR, longer time signals (more relevant data samples) for CAF processing increases the SNR of target echoes.

The simplified architectures of PWR is shown in Figure 2.3. It can be observed that the reference and surveillance channel design adopt the similar architecture as the one used in pulse radar receiver (see Figure 2.2). In general, the only big difference between PWR and the active pulse radar is the way to collect reference pulse waveform. Due to the fact that the transmitter of illuminator transmits the arbitrary waveform with the purpose of communication, the reference signal is collected through reference channel receiver, rather than the pre-collected and well-defined waveform samples. Note that we emphasize the SYNC module in Figure 2.3 which synchronizes i) the reference clock signal between two channels; ii) the Pulse Per Second signal of the digitisers. The SYNC module is usually developed by the MIMO cable or the Octo-Clock device with the GPS-DO signal.

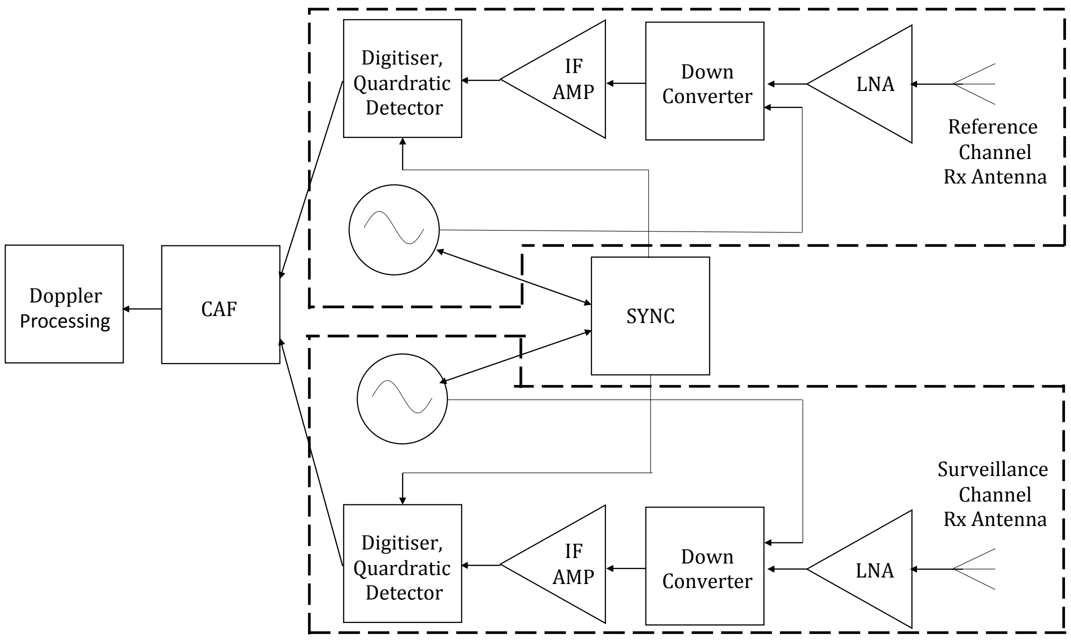

Digitiser, Quardratic Detector
IF AMP
Down Converter
LNA
Reference Channel Rx Antenna
Doppler Processing
CAF
SYNC
Surveillance Channel Rx Antenna
Digitiser, Quardratic Detector
IF AMP
Down Converter
LNA

Figure 2.3: Architecture of the PWR.

.

2.2 µ-DS Analysis for Activity Monitoring

In this section, we assume the IF signals have been ideally collected based on the introduced operations in Section 2.1. Here, we focus on the processing methods for µ-DS analysis in order to extract the useful information for activity monitoring. To monitor the activity of human targets based on radar, we introduce the most essential human µ-DS in Section 2.2.1. To generate the µ-DS from the IF signals, we review the matched filter and the time-frequency analysis in Sections 2.2.2 and 2.2.3 respectively.

In fact, the previously mentioned two steps can be replaced by the Ambiguity Function (AF) which approximately integrates the two steps into a single function. We briefly introduce AF here and describe why its variant CAF is not ideal for the µ-DS analysis in later paragraphs. The AF presents the basic waveform properties, such as the range and Doppler resolution and the ambiguity peaks in the range and Doppler side-lobes etc. A good waveform choice should have the “thumtack” shape, as there will be no correlation response unless the returned echo is closely matched with the transmitted waveform pulse. The AF is a 2-D function with two variables, the delay τ and the Doppler frequency f_d, which calculates the matched filter response using different time delays and Doppler frequencies. The AF result of simulation data using a single pulse rectangular pulse waveform is in Figure 2.4 and the AF results using experimental PWR signal is shown in Figure 2.5. In addition, we review the two important mathematical properties AF [1]:

- The maximum value of AF occurs at $(\tau, f_d) = (0,0)$.
- AF is symmetric in the sense that $AF(\tau, f_d) = AF(-\tau, -f_d)$.

Suppose that the IF signal from the radar receiver is $x_r(t)$ and the reference waveform of the pulse radar is $x_t(t)$ ($x_{ref}(t)$ for PWR). The AF calculates the spectral power of each Doppler frequency f_d and delay shift τ in the following Eq.(2.12), where $*$ indicates the complex conjugate operation.

$$AF(\tau, f_d) = \int_{-\inf}^{+\inf} e^{-j2\pi f_d t} x_t^*(t-\tau) x_t(t) dt \tag{2.12}$$

To compare the echo with the transmission waveform, AF can be extended to the

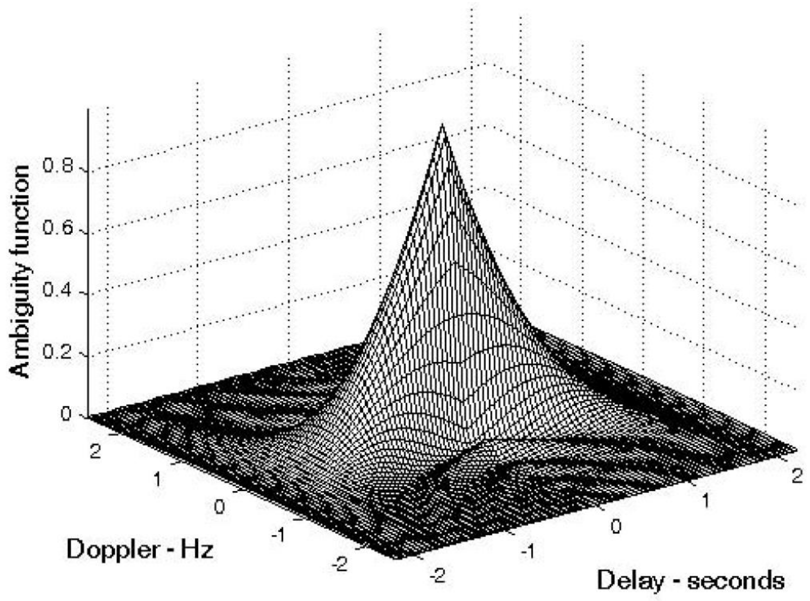


Figure 2.4: AF results using a simulated single rectangular pulse waveform of 2 second from [1].

.

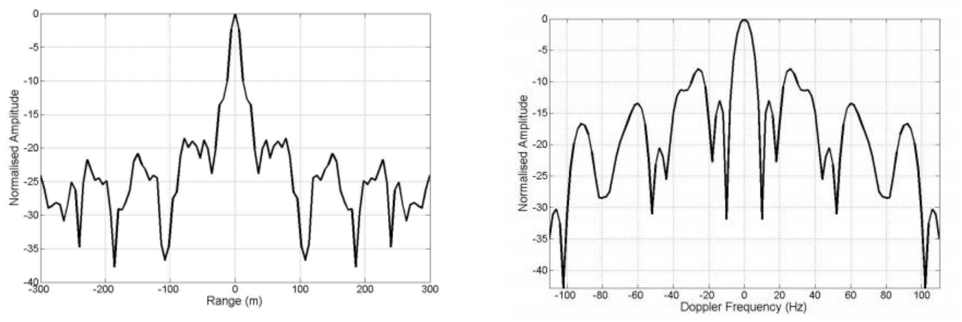


Figure 2.5: AF result using experiment data of a bi-static PWR: (a) zero Doppler cut, (b) zero range cut.

CAF [13, 11] by replacing the $x_t(t)$ by $x_r(t)$ in Eq.(2.13). We argue that although AF is a well-known tool for analysing the fundamental waveform properties [13], CAF is not ideally designed for μ-DS analysis, because adopting discrete fourier transform basis in CAF may not reveal the fine-grained μ-DS structures compared with the more advanced time-frequency analysis methods in Section 2.2.3.

$$CAF(\tau, f_d) = \int_{-\inf}^{+\inf} e^{-j2\pi f_d t} x_t^*(t-\tau)x_r(t)dt \tag{2.13}$$

2.2.1 Human μ-DS

The μ-DS is a mixture of the modulated Doppler frequencies caused by the backscattered signals from different scatters of the target object [5, 6]. As reviewed in Section 2.1.3 and Eq.(2.9) to Eq.(2.11), as the time derivative of the phase, μ-DSs are mixed due to the different velocities (instantaneous ranges) of various scatters of the human target. Highly related to the human motions, the μ-DS is a useful data to infer and classify the activities.

To detect and classify human activities, a model is necessary based on the main scattering sites of human body parts. The widely used point scatter model is used to approximate the continuous scatter sites of the human body into key scattering centres for simplification [14]. Intuitively, the greater the number of centre points chosen, the more accurate model to describe the complex motion patterns. Generally, five centre points are chosen, representing the torso centre f_{torso}, the left arm f_{arm1}, the right arm f_{arm2}, the left leg f_{leg1} and the right leg f_{leg2}. Figure 2.6 illustrates these scattering centres in a situation where a human target approaches the radar antenna. If a more complex model is utilized, such as the example in the work of Chen et al. [5, 6], the μ-DS can present more information. Figure 2.7 shows the μ-DS of a walking target simulated in X-band radar with large bandwidth of 500MHz.

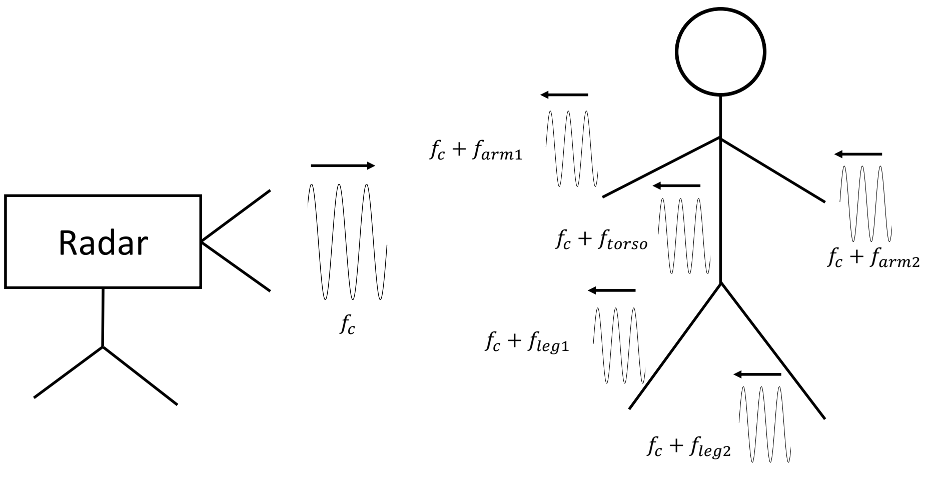


Figure 2.6: Human target point scatter model. Radar sends the waveform at the frequency of f_c; When a target walks approaching to the radar antenna, the Doppler modulated frequencies $f_c + f_{arm1}$, $f_c + f_{arm2}$, $f_c + f_{leg1}$, $f_c + f_{leg2}$ and $f_c + f_{torso}$ represent the velocities of different body parts.

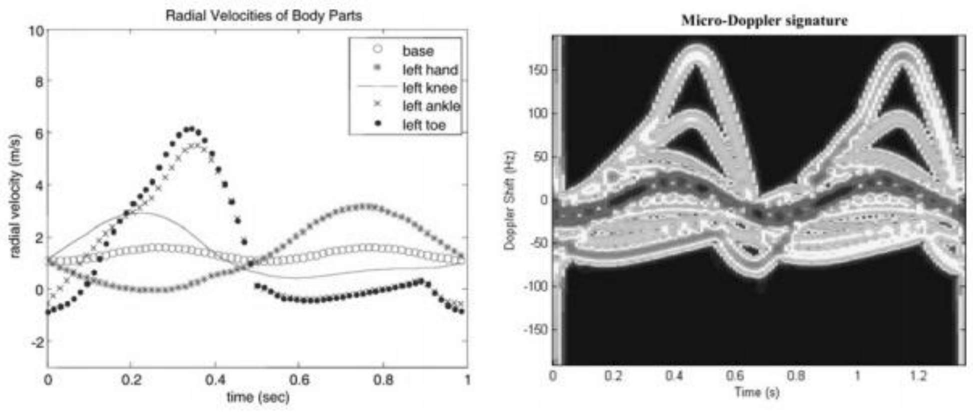


Figure 2.7: Simulated μ-DS of a walking target, from [26, 90]

.

2.2.2 Matched Filter

The filter response measures the changes in the frequency, phase and amplitude of the output compared with the input. Matched filter is a filter which generates the maximum SNR as long as the input $x_r(t)$ matches completely with the transmission waveform $x_t(t)$. Denoting the i^{th} received and transmitted pulse by $x_r^i(t)$ and $x_t^i(t)$, we can perform the matched filter operation defined in Eq.(2.14) based on cross-correlation between delayed waveform $x_t^i(t-\tau)$ and the return echo $x_r^i(t)$. The output $MF^i(\tau)$ is the matched filter response w.r.t. the delay τ for i^{th} pulse.

Assume that we have sampled N_{pulse} pulses (at the sampling rate of f_{sample}) and we only consider the maximum delay as τ_{max}, the matched filter response can be described as a matrix MF, denoted by $MF = [MF_1, MF_2, ..., MF_{N_{pulse}}]$, with each vector $MF_i \in \mathbb{C}^{N_{delay}}$ indicating the MF response of the i^{th} pulse. This matrix MF can be more clearly illustrated in the Figure 2.9 (a).

$$MF(\tau) = \int_0^{T_{pulse}} x_t^*(t-\tau)x_r(t)dt \tag{2.14}$$

Next, we introduce the concept of slow-time and fast-time in radar signal processing, which are useful to understand the μ-DS and time-frequency analysis. Generally speaking, the time within a single pulse sampled at f_{sample} is referred to the "fast-time" while the time between pulses sampled by PRF is the "slow-time". Figure 2.8 (a) illus-

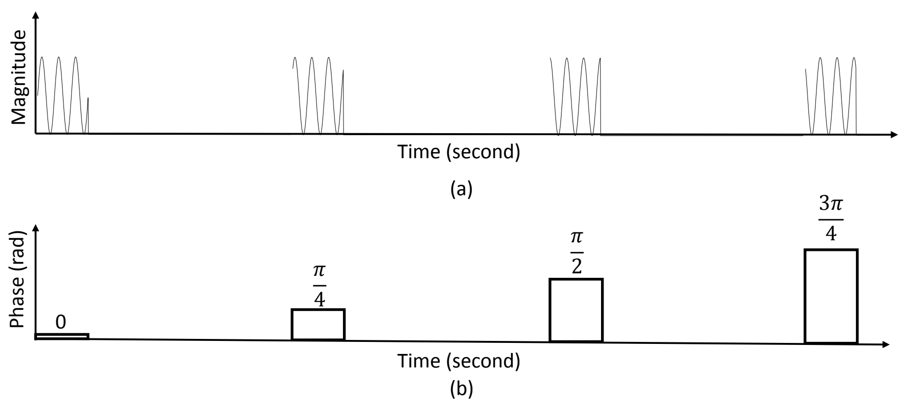


Figure 2.8: (a) Fast-time four pulses modulated by Doppler frequency. (b) The phases of Doppler frequency within the four pulses.

.

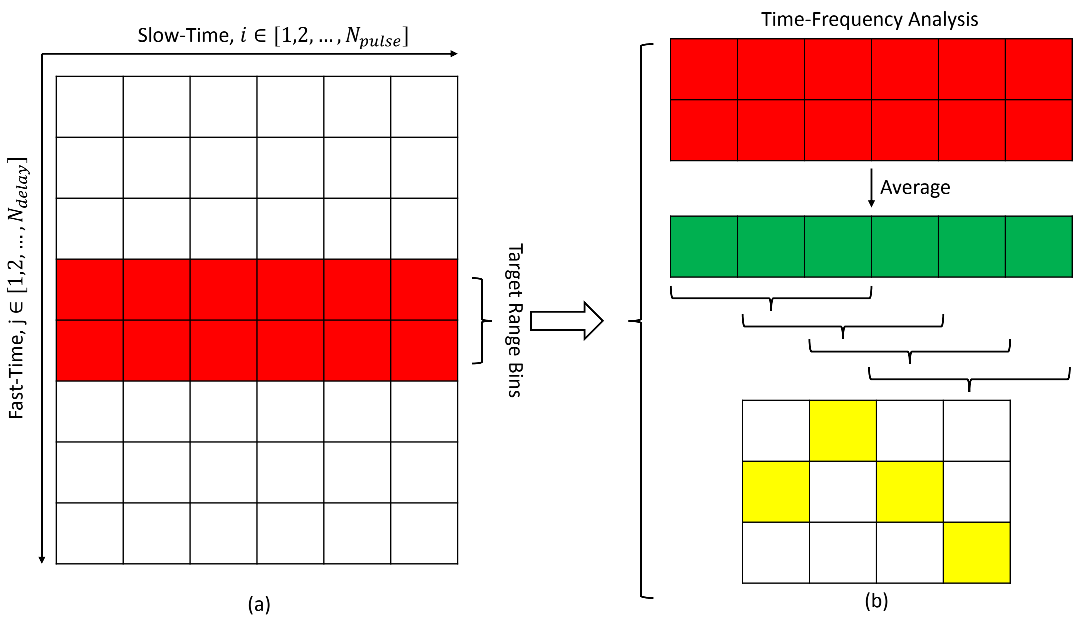

Slow-Time, $i \in [1,2,\dots,N_{pulse}]$
Fast-Time, $j \in [1,2,\dots,N_{delay}]$
Target Range Bins
Time-Frequency Analysis
Average
(a)
(b)

Figure 2.9: (a) Matched filter response matrix MF, the column index $j \in [1,2,...,N_{delay}]$ indicates the fast time index; the row index $i \in [1,2,...,N_{pulse}]$ indicates the slow time index. (b) Time-Frequency analysis, taking the range-time history from (a) to obtain the frequency-time history responses; green vector indicates the averaged range-time history at target's range bins.

.

trates the example of four pulses, modulated by the Doppler frequency $f_{dop} = \frac{PRF}{8}$ and Figure 2.8 (b) plot the phase of the relevant matched filter responses. Note that these measures are all in fast-time domain.

2.2.3 Time-Frequency Analysis

Suppose that we obtain the matrix MF in Figure 2.9(a) after the matched filter processing. First the constant false alarm rate (CFAR) method [15] is applied to identify the target range bins or the delay time index $j \in [1,2,...,N_{delay}]$. Since this sub-section is focused on time-frequency analysis, we assume that we can obtain the range-time signal related to the target's activity, as highlighted in red in Figure 2.9(a). We denote this signal as the matrix $MF_{target} = [MF_{target,1},...,MF_{target,N_{pulse}}]$, with i^{th} column $MF_{target,i} \in \mathbb{C}^{N_{target}}$, with N_{target} the number of active range bins where the target is. Conventional methods take average of MF_{target} along the range bins and obtain the time-domain signal $x_{MF} \in \mathbb{C}^{N_{pulse}}$ as illustrated in Figure 2.9(b).

Time-frequency analysis aims to transform the time-domain signal to the joint time-frequency feature space; the Short Time Fourier Transform (STFT) [16] is the most widely used while others include but are not limited to the Wigner Ville Distribution [17] and Empirical Mode Decomposition [18, 19].

STFT uses the sliding window on time domain signals x_{MF} and the FFT operation on each window to generate the frequency features. This operation is formally defined in the Eq.(2.15) and illustrated in Figure 2.9(b). First a window function $h(u)$ is applied to extract the signal atoms in the neighborhood of the time t. Next the Fourier transform is performed on the extracted atoms to generate the frequency responses at the specific time t and $x(u)h^*(u-t)$ is defined as the atomic element. The performance of STFT depends on the window length, choice of window function and the overlapping length when sliding the window.

$$JTF(t,f;h) = \int_{-\infty}^{+\infty} x_{MF}(u)h^*(u-t)e^{-j2\pi fu}du \tag{2.15}$$

Wigner Ville Distribution utilizes the auto-correlation between signals with different delays and transform the time domain signal into the instantaneous frequency space by FFT. Empirical Mode Decomposition is aimed to decompose the mixed signal into different frequency atoms using the Hilbert-Huang Transform, which is equivalent to extract signals using different filter banks [18]. In sum, they all provide the joint time-frequency features $JTF(t,f;h)$ for the feature exaction and classification. Since we are focused on machine learning methods for µ-DS, we adopt the most widely used STFT for generating all the µ-DS in this thesis.

2.2.4 µ-DS Classification

Although the µ-DS $JTF(t,f;h)$ in Eq.(2.15) contains the fundamental features to infer the activities, they may not be the optimal one for activity classification. This sub-section focuses on the feature extraction methods of the µ-DS and the classification methods. Suppose that we have accessed the µ-DS $JTF(t,f;h)$ and denote its discrete data sample as $x_{\mu D} \in \mathbb{R}^{N_{freq} \times N_{time}}$, where N_{freq} and N_{time} are the frequency and time bins we are interested in. Since we adopt the most commonly used STFT processing, we only review the following feature extraction and classification methods based on the matrix $x_{\mu D}$.

2.2.4.1 Feature Extraction

A large number of features have been proposed and the ones based on the STFT outputs can be divided into the following three categories.

- **Hand-Crafted Features:** these features include the fundamental features directly related to the human motions, for example, the maximum velocity of the movement, the time width of the µ-DS, the RCS values etc.

- **Component Analysis Features:** these features are based on the statistical component analysis, such as Principle Component Analysis (PCA) and Singular Value Decomposition (SVD) of the spectrogram matrix. Then the useful statistics like mean average and standard deviation values of the projected data are regarded as useful features. These methods aim to eliminate the noise and project data vectors to different orthogonal subspaces or transformed into statistically independent vectors.

- **Representation Learning Features:** without identifying specific feature extraction operations, the whole spectrogram matrix is regarded as the only input and we aim to learn the optimal features in an end-to-end learning framework like DCNN, which automatically learns the feature and classifier within a unified optimization function.

2.2.4.2 Classification Methods

Various classifiers in the machine learning community have been investigated for µ-DS classification which can be summarized into three basic categories: distance, representation and the statistical model based classifier.

- **Distance Measurement Based Classifier** Nearest Neighborhood (NN) [20] and Support Vector Machine (SVM) [21] are well-known examples in this category, which aim to measure the distance between test samples with the labelled training samples. The distance varies in both the Kernel space and the raw feature space.

- **Representation Learning Based Classifier** DCNN aims to represent the raw data by non-linear or linear combination of different layers, with the final layer's output the probability vector of the predicted classes. Besides the hierarchical and complex representations learned by DCNN, another popular classifier is the SRC [22], which aims to extract the sparse representation of the test sample using linear combination of training data. The SRC classifier picks up the class with the minimum representation residue.

- **Statistical Model based Classifier** Naive Bayesian classifiers [20] and the Gaussian Mixture Model [23] are the well-known classifiers in this category. They both utilize the principle of maximum likelihood to make decisions according to their probability density function fitted by the training data. The density function of Gaussian Mixture Model is always estimated using the Expectation Maximization algorithms while for the Naive Bayesian classifier, the density function is assumed to follow the Gaussian distribution. In this way, the mean and standard deviation of the Gaussian distribution can be estimated easily based on the training data.

2.3 Sparse Coding and Dictionary Learning

The past decades witnessed both the theoretical developments and applications of sparsity prior in machine learning, signal processing and computer vision communities [24, 25, 26]. Sparsity in the data representation encourages that only a few elements of the representation is non-zero. Rather than using the observed sensor data which are complex, redundant and even noisy, the sparse representation is efficient and flexible, concise and robust to noise [24]. In this section, we review the sparse coding and DL methods required to generate the sparse representation, which may help understand contributions when applying them in μ-DS classification.

Sensor signals are dense and we tend to represent them under a basis (decompose the sensor data under a set of basis) so that the more concise and sparse representations are obtained for advanced applications. More formally based on the sparse coding theory [24] and the Eq.(2.16), given the observed signal or data $y \in \mathbb{R}^M$ and the dictionary or basis $D \in \mathbb{R}^{M \times N}$, sparse coding methods are aimed to find the maximally sparse code $s \in \mathbb{R}^N$ as the representation of y, leveraging the linear combination of the atoms in D. Here, $||s||_0$ indicates the number of non-zero elements in s. In the context of DL, we are aimed to not only find the sparse code but also learn the optimal dictionary D based on the following Eq.(2.17), where ε is the upper bounds of the representation error.

Solving the Eq.(2.16) and 2.17 is NP-hard [24], as searching the best sparse code s with the least non-zero elements is exhaustive, not to say when its dimension is large. Therefore, some methods have been proposed for either relaxing Eq.(2.16) or aggressively obtaining an approximate but tractable solution. We review some existing methods in the following sub-sections and discuss the usage in μ-DS classification.

$$s = arg\min ||s||_0 \;\; s.t. \;\; y = Ds \tag{2.16}$$

$$s, D = arg\min_{s,D} ||s||_0 \;\; s.t. \;\; ||y - Ds||_2^2 \leq \varepsilon \tag{2.17}$$

2.3.1 Sparse Coding

This section only focuses on reviewing the convex optimzation based and the greedy methods to solve Eq.(2.16) and discuss their applications in μ-DS classification. Convex optimization methods use ℓ_1 norm penalty as the replacement of Eq.(2.16) but greedy methods aim to solve the ℓ_0 norm penalty in an aggressive manner. Besides these two categories, methods developed also include concave optimization, Baysian approaches etc; interested readers are suggested to find out more details in [27, 28, 29, 29, 30].

2.3.1.1 Convex Optimization based Methods

In the convex optimization methods, the best approximation of Eq.(2.16) utilizes the ℓ_1 norm penalty, as shown in Eq.(2.18) [31]. This approximation is beneficial because it guarantees the global minimum solution rather than the local minimums. Note that the community also refer to this ℓ_1 norm based methods as the Basis Pursuit methods [24] and we refer the formulation of Eq.(2.18) the Basis Pursuit De-Noising. To further guarantee the equivalence between the ℓ_0 and ℓ_1 norm formulations, the theoretical development called restricted isometry properties [32, 33] is made and this is shown in Appendix A.1. Note that the restricted isometry properties measure the worst situation of the sparse coding problem, however in practice, Basis Pursuit based algorithm works well in many engineering applications.

In addition, the formulation of the Least Absolute Shrinkage and Selection Operator [31] has been proposed in Eq.(2.19) where ϑ is the upper bound of the sparsity penalty. Note that this can be equivalently derived to the well-known formulation without constraints in Eq.(2.20).

$$s = arg\min_{s} ||s||_1 \;\; s.t. \quad ||y - Ds||_2^2 \leq \varepsilon \tag{2.18}$$

$$s = arg\min_{s} ||y - Ds||_2^2 \;\; s.t. \quad ||s||_1 \leq \vartheta \tag{2.19}$$

$$s = arg\min_{s} ||y - Ds||_2^2 + \lambda ||s||_1 \tag{2.20}$$

Under the regime of convex optimization based sparse coding methods, popular methods based on the gradient descent have been proposed and adopted due to the fast yet accurate implementation and we also refer them to ℓ_1-solvers. These solvers are efficient since they are designed for specific optimization problems. These efficient ℓ_1 minimization solvers include, but are not limited to, IST[34], SpaRSA[25], TwIST [35] and GPSR[36], the Fast ISTA [37] and the Learned ISTA (LISTA) algorithms [38] based on learning the parameters in the ISTA. Since the work in Section 3.5.6 is inspired by LISTA algorithm, we mainly review the most relevant developments in the following:

- **ISTA:** ISTA algorithm aims to calculate the optimal s_{t+1} at the $(t+1)^{th}$ iteration based on the conventional gradient descent algorithm in Eq.(2.21). For convenience, we denote the total loss function as $F(s) = f(s) + g(s)$, composed of the smooth sub-function $f(s) = \|y - Ds\|_2^2$ and the non-smooth part $g(s) = \|s\|_1$, with the learning rate γ. Note that ISTA requires that both f and g are convex and g is known and separable. Besides the black-box algorithms in Eq.(2.21), ISTA uses a more efficient algorithm by optimizing the proximal term in Eq.(2.22), where $\frac{1}{\eta} = L(f)$ and $L(f)$ indicates that the function f is $L(f)$-smooth. The solution s_{t+1} can be derived in Eq.(2.23) and (2.24), where the function $h_{\eta\lambda}(\cdot)$ is the element-wise shrinkage function based on $g(s)$ and the threshold is defined as $\eta\lambda$. The ISTA framework can be more intuitively explained in Figure 2.10(a) with the function H defined as $H = I - \eta D^T D$. Additionally, since the $g(s)$ is a simple and element-wise separable function, we can further decompose the Eq.(2.24) to N 1-dimensional problems assuming that $s \in \mathbb{R}^N$.

$$s_{t+1} = s_t + \gamma \nabla F(s_t) \tag{2.21}$$

$$s_{t+1} = arg \min_{s \in \mathbb{R}^N} f(s_t) + \nabla f(s_t)^T (s - s_t) + \frac{1}{2\eta} \|s - s_t\|_2^2 + g(s) \tag{2.22}$$

$$s_{t+1} = arg \min_{s \in \mathbb{R}^N} \frac{1}{2\eta} \|s - (s_t - \eta \nabla f(s_t))\|_2^2 + g(s) \tag{2.23}$$

$$s_{t+1} = h_{\theta = \eta\lambda}(s_t + \eta D^T (y - Ds_t)) \tag{2.24}$$

- **LISTA:** LISTA algorithm aims to solve the problem in Eq.(2.19) more efficiently by learning the parameters of a feed-forward network in the Figure 2.10(b). Compared with ISTA, LISTA can be viewed as a time-unfolded recurrent neural network and the example in Figure 2.10(b) unfolds the three iterations in ISTA algorithm with learnable parameters. Note that the back-propagation (see Section 2.4.1.6) operation is used for optimization and the parameters under estimate include the dictionary D, functional matrix H and the threshold $\theta = \eta\lambda$. Since LISTA is originally applied to solve the sparse code s, the loss function L_{lista} is the mean square error between the ground-truth sparse code s_{opt} and the estimated one $\hat{s}$ in Eq.(2.25).

$$L_{lista} = \|s_{opt} - \hat{s}\|_2^2 \tag{2.25}$$

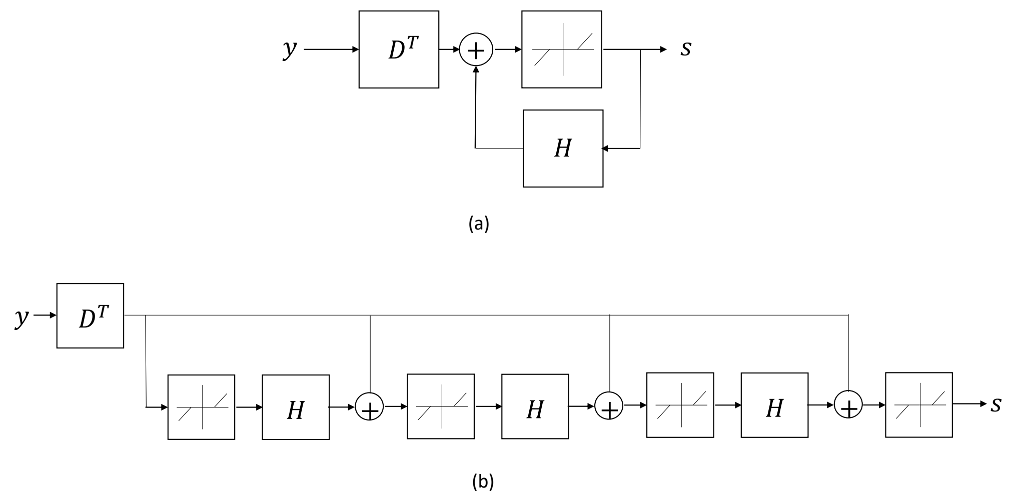


Figure 2.10: (a) ISTA framework to calculate sparse coding s of the input y using dictionary D and linear operator H. (b) LISTA framework to calculate the sparse code s by unfolding the iterative methods in ISTA to recurrent units.

.

2.3.1.2 Greedy Methods

Instead of replacing the ℓ_0 norm with the ℓ_1 norm penalty, greedy algorithms, also referred to matching pursuit (MP) [39], attempt to minimize the ℓ_0 norm by selecting the active support set of the sparse code s in an iterative and aggressive manner. A large number of MP algorithms have been proposed including but are not limited to MP, Orthogonal MP (OMP) [40, 41, 42] and its variants and subspace pursuit [43] etc.

Without considering the gradient and convexity, MP algorithms aim to select the best *active support set* of s_{act} in the *current* step so that the representations $D \times s_{act}$ best matches the original observation y. Although MP based methods usually provide less accurate and worse estimations compared with the Basis Pursuit counterpart, it is advantageous because of its easy implementation and light-weight computational load. It is usually preferred by many engineering applications when computation resources are limited or the reconstruction accuracy $||y - Ds||_2^2$ is not the critical requirement. More specifically, the MP algorithm can be illustrated in Eq.(2.26), where $|\Omega|$ indicates the length of the index set, $\Omega \subset [1, 2, ..., N]$ and N_s is the maximally allowed active support number or the sparsity level of s. Conventionally, the required N_s is fixed based on the applications and MP methods select the best support index $i \in \Omega$ based on the highest correlation results between the atom D_i and the residual r in the *current* step. However, variants of MP differ in searching for and selecting the optimal support set Ω.

$$s, \Omega = arg \min_{s,|\Omega|=N_s} \|y - \sum_{i\in\Omega} D_i s_i\|_2^2 \tag{2.26}$$

- **MP:** MP algorithm is the simplest type of greedy methods. It first selects the best-matching dictionary column/atom D_i to the current residual r_k at k^{th} iteration by finding out the highest correlation results. Then it also uses this correlation result as the i^{th} support value s_i. The pseudo-code is shown in the Appendix A.2 Algorithm 1.

- **OMP:** The OMP algorithm adopts the same way to select the active atoms as MP, but differs when calculating the support value and the residual at each iteration. OMP ensures that the same atom in the dictionary cannot be selected twice, as the residual r_k at the k^{th} iteration is always orthogonal to the selected columns of dictionary D. Specifically, OMP recognizes and adds a new atom to the active support set in each iteration and calculates the relevant support values of the set by minimizing the reconstruction error. The details of the algorithm can be found in Appendix A.2 Algorithm 2. Many variants of the OMP algorithms have been proposed, including StOMP [44], CoSaMP[45] and the Subspace Matching Pursuit (see details in Appendix A.2 Algorithm 3) [43].

2.3.1.3 SRC for µ-DS classification

ℓ_1-solver and the greedy methods can be applied directly to µ-DS classification, leveraging the robust sparse representation of the µ-DS data. This is because the more compact representation of the data is i) robust to noise and ii) intrinsically discriminative [22], which are suitable for classification problems. Wright et al. [22] may be the first to adopt SRC method for robust classification of facial images. In this subsection, we apply the SRC in the context of classifying µ-DS data.

The core idea of the SRC is that we do not use conventional dictionary, such as the Fourier basis, but utilize the labelled training µ-DS samples as the dictionary D. Suppose that the dictionary $D \in \mathbb{R}^{M \times N}$ is composed of N µ-DS samples, within C classes, denoted as $[x_1, x_2, ..., x_N] = [D_1, D_2, ..., D_C]$. SRC first represents the test µ-DS data, denoted as y as the linear combination of the training sample dictionary D constrained by moderate sparsity level. Secondly the test sample is assigned to the label c if and only if the residual r_c using all atoms from the class c is the smallest. More specifically, the SRC can be interpreted as two steps, including sparse coding and classification.

- **Sparse Coding:** The sparse representation of y shown in Eq.(2.27) and 2.28 can be obtained using the solvers introduced in Section 2.3.1.1 and 2.3.1.2.

$$s = arg \min_{s} ||y - Ds||_2^2 + \lambda ||s||_1 \tag{2.27}$$

$$s = arg \min_{s} ||y - Ds||_2^2 + \lambda ||s||_0 \tag{2.28}$$

- **Classification:** After the optimal s is estimated, the classification step assigns the class c to the test sample y by selecting the best class c_* to reconstruct the y with the minimum residual, as shown in Eq.(2.29). Note that D_c refers to the atoms in class c and s_c indicates the support values of these atoms.

$$c_* = arg \min_{c} ||y - D_c s_c||_2^2 \tag{2.29}$$

2.3.2 Dictionary Learning

Selection of dictionary plays an important role in sparse coding and the classification performance. To learn the optimal dictionary in the context of classification, we review the methods to learn a discriminative dictionary from the training data.

Based on the formulation of SRC in Eq.(2.27), (2.28) and (2.26), we may observe that solving the sparse code s under a fixed dictionary D is constrained by both the reconstruction error $||y - Ds||_2^2$ and the sparsity level penalty $||s||_1$. Balancing these two terms and obtaining the optimal trade-off sometimes is limited and may not be beneficial for classification. However, if we learn the dictionary D, the burden of solving the optimal sparse code s is reduced and the classification performances can be further improved.

$$s,D = arg\min_{s,D} ||Y - DS||_2^2 \;\; s.t. \;\; \begin{cases} f_{Prior}(D) \\ g_{Prior}(S) \\ ||s_i||_1 \leq \eta \quad \forall i \end{cases} \tag{2.30}$$

Facing the classification task, DL usually adopts the ideas of incorporating discriminative prior knowledge and the structural constraints when updating the dictionary D and the sparse code s simultaneously. A unified formulation of DL is in the Eq.(2.30), where training samples are Y and the dictionary is D, with the sparse codes denoted as S. Besides the sparsity prior, the $f_{Prior}(D)$ and $g_{Prior}(S)$ are additional constraints which may further increase the discrimination capability and improve the classification performance in the following two ways:

- **Constraints on the dictionary D:** Dictionary can be learned constrained by different structural priors. For example, we may directly divide the dictionary D to multiple sub-dictionaries $D_c, c \in [1,2,...,C]$ [46]. In addition, further constraints can be integrated to increase the incoherence between sub-dictionaries from different classes, which has been proved to improve the classification results [47].

  Although we have neither developed nor applied DL methods in this thesis, we are inspired by DL when designing the DLL layer in DTN Section 3.5.6. Therefore we discuss the connections of DL with the DCNN in advance, emphasizing our arguments that the whole DCNN design can be interpreted as a special prior knowledge for learning the dictionary (network weights) D. The ultimate goal of DCNN's layer-wise functions, including the nonlinear active functions, convolution layers and the FC layers, is to learn a unified function projecting from the observation data Y to the predicted label embedding S. Learning the DCNN parameters performs in the similar way of updating the dictionary D. However, the DCNN applies special prior knowledge on the dictionary D, for example, they used hierarchical prior knowledge and the whole parameter set is layer-wised. In addition, they incorporate some non-linear properties on the dictionary D as well.

- **Constraints on the sparse code S:** Since sparse codes S in the SRC framework directly indicate the class information, we may enforce the S to be discriminative based on some criterion: for example, smaller intra-class distance between S from the same class and larger inter-class distance between the sparse codes from different classes in FDDL[48], LC-KSVD [49] and the SDDL [50].

2.4 Deep Convolutional Neural Network

As discussed in the Section 2.3.2, success of DCNN witness the superior performance when incorporating hierarchical prior knowledge in network design by stacking multiple linear and non-linear functions. In this section, we first review the fundamental layers and operations of the Convolutional Neural Network (CNN) in Section 2.4.1. Next, useful network training strategies are reviewed in Section 2.4.2. Finally we introduce some successful developments of the DCNN architectures in Section 2.4.3.

2.4.1 Convolutional Neural Network

CNNs are made up of 3 essential layers: Convolutional layers (Conv), FC layers (FC) and the Pooling layers (Pool). We first review their design and the mechanism. Next, to increase the network capacity, different non-linear activation functions are reviewed. Finally, we review the loss function and optimization method in CNN.

2.4.1.1 Fully Connected Layers

The FC layer uses linear matrix multiplication and vector summation, with the following tunable parameters: weights $W \in \mathbb{R}^{M \times N}$ and the bias $b \in N \times 1$. The basic operation is shown in the Eq.(2.31), where $x \in \mathbb{R}^{M \times 1}$ is the input to the FC layer and $y \in \mathbb{R}^{N \times 1}$ is the output. As no spatial information and prior knowledge is contained in the FC layer weight matrix W, it is usually utilized to transform the low-level local features, such as edge and blob features to the higher-level representations.

In CNN, we usually measure the spatial extent of the connectivity between each neuron and the input data. Therefore we introduce a hyper-parameter called receptive field *RecF* [51] to describe such spatial extent. This can also be regarded as the spatial size of the input under consideration for calculating higher-level layers' responses. Note that the receptive field $RecF$ of the FC layer, is the size of the whole input.

$$y = f(x) = Wx + b \tag{2.31}$$

2.4.1.2 Convolution Layers

To extract the local features preserving the spatial information of the sensor data, Conv layers are widely utilized. After all, it is extremely memory and processor intensive to process the high-dimensional sensor input using FC layers, as they are fully connecting each input element directly to each element of the output.

However, utilizing the local filters design and the window sliding operations, Conv layers are feasible and tractable. The basic Conv layer operation is illustrated in Figure 2.11, where the input sensor data is $x \in \mathbb{R}^{H_{in}(height) \times W_{in}(width) \times C_{in}(channel)}$ and the i^{th} local filter under optimization k_i, $i \in [1, 2, ..., C_{out}]$ is convoluted with the input x.

Note that the i^{th} kernel filter k_i is of the size $k_i \in \mathbb{R}^{h \times w \times C_{in}}$ and the output size is $[H_{out}(height) \times W_{out}(width) \times C_{out}(channel)]$. Mathematically, the output response of a Conv layer is defined in Eq. 2.32 [52].

$$y(i,j) = x(i,j) * k(m-i, n-j) = \sum_m \sum_n x(i,j) \times k(m-i, n-j) \tag{2.32}$$

2.4.1.3 Pooling Layers

The Conv layers are memory efficient, due to the sliding window operation. However, the output is still translation-variant and sensitive to the spatial local distortions [53]. To alleviate this problem, the pooling layer is inserted between consecutive Conv layers, as an non-linear sub-sampling function. The basic operation can be shown in Figure 2.12 where the input is a 4×4 matrix, the pooling window size is 2×2 and with the sliding window strides of 2 both horizontally and vertically.

In the pooling operation, the stride indicates the step size of the sliding window and it is applied in all-channel responses. Two well-known methods are max-pooling and average-pooling [54]. The max-pooling mechanism by preserving the maximum number

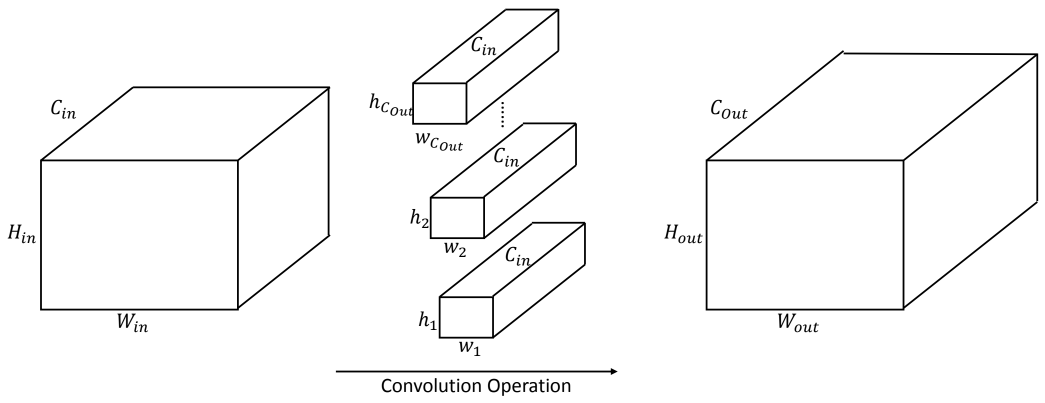


Figure 2.11: Convolutions in Conv layer: input feature map with the size of $H_{in} \times W_{in} \times C_{in}$ is convoluted by C_{out} filters with the size of $h_i \times w_i \times C_{in}$, where $i \in [1, 2, ..., C_{out}]$. The output is with the size of $H_{out} \times W_{out} \times C_{out}$.

.

and deleting all others in each sliding window operation. In addition to the reduced feature dimension after the pooling layer, it also functions to reduce the parameters under estimation.

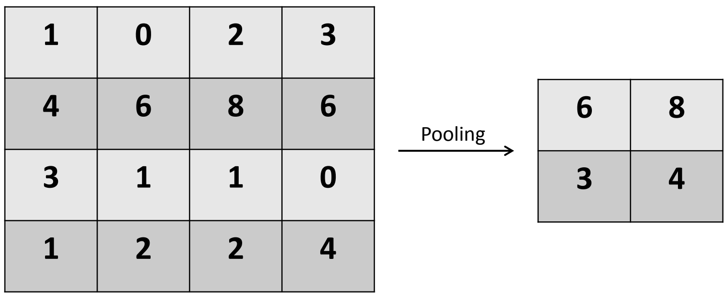


Figure 2.12: Max-pooling operations.

.

2.4.1.4 Activation Functions

Activation function is essential in modelling the DCNN since it increases the representation capability by adding the non-linearity to the features. After all, without the non-linearity, cascading multiple linear functions is equivalent to modelling using a single linear function. Activation function comprises two categories: **i)** the piece-wise non-linear function including ReLU [55, 56], Leaky ReLU [57] and the Exponential Linear Unit [58]; **ii)** the smooth functions, including sigmoid and tanh as shown in Figure 2.13.

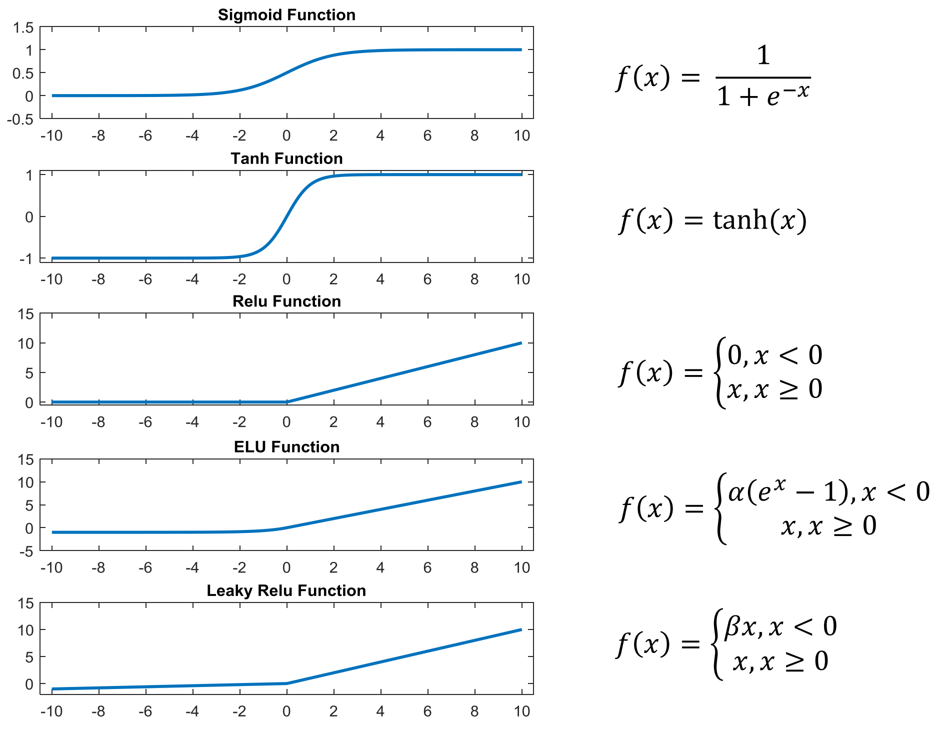


Figure 2.13: Activation functions in DCNN: input is $x \in \mathbb{R}$ and output is $f(x)$

.

In early CNN research, smooth functions are widely adopted due to the existing gradients within all range of input values and the fact that they limit the output values within a specific range. Despite these advantages, inserting them in DCNN is problematic as the gradients are small over a large portion of its input range (also termed as the neural saturation problem). In addition, stacking multiple layers tends to generate vanishing gradients when updating low-level layers. To tackle this, the piece-wise non-linear functions are proposed in the Eq.(2.33). Among these, the most widely used is the ReLU function, which has proved its superior efficiency and robust classification performance in [59]. ReLU does not suffer the neural saturation problem mainly due to its linear function design. Additionally ReLU also reduces the computational complexity since all the negative parts are assigned as zeros. Other advanced variants including the Leaky ReLU and the Exponential Linear Unit functions differ from ReLU only when the input is negative. The designs for the negative input are smoothly changing the function values close to zero rather than directly assigning zero to any negative inputs.

$$f(x) = \max(0, x), \tag{2.33}$$

Besides the activation functions introduced above, softmax function is also widely used for projecting the predicted features $x \in \mathbb{R}^{N_c}$ to the discrete multi-nomial distribution $y \in \mathbb{R}^{N_c}$ with $y_i = P(Y = y_i)$ and $i \in [1, 2, ..., N_c]$ in Eq.(2.34). Softmax gives the following ideal properties for classification task: the sum of the output vector y is one, with each element positive. This enables the output as the predicted N_c-class label under comparison with the ground-truth one-hot labels.

$$y_i = softmax(x) = \frac{exp(y_i)}{\sum_{i=1}^{N_c} exp(y_i)}, \tag{2.34}$$

2.4.1.5 Loss Function

As we mainly tackle the supervised learning problem, only the relevant classification loss function in DCNN is reviewed. Given the predicted label $\hat{y}$ and the ground-truth label y, the conventional operation is to minimize the cross-entropy [60] loss L_{CE} in the Eq.(2.35) for network training. This function is actually measuring the discrepancy between the predicted label distribution with the ground-truth one which can be derived using the definitions of entropy and the KL divergence shown in Eq.(2.36) and (2.37).

$$L_{CE} = \sum_{i=1}^{N_c} -y_i log(\hat{y}_i) \tag{2.35}$$

$$H(y) = H(y_1, y_2, ..., y_{N_c}) = -\sum_i y_i log(y_i) \tag{2.36}$$

$$\begin{aligned} D_{KL}(y||\hat{y}) &= \sum_i^{N_c} y_i log(\frac{y_i}{\hat{y}_i}) \\ &= \textstyle\sum_i^{N_c} y_i log(y_i) - y_i log(\hat{y}_i) \\ &= \textstyle\sum_i^{N_c} -H(y) + L_{CE} \end{aligned} \tag{2.37}$$

2.4.1.6 Optimization in DCNN

Due to the non-linear activation functions between layers, optimization of DCNN is not tractable and there is no guarantee for the global minima. The most widely used method is the Stochastic Gradient Descent (SGD) for minimizing the loss functions defined in the Section 2.4.1.5.

The idea of SGD can be formalized as follows: suppose that we are given N_t training samples x^i with their labels y^i defined in the set $\{(x^i, y^i), i \in [1, 2, ..., N_t]\}$, we can randomly choose a subset $\{(x^i, y^i), i \in [1, 2, ..., M]\}$, including M samples and term it a mini-batch. SGD methods then perform gradient descent based on the randomly selected mini-batch per iteration. More specifically, SGD based algorithm comprises the following two steps:

- **Gradient Computation:** suppose the DCNN is denoted as a unified function f_θ with parameters θ, the gradient of the loss function to the parameters θ is shown in Eq.(2.38):
$$\frac{\partial L}{\partial \theta} = \frac{1}{M} \sum_{i=1}^{M} \frac{\partial L(x^i, y^i)}{\partial \theta} \tag{2.38}$$

- **Weights Update:** this step updates the network weights following the direction to minimize the loss function L, as shown in the Eq.(2.39), with γ the learning rate.
$$\theta \leftarrow \theta - \gamma \frac{\partial L}{\partial \theta} \tag{2.39}$$

To update multiple layers' weights, we first reformulate the N_l-layer DCNN function f_θ as $f_\theta = f_{N_l} \circ f_{N_l - 1} \circ f_2 \circ f_1$ in the compound way in Eq.(2.40). Next, given the network input x and the i^{th} layer function f_i, the back-prop is proposed to update network weights efficiently by computing the layer-wise gradients w.r.t the final loss function based on the chain rule [61, 62]. More specifically, gradients w.r.t. weights of the i^{th} layer $\frac{\partial L}{\partial \theta_i}$ can be computed in the Eq.(2.41) and (2.42), where $x_j^{(i)}$ indicates the j^{th} element of the i^{th} layer's output. We introduce a concrete example to illustrate the whole optimization procedure in Appendix B.1.

$$\begin{aligned} f_\theta(x) &= f_{N_l} \circ f_{N_l-1} \circ f_2 \circ f_1(x) \\ &= f_{N_l}(f_{N_l-1}(...(f_2(f_1(x))))) \end{aligned} \tag{2.40}$$

$$\frac{\partial L}{\partial \theta_i} = \frac{\partial x_j^{(i)}}{\partial \theta_i}^T \frac{\partial L}{\partial x_j^{(i)}} \tag{2.41}$$

$$\frac{\partial L}{\partial x_j^{(i)}} = \frac{\partial L}{\partial x^{(i+1)}}^T \frac{\partial x^{(i+1)}}{\partial x_j^{(i)}} \tag{2.42}$$

2.4.2 Network Training Strategies in DCNN

In DCNN, some training strategies and methods are proposed for efficient and robust optimization. In this section, we review the weight initialization methods and the dropout operations.

2.4.2.1 Weight Initialization

Weight initialization is essential as it strongly influences the final optimisation result. This is mainly because the gradients of the network weights in DCNN either vanishes or explodes easily. This problem and its potential reasons are illustrated clearly in Eq.(2.40), by assuming the identical layer-wised functions $f_{N_l} \approx f_{N_l-1} \approx, ..., \approx f_1 \approx \beta$ in Eq.(2.43). Note that in this simplified case, β is a scaling scalar for the purpose of illustration.

$$\begin{aligned} f_\theta(x) &= f_{N_l} \circ f_{N_l-1} \circ f_2 \circ f_1(x) \\ &= x\beta^{N_l} \end{aligned} \tag{2.43}$$

Obviously, depending on the selection of β, the final unified function $f_\theta(x)$ will approach to either zero or the limit when increasing the number of layers, as shown in Eq.(2.44). To tackle this problem, the most intuitive solution is to initialize the network weights according to the zero mean Gaussian distribution with the standard deviation σ, so that the expectation $\mathbb{E}(\beta)$ is close to 1. Let us further denote the layer's input and output number are denoted as n_{in} and n_{out} respectively. Conventionally, if the activation function is the tanh, we use the Xavier initialization for σ in Eq.(2.45) [55]. If the ReLU is used, the widely used initializer is based on Eq.(2.46) [63].

$$\lim_{N_l \to \infty} f_\theta(x) = x\beta^{N_l} = \begin{cases} 0 & if \;\; \beta < 1 \\ \infty & if \;\; \beta > 1 \end{cases} \tag{2.44}$$

$$\sigma = \sqrt{\frac{2}{n_{in} + n_{out}}}. \tag{2.45}$$

$$\sigma = \sqrt{\frac{2}{n_{in}}}. \tag{2.46}$$

2.4.2.2 Dropout Operation

Dropout operation is designed to prevent the overfitting problem [64] by reducing the number of network weights in a stochastic manner. This has been widely recognized as a stochastic regularization technique which randomly assigns part of the activation x to zero with the probability of $1-q,\ 0 < q \leq 1$. In general, the dropout operation works similarly to a network/output sampler which reduces the number of network weights under estimation for each mini-batch training samples. Another perspective of understanding the dropout operation is via the model ensembling, where multiple smaller networks are ensembled together stochastically in the optimization. In the test stage, the dropout probability q is set to 1 and we are calculating the average of multiple smaller networks' predictions of the test data. Note that in order to preserve the expectation of the output, the activations are scaled by $\frac{1}{q}$ in the training stage.

2.4.3 DCNN Architectures

This section reviews some well-known DCNN architectures designed for object classification. These models formed the basis of most of the deep transfer networks, deep adaptation networks and the deep ensemble networks. We only introduce the adopted ones in this thesis but we refer the references of other developments [65, 66].

2.4.3.1 AlexNet

AlexNet may be the first DCNN architecture demonstrating the success in large-scale object recognition task, e.g. to classify the objects in ImageNet[59]. Architecture of AlexNet is shown in Figure 2.14(a) which is composed of Conv and FC layers, with appropriate weight initialization [67], drop-out techniques [64] and the Relu activation functions [68]. This may be the first DCNN network trained in an end-to-end manner that has demonstrated the superior performance over the hand-crafted features.

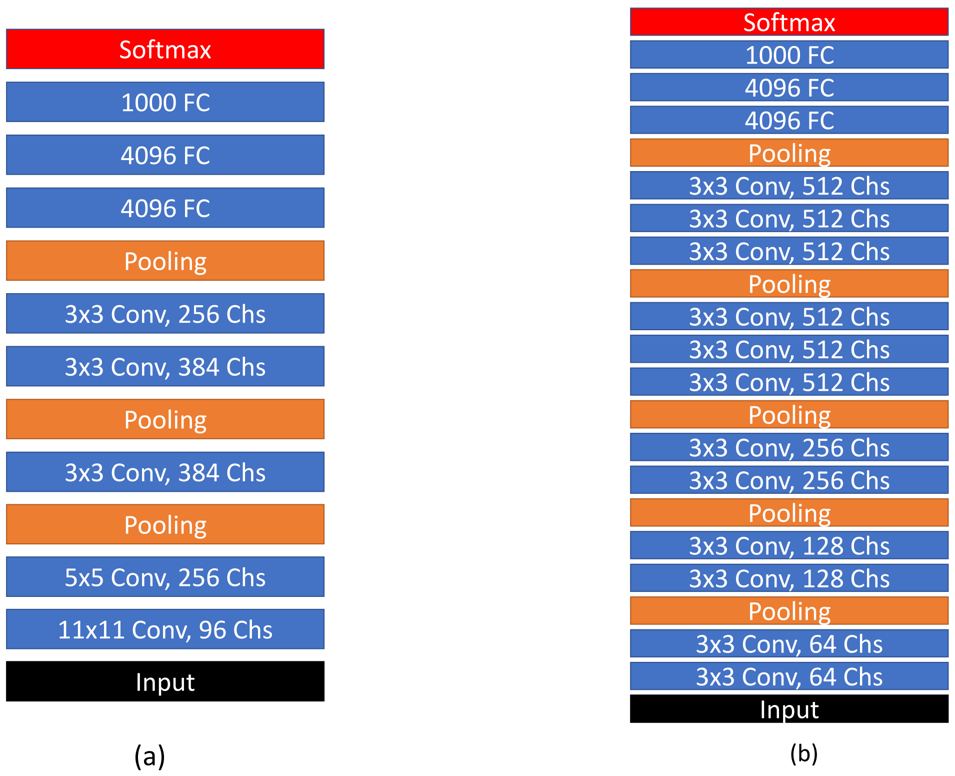


Figure 2.14: DCNN architectures: 1000 FC indicates the output dimension of the FC layer is 1000; 3×3 Conv, 256 Chs indicates the convolutional filter size is 3×3 and we use 256 filters. Softmax indicates the softmax function. (a)Architecture of AlexNet; (b)Architecture of VggNet-16.

2.4.3.2 VggNet

Compared with AlexNet, VggNet [69] in Figure 2.14(b) designs the Conv layer filters with the same size of 3×3, which are smaller than the one used in AlexNet. However, VggNet stacks more layers in their design.

More specifically, stacking three 3×3 Conv layers with the stride size 1 has the same receptive field as the one using a 7×7 Conv layer. Therefore they argue that stacking more Conv layers but with smaller filter size can improve the classification results by reducing the number of parameters compared with AlexNet. Another reason of the outperformance is that VGGNet integrates more non-linear activation functions due to stacking more Conv layers and adopting deeper architectures.

2.5 Transfer Learning in Deep Neural Networks

Transfer learning aims to effectively boost the classification performance of the new task (e.g. learn to recognize a new object) by transferring the knowledge gained from old ones (e.g. the network weights used to recognize other objects). This is inspired by the human recognition mechanism, where we tend to link the observed properties of the new task with the old task stored in the memory. Taking Figure 2.15 as an example where given such an image, we may preferably describe the animal as: the head of giraffe, the body of the horse and the zebra's legs. In our recognition system, these useful features stored from previous observations of other animals are combined in a particular way to recognize this "unpopular" animal. The example indicates that to learn a new task effectively, we may tend to leverage the gained knowledge learned from previous old tasks.

Machine learning algorithms, especially classification methods make assumptions about the model and task. The most widely adopted one is that the training and test data, features or joint feature and label representations are drawn from the same distribution. Unfortunately due to the various environmental perturbations of the data, different sensor types and potential human manipulations in the pre-processing methods, this assumption is rarely held in practical scenarios. In transfer learning, the data sampled from the original distribution is termed the source data or source domain while the data from the new distribution is referred to the target data or target domain. To tackle the distribution discrepancy, transfer learning framework is utilized to classify target data effectively leveraging the gained knowledge from the source data.

Transductive learning (TL) and inductive learning (IL) are two categories of transfer learning [70]. Later, the communities also refer to the TL as the *domain adaptation* problem. In the following, we review the definitions and discuss the differences.

- **Transductive Learning:** given the source and the target domain (data,label) pair $\mathbf{D_s} = \{(x_s, y_s)\}$ and $\mathbf{D_t} = \{(x_t, y_t)\}$, TL assumes that both domains share the same classification categories, more specifically $y_s \in [1, 2, ..., N_c]$ and $y_t \in [1, 2, ..., N_c]$.

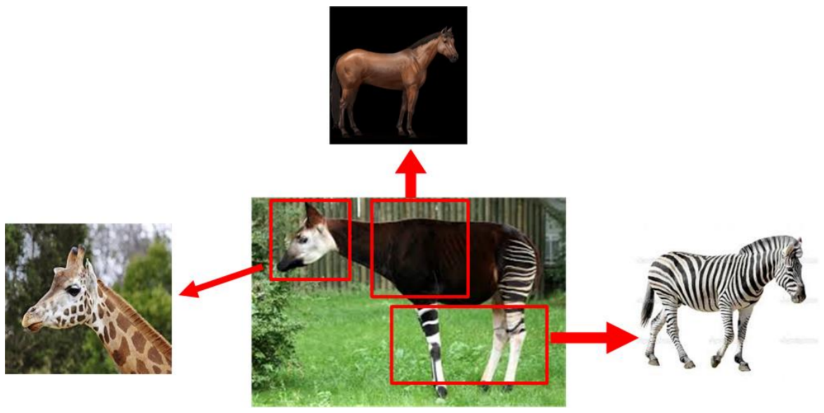


Figure 2.15: An "unpopular" animal called Okapi, with Giraffe's head, Horse's body and Zebra's leg.

.

The random variables representing the image and label in general are denoted as X and Y. In spite of this, the joint distributions of data and label can still be different: $P(X_s,Y_s) \neq P(X_t,Y_t)$. The aim of TL or domain adaptation is to learn transformation T so that the joint distribution of feature and label can be similar: $P(T(X_s),Y_s) = P(T(X_t),Y_t)$. In other words, the target domain classification performance can be improved by reducing distribution divergence between the two domains. A more practical yet challenging scenario is the unsupervised domain adaptation (UDA) where no label information in target domain is available. As we make contributions in this particular scenario, we review the relevant UDA methods in Section 2.5.2.

- **Inductive Learning:** in IL, we consider a more generalized scenario where labels from two domains y_s and y_t are from different classification categories. Although $\mathbf{D_t} \neq \mathbf{D_s}$, IL considers to improve the network training of $\mathbf{D_t}$ using the gained knowledge from source domain $\mathbf{D_s}$. If both the source and target data are well-annotated and labelled, the IL is often referred to the *multi-task learning*. In this thesis since we are handling supervised learning for μ-DS classification, we review the relevant multi-task learning methods in Section 2.5.1.

2.5.1 Multi-task Learning in Deep Neural Networks

Multi-task learning aims to transfer knowledge from well-labeled source domain to the well-labeled target one when $\mathbf{D_s} \neq \mathbf{D_t}$. We mainly review the methods in the following two scenarios:

- **The source and target data are drawn from different distributions, $P(X_t) \neq P(X_s)$ meanwhile, the classification categories are different.** In this scenario, the community designs DTN for learning new tasks by utilizing the state-of-the-art DCNN architectures for the old tasks, for example object recognition in computer vision, as introduced in Section 2.4.3. The very high-level illustration can be found in the example in Figure 2.16, where source domain data is the large-scale ImageNet dataset while the target domain data is the small-scale PWR µ-DS. Although with different tasks, the DTN can transfer the pre-trained weights to the new µ-DS classification.

- **The source and target data are drawn from the same distribution, $P(X_t) = P(X_s)$, however the classification categories are different.** As the data are sampled from the same distribution, we are actually classifying multiple factors of the same data. Taking the µ-DS Classification as an example, for each data sample, we may access labels of three factors, indicating motion, human target and the aspect angle respectively. To transfer knowledge among multiple tasks, the DEN is designed by sharing information with each other. A simple example of classifying three factors of the µ-DS is shown in the Figure 2.17, where the feature extractor network T is shared by the three tasks and the classification of each factor can be regarded as the source domain while the others are the target domains. Leveraging the transferred knowledge between tasks, the final motion classification may outperform the one without gaining knowledge from the other tasks.

2.5.1.1 Deep Transfer Network

DTN is always utilized to classify small-scale dataset leveraging the pre-trained weights from the large-scale dataset, for example the ImageNet [71]. DTN is normally composed

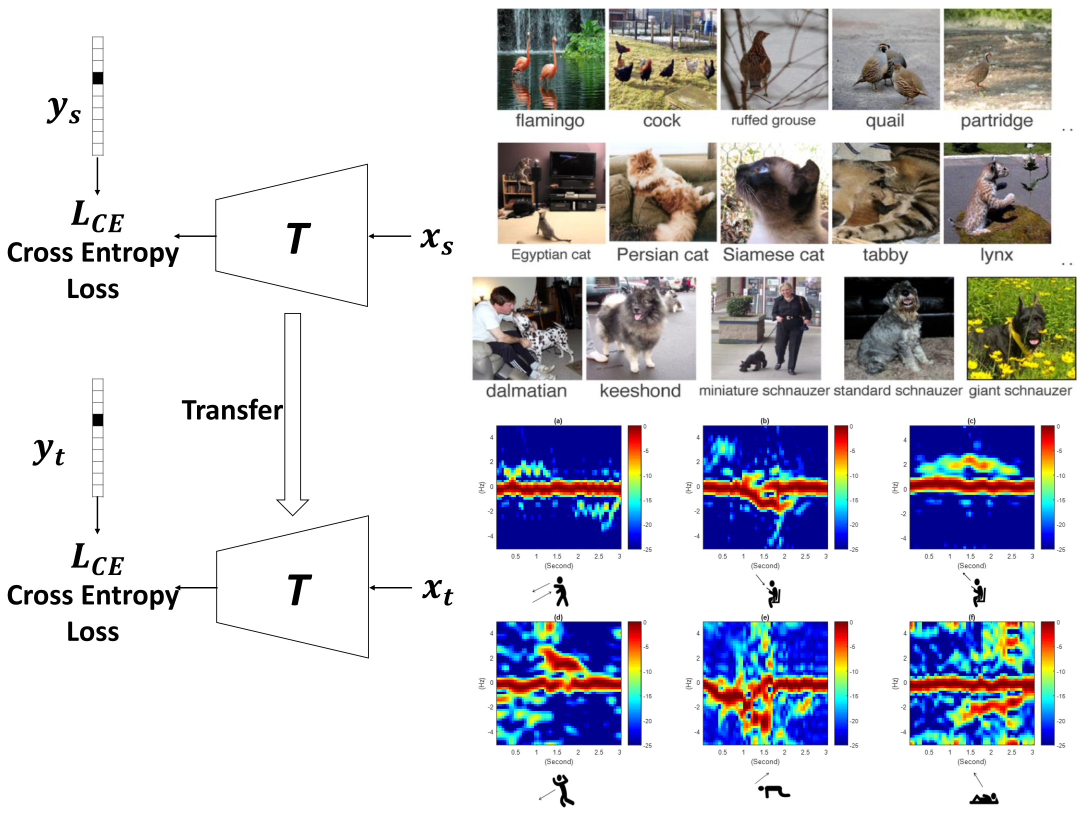


Figure 2.16: An example of DTN: transferring knowledge from object recognition in computer vision to μ-DS classification in PWR. Source domain (data, label) pair x_s, y_s from ImageNet is fed to the DCNN T and network training is based on the CE loss L_{CE}. We transfer network and domain knowledge T to the μ-DS classification task. Fine-tuning the network T using the target domain (data, label) pair x_t, y_t and CE loss.

.

of two steps: high-level layer training and the low-level layer fine-tuning. DTN is inspired by the fact that low-level and local features are shareable between multiple tasks. Therefore, we may first train the high-level layers to minimize the loss function with the low-level Conv layers fixed. Next, we fine-tune the low-level layers and the high-level layers together but using smaller learning rates.

2.5.1.2 Deep Ensembling Network

DEN is adopted by ensembling multiple tasks so that the features extracted can be task generic and more discriminative. From other perspective, multiple tasks can be regarded as the regularization terms for others and prevents the overfitting problem. The commonly utilized multiple tasks comprise the classification task, the data encoding and decoding task [72], the detection task [73] and classifying multiple factors[74].

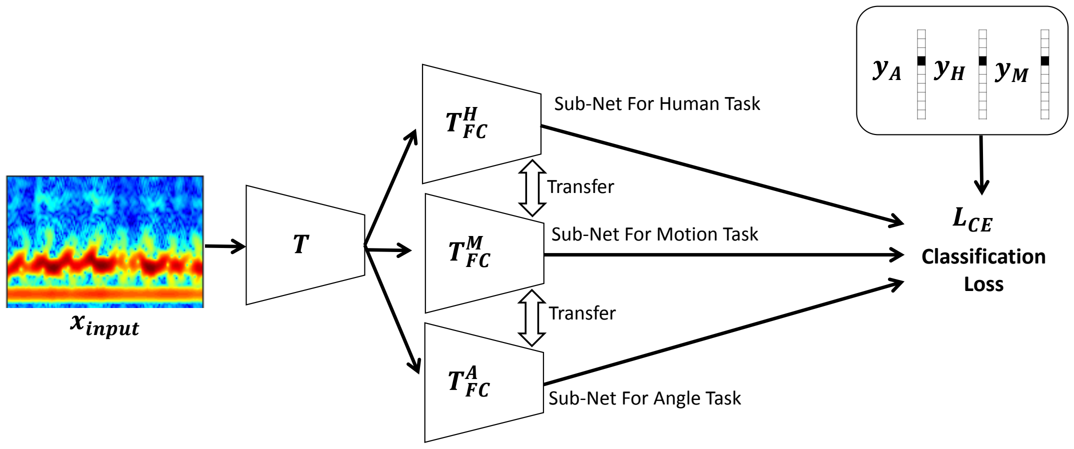


Figure 2.17: An example of DEN: ensembling multiple factors (aspect angle and target personnel) recognition in µ-DS based activity recognition. Given the µ-DS data x_{input}, feature extractor T extracts useful feature $T(x_{input})$ based on three tasks, namely the motion, target personnel and the aspect angle recognition.

.

2.5.1.3 Applications in µ-DS Classifications

Since the µ-DS dataset is relatively small compared with the large scale datasets in computer vision, it is difficult to train the whole DCNN architectures introduced in Section 2.4.3. However, shallow methods are not as powerful as the representation capability compared with the DCNN. In this way, the DTN may be utilized leveraging the pre-trained weights from ImageNet and the fine-tuning techniques introduced in Section 2.5.1.1 are usually utilized.

To design a robust system for the µ-DS classification, two factors of variations should be accounted for, namely the target personnel and aspect angle. Variations of the two factors may give rise to data sample variations and these can destroy the pre-trained classifiers. One solution is to utilize the DEN by ensembling more factor labels and design multiple factor classification tasks shown in the Figure 2.17. Through incorporating multiple tasks and more regularizations for the network training, the variations caused by the factors may be reduced in the feature space, for example, the features extracted for motion classification may be invariant to the human target variations.

2.5.2 UDA in DCNN

As mentioned in Section 2.5, even though with the same classification task, it is almost impossible to assume that training and test data, or the source and the target data are drawn from the same distribution. In addition, as the test (target) data are always assumed unlabeled, it is fairly important to learn the classifier and feature extractor from the training (source) data by considering the effect of test (target) data in an unsupervised manner. We always refer to this the UDA problem, especially for DCNN models, as DCNN is very sensitive to small perturbations of the input [75]. DCNN models for UDA are composed of two steps: i) feature and classifier design and ii) reducing the divergence of feature distribution between domains. In the following, we review the main divergence measurements and the deep adaptation networks to tackle UDA. Finally, we introduce the potential applications of UDA in μ-DS Classifications.

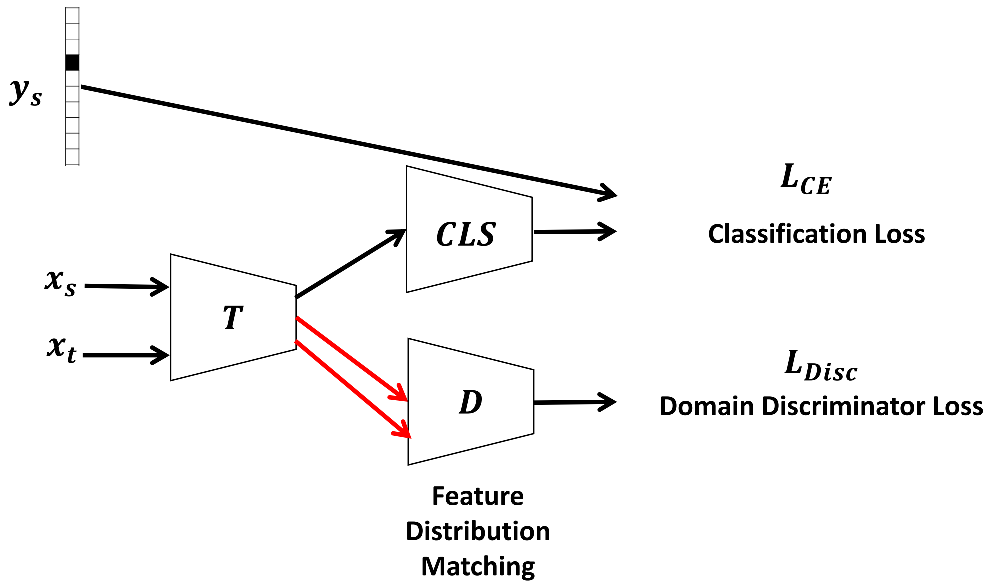


Figure 2.18: Adversarial training in UDA networks. Given the source (data, label) pair x_s, y_s and target data x_t, DCNN network T aims to extract useful features $T(x_s)$ and $T(x_t)$ so that the domain discriminator D cannot differentiate which domain they are from. The classifier is represented as CLS and learned by CE loss using x_s, y_s.

.

2.5.2.1 Divergence Measurement

This section introduces divergence measurements commonly used in adaptation tasks in DCNN, including Maximum Mean Discrepancy (MMD), Jensen-Shannon (JS) divergence and the Optimal Transport (OT) divergence. Let us further assume that the DCNN feature representations from two domains are denoted as $T(x^s)$ and $T(x^t)$ and N_s and N_t samples are drawn from the distribution $P_s(T(X^s))$ and $P_t(T(X^t))$ respectively.

- **MMD:** The MMD is usually considered as the non-parametric estimate of the distance between two distributions, by projecting the input into the Reproducing Kernel Hilbert space (RKHS) [76]. The projection function can be more formally defined as $\phi(x) : \mathbb{R}^d \to \mathscr{H}$ with the $\mathscr{H}$ a RKHS and the MMD is formalized in the Eq.(2.47) where the distribution discrepancy is measured via the difference of the feature sample means in the RKHS. This measurement can be efficiently calculated using the inner product of mapped feature $\phi(T(x_i^s))$ and $\phi(T(x_j^t))$ via the kernel function trick. In addition, MMD will only be close to zero if the two distributions under estimate are very similar $P_s(T(X^s)) \approx P_t(T(X^t))$, as proved in [76].

$$MMD(T(x^s),T(x^t)) = \|\frac{1}{N_s}\sum_i^{N_s}\phi(T(x_i^s)) - \frac{1}{N_t}\sum_j^{N_t}\phi(T(x_j^s))\|_H^2 \tag{2.47}$$

- **JS:** JS divergence may be the first measurement implemented based on the adversarial training strategy. This has been widely used in the well-known Generative Adversarial Network (GAN) [77] and relevant applications. We formulate the JS divergence $JS(P_s||P_t)$ in Eq.(2.48) and its implementation using adversarial training for UDA.

  The most relevant example in UDA using adversarial learning is in Figure 2.18 where we add the discriminator network D to discriminate whether the features $T(x^t)$ and $T(x^s)$ are from the same distribution, while the network T aims to confuse D. The adversarial learning is more formally defined in Eq.(2.49) [77], where the output of discriminator $D(T(x^s))$ and $D(T(x^t))$ are scalars, indicating the probability that the output is from the source domain. The equivalence between Eq.(2.49) and the JS divergence Eq.(2.48) is shown in the Appendix B.3.

$$JS(P_s||P_t) = KL(P_s||\frac{P_s+P_t}{2}) + KL(P_t||\frac{P_s+P_t}{2}) \tag{2.48}$$

$$\min_T \max_D \mathbb{E}_{T(x^s)\sim P_s(T(x^s))}[log(D(T(x^s)))] + \mathbb{E}_{T(x^t)\sim P_t(T(x^t))}[log(1 - D(T(x^t)))] \tag{2.49}$$

- **OT:** OT measures the distribution divergence via the perspective of finding the optimal transportation plan between domain samples [78]. In general, to reduce feature distribution divergence, OT based methods first estimate the OT plan between two distributions and then learn the feature transformation T to minimize the total cost of such a plan. Therefore, we need to define two terms, namely the transportation plan and the distance of each transportation.

  Let us first define the transportation plan between feature distributions in source and target domains as γ with the marginals P_s and P_t respectively. In the discrete version, the set of probabilistic couplings **B** can be defined as the following Eq.(2.50). Ideally, the cost of the transportation from $T(x_i^s)$ to $T(x_j^t)$ is measured by the distance function $c(x_i^s, x_j^t)$ and based on this the distance matrix C between two discrete distributions can be defined.

  Finally, we may define the metric $J(P_s, P_t)$ in Eq.(5.23) to measure the total cost of transporting probability masses from target to source domains. The network T is updated by minimizing the OT divergence measurements in Eq.(2.52).

$$\mathbf{B} = \left\{\gamma \in (\mathbb{R}^+)^{n_s \times n_t} | \gamma 1_{n_t} = \mu^s, \gamma^T 1_{n_s} = \mu^t\right\} [79]. \tag{2.50}$$

$$J(P_s, P_t) = \langle\, \gamma, C\rangle_F,\ with\ \gamma \in \mathbf{B}, \tag{2.51}$$

$$\min_T J(P_s, P_t) = \min_T \langle\, \gamma, C\rangle_F,\ with\ \gamma \in \mathbf{B}, \tag{2.52}$$

2.5.2.2 Deep Adaptation Network

Based on the reviewed measurements in Section 2.5.2.1, the source domain classification and the feature distribution matching are the two tasks in the DAN. Taking the Figure 2.18 as an example, two loss functions related to the two tasks are designed in the DAN, including the source domain CE loss L_{CE} and the domain discrepancy loss $L_{divergence}$ indicating the feature distribution divergence.

2.5.2.3 Applications in μ-DS Classifications

Target personnel and aspect angle are two essential factors of variation in μ-DS which make the classification task more challenging. In the practical scenarios, we may only access a portion of the data from certain target personnels and aspect angles, however the test data may comprise unseen targets and angles. Facing the distribution divergence between training and test data, the DAN can be successfully applied to address this UDA problem where we aim to learn domain-invariant features by reducing the distribution divergence. More specifically, suppose the training (source domain) data x_s are from target A from 30 degree aspect angle, but the test (target domain) data x_t may com from the target B from 60 degree aspect angle. DAN can be directly applied to solve this problem.

2.6 Activity Recognition in Diverse Modalities

This section reviews activity recognition methods using diverse modalities, including camera video based methods, wearable sensor based methods and the radar based methods.

2.6.1 Camera Video based Methods

Although RGB video sequences in the temporal domain provide a lot of and sufficient information for action recognition, RGB data encode irrelevant information as well, for example, the background of the scene and the weather and light condition [80]. The activity recognition requires to model the temporal information of the video sequences, design features and classifiers accordingly.

Various hand-crafted features have been adopted in conventional methods, but these can be divided into two categories: *holistic* and *local* representations [80]. *Holistic* representation methods regard the whole pixels within the human bounding box as the Region of Interest (ROI). Such methods include two steps, person detection and the feature computation of the ROIs. The commonly used descriptors include but are not limited to silhouettes[81, 82], edges [83] and the optical flow based features [84] in the spatial-temporal space. *Local* representation based methods, do not require the person detection but choose to detect the Point of Interest (POI) and feature computation of the

POI. The POI detection is in general based on finding a video region (spatio-temporal) which maximizes the saliency. The commonly used local descriptors include but are not limited to HOG[85], HOF[85], brightness[86], gradient[86], optical flow[86] and dense trajectory methods[87, 88].

With the recent successful development of deep learning, features and classifiers have been integrated in the end-to-end training pipeline using DCNN, which improves the previous benchmark by a large margin. For the action recognition task, DCNN aims to learn specialized spatio-temporal descriptors (convolution filters) for the video sequences and the classifiers (decisioni boundaries) at the same time. The commonly used and popular network structures include but are not limited to two-sequence network using two DCNNs for still images and the optical flow results[89, 90], various RNN based models to integrate the CNN features based on multiple frames [91, 92] and the spatial-temporal 3D CNN [93, 94] including the C3D and I3D networks. Only until recent work of I3D, DCNN based methods have outperform the conventional hand-crafted features by a large margin.

2.6.2 Wearable Sensor based Methods

Compared with the RGB video based methods to describe the activity, wearable sensors including but are not limited to accelerometers [95, 96], radio-frequency identification (RFID) tags[97], switch sensors and motion sensors [98]. Although these wearable sensors provide less information compared with the RGB images, they describe the most important state change information of the activities. Body sensor networks[98] have been developed by various research to capture state of user using heterogeneous sensors[99, 100]. Such network can be used for continuously monitoring the numerous physiological signals.

To extract relevant features, discrete cosine transform [96] and sparse coding techniques [101] combined with principle component analysis are commenly adopted methods for modelling the human activities. For the classifiers, hidden markov modelling [102] and support vector machine [96] are usually utilized. Recent DCNN methods [103] have been investigated for extracting features and designing classifiers using wearable sensor data, which outperforms the conventional methods.

2.6.3 Radar Sensor based Methods

Radar systems, from Doppler to the imaging radar, can detect human activities using either the Doppler by measuring the velocity of body parts, or providing the images of the human targets. However, to robustly detect the activities, µ-DS is the most commonly used measurement.

Various radar systems have been demonstrated to successfully measure the µ-DS of human activities, including the CW radar at various frequencies [104, 105, 106], FMCW radar [107, 108] which provides robust ranging detection of the human target as well and the passive Wi-Fi radar for monitoring the activities in both the healthcare and security applications [109, 110]. As for feature extraction methods, the spectrogram and wavelet transformation based methods have been most widely utilized to extract the various frequency components of the µ-DS[111, 6]. Other time-frequency analysis methods have been suggested as well, including the hilbert huang transform and the empirical mode decomposition. These frequency descriptors are then processed by the dimension reduction methods, such as principle component analysis and the independent component analysis methods or some research focus on designing the hand-crafted features including the power of the frequencies, the RCS of the radar reflections etc [3, 112]. With the development of DCNN based methods, more research are investigating to use the spectrogram images as the input for the DCNN and learn feature extractor and classifiers simultaneously[113].

Chapter 3

Passive Wi-Fi Radar for Activity Classification and Signs-of-life Detection

3.1 Introduction

This chapter investigates PWR methods for signs-of-life detection and activity recognition, which have attracted attentions from both healthcare and security communities. Signs-of-life detection can detect breathing patterns in a contactless way in healthcare applications and can also be used for detecting people dead or alive inside a building in security applications. Activity recognition can contribute to detect fall motion in the context of elderly home and also may help identify whether someone gets hit and falls down etc.

In the era of big data, communications among devices exist almost everywhere and therefore the Wi-Fi access points are deployed ubiquitously. Because of that, it is more feasible to deploy PWR sensors utilizing existing Wi-Fi routers for activity monitoring. More specifically, PWR system comprises two receiver channels, namely the reference and surveillance channels. Ideally the reference channel is designed to receive signals from the Wi-Fi transmitter directly and isolate the reflected echoes from the surveillance channel. On the other hand, the surveillance receiver should pick up the reflected echoes from targets, also isolating the direct arrivals from the transmitter (see more details in

Section 2.1.4.3). PWR sensors are able to detect the μ-DS caused by targets' movements and indicate the relevant activities by comparing the surveillance channel echoes with the reference waveforms.

Driven by demands "which the aging population are placing on the healthcare services", the Ambient Assistant Living has been a widely accepted concept whereby various E-healthcare technologies are employed to monitor elderly and disabled people. Within the Ambient Assistant Living framework, activity recognition has been an important research topic to facilitate enhanced situational awareness. Current activity monitoring sensor technologies including wearables [114, 115, 116], mobile phones [117], RFID [118], passive infrared sensors [119, 120], ultra-wide bandwidth based [114] and vision based sensors [121, 122] are being investigated to detect, recognize and monitor human activity [123]. Among these, sensors embedded in wearables and mobile phones such as accelerometers and gyroscopes are able to provide some physical information about the subjects, but suffer from low movement update rates of typically less than 5Hz. In addition, people may forget to wear or drop their on-body sensors due to the physical discomfort. Passive infrared sensors are able to only provide the coarse-grained room level existence [124] while RFID based devices employ complex transmitters and receivers, and require pre-planning in order to optimally site the positions of the nodes [123]. Similar to on-body sensors RFID tags or transmitters can also be easily damaged, lost or forgotten [125]. In a similar manner to RFID, ultra-wide bandwidth activity recognition systems need heavy pre-deployment set up and ultra-wide bandwidth components are more expensive than other technologies. Video system such as MS Kinect and Intel RealSense have been investigated in some healthcare projects [126]. However, in general, the video camera systems require optimal lighting conditions and the acceptability of deploying video cameras in home environments raises many privacy issues.

Besides normal activity recognition, signs-of-life detection is challenging in the healthcare applications as it requires very sensitive detection methods. It is even more challenging to use wireless or other low bandwidth signal, because we want to detect small chest-wall movements as an indicator of the respiration. In fact, most techniques in this category are bespoke active systems with large bandwidth and may require complex antenna arrays or the use of wearable devices. However, small movement detection is very important in both security and healthcare applications, e.g. in security applications, it is important to detect the hidden human targets, ensuring that no one is behind the door when raiding a house during a policing operation; in healthcare applications, it is essential to monitor the sleeping quality and analyze the breathing pattern.

In this chapter, four key contributions are made:

- A real-time phase-sensitive breathing detection methods is proposed and evaluated for signs-of-life detection.
- A μ-DS dataset involving six motions of interest were collected experimentally and the key features of the signatures are analyzed.
- We propose a novel μ-DS classification pipeline including a detection scheme, data sample alignment method and a suite of classifiers. The classification methods are SRC and the DTN pre-trained by the ImageNet dataset [7] but fine-tuned using μ-DS datasets.
- Finally, we proposed the DLL layer implemented as a series of recurrent units to control the moderate sparsity of neural activations in DTN. This is achieved by replacing conventional FC layer with our newly proposed DLL which improved the results of conventional DTN [8].

3.2 Related Works

In this section, we mainly review the most related works for signs-of-life detection and activity recognition using PWR μ-DS.

3.2.1 Signs-of-life detection using PWR

Tang et al. designed the new analog circuit for signs-of-life detection [127], namely the injection lock oscillator to compare the surveillance signal with the reference one and to extract the phase difference between two channel data. In [128], Li et al. proposed to extract statistics of the Doppler frequency bins between adjacent CAF batch results to detect the relative changes of the frequency spectrum. They argue that the statistics and relevant patterns can indicate the breathing. The most similar work to us is [129] where Khan et al. utilized and extended our phase extraction framework for breathing detection and added to detect large movements by measuring whether the phase value is periodic or not. To the best of author's knowledge, we may be the first to investigate using the phase information in PWR signs-of-life detection.

3.2.2 Activity recognition using PWR µ-DS

Human µ-DS classification has widely been investigated for activity recognition, especially in healthcare applications. Li et al. [110] proposed to use the energy power of PWR Doppler as features for hidden markov model and adopted the time-series feature for clustering. Since the limited work in activity recognition using PWR µ-DS, we also compare our proposed DTN with DLL and the SRC with related feature learning and classification methods from other radars. Li et al. [130] proposed to use sparse representation for feature extraction of the dynamic hand gesture using the 24GHz radar. Their sparse representation framework utilizes the basis of Gabor filters design, however, our SRC is a more self-contained classifier applied directly to the PCA features using the dictionary composed of training data features, without the memory and time-consuming, pre-defined Gabor filter dictionary. Seyfiolu et al. [131] also handles the small scale µ-DS dataset using DTN and analyzed the optimal DCNN initilization methods.

3.3 Signal Model and Processing of PWR

PWR utilizes the existing Wi-Fi access points as transmitters of opportunity. The reference signal $ref(t)$, can be described as a linear combination of the "clean" transmitted Wi-Fi signal $x_{source}(t)$ and the reflections from static objects, characterized by p^{th} path delay τ_p and the corresponding complex magnitude and static phase A_p^{ref}:

$$ref(t) = \sum_p A_p^{ref} x_{source}(t - \tau_p) \tag{3.1}$$

Similarly, the signal in the surveillance channel $sur(t)$, is composed of echoes from all moving targets in the illuminated area of interest, which can be characterized as the p^{th}

path delay τ_p, its Doppler shift $f_{d,p}$ and relevant complex magnitude and phase A_p^{sur}:

$$sur(t) = \sum_p A_p^{sur} x_{source}(t - \tau_p) e^{j2\pi f_{d,p} t}. \tag{3.2}$$

It is assumed that the reference and surveillance signals are separated via the spatially directional antennas. In general, a target can be identified by cross correlating the reference and surveillance signals and using the Fast Fourier Transform (FFT) to find the exact delay τ and frequency shift f of the reflected signal. This can be represented by the CAF in [132, 13] Section 2.2 and also similarly in Eq.(2.13):

$$CAF(\tau, f) = \int_{-\infty}^{\infty} e^{-j2\pi f t} ref^*(t - \tau) \times sur(t)\, dt. \tag{3.3}$$

3.4 Breathing Detection using PWR

For the small movement detection, we propose a phase extraction method evaluated by experimental data, assuming that there is no translating motion while the target is breathing during measurement recordings.

As introduced in Section 2.1.3, Doppler information is directly related to the range changes. Due to the slow and tiny chest-wall movements, the induced Doppler frequencies are too small to be detect, not to mention the interference near the zero Doppler frequencies. However, phase information is ideal for the signs-of-life detection as it is the instantaneous Doppler measurement which is able to detect tiny movements smaller than the wavelength. In the following, we first discuss the limitations of CAF and propose the phase-sensitive detection in signs-of-life detection. Second, we propose the Hampel filter to guarantee the phase stability.

3.4.1 Limitations of CAF and Phase-Sensitive Detection

In normal human breathing, the chest moves slowly and the amplitude of the movement is small. Detecting breathing using PWR is therefore limited by the following factors: firstly, the bandwidth of the Wi-Fi signal limits the range resolution to 17 meters [133, 134] which is too coarse for this application; secondly, as the chest movement is slow, the ideal Doppler resolution for detection requires a long integration time; third, this small Doppler signature can be masked by the Direct Signal Interference (DSI) and the background noise related with the phase variation of the system noise. Later parts focus on analyzing the effects of the non-stationary background phase variation noise, related with the systematic phase variation, the long integration time of the Wi-Fi signal or the specific PWR waveform in the CAF processing.

We analyze breathing detection in more detail and denote the range of chest movements as $d(t)$. Accordingly, the phase of the received echo signal from the chest, $\phi(t)$ can be represented by:

$$\phi(t) = \frac{2\pi \times d(t)}{\lambda}. \tag{3.4}$$

Due to the limited range resolution of the PWR, we can therefore assume that the target is in one range bin during each CAF operation. Then the correlation result $x[m], m \in [0, 1, ..., M-1]$ in the time domain within a range bin of largest power l_{max} is given by:

$$x[m] = \sum_{n=0}^{N_m - 1} ref^*[i_m + n - l_{max}] \times sur[i_m + n], \tag{3.5}$$

where M is the number of batches we divide the signal into and $*$ is the Hermitan operator. N_m is the number of data samples in m^{th} batch and i_m is the starting sample index of each batch [135]. The reason we are using the batch processing is for the computation efficiency required in real-time processing. In the indoor environment, as there are echoes from other reflectors as well, the phase of $x[m]$ could be represented as:

$$\phi[m] = \frac{2\pi \times d[m]}{\lambda} + \phi_{static}, \tag{3.6}$$

where $\phi_{static}[m]$ is the combination of echo phases from static reflectors. Since the slow breathing process, $\phi_{static}[m]$ cannot be larger than half the wavelength and ϕ_{static} is time-invariant between two consecutive CAF operations, we are able to simultaneously monitor the phase variation corresponding to the chest movement, without considering the distortion from DSI and the long integration time involved in FFT operation.

Note also that since the modulation scheme of the Wi-Fi signal is not ideally designed for the radar detection, the test of the background phase stability is necessary.

Since the background phase variation may cause the ϕ_{static} not the constant, or at least probably larger than the phase variation caused by the chest movement. Another cause for the phase non-stability is the uncertain starting and ending point of the Wi-Fi signals under integration among different batches. The final phase variation may be caused by the radar system components, where the systematic noise may be severe as well. These all can generate the non-stable ϕ_{static}, under which the expected phase variation of the chest movement may be buried and difficult for the detection.

3.4.2 Hampel Filter and the Phase Stability Analysis

Due to modulation characteristics of the Wi-Fi signal, extracting stable and continuous phase output from cross-correlation is a challenge. The case is even worse when the beacon signals are transmitted, because the low power and burst rate give rise to less effective target echoes for integration and the low SNR matched filter results. In this scenario, the phase caused by the system noise or the clutters will dominate, as explained in more details in Section 3.7.1.1. When $x_{source}(t)$ is sparse in the time domain, discontinuous phase samples are frequently observed and the phase output has relatively large distortions, as shown in Figure 3.16.

To tackle this problem, a Hampel filter is utilized as a post-processing step to eliminate the outliers caused by the discontinuous phase output [136]. Suppose we have obtained a time series of phase samples, denoted as:

$$\phi_k = [\phi_{k-\omega}, \phi_{k-\omega+1}, ..., \phi_k, ..., \phi_{k+\omega-1}, \phi_{k+\omega}], \tag{3.7}$$

then we would like to check whether ϕ_k is an outlier. Within a certain window size, ω, the median absolute deviation scale estimate is calculated as follows:

$$MAD = median\{|\phi_k - median\{\phi_k\}|\}. \tag{3.8}$$

Then, with the assumption that the phase data within the window are sampled according to the following distribution:

$$\phi_k \sim \mathcal{N}(\mu_k, \sigma_k) \tag{3.9}$$

the estimated deviation of the distribution, σ_k can be approximately given by the criterion from [136]:

$$\sigma_k = 1.4826 * MAD, \tag{3.10}$$

where 1.4826 is the calculated in the Gaussian distributions [137]. Given a threshold t, we can classify the phase as an outlier if $\|\phi_k\| \geq t \times \sigma_k$. In this case, the outlier will be set as the median of the series of phase samples. Although outliers can be eliminated by above methods, phase shifts caused by the background static reflections still remain. Due to the time invariant characteristics during consecutive CAF operations, we can subtract the mean of phase outputs for background elimination, represented by the following equation:

$$\phi_{true,k} = \phi_k - \frac{1}{L} \sum_{k-L+1}^{k} \phi_k, \tag{3.11}$$

where ϕ_k is the k^{th} slow-time sequence of phase information, L is the window length and $\phi_{true,k}$ is the phase sequence after mean subtractions.

3.5 Activity Recognition using PWR

3.5.1 Problem Formulation

The PWR basics have been introduced in Section 2.1 and a complete description of the PWR signal model is given in Section 3.4.1, Eq.(3.4) and the research work [135]. As introduced in previous Section 2.2, the µ-DS is the time history of the frequency vectors at specific delay induced by the moving target. These vectors are concatenated along the time axis to generate the time-Doppler history signatures, which are usually regarded as the preliminary data of the µ-DS dataset.

More specifically, suppose that we obtain the original µ-DS dataset $\mathbf{D_{PWR}} = \{(X_i, y_i)\}_{i=1}^{N_{PWR}}$ with N_{PWR} samples, $X_i \in \mathbb{R}^{N_f \times N_t}$ indicates the i^{th} time-frequency representation and $y_i \in \mathbb{R}$ indicates its n_{cls}-class label. Note that since different motions may induce different time periods, X_i is the 2-D time-frequency representations of dynamic size N_t in the time domain. For each recording X_i, we normalize it to the range of $[0,1]$ following the widely adopted techniques. The aim of the PWR based activity recognition is to predict the activities using µ-DS the dataset $\mathbf{D_{PWR}}$.

3.5.2 Capability of PWR for Human Activity Recognition

As the velocity of movement is an essential factor of the µ-DS, it is necessary to analyze the velocity profile of daily motions in the within-home environments. In general, the velocities associated with sitting down on a chair, falling down and even walking, exhibit a maximum velocity of around 2 m/s [138]. Given a PWR in 2.4GH sepctral band, the maximum speed limit induces the maximum Doppler shift of approximately 32Hz. To summarize, the PWR system is capable of accurately detecting different motions via the Doppler frequency estimation. At some specific time instant, although the Doppler frequencies from two motions might be similar, the whole temporal Doppler trace may discriminate different activities. To sum up, one activity will induce a particular velocity-time pattern, which exhibits fruitful features for activity differentiation. The concatenated µ-DS time history is regarded as the main feature for recognition.

3.5.3 µ-DS Pre-processing Methods

In this section, pre-processing methods are proposed for activity recognition including the following two main steps: (i) align and adjust µ-DS, (ii) pre-processing methods, as shown in Figure 3.1.

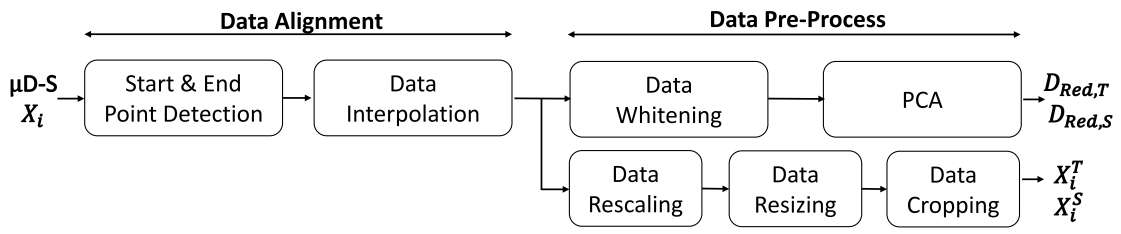


Figure 3.1: Pre-processing steps for activity recognition using PWR.

3.5.3.1 Align and Adjust µ-DS

To set up a dataset, the most important requirement is to keep the data samples of the same size. It is straightforward to maintain the same number of frequency bins in X_i through CAF operation by dividing signal samples within the certain time period into a fixed number of batches. However, due to the unpredictable time periods of different motions, it is difficult to keep the same number of time bins for each µ-DS sample. To tackle this, we design the following two procedures:

- Automatic start and end point detection

- Adjust the data sample size by interpolation

Before illustrating the details of the two procedures, it is necessary to introduce the normalization method to construct the spectrogram data in this chapter. Suppose that we obtain the frequency vector $x_t \in \mathbb{R}^{N_f}$ at time t, before we do the concatenation in the temporal field, we normalize the x_t in the range of $(0,1]$.

To illustrate the two steps clearly, an example is shown in Figure 3.2 (a) and (b). In Figure 3.2 (a), the start and end points of an activity are identified by the red arrows and we denote the μ-DS between them as the active μ-DS. Next in Figure 3.2 (b), the active μ-DS is extracted. and is interpolated to a fixed size using bi-cubic interpolation methods [139].

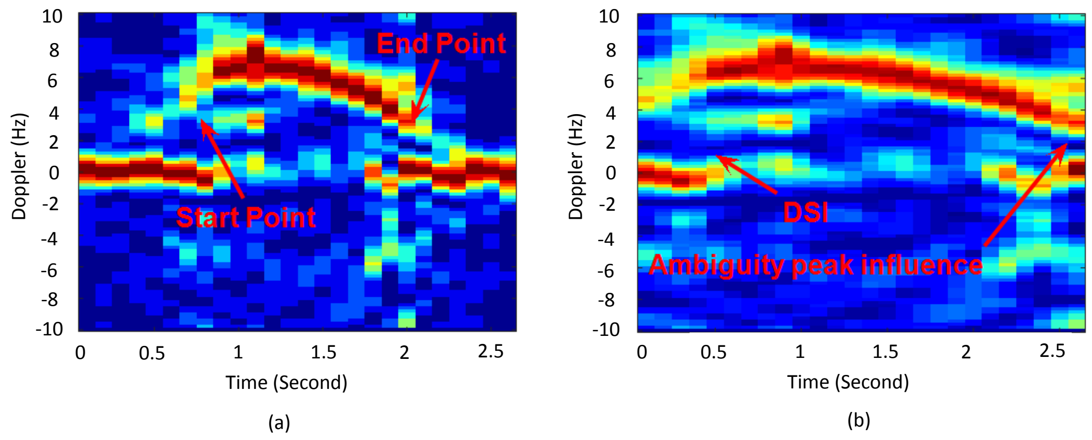


Figure 3.2: (a) Start and end point illustration; (b)The useful part of the μ-DS is adjusted into the same size, with the DSI and the ambiguity peak examples.

Specifically, we propose a method for μ-DS alignment using the standard deviation of frequency vector, $x_t \in \mathbb{R}^{N_f}$ at time t. The conventional CFAR based detection may not be suitable in the indoor environment, as shown in Figure 3.2 (b) due to the following two reasons:

- The ambiguity peaks in μ-DS might mislead CFAR to provide false detections, as shown in Figure 3.2 (b) as they are similar to the Doppler bins induced by the target movements.

- DSI effect will generate strong peaks on the zero Doppler line, which might mislead the CFAR detector, as shown in Figure 3.2 (b). In addition, as DSI will be an important feature to distinguish μ-DS (details about this point are in Section 3.6.3.1), we have not performed any DSI elimination method, such as CLEAN algorithm [11].

$$Mean(x_t) = \frac{\sum_{j=1}^{N} I[j] \times x_t[j]}{N} \tag{3.12}$$

$$std(x_t) = \frac{\sum_{j=1}^{N} (I[j] \times abs(x_t[j] - Mean(x_t)))^2}{\sum_{j=1}^{N} I[j]} \tag{3.13}$$

Without ambiguity elimination and the DSI, an intuitive way to identify the active μ-DS periods is to check whether the non-zero frequency bins have large powers; from another perspective, the variance of the Doppler spectrum is a good feature to distinguish active or inactive μ-DS. To eliminate the effects of zero Doppler clutters when calculating the distribution variances, we adopt the weighted standard deviation in Eq.(3.12) and (3.13), where I is the weights of the frequency bins, which are larger on higher frequency bins. We choose the indicator function as $I[j] = j^2$, where $Mean(\cdot)$ calculates the vector average and $abs(\cdot)$ calculates the absolute value.

In general, we firstly select the active μ-DS and then choose the smallest and largest time bin as the start and end point. The active time bins are selected once the weighted standard deviations of continuous three time bins are larger than the threshold $Thres = \gamma \times min(std(x_t))$, where γ indicates the scaling factor and $min(std(x_t))$ calculates the minimum of the Doppler energy time history. To illustrate the detection process clearly, we plot the example results using the proposed energy Doppler detection in Figure 3.13 and 3.14.

As the active μ-DS denoted by $X_{i,ext}$ may have different number of time bins, we interpolate them in the time domain to a fixed number using the conventional bi-cubic interpolation methods [140]. In this way, the transformed data can be denoted as $X_{i,Fix} \in \mathbb{R}^{N_{freq} \times N_{time}}$, where N_{freq} is the number of Doppler bins and N_{time} is the interpolated number of time bins. In fact, we prefer to represent the well-aligned dataset using the following two formats, fitting the SRC and DTN classification methods respectively.

- **Matrix Format:** we transform the data sample by simple concatenating $X_{i,Fix}$ to a vector $d_i \in \mathbb{R}^{N_{total}}$, where $N_{total} = N_{freq} \times N_{time}$ in Eq.(3.14). Next, we again concatenate d_i to matrix form $D_{Fix,PWR} \in \mathbb{R}^{N_{total} \times N_{PWR}}$.

- **Set Format:** the original set $\mathbf{D_{PWR}} = \{(X_i, y_i)\}_{i=1}^{N_{PWR}}$ is transformed to another set: $\mathbf{D_{Fix,PWR}} = \{(X_{i,Fix}, y_i)\}_{i=1}^{N_{PWR}}$, where $X_{i,Fix} \in \mathbb{R}^{N_{freq} \times N_{time}}$.

$$d_i = vec(X_{i,Fix}) \tag{3.14}$$

3.5.3.2 Pre-processing Methods and Feature Selection for SRC

We adopt the PCA method to reduce the dimension and extract features ready for the SRC classifier and for convenience we use the matrix format $D_{Fix,PWR}$ and randomly divide it into training and test datasets, denoted as D_T and D_S with N_T and N_S samples respectively. More details can be found in Section 3.6.3.

Specifically, the PCA implementation is summarized in the following: *first*, as Eq.(3.15) and (3.16) show, we whiten the datasets D_T and D_S to $D_{diff,T}$ and $D_{diff,S}$ respectively by subtracting the average of each column. *Second*, through Singular Value Decomposition (SVD) [141], the left singular vectors *eigenVec* and the eigenvalues *eigenVal* are obtained in Eq.(3.17). *Finally*, we choose the largest K eigenvectors from the training database and project $D_{diff,T}$ and $D_{diff,S}$ onto the subspace spanned by the K eigenvectors in Eq.(3.18) and (3.19).

The dimension-reduced datasets, denoted as $D_{Red,T}$ and $D_{Red,S}$, are ready to be fed into the classifiers. Note that we used the eigenvectors of the training set to project the test samples, as it is assumed that we cannot know all the test samples a priori.

$$D_{diff,T} = D_T - Mean(D_T) \tag{3.15}$$

$$D_{diff,S} = D_T - Mean(D_S) \tag{3.16}$$

$$[eigenVal, eigenVec] = SVD(D_{diff,T}) \tag{3.17}$$

$$D_{Red,T} = eigenVec^T \times D_{diff,T} \tag{3.18}$$

$$D_{Red,S} = eigenVec^T \times D_{diff,S} \tag{3.19}$$

3.5.3.3 Pre-processing Methods for DTN

Since DTN adopts the end-to-end training pipeline composed of both feature extraction and classifier design, we represent the interpolated µ-DS dataset using set format $\mathbf{D_{Fix,PWR}}$. The three-step pre-processing method consists of data *re-scaling*, *re-sizing* and *cropping*, as illustrated in Figure 3.1.

1. Data Re-scaling: since the original DTN is designed for RGB images, this step linearly re-scales original values of $X_{i,Fix}$ to the range of $[0, 255]$.

2. Data Re-sizing: Suppose that DTN requires the input size of $N_{height} \times N_{wid} \times 3$, we first interpolate the µ-DS data $X_{i,Fix}$ to the size of $N_{height} \times N_{wid,ext} \times 3$, where $N_{wid,ext} > N_{wid}$. Both N_{wid} and $N_{wid,ext}$ indicate the number of time bins and N_{height} indicates the number of frequency bins. Taking the AlexNet as an example, the input size is $227 \times 227 \times 3$ so we interpolate $X_{i,Fix}$ using bi-linear method to $227 \times 256 \times 3$ and assign the other two channels using the same µ-DS.

3. Data Cropping: For training data, we randomly crop the input in the time domain from $N_{height} \times N_{wid,ext} \times 3$ ($227 \times 256 \times 3$ in AlexNet) to $N_{height} \times N_{wid} \times 3$ ($227 \times 227 \times 3$ in AlexNet). For test data, we centrally crop them to the size $N_{height} \times N_{wid} \times 3$ ($227 \times 227 \times 3$ in AlexNet). The reason for not directly interpolating to $227 \times 227 \times 3$ in the AlexNet case is that randomly cropping from the 256 to 227 time bins increases the training data diversity and the generalization capability of the model. This augmentation technique is useful to extract features that are invariant to time-domain shifts and perturbations.

Finally, we denote the output of the pre-processing method as the training and test sets: $\mathbf{D^T_{PWR}} = \{X_i^T, y_i^T\}_{i=1}^{N_T}$ and $\mathbf{D^S_{PWR}} = \{X_i^S, y_i^S\}_{i=1}^{N_S}$ respectively.

3.5.4 SRC for Activity Recognition

In this section, we apply the SRC on the PCA features extracted in Section 3.5.3.2. Note that SRC for µ-DS classification has been generally introduced in Section 2.3.1.3.

For a test sample $d_{test} \in \mathbb{R}^K$ from $D_{Red,S}$, if a small number of samples in the training dataset $D_{Red,T}$ can be utilized to reconstruct the test sample d_{test} with minimum residuals, the training sample with the bigger support and minimum residual may belong to the same class as the test sample. This principle results in the following optimization problem in Eq.(3.20), where s is the sparse coding vector and $||s||_0$ is the ℓ_0-norm operator. As introduced in Section 2.3.1.1 and 2.3.1.2, the ℓ_1-solver is computationally expensive and the OMP is noted for its low convergence rate and inaccuracy [43]. Therefore, we choose subspace pursuit [43] as it has a faster convergence rate than the ℓ_1-solver without loss of accuracy. The pseudo-code of subspace pursuit is in Appendix A.2.

The classification task is operated based on residual comparisons of the reconstruction error using the i^{th} class samples. Specifically, reconstruction error of the i^{th} class can be calculated by subtracting the linear combinations of atoms from the i^{th} class, as indicated by Eq.(3.21). The test sample label can then be predicted based on the by looking for the minimum of the reconstruction residuals among all classes.

$$\underset{s}{\operatorname{argmin}} ||d_{test} - D_{Red,T} \times s||_2^2 + ||s||_0 \tag{3.20}$$

$$\underset{i}{\operatorname{argmin}} ||d_{test} - D_{Red,T} \times s_i||_2^2 \tag{3.21}$$

3.5.5 DTN for Activity Recognition

The aim to use DTN for PWR µ-DS classification is two-fold: first, low-level features and kernel filters in vision tasks are transferable to µ-DS data since they are both 2-D data and the complexity of µ-DS is less than the RGB image (no fine-grained texture features in RGB images); second, utilizing the fine-tuning techniques and pre-trained weights from the ImageNet, we may extract the hierarchical and high-level features using very small number of training samples. To improve the results of non-deep methods, we apply the most conventional and light-weighted Alexnet for µ-DS classification using pre-trained weights from ImageNet. As the AlexNet architecture has been introduced in Section 2.4.3, this section focuses on applying and fine-tuning AlexNet for µ-DS classification.

3.5.5.1 Transfer Vision Knowledge to µ-DS Classification

The simplified AlexNet architecture is shown in Figure 3.3, assuming the AlexNet takes µ-DS X_i^S and the label y_i as inputs.

The convolutional and pooling layers in AlexNet from aim to extract convolutional features of the µ-DS data, denoted as $T_{Conv}(X_i^S)$. Next, in order to predict the class categories explicitly, these spatial sensitive convolutional features are transformed to $Logits(X_i^S) = CLS \circ T_{FC7} \circ T_{FC6} \circ T_{Conv}(X_i^S)$ by two Fully-Connect (FC) layers T_{FC6}, T_{FC7} and the classifier CLS. Applying soft-max function in Eq.(3.22) on $Logits(X_i^S)$, the i^{th} element of the output $Logits_{soft}$ indicates the probability that the input X_i^S belongs to the i^{th} class.

Since the ground-truth label y_i is available, the CE loss L_{CE} in Eq.(3.23) is used to train the networks with the regularization terms on network weights L_{Reg} shown in background Section 2.4.1, where $[j]$ are operations to select the j^{th} element of the vector.

$$Logits_{soft} = \frac{exp(Logits)}{\sum_{j=1}^{n_{class}} exp(Logits[j])} \tag{3.22}$$

$$L_{CE} = -\sum_{j=1}^{n_{class}} y_i[j] \times log(Logit_{soft}[j]) \tag{3.23}$$

$$\min_{T_{Conv}, T_{FC}, CLS} L_{Total} = L_{CE} + L_{Reg} \tag{3.24}$$

$$i_{class} = \underset{j}{\operatorname{argmax}}\ Logits_{soft}[j] \tag{3.25}$$

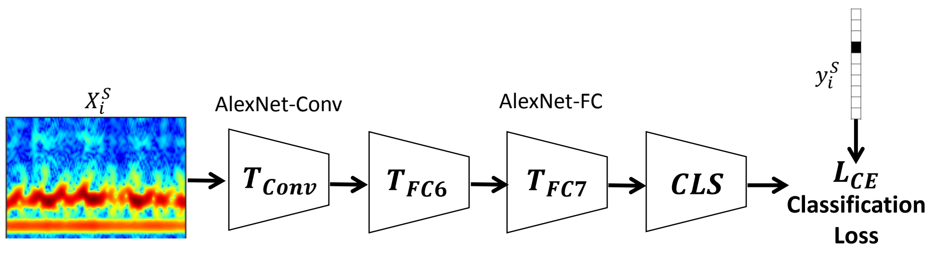


Figure 3.3: DTN for PWR based µ-DS classification using AlexNet.

3.5.5.2 Fine-tuning Techniques in DTN

In this section, we introduce the adopted fine-tuning technique, including the following three phases: *firstly*, we initialize the network using the weights trained on ImageNet, except the *CLS* network (the weight of *CLS* is initialized based on techniques in Section 2.4.2.1); *secondly*, we update network T_{FC7} and *CLS* by back-propagation but shut off the paths to T_{Conv} layers and stop updating them until convergence; *finally*, we update the whole network using a smaller learning rate.

3.5.6 Integrate Sparsity Prior in DTN

In this section, we focus on integrating the sparsity prior in DTN, as sparsity has been investigated and proved beneficial to prevent over-fitting problem and robust to input perturbations [142]. Specifically, we investigate the way to incorporate sparsity prior for automatic controlling the number of neural activation in AlexNet, especially for FC layers. To achieve this, we design the DLL implemented via a serious of recurrent units and replace the conventional FC layer with the newly proposed DLL. Note that we adopt the same fine-tuning technique and details of them can be found in Section 3.5.5.1 and 3.5.5.2.

3.5.6.1 DLL in DTN

Similar to most recognition methods based on DCNN, our network architecture is designed to learn the transformation from the original µ-DS X_i^S to its corresponding label vector y_i in an end-to-end manner. The overall network structure with the replacement of FC7 layer by DLL layer is illustrated in Figure 3.4, where arrows indicate the feed-forward direction. The details of each layer and recurrent units are discussed as follows.

To extract the local features of X_i^S, it first goes through the a stack of conventional convolution and pooling layers. To measure the improvements brought by our proposed DLL and the novel loss function in a fair setting, the configurations of all Conv layers are designed based on the same principles and parameter settings used in AlexNet, as introduced in Section 2.4.3, 3.5.5.1, 3.5.5.2 and the well-known AlexNet paper [8]. In the following, we use the example of replacing the $FC7$ by DLL but it can be used to replace all FC layers (see the results of different replacements in Table 3.4). We denote the features obtained from the $Conv$ and $FC6$ layer as $T_{FC6,i} = T_{FC6} \circ T_{Conv}(X_i^S) \in \mathbb{R}^M$, where $T_{FC6} \circ T_{Conv}$ represents the cascaded transformation in the convolution, pooling and FC6 layers.

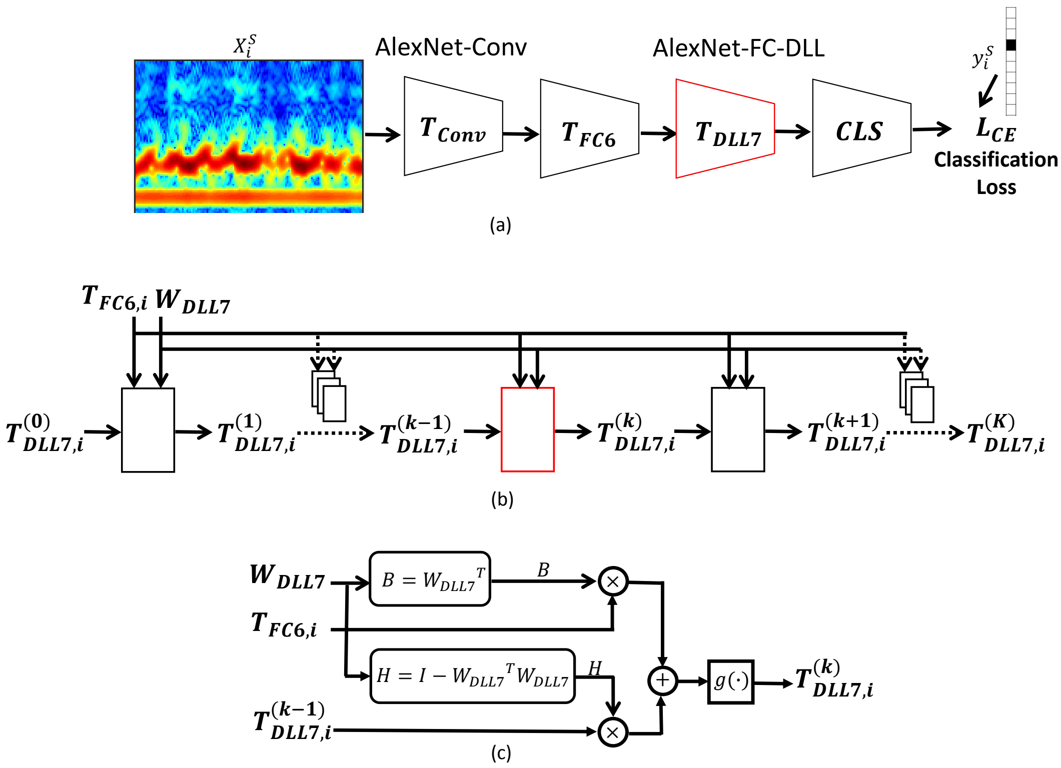


Figure 3.4: (a) DTN for PWR using AlexNet and DLL; (b) Recurrent network design of the highlighted DLL7 layer; (c) Detailed architecture of the highlighted unit in DLL7 layer.

To learn the optimal combination of local features, instead of forwarding the features $T_{FC6,i}$ into FC layers (followed by ReLu), we design the non-linear DLL so as to obtain a sparse feature representation. The DLL is parameterized with a dictionary $W_{DLL7} \in \mathbb{R}^{M\times N}$, composed of a finite number of recurrent units to mimic the sparse coding procedure. Specifically, the dictionary will be ideally learned if the following equation is satisfied for the given extracted feature representation $T_{FC6,i}$:

$$T_{DLL7,i} = arg \min_{T_{DLL7,i}} \left\|T_{FC6,i} - W_{DLL7}T_{DLL7,i}\right\|_F^2 + \lambda \left\|T_{DLL7,i}\right\|_1. \tag{3.26}$$

where $T_{DLL7,i}$ is the corresponding sparse feature representation and $\|\cdot\|_1$ is the ℓ_1-norm function of the input.

In the proposed network, we replace the conventional FC7 layer by the DLL7 in AlexNet, which is a sub-network aimed at enforcing the sparsity prior on the representation besides the CE loss L_{CE} in Eq.(3.23). Specifically, we formulate our loss function used for integration DLL and DCNN in Eq.(3.27) and 3.28, where Θ includes all parameters in AlexNet except the FC7 layer and the parameters in the DLL layer. λ_{recon} and $\lambda_{sparsity}$ are weights of relevant loss functions and the new term $Logits(X_i^S)$ is defined in 3.28. Conventionally, as in [59], a generic ReLu is used for nonlinear mapping. Since the DLL is designed based on domain knowledge from sparse coding, we are able to adjust the sparsity level of the neuron activation based on the training set, consequently, obtaining a better interpretation of the layer response. The implementation of this layer should be such that it is capable of passing the error differentials from its outputs to inputs during back-propagation to update the dictionary. The detailed description of the DLL and its relevant optimization rules are discussed in the Section 3.5.6.2.

$$\min_{\Theta} L_{overall} = L_{CE} + \lambda_{recon} \left\| T_{FC6,i} - W_{DLL7} T_{DLL7,i} \right\|_F^2 + \lambda_{sparsity} \left\| T_{DLL7,i} \right\|_1 . \tag{3.27}$$

$$Logits(X_i^S) = CLS \circ T_{DLL7} \circ T_{FC6} \circ T_{Conv}(X_i^S) \tag{3.28}$$

3.5.6.2 Integration of DLL with DCNN

This section introduces details of the fast yet accurate DLL which is inspired by the LISTA [38] and the dictionary learning method [143] (see background Section 2.3 for details). As shown in Figure 3.4(b), given the input data from previous layers $T_{FC6,i}$, the DLL is able to compute the final output sparse codes $T_{DLL7,i}{}^{(K)}$ efficiently using K stacked recurrent units (rectangular boxes), each of which exhibits a similar function to that of an iteration in the ISTA algorithm. In Figure 3.4 (c), we represent such adjacent recurrent units in a data flow graph, where the rectangular box represents the recurrent unit, a node represents a variable and the directed edge represents the flow between two variables. The operations between the adjacent $(k-1)^{th}$ and $(k)^{th}$ units in the data flow graph can be represented by Eq. (3.29) to (3.31), where $t_i^{(k)}$ represents the learned threshold vector in k^{th} unit, matrix B and S are the linear operator matrix parameterized by weight matrix of FC6 layer.

$$B = W_{DLL7}^T,\ H = I - W_{DLL7}^T W_{DLL7} \tag{3.29}$$

$$z_i{}^{(k)} = H \times T_{DLL7,i}^{(k-1)} + B \times T_{FC6,i} \tag{3.30}$$

$$T^{(k)}_{DLL7,i} = g(z_i{}^{(k)}) = max(\left|z_i^{(k)}\right| - t_i^{(k)}, 0)\frac{z_i^{(k)}}{\left|z_i^{(k)}\right|} \tag{3.31}$$

Since the dictionary is shared among recurrent units within a DLL, the parameters in each DLL can then be represented as $\Theta_{DLL7} = \{W_{DLL7}, \{t_i^{(k)}{}_{k=1}^{k=K}\}\}$. To further ensure the effective non-linear properties of the function $g(\cdot)$, we enforce $t_i^{(k)}$ to be positive by taking the absolute value as the thresholds. From the implementation side, we consider the dictionary W_{DLL7} as a shared parameter implicitly in each recurrent unit and propose to compute the gradient of the loss function $L_{overall}$ with respect to W_{DLL7} using the chain rule and back-propagation:

$$\frac{\partial L_{overall}}{\partial W_{DLL7}} = \sum_{k=1}^{K} \frac{\partial L_{overall}}{\partial T^{(k)}_{DLL7,i}} \frac{\partial T^{(k)}_{DLL7,i}}{\partial z_i^{(k)}} \frac{\partial z_i^{(k)}}{\partial W_{DLL7}}, \tag{3.32}$$

where

$$\frac{\partial L_{overall}}{\partial T^{(k)}_{DLL7,i}} = \begin{cases} \prod_{k+1}^{K} \frac{\partial L_{overall}}{\partial T^{(k)}_{DLL7,i}} \frac{\partial T^{(k)}_{DLL7,i}}{\partial z_i^{(k)}} \frac{\partial z_i^{(k)}}{\partial T^{(k-1)}_{DLL7,i}} & if\, k < K \\ \frac{\partial L_{overall}}{\partial T^{(k)}_{DLL7,i}} & if\, k = K \end{cases} \tag{3.33}$$

It is also worth noting three points: first, $\frac{\partial L_{overall}}{\partial T^{(k)}_{DLL7,i}}$ can be calculated based on the overall objective function (3.27); Second, the q^{th} element of $\frac{\partial T^{(k)}_{DLL7,i}}{\partial z_i^{(k)}}$ is set to 0 if the q^{th} element of $T^{(k)}_{DLL7,i}$ is zero, otherwise, it is set to 1. Thirdly, $\frac{\partial z_i^{(k)}}{\partial T^{(k-1)}_{DLL7,i}}$ can be calculated based on Eq. (3.30) and (3.31) and we use the ℓ_1-norm as an approximation of ℓ_0-norm in Eq.(3.31) to enable standard back propagation in our implementation.

Once $\frac{\partial L_{overall}}{\partial W_{DLL7}}$ is calculated, the dictionary W_{DLL7} in the DLL layer can be updated by SGD. To make this new layer compatible with the other layers in the current DCNN framework, we need to consider how the loss function can be back-propagated through this DLL layer to the previous layers. We need to calculate $\frac{\partial L_{overall}}{\partial T_{FC6,i}}$, and once it is obtained, we can perform standard back propagation [59] to update the CNN parameters in the previous layers. In fact, $T_{FC6,i}$ is similar to the case of dictionary W_{DLL7}, and can also be obtained by use of the chain rule.

From the perspective of the overall network architecture, different DLLs (cascaded in series) are parametrized by separate dictionaries and all the parameters within the overall network can be trained by standard backpropagation. It will be shown in the experiments that with the integration of the newly proposed DLL, the overall structure is able to generate better recognition performance than can the conventional DCNN structure.

3.6 Results

In this section, the system design of PWR and the implementation of the real-time phase extraction method are described in Section 3.6.1. Experiments and results in four scenarios are described in Section 3.6.2, including indoor and through-wall scenarios using both the Wi-Fi beacon and data transmission signals. Finally, we present the activity recognition results in Section 3.6.3 and evaluate the SRC and the novel DTN methods using the six-motion dataset.

3.6.1 PWR System Design and Implementation

The PWR used in this section utilizes a Software Defined Radio (SDR) architecture. In this system, two Universal Software Radio Peripheral (USRP) N210s synchronized with a MIMO cable are used to down-convert the RF Wi-Fi signals centered at 2.462GHz, as shown in Figure 3.5 (a) and (b). An FPGA (Xilinx Spartan 3A-DSP 3400) and a 100Ms/s, 14-bit analog digital conversion are used to digitize the signal and with the FPGA function diagrams shown in Figure 3.5 (c). The data are recorded and transferred into a laptop via gigabit Ethernet port for real time processing.

The reference and surveillance channel antennas used in the indoor experiments are both log-periodic PCB antennas, with 5 dbi gain and 60 degree beam-width [144]. In the through-wall experiments, we used two 15 dbi gain Yagi antennas, with a beam-width of 32 degree [145]. For the Wi-Fi access points, we used an Edimax EW-7416APn, with two omni-directional antennas of 3 dbi gain [146].

In the case of both beacon and data signals, we sample the Wi-Fi signal using 2 MHz sampling rate and use a 0.5 second integration time with 0.4 second overlapping time for cross-correlation operations. These parameters are chosen based on trials and error in the real-world experimental environment, by balancing the integration time and

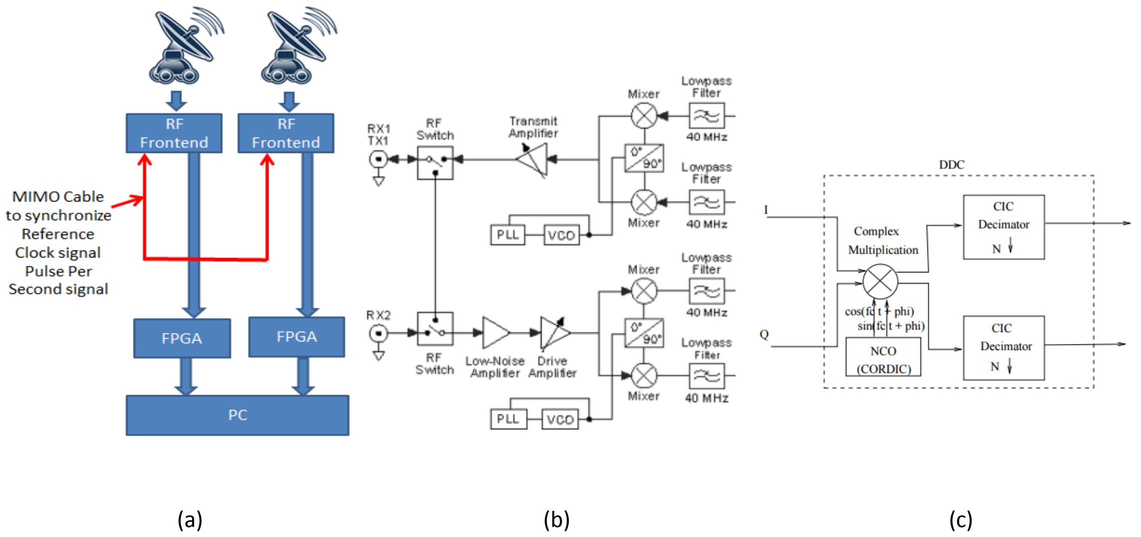


Figure 3.5: (a) Two-channel PWR system; (b) RF front-end daughter board; (c) FPGA of USRP device.

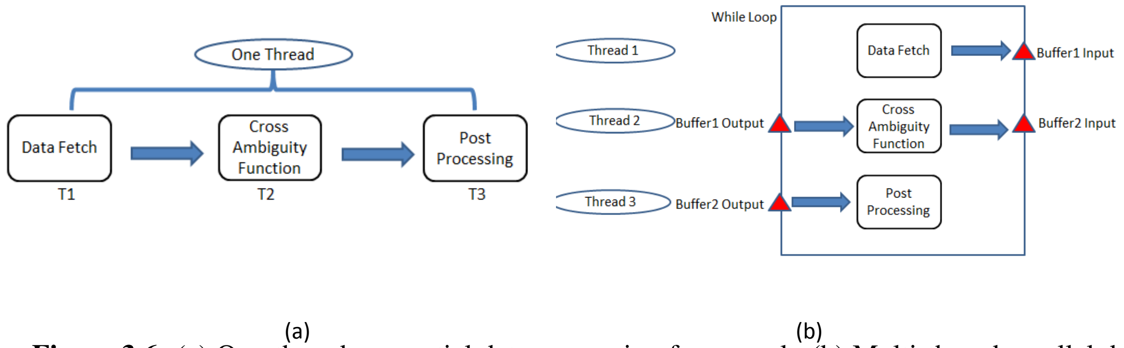


Figure 3.6: (a) One-thread sequential data processing framework; (b) Multi-thread parallel data processing.

the output frame rate. The batch processing with a pipeline flow in Labview is used for real-time implementation [135] including the data buffer storage and the multi-thread processing. The whole framework can be achieved efficiently by the shift register and the while loop without missing any data. More specifically, Figure 3.6 (a) shows the one-thread processing stream, including three sequential procedures: Data Fetch, Cross Ambiguity Function and the Post Processing. In such sequential procedures, the total time for each cycle will be $T_1 + T_2 + T_3$. While in Labview, a multi-thread parallel processing framework can be set up as shown in Figure 3.6 (b), where in the while loop, three threads are assigned to three sub-functions respectively. In addition, two buffer registers are utilized to store the data for next-stage processing. Using this framework, the delay time is reduced in a large margin to $max(T_1, T_2, T_3)$, without missing any data.

For detections based on beacon signals, only 10 batches are used while for the one based on data signal, 50 batches are used. It is worth mentioning that the beacon burst rate in conventional access point setting can be less than 20Hz but for investigating the breathing detection capability, we select the highest rate in experiments.

3.6.2 Results of Breathing Detection using PWR

3.6.2.1 Line of Sight (LoS) Breathing Detection

The indoor experimental scenario is shown in Figure 3.7 (a). There are four test positions (from P1 to P4) to allow investigation of different geometries and we plot the bi-static triangles and the aspect angles related to the four positions in Figure 3.8. During the experiments, the same subject sat on a chair and breath normally in each position.

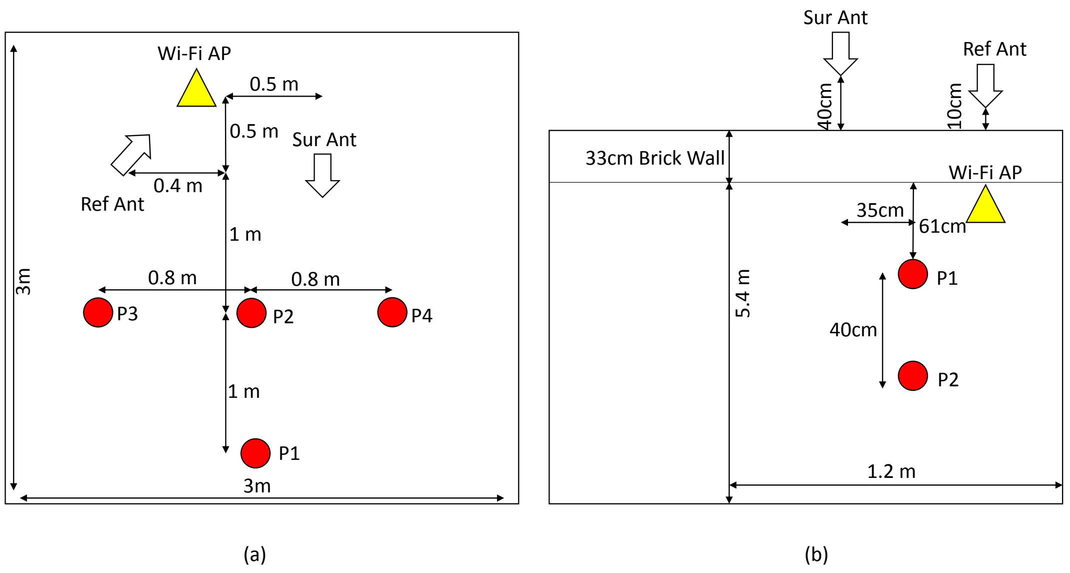


Figure 3.7: (a) Indoor experiment scenario; (b) Through-wall experiment scenario.

Table 3.1: Phase variations of the four test positions from P1 to P4 based on Figure 3.10 with the corresponding aspect angle and cosine value of the aspect angles.

Positions	Average Phase Variations (rad)	Aspect Angle, θ_{aspect} (degree)	$cos(\theta_{aspect})$
P1	1.05	7	0.993
P2	0.96	13	0.974
P3	0.80	41	0.755
P4	0.86	22	0.927

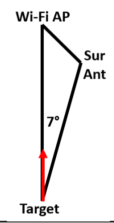
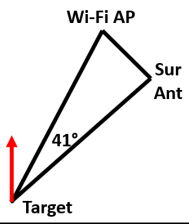
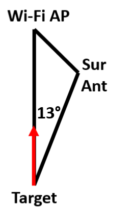
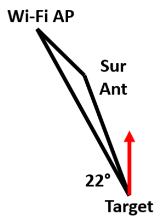
Position	Bistatic Triangle	Angle	Position	Bistatic Triangle	Angle
P1		7°	P3		41°
P2		13°	P4		22°

Figure 3.8: Bi-static triangle and the bi-static aspect angles in the four test positions.

Figure 3.9 shows the phase extraction result for breathing detection using the Wi-Fi beacon signals only. The phase variations caused by the chest-wall movements can be clearly seen in the text part of Figure 3.9 (a) . The pattern is examined and verified that inhaling is found to correspond to decreases in the phase while the increasing phases are observed along the exhaling. It can also be seen that approximately 8 to 9 cycles of breathing are detected within around 40 seconds for all testing positions. These correspond to normal breathing rates and have been verified by video recordings. Although the targets are breathing normally, the magnitudes of the phase variations from the four positions are different. In addition, it can be seen that even for the same position, the phase variations vary slightly from one breathing cycle to the next. Here, a number of potential reasons for these variations are described in the following. *Firstly*, the wavelength of Wi-Fi signal is around 12 cm so even slight changes in the geometry due to body movement will change the phase variation. For example, from Figure 3.9 (a) and (b), the difference between phase deviations is approximately 1 rad, which equates to 1 cm difference in chest movement. *Secondly*, different parts of the chest may be detected

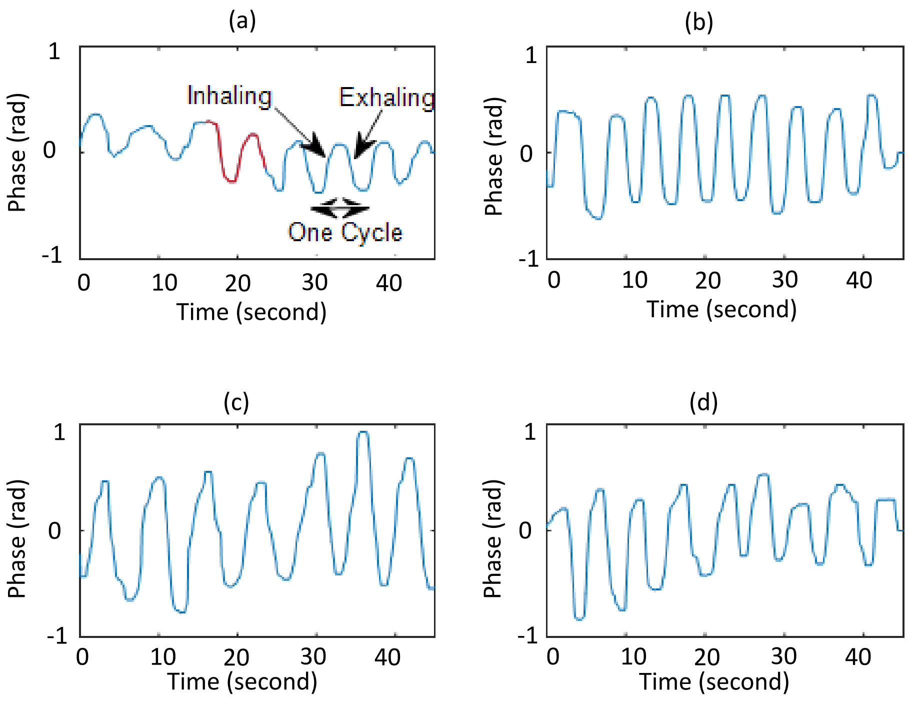


Figure 3.9: Breathing detection using beacon signals at four positions; (a),(b),(c) and (d) correspond to results obtained from P1 to P4 respectively; red curves in (a) and (c) indicate the phase variations caused by irregular chest-wall movements.

which may alter the phase deviation somewhat. Finally, although the targets are breathing normally, the chest movement is unlikely to be absolutely regular from one cycle to the next and bulk body movements will cause larger immediate phase changes, such as the red curves in Figure 3.9 (a) and 3.10 (c).

Figure 3.10 shows the same experiment using data signals. In general, the results are similar to those found using the beacon signals. However, it seems that the phase variations are smoother when using data transmissions as illustrated by comparing Figure 3.9 (a) and 3.10 (a). The likely reason for this has been discussed in previous paragraph.

Based on the experimental results, we estimate the average maximum phase variation within a breathing cycle in the four positions in Table.3.1 and compare with their aspect angles in Figure 3.8. These values are calculated manually based on estimating the difference between every two consecutive maxima and minima phase values. In general, it seems that increasing the bi-static aspect angle decreases the phase variations, which agrees with the bi-static Doppler theory. More specifically, phase variation at

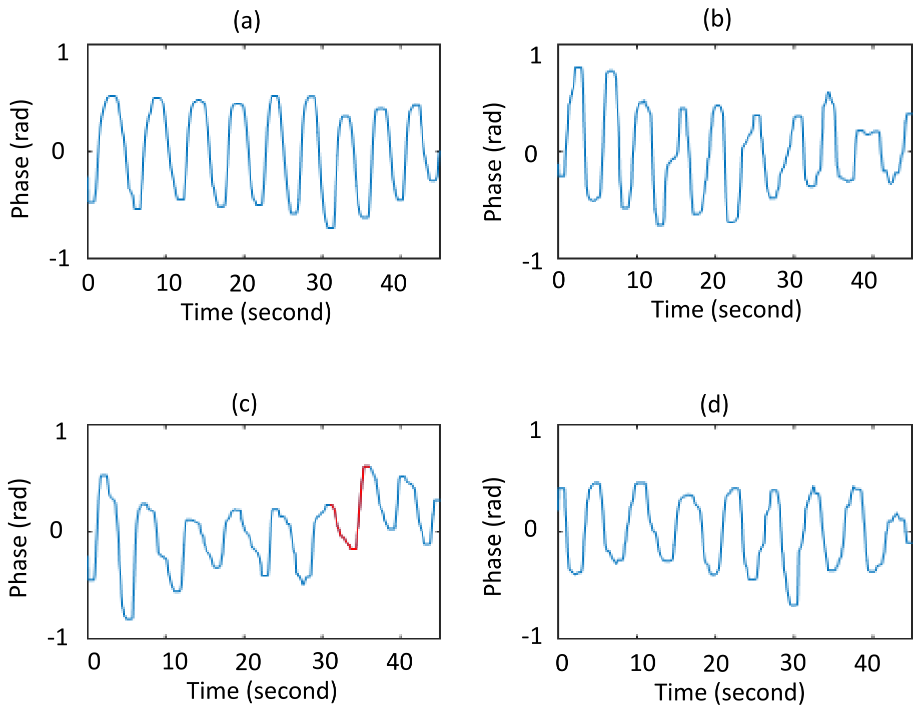


Figure 3.10: Breathing detection using data signals at four positions; red curve in (c) indicates the phase variations caused by irregular chest-wall movements.

position P1 is the biggest corresponding to the smallest aspect angle which is 7 degree.

3.6.2.2 Through-Wall Breathing Detection

The through-wall experimental scenario is shown in Figure 3.7(b) and the house with a standard brick/block cavity wall is used (33cm thickness). The detection zone is approximately 1.2m by 5.4m inside the house and a continuously transmitted data Wi-Fi signal is used. Both the reference and surveillance channel antennas are placed outside the house, with 10 cm and 40 cm distances from the wall respectively. The Wi-Fi access point is located inside the room, at a height of 1.15m and two test positions (P1 and P2) are located in the room at a range of 61 cm and 101 cm respectively from the wall. It is worth noting that both P1 and P2 are horizontally adjusted to be at approximately on the bore sight of the surveillance and reference antennas. During these experiments, the target is stationary and breathing normally. The detection results using data burst signals are shown in Figure 3.11. The phase variation patterns are similar to those seen in Figure 3.9 and 3.10 and the breathing pattern can still clearly be seen. However, it seems that

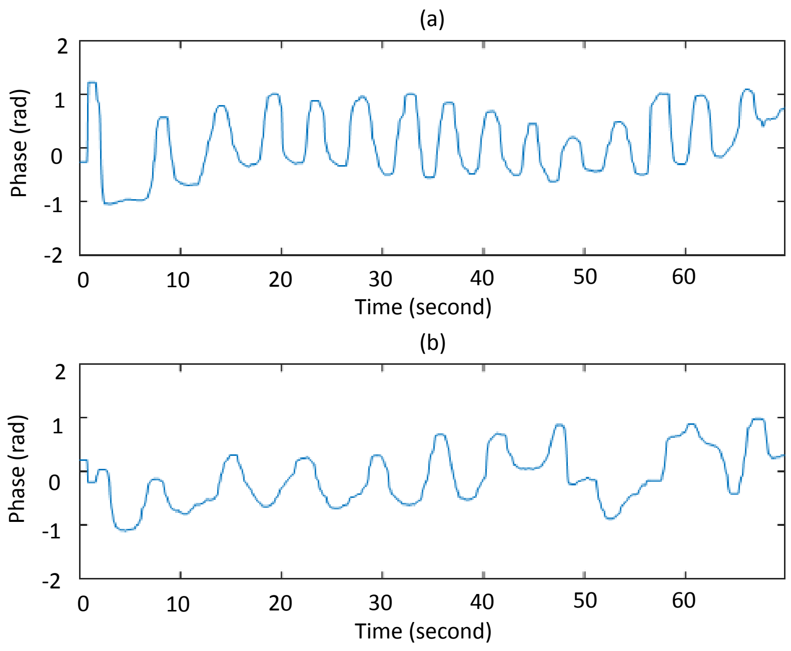


Figure 3.11: Breathing detection results in through-wall scenario (a) at P1 (b) at P2.

when target is at P2, the phase variation pattern is not as stable as the one at P1, which is likely to be because of the weaker signal strength caused by the longer distance of P2.

3.6.3 Results of Activity Recognition using PWR

In this section, we first show the PWR µ-DS and give a brief introduction. Next, we illustrate the results of the proposed start and end point detection method. Next, we design three experimental settings where 20%, 40% and 60% of the dataset are used for training. Finally, for fair comparisons, we evaluate the SVM, SRC with PCA features and the DTN based methods listed in Table 3.2.

3.6.3.1 Overview of PWR µ-DSs and Analysis

In this section, a µ-DS dataset is collected including six daily activities defined in Table 3.3 and we show the corresponding µ-DS in Figure 3.12. The six µ-DS exhibit different patterns and the following describe the visual discriminative characteristics:

- The maximum Doppler shift
- Time duration of the µ-DS

Table 3.2: Methods under evaluation for μ-DS classification and their descriptions.

Methods	Description
PCA Feature + SVM	We extract PCA feature as introduced in Section 3.5.3.2 and apply SVM classifier.
PCA Feature + SRC	We extract PCA feature as in Section 3.5.3.2 and apply SRC classifier.
AlexNet+FC+FT	We only fine-tune the final FC layers and classifier and keep the Conv layers' weights unchanged.
AlexNet+WholeNet+FT	We fine-tune the whole network using the strategy introduced in Section 3.5.5.2.
AlexNet+DLL6	We fine-tune the whole network and replace the FC6 with our newly proposed DLL6.
AlexNet+DLL7	We fine-tune the whole network and replace the FC7 with our newly proposed DLL7.
AlexNet+DLL6+DLL7	We fine-tune the whole network and replace FC6 and FC7 by our DLL6 and DLL7.

Table 3.3: Dataset description of the six activities in PWR experiments. M1(20) indicates the first motion including 20 μ-DS samples.

Activity and Index	Description
M1(20)	Subject pickes up from the ground and stand up
M2(20)	Subject sits down on a chair
M3(20)	Subject stands up from a chair
M4(57)	Subject falls down onto the mattress
M5(62)	Subject stands up after falling
M6(10)	Subject lies on a mattress first then gets out of it

- Does Doppler frequency traverses from negative to positive or just negative / positive
- The strength of the zero Doppler line caused by the DSI or multipath.

In general, the maximum Doppler frequencies of these six μ-DS range from 2Hz to 4.5Hz and motions with different maximum Doppler frequency shifts will help distinguish different signatures. The second discriminative feature relates to the relative direction of motion, indicated by the sign (positive or negative) of the Doppler shift frequency: some motions induce Doppler frequencies that transverse from positive to negative, (e.g. M1 and M2), while others induce only positive or negative Doppler frequencies (M3, M4, M5 and M6). Although M1 and M2 both have the similar patterns (from positive to negative), the time duration of each signature segment increases the discrimination, such

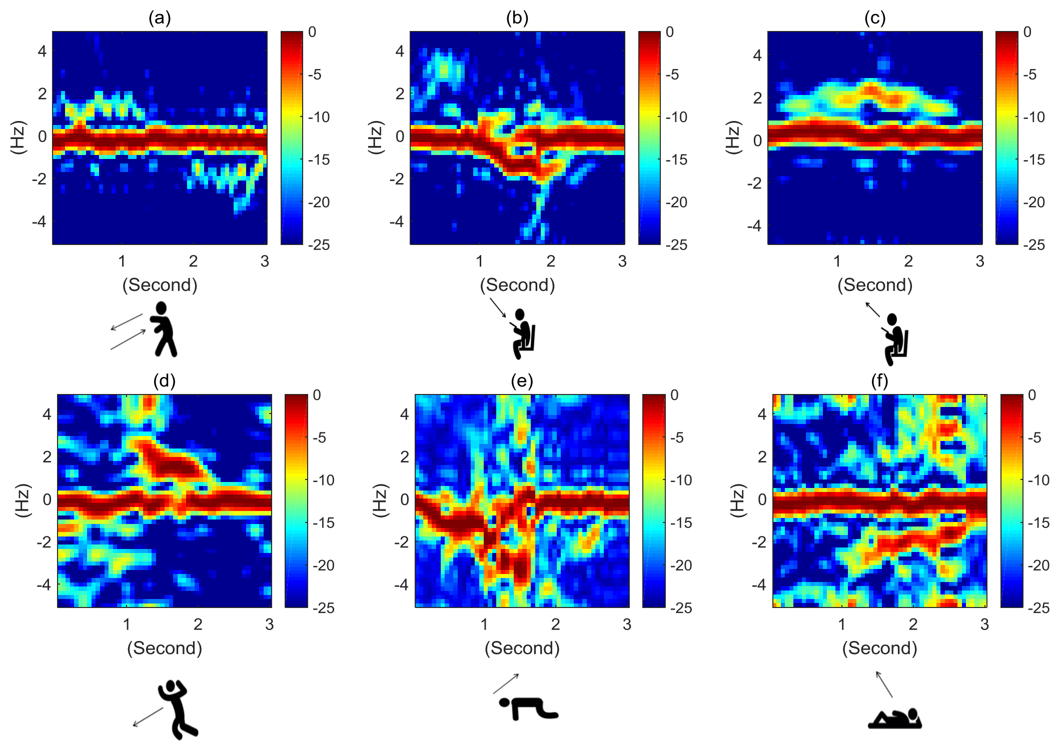


Figure 3.12: Overview of PWR µ-DS in dB scale. (a) to (f) are the µ-DS of motion 1 (M1) to 6 (M6) in Table 3.3.

as the shorter duration of positive Doppler frequency in M2 than the positive Doppler frequency in M1. The final distinguishable feature is the presence of the zero Doppler line during the motion. A clear example is the comparison between the M5 and M6, where the Doppler signature patterns are similar, but the latter has a strong zero Doppler line while former does not. The reason why M5 exhibits no zero Doppler line is when the target gets out of the floor, the bulk motion blocks the direct signal to the receiver. Although these selected empirical features agree closely with the intuitive visual interpretation, obtaining these features accurately requires complex feature selection methods such as detecting accurate Doppler patterns and estimating the mentioned empirical features. In addition, these methods are prone to be erroneous and have a big influence on the classification outcome. Anyway, it will always be difficult to represent a high-dimensional dataset using just four to six empirical features. Note that we further analyze why the DSI effect is missing at some time in the spectrogram, especially in 3.12 (e). This is because we are normalizing the frequency vector at each time instance and frequency component related to the Doppler movement is stronger than the power of the zero Doppler frequency. This is possible, since when the human target may block the direct path from transmitter to the receiver of the PWR system. This also can be regarded as one of the feature to differentiate different activities.

For the classification scheme using SRC, we utilize the reduced-dimension data vectors defined in Eq.3.18 and 3.19 as input. For selecting features for DCNN, we simply input the original μ-DS according to methods introduced in Section 3.5.5.1.

3.6.3.2 μ-DS Start and End Point Detection Result

In this section, we described the start and end point detection results in Figure 3.13, corresponding to the μ-DS results in Figure 3.12. From the observation and comparison between Figure 3.12 and 3.13, Figure 3.13 (a), (b), (c) and (d) give good and continuous detection results (red dots), however, the detections are not continuous in (e) and (f). Through comparisons of μ-DS and detection results in Figure 3.13 (e), it seems that μ-DS during 1.6 to 2.1 second are of low energy although we are certain that the target is moving. The potential reasons may be that the target torso is out of the antenna beam during this time period. In the detection result in Figure 3.13 (f), we obtain some outliers around 0 to 0.2 second and the Doppler energies within that period are relatively high which corresponds to the high energy around 4.2Hz around 0.1 second in Figure 3.12 (f). Since we know no movements during that period, we hypothesize that this is the frequency flipping phenomenon, mainly caused by the relatively small slow-time batch numbers in calculating the Doppler spectrogram or the cyclic properties of the Wi-Fi signals.

In addition, we analyze the number of the total false detection points (including both missed and false alarm detection) using different threshold scaling factors, compared with the ground-truth one generated by manual labeling. Therefore, we use the average number of missed and false alarm detection bins as the evaluation metrics in Figure 3.14, with the scaling factor γ in the following set $\{1.05, 1.1, 1.15, 1.2, 1.25, 1.3\}$. The general trend is that with the increasing scaling factors, the missed detection bin number increases while the false alarm detection bins decreases. In addition, we plot the total false detection in green dots as the sum of the missed and false alarm detection numbers. If the system requirements focus more on low missed detection, we'd better choose the small scaling factor like 1.05, however if the low false alarm rate is essential, we'd better choose 1.15 as the proper one, which is the trade-off solution and balances the relatively small total false and the false alarm detection bin number. The discontinuous problem may be solved better if some tracking filters are applied on the results, which is out of the scope of the thesis. The results generated in this section adopts the 1.15 as the final scaling factor. For detecting the start and end point index, we choose the smallest and the largest index of the detected set as the start and end point index.

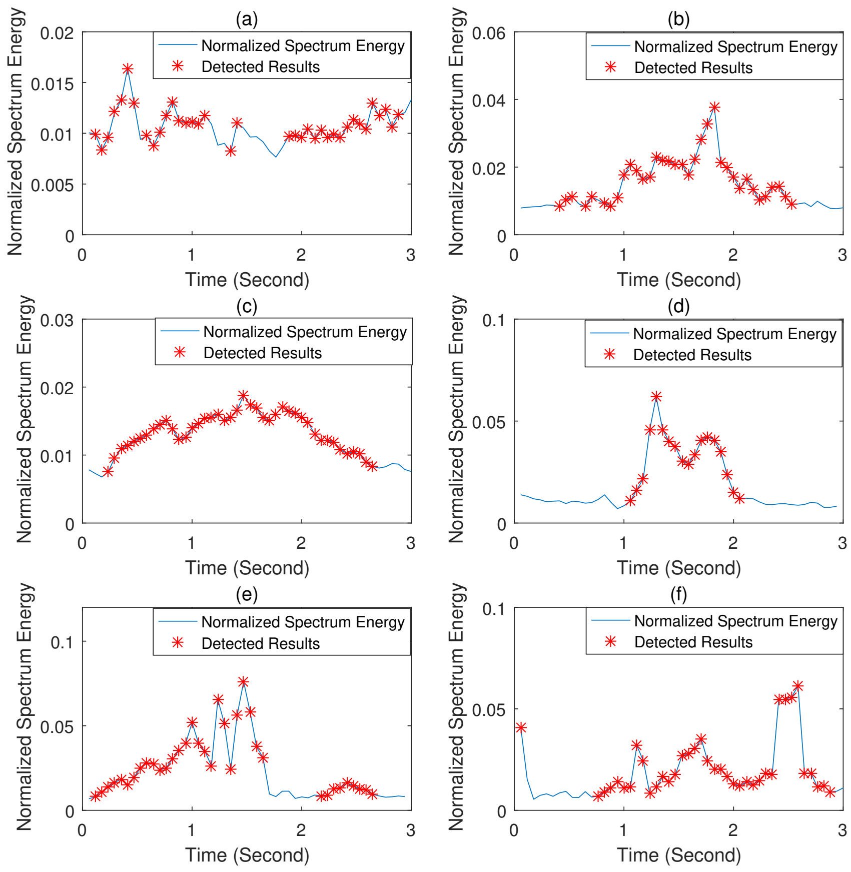


Figure 3.13: Detection results of the proposed start and end point detection method. Blue line indicates the time history of normalized spectrum energy of the µ-DS in Figure 3.12. The red star represents the detected time bins with active µ-DS.

3.6.3.3 Activity Recognition Results

In Table.3.4, compared with the widely utilized SVM method, SRC with PCA features outperforms SVM in average 28% in the recognition results. Compared with the SRC using PCA features, *AlexNet+FC+FT* method achieves better performance than SRC

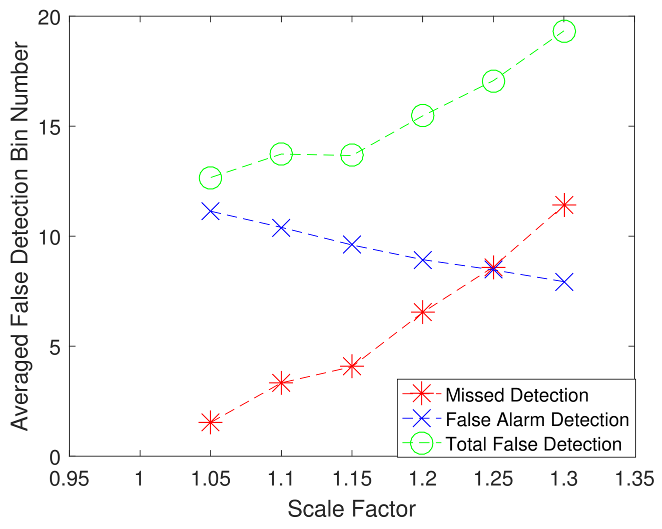


Figure 3.14: Overview of false detection results: red, blue and gree line and markers indicate the missed, false alarm and the total false detection numbers respectively.

using 40% and 60% of the data for training, however SRC outperforms it by around 1.7% using 20% of the dataset for training. In addition, other DTN methods e.g. *AlexNet+WholeNet+FT* fine-tuning the whole networks achieve at least 11% improvement than SRC. This may suggest that shallow method, like SRC is suitable for handling small training dataset and DTN framework achieves the superior performance compared to shallow methods only if the whole network is fine-tuned. This may further indicate that adapting and changing the local feature in Conv layers is essential for DTN.

Table 3.4: Activity recognition results for different features and classifiers, percentage in (%).

Feature + Classifier	Train on 20%	Train on 40%	Train on 60%
PCA + SVM	32.9	57.0	60.0
PCA + SRC	82.0	81.0	88.0
AlexNet+FC+FT	80.3	90.5	93.5
AlexNet+WholeNet+FT	86.3	94.7	98.7
AlexNet+DLL6	**89.5**	**96.0**	99.5
AlexNet+DLL7	88.9	95.3	**100.0**
AlexNet+DLL6+DLL7	88.7	95.7	99.2

For all experimental settings and methods in Table.3.2, AlexNet using DLL layers achieves the best performance, outperforming the second best method without DLL replacements by 3%, 1.3% and 1.3% for training on 20%, 40% and 60% of dataset respectively. This proves our assumption that moderate sparsity of neural activation induce better discriminative capability and prevent the potential over-fitting problem. The reason why the result improvements of DLL layers decrease is obvious, since more training data ease the recognition task. In addition, we conduct the ablation study investigating replacement of different FC layers with DLLs and verify the performance of cascaded DLL6 and DLL7. In comparison of the three DLL replacement strategies, including *AlexNet+DLL6*, *AlexNet+DLL7* and *AlexNet+DLL6+DLL7*, their recognition differences are only 0.6% in average of all experimental results.

We also provide confusion matrix related to these methods in Table 3.5 and due to the space limitations, we only report the confusion matrix using 20% training data. In Table 3.5, SVM with PCA feature achieves the worst results where all test data are assigned to class 6, which gives rise to the lowest recognition results. Through observations of the confusion matrices from Table 3.6 to 3.9, both SRC and DTN methods are confused by μ-DS among [C1, C2, C3] and between [C5,C6]. These can be explained by their visual similarities in among Figure 3.12 (a),(b),(c) and between Figure 3.12 (e),(f). Specifically, μ-DS in C1 and C2 both transverse from positive to negative and with very similar maximum Doppler frequency. In addition, μ-DS in C5 and C6 are similar as their Doppler periods are both 1 second and the maximum frequencies are around 3Hz.

Table 3.5: Confusion matrix of SVM method with PCA features using 20% data for training, percentage in (%).

	C1	C2	C3	C4	C5	C6
C1	**0.0**	0.0	0.0	0.0	0.0	100.0
C2	0.0	**0.0**	0.0	0.0	0.0	100.0
C3	0.0	0.0	**0.0**	0.0	0.0	100.0
C4	0.0	0.0	0.0	**0.0**	0.0	100.0
C5	0.0	0.0	0.0	0.0	**0.0**	100.0
C6	0.0	0.0	0.0	0.0	0.0	**100.0**

3.7 Analysis and Discussion

The following sections first analyze and discuss the following two problems for the proposed phase-sensitive signs-of-life detection method:

- The limitations of the phase extraction method.

Table 3.6: Confusion matrix of SRC with PCA features,using 20% data for training, percentage in (%).

	C1	C2	C3	C4	C5	C6
C1	**100.0**	0.0	0.0	0.0	0.0	0.0
C2	18.8	**75.0**	6.2	0.0	0.0	0.0
C3	50.0	6.3	**43.8**	0.0	0.0	0.0
C4	6.5	0.0	4.3	**89.1**	0.0	0.0
C5	0.0	0.0	0.0	2.0	**96.0**	2.0
C6	12.5	0.0	0.0	0.0	75.0	**12.5**

Table 3.7: Confusion matrix of AlexNet+FC+FT using 20% data for training, percentage in (%).

	C1	C2	C3	C4	C5	C6
C1	**88.2**	11.8	0.0	0.0	0.0	0.0
C2	6.3	**43.8**	43.8	0.0	0.0	6.3
C3	0.0	0.0	**100.0**	0.0	0.0	0.0
C4	6.5	2.2	0.0	**76.1**	15.2	0.0
C5	0.0	0.0	0.0	10.0	**78.0**	12.0
C6	0.0	0.0	0.0	0.0	50.0	**50.0**

Table 3.8: Confusion matrix of AlexNet+WholeNet+FT, using 20% data for training, percentage in (%).

	C1	C2	C3	C4	C5	C6
C1	**94.1**	5.9	0.0	0.0	0.0	0.0
C2	0.0	**68.8**	31.2	0.0	0.0	0.0
C3	0.0	0.0	**100.0**	0.0	0.0	0.0
C4	6.5	0.0	2.1	**84.8**	2.1	4.3
C5	0.0	0.0	0.0	2.0	**86.0**	12.0
C6	0.0	0.0	0.0	0.0	12.5	**87.5**

Table 3.9: Confusion matrix of AlexNet+DLL7 using 20% data for training, percentage in (%).

	C1	C2	C3	C4	C5	C6
C1	**88.2**	5.9	0.0	0.0	0.0	5.9
C2	12.5	**75.0**	12.5	0.0	0.0	0.0
C3	0.0	0.0	**100.0**	0.0	0.0	0.0
C4	6.5	0.0	0.0	**84.8**	4.3	4.3
C5	0.0	0.0	0.0	0.0	**94.0**	6.0
C6	0.0	0.0	0.0	0.0	12.5	**87.5**

- The stability and continuity of phase information to detect breathing.

Secondly, we analyze and discuss the DLL in DTN framework including sparsity results of neural activations and the effect of recurrent unit number in DLL.

3.7.1 Limitations of the Phase Extraction Method

First, we use an ideal model to explain the phase differences between the reference and surveillance signals as follows:

$$sur(t) = ref(t-\tau) \times A \times e^{j\phi(t)}, \tag{3.34}$$

where τ represent the potential time delay, A is the amplitude of the surveillance signal loss, $\phi(t)$ represents the mixed phases from different clutters in Eq.(3.35):

$$Ae^{j\phi(t)} = \sum_{i}^{N} A_i e^{j\phi_i(t)}, \tag{3.35}$$

where $\phi_i(t)$ and A_i are the phase and amplitude from individual i^{th} reflected clutter. In the model of phase mixture in Eq.(3.34), we analyze the limitations of the phase extraction method and therefore propose the two arguments:

- The success of breathing detection relies on the fact that there are no large movements in the same range bin.

- The success of breathing detection does not depend on the static phase influences, such as DSI and multipath reflections.

Next, we explain the two arguments by presenting the phase mixture phenomenon in Figure 3.15 (a) and (b). In Figure 3.15 (a), we can see that although the magnitude from the wall or multipath (highlighted in red arrow) is stronger than the breathing reflection (highlighted in green arrow), the mixed phase (highlighted in blue arrow) is still able to represent the chest movement but with the average of the static phase. However, in the following Figure 3.15 (b), if there are large movements occurring at the same range bin, then the mixed phase changing circle is combined by two circles including the breathing one and the other representing the changes induced by large movements.

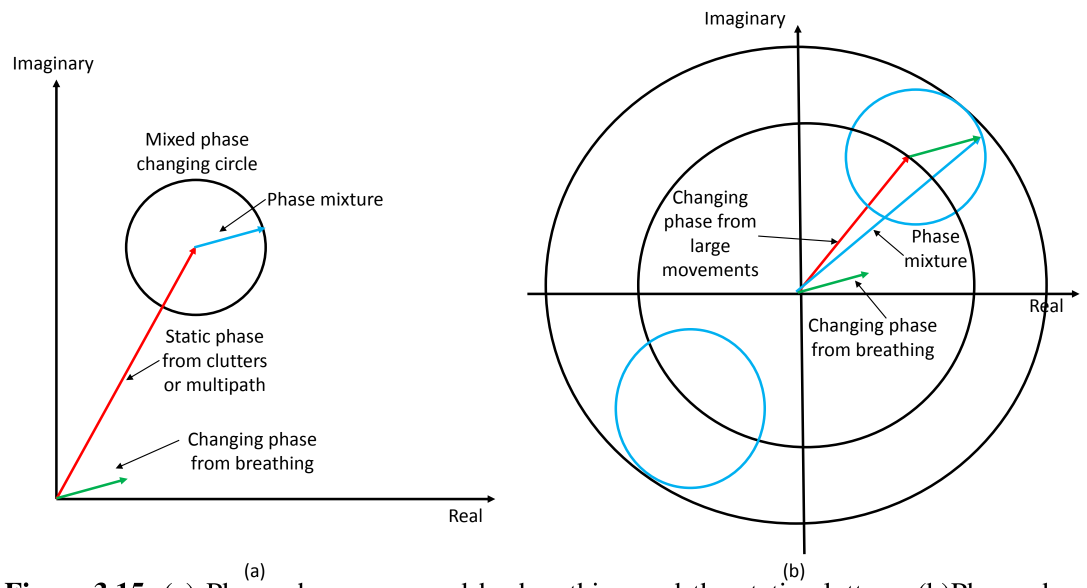

Imaginary
Mixed phase changing circle
Phase mixture
Static phase from clutters or multipath
Changing phase from breathing
Real
(a)
Imaginary
Changing phase from large movements
Phase mixture
Changing phase from breathing
Real
(b)

Figure 3.15: (a) Phase changes caused by breathing and the static clutters; (b)Phase changes caused by breathing and large motions.

In this scenario, it is difficult to separate these two circles, unless using phased array and beamforming techniques.

3.7.1.1 The phase stability for breathing detection

Ten seconds of data relating to a target sitting at position P1 in Figure 3.7 and breathing for around 2.5 cycles were collected for detailed analysis. The relevant phase time history plot is shown in Figure 3.16 and note that we represent the batch number of CAF processing in the x-axis for easily identifying which batch generates the discontinuous phase output. First we divide the 10 seconds (sampled at 2MHz) reference and surveillance signal into 1000 batches as shown in Figure 3.16 (a). For each batch, we subtract the mean of the signals and plot their phases. It is very clear that the up and down of the phase output represent the chest movement forward and backward relative to the surveillance antenna. In addition, we can also observe the three discontinuous points at batch 155, 157 and 874.

Plotting the amplitudes of the segmented signal from these three batches in Figure 3.17 may explain the potential reasons. This figure shows that: fewer high-power data bursts are transmitted in these batches. Based on this, it can be concluded that, the phase output will be able to reflect the chest movement if and only if enough continuous burst signals are collected during each batch processing of CAF.

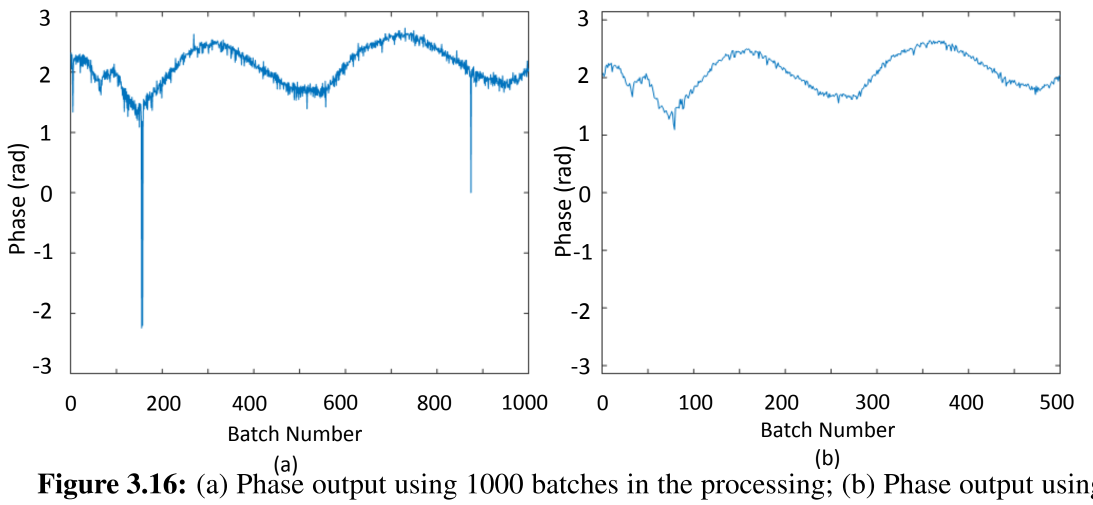


Figure 3.16: (a) Phase output using 1000 batches in the processing; (b) Phase output using 500 batches in the processing.

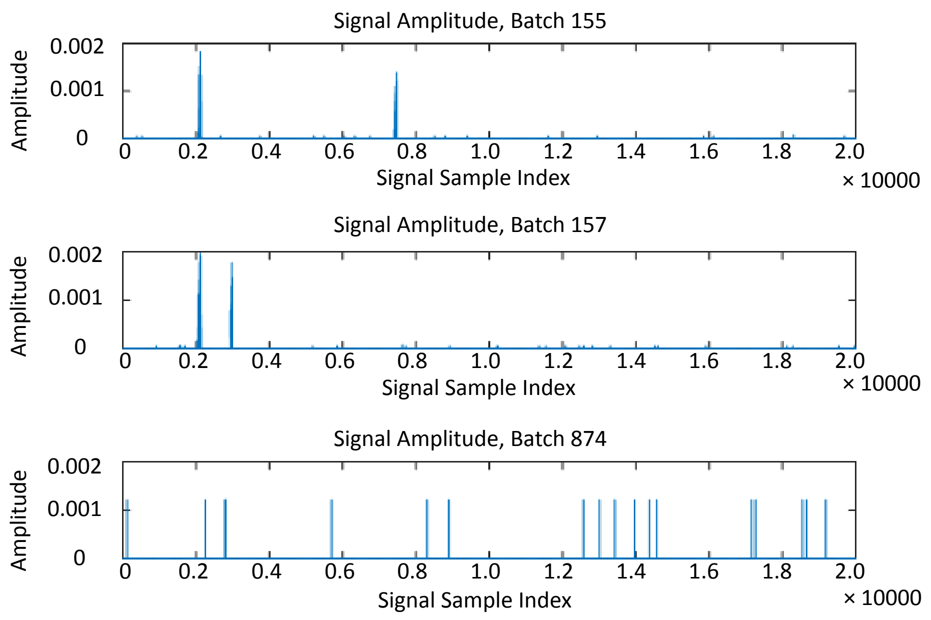


Figure 3.17: Signal amplitude of the three batches that generate discontinuous phase outputs in Figure 3.16(a).

To solve this problem, we divide the whole 10 seconds' data into fewer batches (this ensures to keep more high-power signals in every batch), say 500 and perform the same operations. The phase output is shown in Figure 3.16 (b), where the three discontinuous phase disappear and more continuous phase outputs are obtained.

To sum up, to make the phase output stable and continuous in PWR, we need to be careful about choosing the proper batch number for CAF processing. Based on the Wi-Fi data transmission characteristics, detection of the start and end point of the burst signal is important to extract the stable phase information. In addition, when applying the phase extraction method in the Wi-Fi beacon signal with a burst rate of maximum 20 Hz, the integration time should be increased to ensure that each batch contains enough valid and high-power beacon signal samples.

3.7.2 Sparsity of Neural Activations and the Effect of Number of Recurrent Units in DLL

We analyze the number of recurrent units in DLL by conducting experiments using AlexNet+DLL7, with other parameters fixed but only increasing the number of recurrent units from 1 to 8. The recognition results are shown in Table.3.10. It seems that with the increasing of recurrent units, the recognition rates increases first and then decreases. It suggests that only the moderate sparsity and number of recurrent units can give rise to the most discriminative results in the DTN.

As shown in Figure 3.18, we plot the ℓ_1 norm of the neural activations of FC7 and DLL7 with different number of non-linear units. These neural activations are based on feed-forwading all the test data along 5000 iterations of training without drop-out operation. It can be observed that increasing the non-linear units controls the sparsity and the ℓ_1 norm is decreasing. With different sparsity levels, recognition result using 4 units achieves the best performance and this further indicates that moderate sparsity can increase the discrimination capability of DTN.

Table 3.10: Recognition rates using different number of recurrent units in DLL, evaluated using 20% data for training.

No.Recurrent Units	0 (FC only)	1	2	4	8
Recognition Rates	86.3%	88.1%	88.9%	89.5%	89.0%

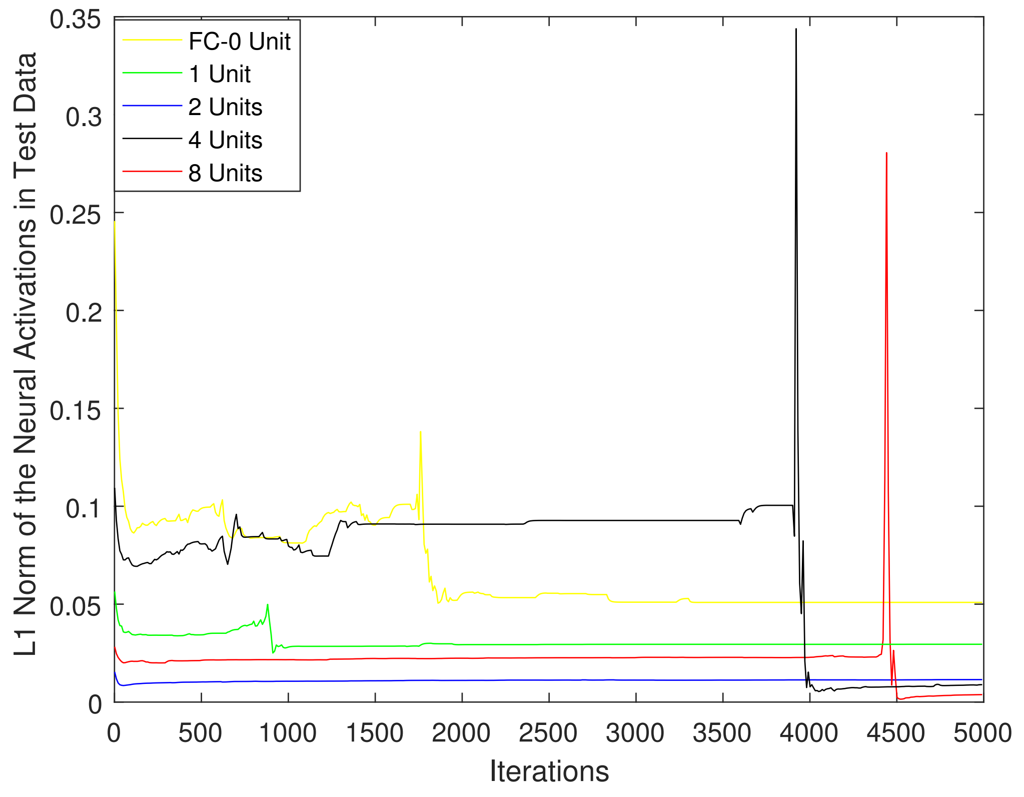


Figure 3.18: ℓ_1-norm of the neural activations of the T_{DLL7} and T_{FC7} using different number of recurrent units.

3.8 Summary

In this chapter, with the aim of advancing PWR Doppler processing for recognizing the daily activities and signs-of-life detection, the work carried out in this chapter makes specific three key contributions to the knowledge base:

1. For signs-of-life detection using PWR, we propose the novel, real-time and phase-sensitive processing to monitor the tiny chest-wall movements. Conventional PWR real-time Doppler processing is limited by the small range changes of chest-wall movements, the required long integration time to achieve the ideal Doppler resolution and the strong DSI effect from the clutters. To tackle these, we utilize the instantaneous Doppler which is the phase output of the cross-correlation result. Our proposed phase-sensitive method is robust to DSI from clutters and does not require long slow-time for the integration in the Doppler processing. This method

is evaluated in various experimental scenarios and we discuss the assumptions, limitations and the potential reasons of these limitations in the discussion section.

2. For activity recognition using PWR, we propose the whole real-time processing pipeline to capture μ-DS of the six daily activities and make them ready for classification. We apply the SRC method with the PCA feature based dictionary. Finally, we compare the SRC with PCA features with other conventional methods for evaluation. Compared with the conventional SVM method with PCA features, SRC outperforms it by 33.7% in average.

3. For activity recognition using PWR, we wish to investigate whether the DCNN method can improve the μ-DS classification performances. It is well-known that applying the DCNN in the small-scale μ-DS dataset is limited by the over-fitting problem. To tackle this, we adopt the DTN framework pre-trained by ImageNet and integrate the sparsity prior in FC layers by replacing it with our newly proposed DLL. This design is beneficial in the following two perspectives:

    - Compared with sparse representation, the dense representations is highly entangled and sensitive to input perturbations [147], as small changes of the inputs will change most of the entries in the feature representation. Therefore, by integrating the the sparse prior in DTN, the FC layers' outputs are robust to perturbations of low-level Conv features. The potential perturbations may be more severe in μ-DS classification as the Conv filters are originally learned for images rather than μ-DS; in addition DTN trained by small-scale training data cannot generalize well to variations of the unseen test data.

    - The DLL integrates the sparsity prior and also implicitly allows variable-size feature representation for different inputs. This is different from the ReLU, which only keeps the positive activation neurons. The variable-size data structure controls the effective dimension of the representation for a given input which regulates the model to prevent over-fitting and eliminate the outlier in small-scale dataset.

The DLL based DTN has been verified to outperform conventional FC layer based DTN by 2% in average.

For signs-of-life detection using PWR, the most recent research [129] partially addressed the limitation of our method discussed in Section 3.7.1. This is achieved by extending our phase extraction framework with an additional module to detect large movements by measuring whether the phase time history is periodic or not. However, no detailed method in [129] has been proposed to separate the small and large phase changes for signs-of-life detection, without losing the small phase changes.

For activity recognition using PWR, we argue that the energy power of Doppler information proposed by Li et al. [110] cannot represent the complete information of the spectrum information and the relevant feature learning is not optimal. The main hypothesized reason is that the 1-D energy power contains much less information compared with the 2-D μ-DS for a classification task. Note that the start and end point detection method developed in Section 3.5.3.1 also relies on the energy power, however, we only use it to detect the start and end point and the later feature learning and classifier design leverage the whole μ-DS. In addition, to address the DCNN initilization methods for radar μ-DS classification with low training sample support, Seyfiolu et al. analyzed the conventional DTN and the auto-encoder methods, however, our newly proposed DLL focuses on improving the DTN performance by incorporating the sparsity prior. In addition, our proposed methods are not only limited to μ-DS but also applicable to other DTN tasks.

Chapter 4

DopNet: A DCNN to Recognize Armed and Unarmed Human Targets

4.1 Introduction

This chapter is focused on training a DCNN from scratch to recognize the armed and unarmed personnel using the multi-static radar μ-DS. Their applications in the context of security, warfare and healthcare have been investigated over a number of years [112, 3, 4]. The challenge of collecting the raw radar data and understanding what action is occurring can be broken down into three key steps. 1) The representation of the raw signals, 2) The features that can be extracted from them 3) The classification algorithm applied to these features. A large number of data representations, features and classifiers methods have been proposed and applied as a series of separate steps.

Due to the data being a time-frequency signal, the spectrogram (details in Section 2.2.3) is the most common method of representing the data via the STFT. This was shown to distinguish human targets movements, e.g. walking, crawling, running etc. or to distinguish human from animals [4, 3, 148]. In addition, other time-frequency representation methods have been applied, e.g. Gabor transform, Wigner-Ville transform, Empirical Mode Decomposition based on Hilbert-Huang transform to extract the time-frequency representation of various human movements [149, 150, 151, 152]. Other approaches have proposed the use of extracting empirical features, such as RCS, Doppler bandwidth, period of motion. In addition, various dimensionality reduction or de-noising methods have been investigated, like SVD method, PCA and sparse representations[3, 4, 112, 109, 108, 149].

As for the selection of classifiers, various research work related to classifiers in machine learning community have been proposed [105, 153, 109, 108]. However, these features and classifiers have not been developed and optimized in the same joint framework. This means that for different applications and conditions, the feature extraction and classifiers may need to be modified and tuned based on subjective experience of the researcher, rather than an objective methodological framework.

In recent years, with the development of hardware facility and computation methods such as Graphic Processing Unit (GPU), DCNN has been proposed firstly to address the ImageNet challenge, to classify an image dataset of more than 10 million images [154]. One of the main advantages is that the feature extraction and the classifier can be jointly learned in the same framework. However, DCNN is well-known for its difficult training from scratch and normally requires large amount of data in the training stage to prevent the overfitting problem.

In this chapter, we propose a modified DCNN trained from scratch called DopNet, to distinguish armed and unarmed walking human targets using the multi-static radar data. The contributions are three-folded:

1. Firstly, we propose two key novel schemes to address the over-fitting problem in training DCNN, including the radar data augmentation in the training stage and a new regularization term balancing the Mahalanobis distance (M-dist) and Euclidean distance (E-dist) of the network weights. We analyze the effect of various factors in the single channel DopNet (SC-DopNet) and evaluate the proposed two schemes. In addition, we compare SC-DopNet results from mono-static radar data with other handcrafted features and classifiers by experimental results.

2. Secondly, we build the multiple channel DopNet (MC-DopNet) similar to SC-DopNet and proposed two fusion methods to jointly optimize the total objective function, called Greedy Importance Reweighting (GIR) method and the ℓ_{21}-Norm method. Note that these two methods are embedded in training the MC-DopNet and parameters of the two methods can be jointly learned under the total optimiza-

tion function. MC-DopNet is an end-to-end learning framework to address the classification of human µ-DS using experimental radar data.

3. Finally, we compare our proposed fusion methods together with MC-DopNet to other conventional data fusion methods with various features and classifiers. Note that we also discuss and conclude in what scenarios the proposed two fusion methods are preferable to be utilized.

4.2 Related Work

In this section, we only review the most relevant work to DopNet, including DCNN based classification methods for human µ-DS. Kim et al. first proposed to train DCNN from scratch to distinguish hand gestures and aquatic movements using mono-static radar [155, 156]. Recently, Kim et al. adopted DTNs for classifying aquatic movements using the trained DCNN network weights from ImageNet dataset and only small part of the network weights are required to be trained [157]. This idea is further evaluated and compared with the one using auto-encoder based pre-training weights in [158] using a small number of radar samples. More recently in healthcare applications, DCNN has also been utilized to recognize falling based on mono-static range-Doppler signatures [159].

To analyze experimental multi-static µ-DS data Using DCNN, Chen et.al [113] proposed to use multiple DCNNs, one for µ-DS from each channel. However, they utilized the DTN framework and fine-tuned only the classification layers. Additionally, when integrating multi-channel representations for the final descision making, they proposed to simply concatenate the representations from multiple channels. Patel et.al [160] proposed to use multi-static µ-DS but adopted a unified auto-encoder network for feature learning.

4.3 DopNet Architecture

There are five main component layers in a DCNN, including the Conv layer, non-linear activation functions, pooling, FC layer and the final Softmax classification layer, as described in Section 2.4. First these layers with their architectures and purposes are illustrated specifically in Section 4.3.1. Next in Section 4.3.2, operations to prevent the overfitting are presented.

4.3.1 DopNet Component and Architecture

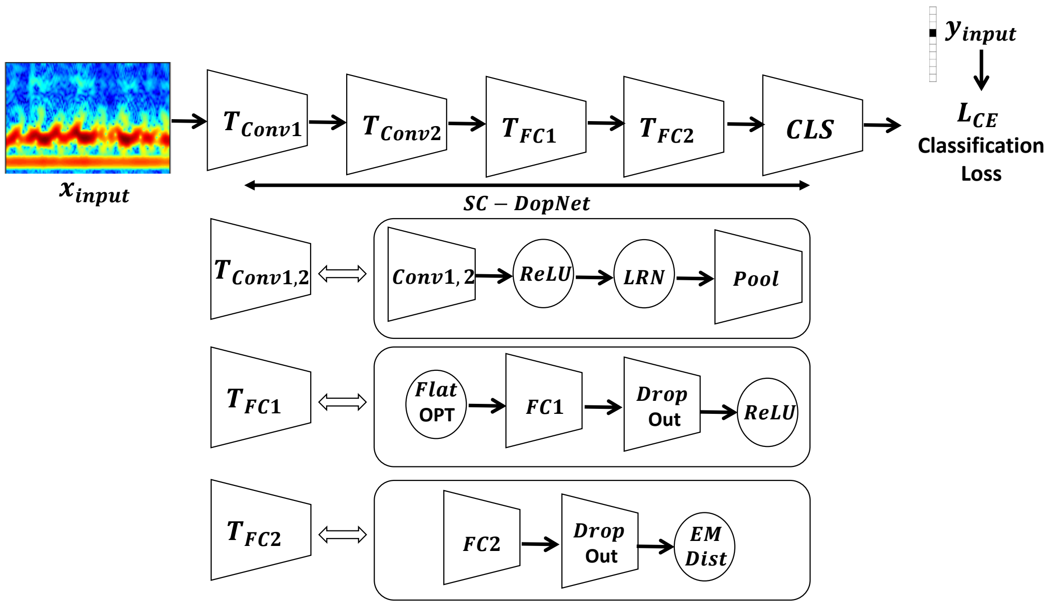


Figure 4.1: Architecture of SC-DopNet.

The network structure of DopNet is illustrated in the Figure 4.1. The first two layers in DopNet are the Conv layers, denoted as *Conv*1 and *Conv*2, composed of 64 and 128 kernel filters with the size of 32×32 and 11×11 respectively. The Conv layers are aimed to select local features of the µ-DS by convolving it with the kernels and generate activation or correlation degree maps. This can be regarded as the Conv layer outputs and these local maps will be concatenated together in a hierarchical manner [161].

To increase the model capacity of the network, we adopt the ReLU in DopNet shown in Eq.(4.1). Note that we apply the ReLU function following each layer output, except the output of the final layer. Additionally, we utilize the Local Response Normalization (LRN) layer developed and verified to reduce the saturation of ReLU function [161]. The idea is introduced by the lateral inhibition, which will normalize the activated map among different kernel filters. The details are shown in Eq.(4.2) where $I^i_{w,h}$ and $O^i_{w,h}$ are the input and output of the LRN layer activated by i^{th} kernel at position w, h after the ReLU layer, R is the radius of the amount of kernels for the normalization, β

can be interpreted as the polynomial parameter chosen empirically from trials and errors. This layer generalizes the network by generating competition among different kernel activations. Note that input of the LRN layer is always chosen as the output after ReLU function.

Even with the non-linear and normalization layer, representation of the *Conv* outputs are redundant and sensitive to the spatial transitions of the inputs. Therefore, the pooling layer is used as a conventional non-linear operation by only remaining the maximum among a small region of the output activation map from Conv layers [161]. It is also mostly used to simplify the network model and to extract the most useful information.

Following the Conv layers are the FC layers which usually transform the local activation maps of Conv layers to the label embedding. In DopNet, we adopt a three-layer architecture with output activation number of 512, 128 and 2 respectively, directly transforming local features to the representation of semantic categories. FC layer is regarded as a conventional linear projection operation, parameterized by the weight w_{fc} and bias b_{fc}, but without convolutions. Due to the sparsity of the label embedding, the output of FC layers O_{fc} can always be added with the ReLU operation, described by the Eq.(4.3), where I_{fc} is the input from the previous layer. Another operation related with the FC layers is the dropout operation in Section 2.4.2.2, which shuts down the gradient flow and the updates of some neurons randomly for each mini-batch of data so that the FC weight matrix can be partially learned in a stochastic manner.

Lastly, the output of the last FC layer is a vector $Logit \in \mathbb{R}^{N_{class}}$ with the class number N_{class}, each of which indicates the probability that the μ-DS input belongs to that class. As a model under optimization, the loss function we used is the CE function, implemented by calculating the divergence between the final FC output and the ground-truth label, $y \in \mathbb{R}^{N_{class}}$ in the Eq.(4.4), (4.5) and in Section 2.4.1.5. $Logit_{soft}$ is the output after the softmax operation (see Section 2.4.1.4) and L_{CE} is the final losses. Besides the softmax CE loss, we also add a regularization term to balance the M-dist and E-dist for better optimization schemes. In the next section, the balancing term, together with the techniques used to prevent overfitting are introduced.

$$ReLU(x) = \max(x,0) \tag{4.1}$$

$$O^i_{w,h} = \frac{I^i_{w,h}}{(\ 1+\sum_{i-R/2}^{i+R/2} (I^i_{w,h})^2\ )^\beta} \tag{4.2}$$

$$O_{fc} = ReLU(w_{fc} \times I_{fc} + b_fc,\ 0) \tag{4.3}$$

$$Logit_{soft} = \frac{exp(Logit)}{\sum_{i=1}^{N_{class}} exp(Logit[i])} \tag{4.4}$$

$$L_{CE} = -\sum_{i=1}^{N_{class}} y[i] \times log(Logit_{soft}[i]) \tag{4.5}$$

$$\begin{aligned} L_{reg} = \gamma_{fc1} \times \|w_{fc=1}\|_2 + \\ \gamma_{fc2} \times \|w_{fc=2}\|_2 + \\ \gamma_{conv1} \times \|w_{c=1}\|_2 + \gamma_{conv2} \times \|w_{c=2}\|_2 \end{aligned} \tag{4.6}$$

$$x_{aug} = x_{width\_win,height\_win} \tag{4.7}$$

$$\begin{aligned} \|y - w_{fc2} \times I_{fc2}\|_2^2 &= \|w_{fc2} \times I_{fc2}^{GT} - w_{fc2} \times I_{fc2}\|_2^2 \\ &= (I_{fc2}^{GT} - I_{fc2})^T w_{fc2}^T w_{fc2} (I_{fc2}^{GT} - I_{fc2}) \end{aligned} \tag{4.8}$$

$$L_{ME} = \|w_{fc2}^T w_{fc2} - I\|_2^2 \tag{4.9}$$

4.3.2 DopNet Operations for Overfitting Preventions

Overfitting problems are common in DCNN models, due to the fact that network parameters are biased or more discriminative to the noise in the training samples instead of being generalisable to the test samples. Prevention of the overfitting problem in DopNet can be solved from the following three perspectives:

1. Simplifying the network capacity: we adopt the drop-out operation of the FC_2 layer and add the ℓ_2 regularization to the weight parameters of the Conv and the FC layers, with details illustrated in Eq.(4.6), where $\gamma_{fc1}, \gamma_{fc2}, \gamma_{conv1}, \gamma_{conv2}$ are regularization weights of the L_2 norm of kernel filters in $FC1, FC2, Conv1, Conv2$

layers respectively.

2. Increasing the diversity of training data by augmentation, i.e. by cropping the original training data into smaller patches, as shown in Figure 4.3. Due to the nature of the time series of the µ-DS data, we generated more training samples by shifting them in the time domain by different stride sizes. As our training µ-DS data is $x \in \mathbb{R}^{w_{input} \times h_{input}}$, the augmented training samples can be represented as the cropped data along the time axis via different strides, as shown in Eq.(4.7), where $width_{win}$, $height_{win}$ are the window sizes in two dimensions. This operation, if stride sizes small enough are chosen, will increase the number of training samples, give additional data diversity, and improve the robustness of the model as data generated under various conditions will be used for training. In practice, this time shifting simulates misalignment in time and small Doppler offsets for the training data and these two situations can practically happen in realistic uncontrolled scenarios.

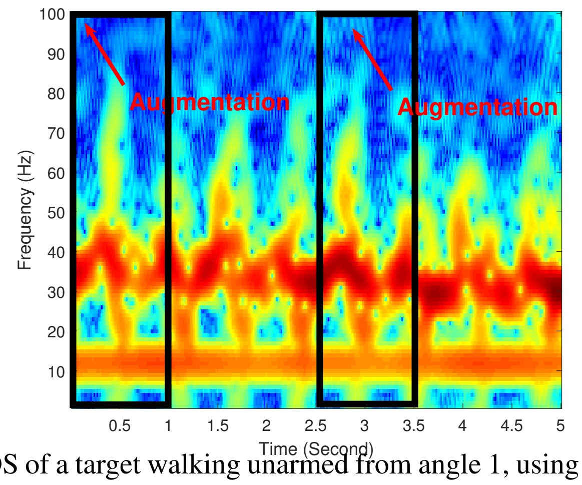


Figure 4.2: Raw µ-DS of a target walking unarmed from angle 1, using receiver node 1 in Figure 4.7; two black boxes indicate the augmented µ-DS, with window size of 1s and window height of 100Hz.

More specifically, as shown in Figure 4.2, an example of a target walking unarmed from angle 1, received by node 1 is illustrated. Here, in the 5-second µ-DS, red and green rectangular boxes indicate two augmented data samples in the training stage. The example shown in Figure 4.2 uses *width_win* as 1.5 second, while the *height_win* chosen as 100 Hz. It seems obvious that, the augmented data samples can be generated by selecting very small stride of the moving window, which will

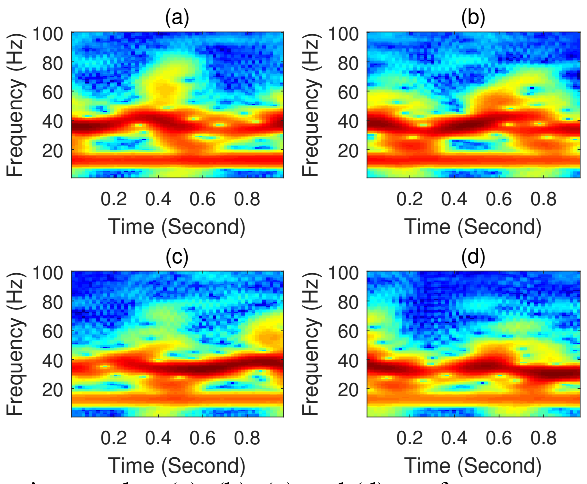


Figure 4.3: Augmentation results: (a), (b), (c) and (d) are four augmented data examples generated from the 5-second µ-DS in Figure 4.2. The augmentation method and parameters are illustrated in Section 4.3.2.

also simulate small misalignment in realistic data.

3. Regularizing the final loss metric: we derive the M-dist and E-dist between the ground-truth and predicted labels in DopNet. Since M-dist is aimed to maintain the discrimination capabilities while the E-dist to provide generalization, a regularization term is designed and incorporated in the DopNet by balancing the generalization and discrimination of the network weights. Note that we balance the E-M distance only in the final FC_2 layer and denote the input, output, weights and bias in FC_2 layer as $I_{fc2}, O_{fc2}, W_{fc2}, b_{fc2}$. If we assumed that the ground truth label y can be transformed from the perfect input I_{fc2}^{GT} using the weight W_{fc2} , then we could write up the simple Euclidean loss between y and $W_{fc2} \times I_{fc2}$ as the Eq.(4.8).

   In this way, we argue that this E-dist term is actually measuring the M-dist between ideal and predicted FC inputs, parameterized by weight matrix w_{fc2}. Since M-dist is designed to ensure the discrimination capability of the matrix, the regularized term denoted as L_{ME}, is added to balance the E-distance and M-dist, which enforces the term $w_{fc2}^T w_{fc2}$ to be close to identity matrix, as shown in Eq.(4.9). By adjusting the balance the E and M distance (see Figure 4.17 for the sensitivity test), we are actually controlling the discrimination and generalization of the DCNN.

4.4 SC-DopNet

In this section, we introduce the SC-DopNet using the components illustrated in Section 4.3.1. We introduce two phases and the relevant loss functions, including the training and testing phase. To sum up, the total loss function in the training stage is shown in Eq.(4.10). Once the network parameters are stored and saved, in the test stage, given a test sample x_{test}, the predicted label can be calculated by the simple max operation of the $Logit_{soft}$ in Eq.(4.11), where $Logit_{soft}$ is the output after the softmax operation Eq.(4.4) in the Section 4.3.1.

$$\min_{w,b,k} L_{Total} = L_{CE} + L_{Reg} + \gamma_{ME} L_{ME} \tag{4.10}$$

$$i_{class} = \underset{i}{\text{argmax}}\ Logit_{soft}[i] \tag{4.11}$$

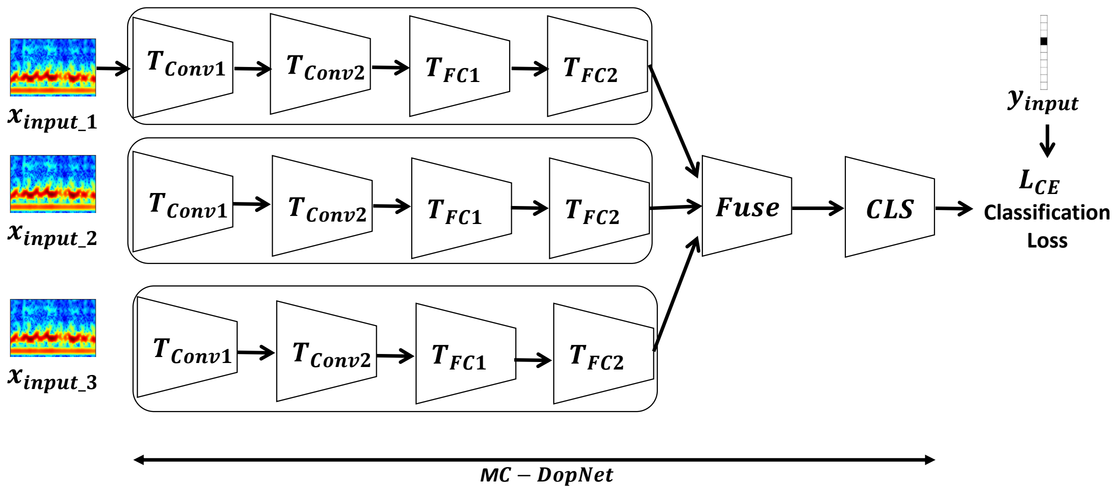


Figure 4.4: Architecture of MC-DopNet

4.5 MC-DopNet

Before describing the MC-DopNet, we introduce the effect of aspect angle in μ-DS and how they affect the classification. Figure 4.5 (a), (b) and (c) represent the μ-DS of a walking personnel from the three angles assuming to use the node 3 only as a mono-static radar (right figure in Figure 4.5). Although the μ-DS are collected from the same motion and the same personnel, the large intra-class variations and small inter-class variations caused by the aspect angles effect may destroy the classifier, especially considering to distinguish the μ-DS in Figure 4.5 (b), from Ang 2, walking without a gun and the one in Figure 4.5 (d) from Ang 1, walking with a gun.

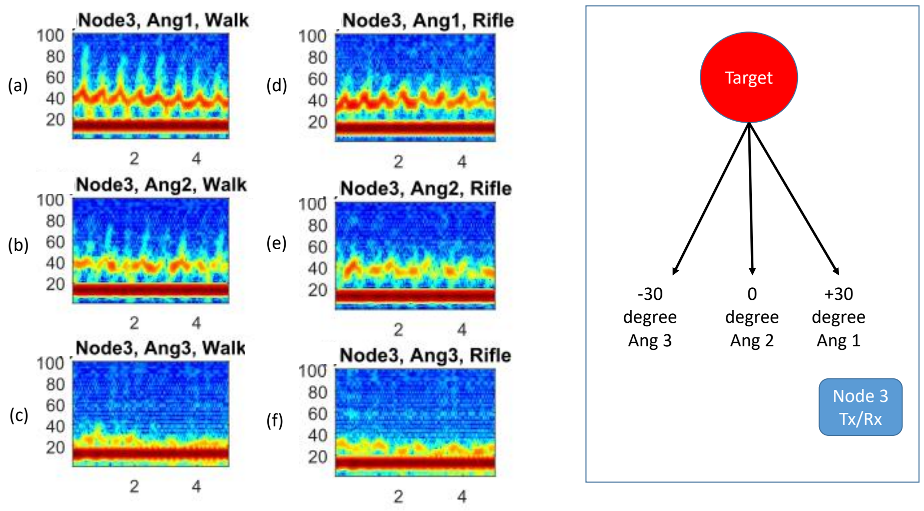


Figure 4.5: μ-DS of two activities from three aspect angles using the mono-static radar. All x-axis represents time with unit of second while all y-axis represents frequency with unit of Hz.

To reduce the intra-class and increase the inter-class variations caused by the diversity of aspect angles, multi-static radar is utilized to increase the μ-DS classification robustness. This implies collecting simultaneous μ-DS of a particular target and activity from different and spatially distributed radar nodes. Since μ-DS from different nodes exhibit diversed local features, it is difficult to train a single SC-DopNet local feature extractor which is sharable for all nodes.

Considering a more realistic scenario where streaming the raw I/Q data across a network from all independent radar nodes may not be feasible but it is highly feasible to share a higher level classification decision into a centralised decision making system. Considering these, we choose to design multiple SC-DopNet for each radar node in MC-DopNet and the way to fuse these data into the multi-channel decision enabled by MC-DopNet is introduced and discussed in the following section. The MC-DopNet architecture is shown in Figure 4.4.

Specifically, two novel schemes are proposed to combine multiple channel μ-DS for recognition. The first has been formed the GIR method and the second is the ℓ_{21}-Norm method, which are both applied to fuse the outputs of individual SC-DopNet in an end-to-end learning framework. Let us assume that data from N_{MC} multiple channels are now available; we propose to build N_{MC} individual SC-DopNets, where input of each SC-DopNet is data from the single channel radar.

When feed-forwarding the multi-channel data samples to their respective SC-DopNet, the j^{th} single channel output after softmax function is obtained using Eq.(4.4), denoted as $Logit^j, j \in [1, N_{MC}]$. In general, we proposed two strategies: (1) *"win by sacrificing worst case"* and (2) *"win by sacrificing best case"*, to guide the design of GIR and ℓ_{21}-Norm method respectively. The *"win by sacrificing worst case"* strategy is to increase the overall recognition rate by sacrificing the performance of channels with average or poor data quality but only enhancing performance of the best single channel. The *"win by sacrificing best case"* strategy is to increase the overall recognition rate by degrading the channel performance with the best data quality a little but improving the channel performance with the worst data quality.

(i) GIR Method: the GIR method fuses multiple outputs into one final result denoted as $Logit^{GIR}$ based on weighted linear combinations of individual SC-DopNet result. Note that the weights are also the parameters under DCNN training, rather than conventional binary voting schemes using the fixed and equal weights. The details are shown in Eq.(4.12). In general, the GIR method is a greedy algorithm, because the higher weight from a given channel will be learned and assigned automatically if, and only if, its corresponding prediction output contributes more to decrease the total loss function than other channels. In addition, due to the sum of weights are forced to one, the lower weights of the other channels will be updated as well. To sum up, GIR method re-weights the weights of multiple channels so that the final loss function is minimized by the greatest amount.

Specifically, our GIR method uses the weighted single channel output as the fused prediction into the CE loss function (see Eq.(4.5)). Therefore, the multi-channel loss function is proposed, which is similar to Eq.(4.5) except that we replace L_{CE} by L_{CE}^{GIR}, with the input $Logit^{GIR}$ in Eq.(4.13), where L_{ME}^j and γ_{ME}^j are the j^{th} channel M-dist regularization and its respective weight. L_{Reg}^j is the j^{th} channel regularization corresponding to network weights.

$$Logit^{GIR} = \sum_{j=1}^{N_{MC}} \beta_j Logit^j, \ with \sum_{j=1}^{N_{MC}} \beta_j = 1, \beta_j \geq 0 \tag{4.12}$$

$$\min_{w,b,k} L_{Total}^{GIR} = L_{CE}^{GIR} + \sum_{j=1}^{N_{MC}} \gamma_{ME}^j L_{ME}^j + L_{Reg}^j \tag{4.13}$$

(ii) ℓ_{21}-Norm Method: Unlike the GIR method, ℓ_{21}-Norm method prefers equal weights on outputs from all channels and tries to enforce similar outputs from different channels. In general, the potential advantage of ℓ_{21}-Norm method is to enhance the output performance from the poor quality channel, constrained by channels with better quality results. More specifically, the ℓ_{21}-Norm method constrain the final data representation of each node share the same structure. In this way, data representation from the node with poor quality is able to be compensated by the one with good quality.

Similar to the GIR method from the perspective of implementation, the ℓ_{21}-Norm method uses a regularization term constrained on the multiple outputs $Logit^j$ from the last FC layer of multiple SC-DopNets. The regularization term can be shown in Eq.(4.14) and (4.15), where $Logit_{soft}^j$ is the output of j^{th} channel after the softmax operation and $Logit_{soft}^j[i]$ infers the probability output that the data from j^{th} channel belongs to the i^{th} class. Finally, the loss function using the ℓ_{21}-Norm method is shown in Eq.(4.16), where $Logit^{L_{21}}$ is the final output by averaging all single channel outputs with equal weights.

$$Logit^{L_{21}} = \frac{1}{N_{MC}} \sum_{j=1}^{N_{MC}} Logit^j \tag{4.14}$$

$$L^{L_{21}} = \sum_{i=1}^{N_{class}} \sum_{j=1}^{N_{MC}} Logit_{soft}^j [i]^2 \tag{4.15}$$

$$\min_{w,b,k} L_{Total}^{L_{21}} = L_{CE}^{L_{21}} + \sum_{j=1}^{N_{MC}} \gamma_{ME}^j L_{ME}^j + L_{Reg}^j + \gamma_{L_{21}} L^{L_{21}} \tag{4.16}$$

4.6 Implementation

In this section, we introduce the NetRad radar systems, the experimental scenarios in the trial and finally the processing methods and details.

4.6.1 Radar System

The radar system used to collect the data presented in this chapter is the three-node multistatic system NetRAD [4, 2], which has been developed over the past years at University College London. Figure 4.6 presents one of the node in NetRad and the basic architecture can be found in [2]. The system is a coherent pulsed radar and operates at 2.4 GHz. The data shown in this work were collected using the following RF parameters: 0.6 μs pulse duration, 45 MHz bandwidth, linear up-chirp modulation, and 5 kHz PRF to include the whole human μ-DS within the unambiguous Doppler region. Five seconds of data were recorded for each measurement in order to collect a multiple periods of the average human walking gait, which is on average approximately 0.6 seconds. The transmitted power of the radar is approximately 200 mW. The antennas have 24 dBi gain and are operated with vertical polarization to effectively interact with human subjects, as the human body shape is such that the vertical dimension is more significant than the horizontal dimension. This is expected to increase the SNR of the targets' echoes in comparison with horizontal polarization

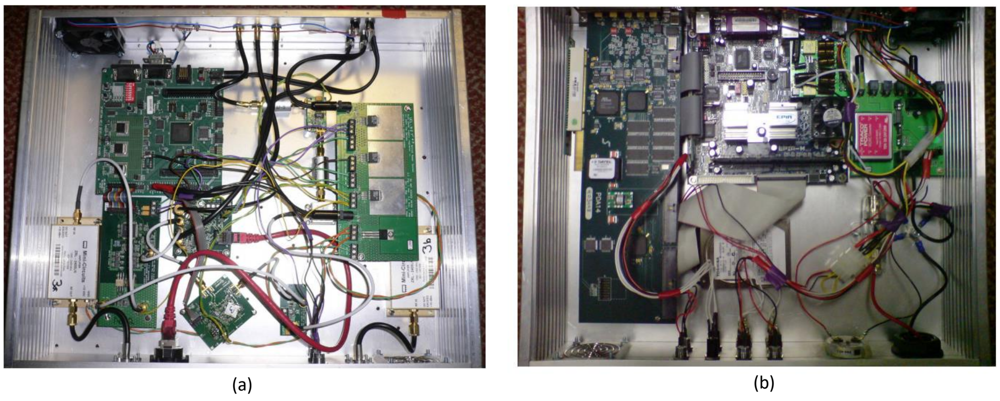

(a) (b)

Figure 4.6: Picture of NetRad System [2]: (a) RF chain and FPGA; (b) Data collection module and mini-PC .

Table 4.1: Processing parameters including the ones to collect and generate the μ-DS, train the DCNN and augmentation.

Recording Time	5s
Overlapping Ratio	0.9
FFT Integration Time	0.3s
Augmentation Time Width	0.15s
SGD Momentum	0.9
Base Learning Rate	0.001 (FC), 0.0005 (Conv)

4.6.2 Experiment

In the following Figure 4.7, the data are collected when targets walk from three different angles roughly around 30 degree, 0 degree and -30 degree, (denoted as Ang 1,2,3). The distance between the target position and the node 1 is 70m and the node 3 is the Tx/Rx pulse radar part, while the other two are receivers. There are two movements in the experiment, which are walking with armed and unarmed weapon (replaced with a metal stick holding or not holding). There are three people involved into the experiments, their height are 1.87m, 1.7m and 1.75m respectively. For each walking, the recorded time is 5 second and in total 270 data samples (5 second recording) are collected.

4.6.3 Processing Method

First the matched filter processing in Section 2.2.2 between the reference and received echo signals and the STFT in Section 2.2.3 is used to obtain the spectrogram. The processing parameters is summarized in Table 4.1, where the overlapping ratio is 0.9 and the integration time of FFT is 0.3 seconds. Each μ-DS sample is recorded for 5 seconds and the stride for cropping the μ-DS samples in the data augmentation operation is chosen as 0.15 seconds. In addition, to increase the challenge of testing, we are also cropping the testing data into different dwell time but the stride is chosen as 0.3 seconds and the cropping starting point is chosen randomly. We argue that this test scheme is a more realistic scenario, where we cannot guarantee where the real-time test data starts, as the radar may have been performing other tasks prior to extracting the μ-DS of a specific target at a specific time.

All DopNet layers and data operations are implemented using the Tensorflow software. The conventional SGD method is used for optimizing the parameters, with the momentum 0.9. The initialized learning rate for FC layers and Conv layers are chosen as 0.001 and 0.0005 respectively. The decay policy for the learning rate is the inverse decay and the decayed learning rate denoted as lr_{decay} is following the Eq.(4.17), where lr_{base} is initialized base learning rate. The batch size is chosen as 50 and training and test samples are shuffled by the Tensorflow FIFO-Queue operation. The regularization weight for the Conv and FC layers are chosen as 0.005.

$$lr_{decay} = lr_{base} \times (1 + 0.001 \times epoch)^{-0.75} \tag{4.17}$$

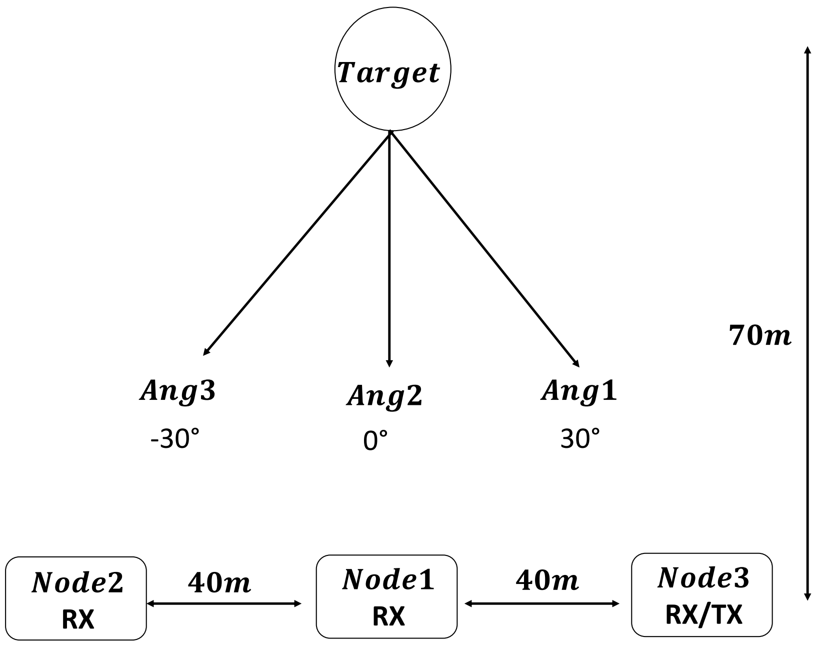


Figure 4.7: Experiment scenario in the field using NetRad.

4.7 Results and Ablation Study of SC-DopNet

This section analyze the various factors of SC-DopNet via ablation study, including the augmentation operations, radar operating parameters and the parameters in the SC-DopNet model. We also give potential reasons and discuss these results. Finally, we compare the SC-DopNet with various features and classifiers.

4.7.1 Raw and Augmented μ-DS

In this section, the raw μ-DS are presented and illustrated and the detailed processing parameters are outlined in the implementation section. In addition, the data augmentation procedure and the diverse μ-DS are presented in Figure 4.3 with the raw input in Figure 4.2.

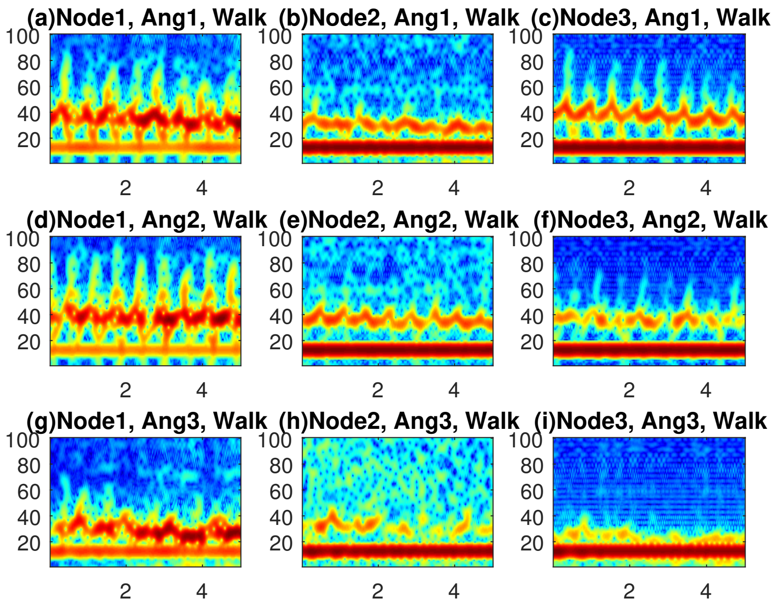


Figure 4.8: Raw μ-DS of walking among three angles and three nodes. All x-axis represents time with unit of second while all y-axis represents frequency with unit of Hz.

In Figure 4.8 and 4.9, the Doppler frequencies related to the bulk movement from angle 1 and 2 are centred at around 42Hz, while the one from angle 3 is around 30Hz. This is due to the relatively larger aspect angle for angle 3 than angle 1 and 2. Either for armed or unarmed gaits, in general, frequency due to arms movement from angle 1 is smaller than angle 2, while the one from angle 3 is much smaller than the angle 1. These can all be explained by the different Doppler aspect angles in the bi-static radar geometry and the velocity components of the bi-static bisector.

It can be clearly seen that the unarmed walking gait from angle 1 and 2 are clearly distinguished from the armed ones by the signatures from 20Hz to 30Hz caused by the swinging arms. From angle 3, as shown in (e) and (f), the difference between unarmed and armed one is not obvious, but some vague differences in the μ-DS from 15Hz to

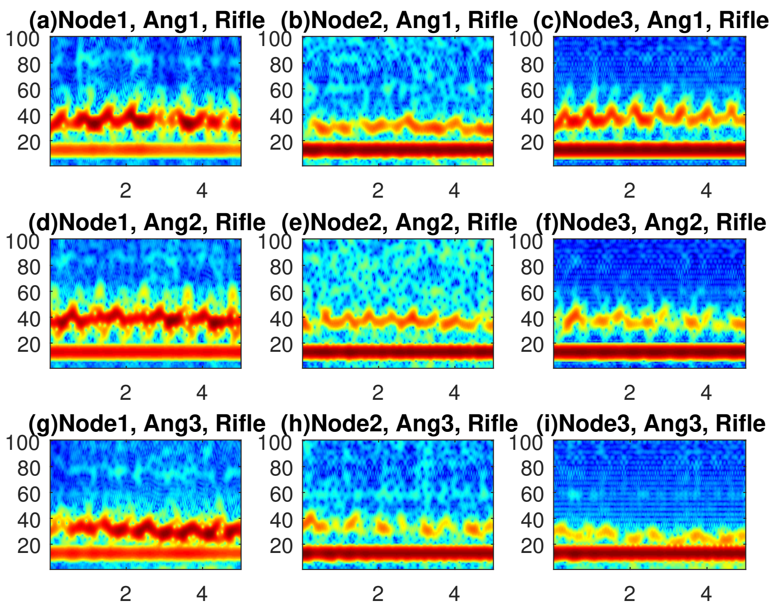


Figure 4.9: Raw Doppler signature of walking with rifle among three angles and three nodes. All x-axis represents time with unit of second while all y-axis represents frequency with unit of Hz.

Table 4.2: Radar operating parameter set used for evaluating in the μ-DS classification experiments.

Training Perc.	{20%,40%,60%}
Dwell Time (Second)	{1,1.5,2,2.5}
Aspect Angles (Degree)	{-30,0,30}
Radar Node	{N1,N2,N3}

20Hz still exist.

4.7.2 Evaluating Radar Operational Parameters

In this section, we analyse the recognition rate using single-channel μ-DS data with different operational parameters such as training percentage, dwell time, node geometry and all the three aspect angles. The parameters evaluated in the experiments are in Table 4.2.

As shown from Figure 4.10 to 4.13, in general, with all other variables controlled, increasing the training percentage increases the recognition rate. The reason is obvious, as it increases the training data diversity and eases the classifier task by decreasing the test data samples. For training ratio at 0.6, no matter at what dwell time, the recognition

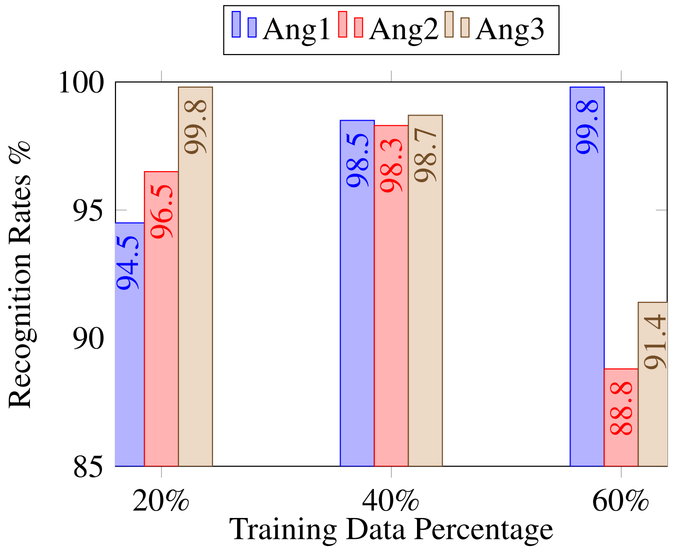


Figure 4.10: Recognition rates of three training data percentages and three aspect angles, using data from node1, dwell time of 1s and SC-DopNet.

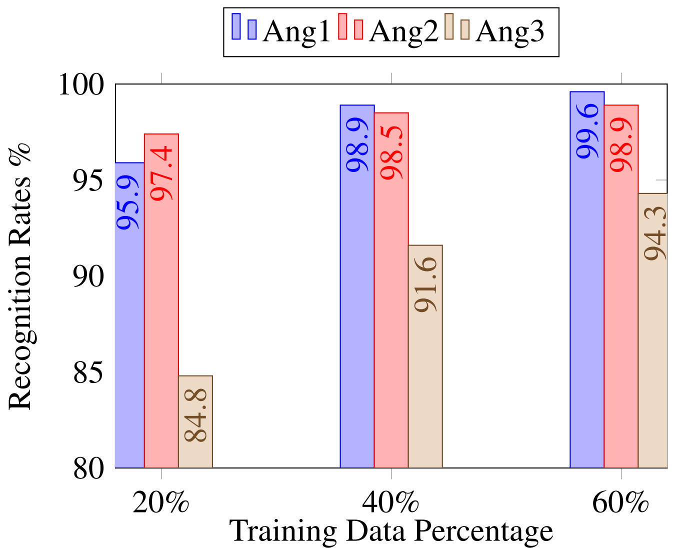


Figure 4.11: Recognition rates of three training data percentages and three aspect angles, using data from node1, dwell time of 1.5s and SC-DopNet.

rates all achieve close to 100%. In addition, with the training ratio at 0.4, recognition rate with dwell time of 2s and 2.5s already achieves very close to 100%. However, the one with dwell time of 1s and 1.5s can only achieve 98% approximately.

The Figure 4.14 shows the averaged recognition rate among all dwell time with respect to different angles and training ratios. Among the three aspect angles, it can be found that the average recognition rate from aspect angle three is the lowest. The

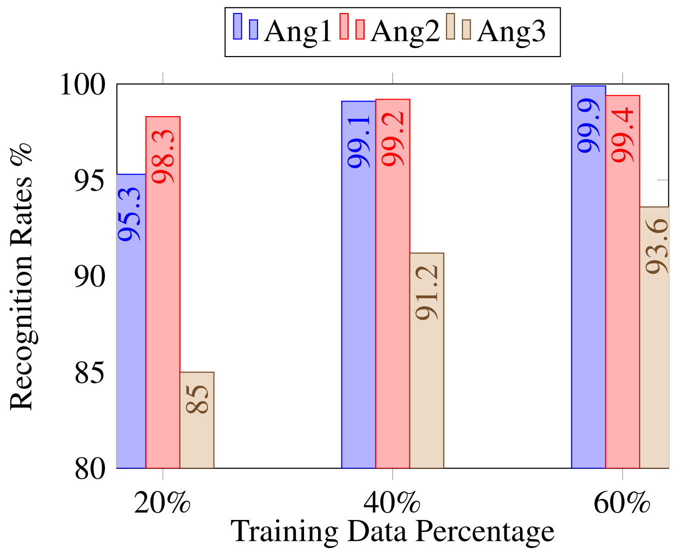


Figure 4.12: Recognition rates of three training data percentages and three aspect angles, using data from node1, dwell time of 2s and SC-DopNet.

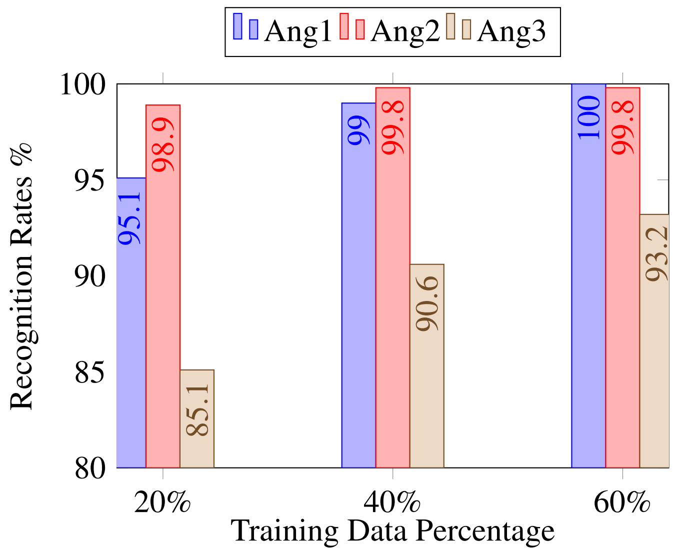


Figure 4.13: Recognition rates of three training data percentages and three aspect angles, using data from node1, dwell time of 2.5s and SC-DopNet.

reason is that under movements from aspect angle 3, the bi-static angle formed by target, transmitter node 1 and the receiver node 2 is the largest that induces the lowest Doppler frequency shifts and SNR of µ-DS. These further lead to the poor discriminative quality of µ-DS from armed and unarmed motions. It can also be inferred that when trained on 20%, under all dwell time, recognition rate of angle 2 outperforms angle 1 around 2%, but with the increasing of training percentage, recognition rate of angle 1 is chasing

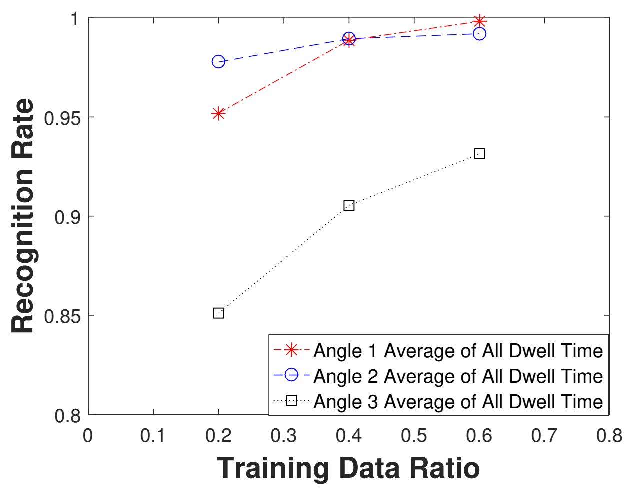


Figure 4.14: Averaged recognition rate among all dwell time with respect to different angles and training ratios.

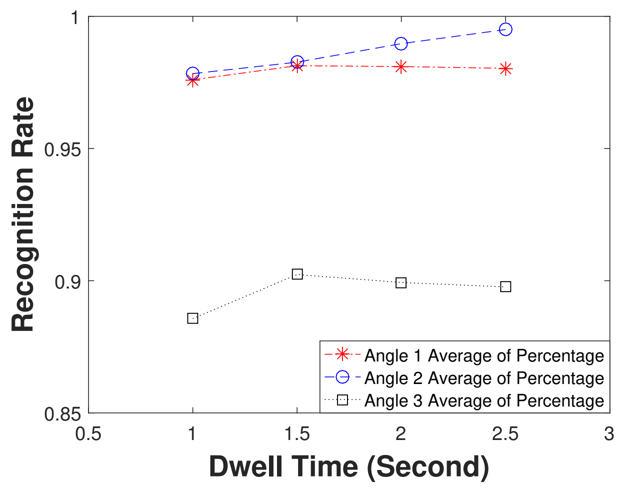


Figure 4.15: the averaged recognition rate among all training data percentages with respect to different aspect angles and dwell times.

up to equally the same as angle 2 and even outperforms angle 2 at training percentage of 0.4 and 0.6 respectively. The main reason can be observed by comparing Figure 4.8 and 4.9, (b) and (d) that the Doppler frequency induced by bulk movements from angle 1 varies larger than the one from angle 2. Note that in our data augmentation steps, we are truncating the whole 5 second spectrogram into smaller parts (with shorter time duration), therefore among different augmented truncations of the dataset, augmentation from angle 2 will induce more variations of bulk Doppler frequency and the more of μ-DS frequency as well. This actually increases the intra-class variations among the train and test datasets. With the increasing training ratio, these intra-class variations can be eliminated as more samples related to different bulk movement Doppler shifts can be used to train the network. Finally, with the training ratio of 0.6, recognition rate of angle 1 outperforms angle 2, with the potential reason that Doppler signatures induced by arms movement at unarmed scenario from angle 2 has higher SNR and higher Doppler frequency shifts than angle 1, as shown in Figure 4.8 and 4.9 (a) and (c). These all relate to the node 1 as the transmitter and the geometry of radar in the experiments.

Figure 4.15 shows the recognition rate of different angles with respect to different dwell time. Due to the average of the training ratios, we can deduce that for all angles, recognition rates increase with the increasing dwell time from 1s to 1.5s. However, with further increasing dwell time from 1.5s to 2.5s, only the recognition rate from angle 2 increases further and the one from angle 1 attain the similar result at dwell time of 1.5s. Note that recognition result from angle 3 drops with the further increase in dwell time. The potential reason for this may be that there are few useful features for signatures of angle 3 due to the large aspect angle.

4.7.3 Evaluating the Drop-out rate

In this section, we analyse the recognition rate with different drop-out rates via experiments. Here, we only choose the data from the following scenario: node 1, angle 1, dwell time of 1s. The drop rate ranges from 0.2 to 0.8 in the step of 0.2 and we evaluate the respective recognition rates. According to Figure 4.16,when trained with 20% and 40% data, increasing the dropping rate from 0.2 to 0.8 on FC layers increases the recognition rates first (when increasing from 0.2 to 0.4) and then decreases them (when increasing the drop-out rate from 0.4 to 0.8). This is due to the fact that the model is first overfitting the data but the generalization capability of the model to the test data cannot be maintained. Increasing drop-out rate from 0.2 to 0.4 then ensures the moderate network capability for better generalization. When the drop-out rate increases further, the network parameters are relatively small to model the training data, which decreases the capability of the model and the recognition rate.

In addition, Figure 4.16 shows that when trained with 60% of the data, increasing the drop-rate from 0.2 to 0.6 continuously increase the recognition rates and the further increasing will decrease the recognition rate. The reason can be attributed to previously

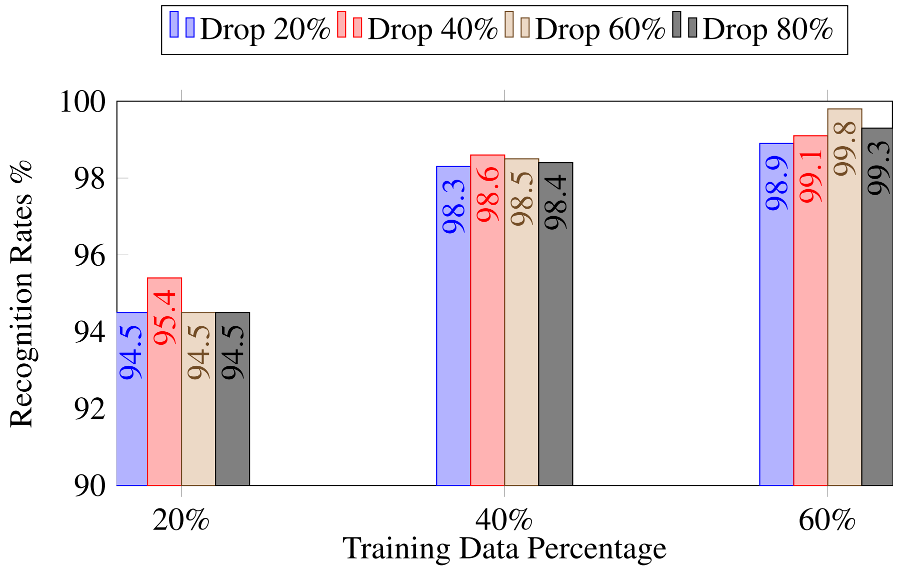


Figure 4.16: Recognition rates using different drop-out rates and training data percentages. All data come from angle 1, node 1 using dwell time of 1s.

described scenario and explanations.

Table 4.3: Recognition rates with different input SNRs. All data come from node 1, dwell time of 1s using SC-DopNet.

	SNR:-10	SNR:-5	SNR:5	SNR:10
Ang1	94.2%	95.5%	95.8%	96.0%
Ang2	96.1%	96.5%	96.9%	96.8%
Ang3	83.6%	84.3%	84.9%	85.3%

4.7.4 Evaluating E-dist and M-dist

In this section, we analyze and evaluate the regularization weight of the balancing E-M distances, by comparing different recognition rates based on whether M or E distance metric dominates. The experiments are conducted using node 1, angle 1, dwell time 1s with 20% training data. As shown in Figure 4.17 increasing the lambda from 0 to 1.5 based on the following set $\{0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 1.5\}$ induces the results increasing from 89.3% to top 96.9% and then dropping to 93.3%. At first, the M-distance dominates, but this is prone to overfitting the data. Then due to the increasing of the balancing weights, the good balance between M and E distance is able to maintain both generalization and discrimination capability of the network. Increasing the balancing weights further will then drop the recognition rate again, as the model generalizes too

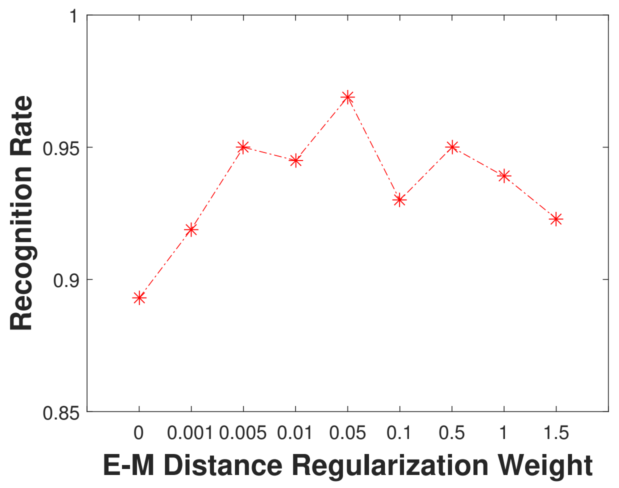


Figure 4.17: Recognition rates among different regularization weights of the E-M distance. All results use data from node 1, angle 1, dwell time of 1s with 20% the training percentage.

much but lose the discrimination capability. In this scenario, we choose 0.05 as our optimal balanced hyper-parameter.

4.7.5 Evaluating the SNRs and Memory Usage

In Table 4.3, recognition rates based on different SNR levels are shown. It can be observed that increasing the SNR from -10 to 5 dB increases the recognition rate by 2% while recognition rate stays stably the same if we further increase the SNR from 5dB to 10dB.

In addition, we discuss memory usage of the network. In the practical scenario, the dataset can be trained on the server or computer cluster therefore only the weights of the network need to be carried onto the site. We also report the memory usage of the network weights in our single channel DopNet as 44MB. The processing time in the test stage is roughly linearly proportional to the number of test data and the ratio is 0.085 second per 100 samples. To sum up, the SC-DopNet is big as 44MB and can predict one test sample in 850 μs.

4.7.6 Comparison with Empirical Features and Classifiers

In Table 4.4, we compare SC-DopNet results with other features extraction methods (including the state-of-the-art SVD and centroid features) and classification methods (including the Naive Bayes classifier, the Discriminative Analysis method and Classifi-

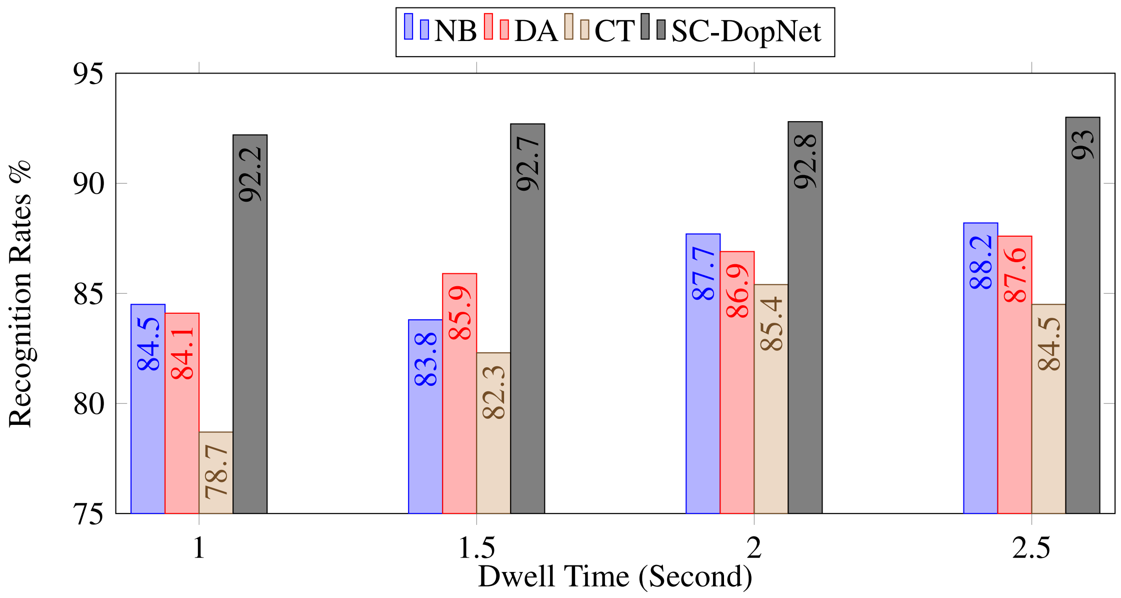


Figure 4.18: Average recognition rates among three aspect angles with respect to dwell time and classifiers. We report the best feature combination results for the non-deep classifiers.

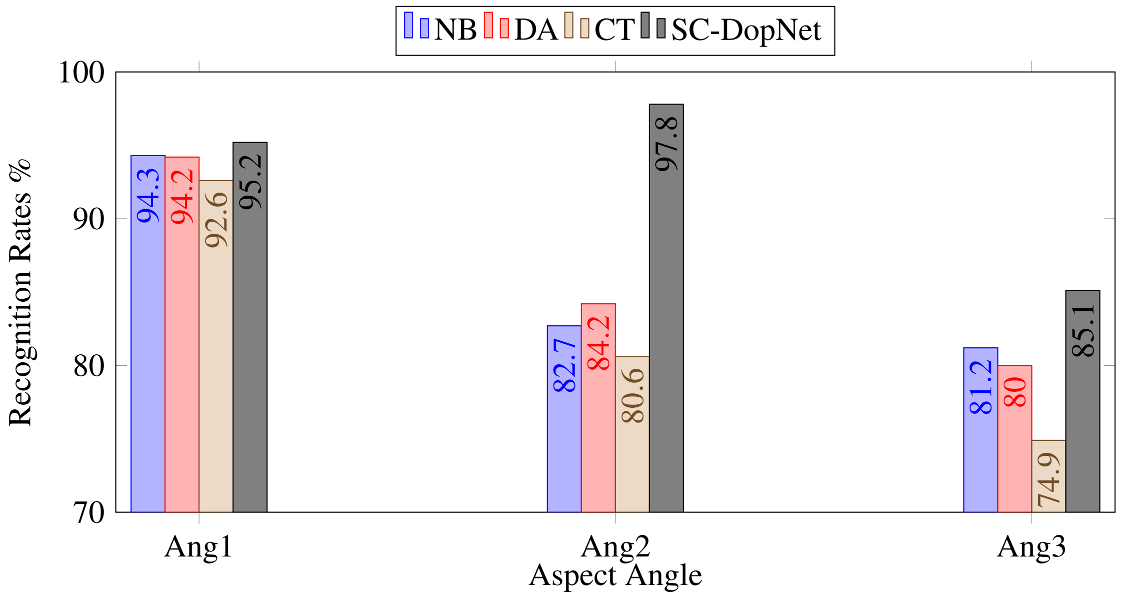


Figure 4.19: Average recognition rates among different dwell time but ranging from different aspect angles. We report the best feature combination result for the non-deep classifiers.

Table 4.4: Recognition results of three non-deep classifiers with the best empirical feature combinations and SC-DopNet results. We use data from node 1 with training percentage of 20%, different dwell time and aspect angles, percentage in (%).

Features	Best Combined SVD and Centroid Features												DopNet			
Dwell Time	1s			1.5s			2s			2.5s			1s	1.5s	2s	2.5s
Classifier	NB	DA	CT	NB	DA	CT	NB	DA	CT	NB	DA	CT	DopNet			
A1,N1	93.1	93.0	88.2	93.5	93.9	91.9	94.5	94.7	94.2	96.1	95.0	96.2	94.5	95.9	95.3	95.1
A2,N1	83.7	83.8	78.6	78.3	84.2	80.2	84.2	83.9	83.1	84.5	84.7	80.5	96.5	97.4	98.3	98.9
A3,N1	76.7	75.4	69.2	79.6	79.5	74.7	84.3	82.0	78.8	84.1	83.0	76.7	85.5	84.8	85.0	85.1
Average	84.5	84.1	78.7	83.8	85.9	82.3	87.7	86.9	85.4	88.2	87.6	84.5	92.2	92.7	92.8	93.0

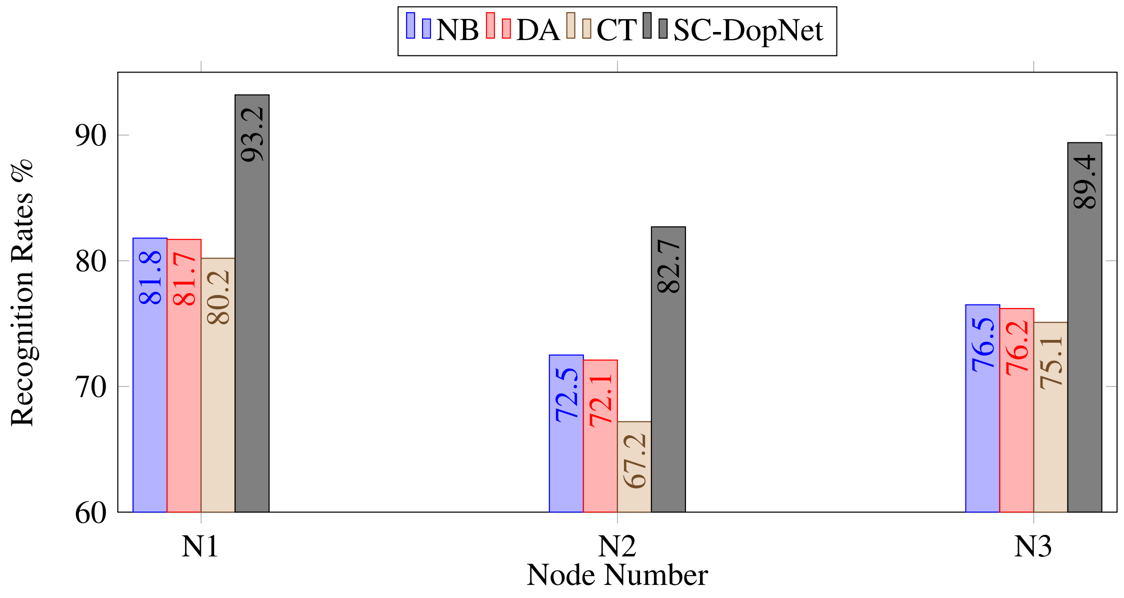


Figure 4.20: Averaged recognition results of four classifiers on aspect angles, ranging different nodes. We report the best feature combination results.

cation Tree method). The experiment setting to generate Table 4.4 is aimed to analyze recognition performance under node 1 and each of the three angles.

In each experiment, we report the best recognition rates selecting different combination of features and we highlight the best recognition rates by red. In addition, we report the averaged recognition rates on three angles but ranging from the all dwell times in Figure 4.18 and also the averaged recognition rates on dwell times but ranging from the aspect angles in Figure 4.19. With increasing of dwell time, under various classification methods, the recognition rates all increase. Obviously, compared with angle 2 and 3, using non-deep features and classifiers achieve the best results 93% in angle 1 but SC-DopNet outperforms by 1-2%. For angle 2 and 3, SC-DopNet outperforms other

Table 4.5: Recognition results of three non-deep classifiers with the best empirical feature combinations and SC-DopNet results. We use data from all aspect angles, each radar node, different dwell times, with training percentage of 20%, percentage in (%).

Features	Best Combined SVD and Centroid Features												DopNet			
Dwell Time	1s			1.5s			2s			2.5s			1s	1.5s	2s	2.5s
Classifier	NB	DA	CT	NB	DA	CT	NB	DA	CT	NB	DA	CT	DopNet			
N1	80.4	80.4	77.4	81.4	81.7	79.4	82.3	81.9	81.2	83.2	82.6	82.6	92.6	93.0	93.7	93.5
N2	70.9	70.0	64.7	73.1	72.8	66.9	73.1	72.7	68.9	73.0	72.8	68.3	80.2	83.6	83.2	83.9
N3	74.2	74.2	72.3	76.2	75.6	75.0	77.5	77.3	77.3	77.9	77.8	77.7	90.1	92.7	93.5	92.9
Average	75.2	74.9	71.5	76.9	77.5	76.7	77.6	77.3	75.8	78.0	77.7	76.2	87.6	89.8	90.1	90.1

methods by 10% and 6.4%. In addition, from the results of angle 2 in Figure 4.19, SC-DopNet seems more robust to the aspect angle variations compared with other empirical features, where other methods achieves in average 82% for angle 2, but SC-DopNet still remains the recognition rate of over 90%. In Figure 4.19, it seems that SC-DopNet outperforms all other methods and the recognition result difference is descreasing from 9.8% to 6.2% with the increasing dwell time from 1s to 2.5s.

In Table 4.5, we design a new experiment where all three angles from a single node are included in the dataset to recognize armed and unarmed movements. In addition, we summarize the results in Figure 4.20. We argue that this is a more realistic scenario in mono-static μ-DS recognition, as μ-DS from all aspect angles are included in the dataset. It can be observed that under this more complicated task, recognition rates of node 1 using non-deep features (the first row of Table 4.5) drop around 5% compared with the average in Table 4.4, however, SC-DopNet only drops 2% achieving averaged 93.5% on all dwell-time settings. For node 2, SC-DopNet still outperforms other methods by approximately 10%. For the most difficult result for node 3, it seems that training using all-angle signatures, SC-DopNet achieves 92.3%, outperforming 15% than the other features.

To sum up, compared with non-deep feature and classifier, SC-DopNet achieved in average 92.7%, outperforming the second best feature and classifier (discriminative analysis method) 4% when evaluating for single angle in node 1. When mixing up μ-DS of all angles, SC-DopNet achieves the most robust results, in average 91.0%, outperforming the second best around 12.6%. This is mainly due to SC-DopNet's large model capacity when handling complex classification tasks, for example the classification of all-angle scenario. From another perspective, since we do not change the hyper-parameters, it is beneficial to train SC-DopNet using larger number of training data.

4.8 Results and Ablation Study of MC-DopNet

In this section, we discuss the feasibility of extending the SC-DopNet to MC-DopNet to process multi-static µ-DS. In addition, we evaluate our proposed two methods, namely GIR and ℓ_{21}-Norm method, by comparing with the conventional binary voting (BV) method. Specifically, the recognition results are shown in Table 4.6, 4.7, 4.8, generated respectively by BV, GIR and the ℓ_{21}-Norm method. These three tables illustrate the the fusion recognition rates, the recognition rates using the single-channel prediction (but trained by fusing multiple-channel data) and the learned weights (for GIR method only) are shown and compared. Finally in Figure 4.8 and 4.9, we have already showed the raw µ-DS but here we link the recognition results to the µ-DS.

4.8.1 Analysis of Node-Angle Combinations in MC-DopNet

This section aims to compare the proposed GIR and ℓ_{21}-Norm with the conventional BV using MC-DopNet. All experiments are conducted using 30 trials with randomly selected training and test samples. To ensure fair testing, experimental settings are the same for the three fusion methods: dwell time of 1s, training with 20%, with the same augmentation scheme and SC-DopNet architecture, in Section 4.7.

Although results from Table 4.6 to 4.8 are trained based on fusing multiple channel data, we argue that recognition results based on different node-angle combinations are essential to understand mechanisms of the fusion methods.

Therefore, we propose to use recognition rates of BV methods as the baseline to measure the performance of certain node-angle combination. The reason for adopting BV as baseline is that the total loss function using BV is simply the sum of the loss functions from three networks with equal weights. There are three main findings by observing the Table 4.6 and corresponding signatures in Figure 4.8 and 4.9.

1. **Finding 1:** From the first three columns in Table 4.6 and 4.8, for all angles independent of the fusing method, recognition rates from node 1 outperform node 3 and those from node 3 outperform node 2. This matches the higher SNR and

Table 4.6: MC-DopNet recognition results of three aspect angles, using dwell Time of 1s, 20% training percentage but using the BV method for fusion.

	Node 1 Accuracy (weight)	Node 2 Accuracy (weight)	Node 3 Accuracy (weight)	BV Accuracy
A1	95.7 (0.333)	65.0 (0.333)	87.1 (0.333)	96.1
A2	96.5 (0.333)	77.3 (0.333)	93.5 (0.333)	**99.0**
A3	85.3 (0.333)	70.8 (0.333)	81.8 (0.333)	91.7

Table 4.7: MC-DopNet recognition results of three aspect angles, using dwell Time of 1s, 20% training percentage but using the GIR method for fusion.

	Node 1 Accuracy (weight)	Node 2 Accuracy (weight)	Node 3 Accuracy (weight)	BV Accuracy
A1	97.3 (0.421)	50.7 (0.289)	49.3 (0.289)	98.2
A2	98.2 (0.434)	50.1 (0.283)	51.2 (0.283)	**98.9**
A3	88.8 (0.462)	51.7 (0.269)	48.3 (0.269)	89.5

more discriminative features of node 1 than node 3 and 2. The main reason of the lowest results for node 2 is the larger bi-static angle formed by receiver node 2 and transmitter node 3, which decreases both SNR and the frequency shifts related to the motions. In addition, the potential reason for node 1 to outperform node 3 may be attributed to DSI in node 3, which degrades the SNR of the μ-DS.

2. **Finding 2:** In Table 4.6, all recognition results from A2 and A1 outperforms A3, no matter which node is selected. This matches the observation by comparing Figure 4.8 and 4.9, where better data quality (SNR) from angle 1 and angle 2 induce better discriminative quality than the angle 3, no matter from which node.

3. **Finding 3:** The recognition results from A2 always outperform the A1. The main reason is that the Doppler frequency induced by bulk movements from angle 1 varies larger than the one from angle 2. This can be observed by comparing (a) and (d), (b) and (e) of Figure 4.8 and Figure 4.9 respectively. More detailed discussions can be found in previous Section 4.7.2.

In Table 4.7, when the GIR method is utilized, it can only be found that recognition rate from Node 1 outperforms the others, but the one of Node 2 and Node 3 are basically the same. Specifically, the first column in Table 4.6, 4.7 and 4.8 all show that node 1

Table 4.8: MC-DopNet recognition results of three aspect angles, using dwell Time of 1s, 20% training percentage but using the ℓ_{21}-Norm method for fusion.

	Node 1 Accuracy (weight)	Node 2 Accuracy (weight)	Node 3 Accuracy (weight)	BV Accuracy
A1	94.9 (0.333)	72.8 (0.333)	85.7 (0.333)	95.9
A2	96.1 (0.333)	81.6 (0.333)	92.7 (0.333)	**99.0**
A3	83.6 (0.333)	72.9 (0.333)	79.9 (0.333)	93.4

recognition results from whatever angles using GIR method outperforms ℓ_{21}-Norm and BV method for around 2%. In addition, from the second and third columns, recognition rates of GIR from node 2 and 3 drop significantly compared to others. The reason is the greedy nature of GIR which aims to increase the weights from the best quality channel node 1 and decrease the ones from other channels in the fusing mechanism. This explanation can also be verified by the extremely unbalanced weights, found in Table 4.7, where weight from node 1 with all angles are larger than other nodes; meanwhile, weights of node 2 and 3 are approximately the same.

It can be found that for all angles, ℓ_{21}-Norm method only outperforms the BV method for node 2 data and performs worse than BV for node 1 and 3. Specifically, comparing the first row in Table 4.6 and 4.8, using the ℓ_{21}-Norm method increases the recognition rate for node 2 (with the worst data quality) from 65% (using BV method) to 72.8% however at the price of decreasing the recognition rate for node 1 and 3 from 95.7% (using BV method) to 94.9% and from 87.1% (using BV method) to 85.7%.

Similar patterns can also be found by observing other rows in Table 4.6 and 4.8. This phenomenon can be interpreted by the nature of ℓ_{21}-Norm, where similar prediction outputs are enforced among multiple channels. From other words, to improve the performance of node 2 with the worst data quality, the ℓ_{21}-Norm sacrifices the single-channel recognition rate for node 1 and 3 but makes improvements in the fused multi-channel recognition rate.

4.8.2 Comparison of Fusion Methods in MC-DopNet

In this section, we focus on comparing the recognition results based on multi-channel data using different fusion methods.

For fusion results based on angle 1, GIR method achieves the best, approaching to 98.2% in average, outperforming the BV method (96.1%) and the ℓ_{21}-Norm method (95.9%). Looking at the weights in Table 4.7, it seems that the greedy-like algorithm assigns the biggest weight to the node 1 (around 0.421) while assigns equally (around 0.289) to the other two. This matches our assumption and understanding of the greedy algorithm in the three findings in Section 4.8.1.

For fusion results based on angle 2, BV, GIR and ℓ_{21}-Norm exhibit similar and the best recognition result (around 99%), due to the data having the highest SNR and the most discriminative features. For results based on angle 3 with the worst data quality, the ℓ_{21}-Norm method achieves the best 93.4% compared with BV method at 91.7% and GIR method at 89.5%. The main reason has been discussed in Section 4.8.1 about the descriptions of ℓ_{21}-Norm method.

In addition, we investigate and discuss when, especially for which angle, GIR and ℓ_{21}-Norm should be utilized to improve fusion results. We generate Table 4.9 by calculating the mean and STD of results using BV method (based on Table 4.6). We argue that the mean and STD of recognition results are useful to determine the preferable method for certain angle. It is easy to conclude from Table 4.7 and 4.8 that when fusing multi-channel results for angle 2, the three fusion methods achieve similar result due to its originally good discriminative quality. However in Table 4.9, when the mean recognition accuracy is relatively high but the standard variation is very large, like angle 1, the GIR method is more suitable for the fusion task. For the low mean accuracy and low STD scenario, ℓ_{21}-Norm might achieve the best to fuse these multi-channel data.

Table 4.9: Mean and standard deviation of the results in Table 4.6 fused by BV method.

	Mean Recognition Rate of All Nodes	STD Recognition Rate of All Nodes
A1	82.6%	12.9%
A2	89.1%	8.4%
A3	79.3%	6.2%

4.8.3 Comparison with Empirical Features and Classifiers

In this section, we compare our results with the state-of-the-art SVD and centroid features, using the threshold voting as the fusion method in Table 4.10. The experiment

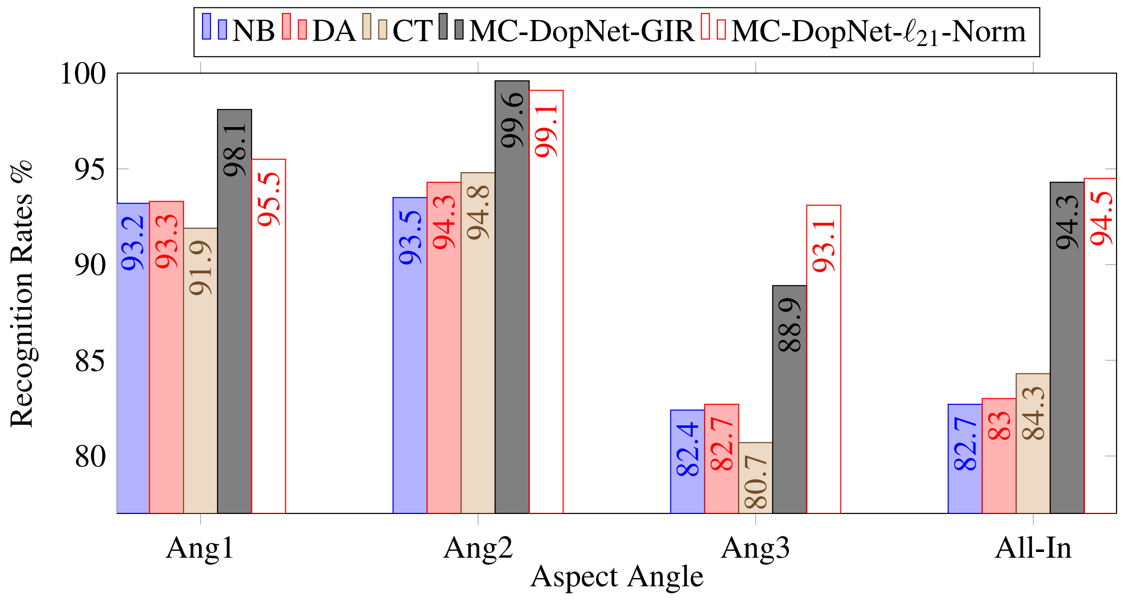


Figure 4.21: Recognition results of five classifiers and different aspect angles. We use all multi-static data with dwell time of 1s. We report the best feature combination results from [3].

setting is to classify the activity from a specific angle using the multi-static μ-DS.

Table 4.10: Comparison of MC-DopNet results with other feature extraction methods and classifiers. For fair comparison, all voting schemes are the best thresholding voting scheme and features are selected the best combinations reported in the paper [4]; percentage in (%).

Feat	Best Combined SVD and Centroid Features												MC-DopNet				MC-DopNet			
DT	1s			1.5s			2s			2.5s			1s	1.5s	2s	2.5s	1s	1.5s	2s	2.5s
CLS	NB	DA	CT	NB	DA	CT	NB	DA	CT	NB	DA	CT	Soft-max(GIR)				Soft-max(ℓ_{21}-Norm)			
A1	91.1	91.3	89.2	92.4	92.3	91.8	93.5	93.7	93.5	95.6	95.7	93.2	98.2	97.2	98.2	98.5	95.9	95.9	95.4	94.9
A2	90.8	91.9	92.0	93.4	93.5	94.8	94.2	94.7	95.9	95.7	96.9	96.6	98.9	99.6	100.0	100.0	99.0	99.0	99.0	99.4
A3	79.4	79.4	77.2	81.7	82.6	80.0	84.3	84.7	82.1	83.4	84.2	83.4	89.5	89.1	88.2	88.7	93.4	92.3	93.5	93.0
All	80.9	80.7	80.7	82.1	82.9	83.9	83.3	84.0	85.6	84.5	84.4	86.9	93.7	94.3	94.9	94.3	93.2	94.7	94.6	95.6

The results are summarized in Figure 4.21. It can be observed from angle 1 and 2 that MC-DopNet with GIR method achieves the best, outperforming other methods by 4% to 5%. For angle 3, MC-DopNet with ℓ_{21}-Norm method outperforms all other methods by 9.4% in average. For the most difficult test scheme with all angle data in, MC-DopNet based methods either using GIR or ℓ_{21}-Norm, outperform 11.1% than others in average. The reason has been explained in the three findings in Section 4.8.2. Note that for all features and methods, recognition rates for all-angle-in setting are lower than the best single-angle result but still better than the worst single-angle result.

4.9 Summary

In this chapter, with the aim of recognizing very close human activities, such as armed or unarmed personnel in security applications, we advance Doppler processing in multi-static radar by proposing two modified DCNNs, namely SC-DopNet and MC-DopNet for mono-static and multi-static μ-DS classifications.

Differentiating armed and unarmed walking personnel is challenging, mainly due to the effect of aspect angle diversities in real-world scenarios, as illustrated in details in Section 4.5, and the overfitting problem caused by small-scale μ-DS dataset. To address these two problems, the work carried out in this chapter makes specific three key contributions to the knowledge base:

1. To prevent the overfitting problem facing a small-scale μ-DS dataset, two effective schemes including data augmentation operation and the regularization term to balance the E-dist and M-dist are proposed so that we can train SC-DopNet from scratch successfully. By balancing the E and M-dist, recognition results of SC-DopNet on node 1 has been verified to outperform the one without the regularization by 7.6% in average.

2. In addition, performances of SC-DopNet and the factor analysis have been conducted based on various operating parameters in both the processing and radar operations. Note that from the authors' best knowledge, no detailed ablation study of DCNN for μ-DS classification has been done. Compared with the best empirical feature selection and classifier design, SC-DopNet outperforms the best-performing method by 12.5% in average.

3. To solve the problem of aspect angle diversities for μ-DS classification, we adopt the multi-static μ-DS, design the MC-DopNet and propose two fusion schemes termed as GIR and ℓ_{12}-Norm methods, embedded in the MC-DopNet. These two schemes are based on two different strategies respectively, namely (1) *"win by sacrificing worst case"* and (2) *"win by sacrificing best case"* and we evaluate

them via extensive experiments. In general, comparisons of GIR and ℓ_{12}-Norm methods largely depend on different geometries of the multi-static radar. However, whatever strategy to use, MC-DopNet results outperform the best-performing non-deep feature selection with fusion methods by at least 9.8% using data from all the aspect angles. The results of different single aspect angle have also been evaluated but they largely depend on different geometries.

Note that we also argue and discuss how to utilize the statistics of SC-DopNet results to infer the selection of fusion strategies for MC-DopNet based on different experimental scenarios.

As reviewed in Section 4.2, the research [113] adopted the multiple DTNs for multi-static μ-DS classification. This is similar to our MC-DopNet, however they simply concatenate multi-channel features to a longer one for classification. We argue that this concatenation makes the classifier more difficult to train considering the small-scale μ-DS. In addition, this concatenation does not leverage the intrinsic prior knowledge in the multi-channel classification task: the extracted features should represent the common factors among multi-channel data. To leverage such prior, the network needs to learn the unified features that are discriminative and meanwhile can tolerate the poor quality single-channel data. Different from their work, we propose two fusion strategies for MC-DopNet in Section 4.5. Patel et al. [160] adopted the fusion strategy to handle multi-static data but their network only used single DCNN to extract multi-static features. We argue that their design ignores the variations of the multi-channel local features, which makes it more difficult to train the feature extractor. This is mainly due to the large intra-class and small inter-class variations, as illustrated in Section 4.5 and Figure 4.5.

Chapter 5

Deep Adaptation Network using Optimal Transport

5.1 Introduction

Recent developments in DNN have yielded state-of-the-art results from supervised learning applications [162, 154, 163]. However, the success of DNN requires a large amount of well annotated training data which is not always feasible to perform manually. Therefore, it has acted as a driver to transfer knowledge from datasets for which labels are well-defined. Another challenge of DNN is its vuneralbility to small perturbations of the input data, which is very common for both the computer vision datasets and human μ-DS in radar. The potential perturbations include but are not limited to: light changes, camera shaking effect and the changes of the geometry view in vision applications; the aspect angle diversity of the μ-DS, SNR changes of target echoes from different experimental scenarios and even the same μ-DS from different target personnels in the radar community. As introduced in Section 2.5.2.3, these perturbations always lead to domain discrepancies between training and test datasets.

The DA problem [164] was proposed in this context where the data distribution between the target domain (a few of labels are available, usually in test dataset) and the source domain (well-annotated labels, usually in training dataset) varies so that the discriminative features and the classifiers in the source domain cannot be transferred to the target domain[164, 165]. Under this regime, UDA is the most challenging problem where no label information in the target domain (test dataset) is available. More detailed discussions about the DA and other related concepts are described in Section 2.5, 2.5.1 and details about UDA have already been introduced in Section 2.5.2 and 2.5.2.3.

In this chapter, we propose two adaptation networks for UDA to reduce disparate domain discrepancies and to eliminate the variation of factors between training and test datasets, namely the Re-weighted Adversarial Adaptation Network (RAAN) and Joint Adversarial Adaptation Network (JAAN). The two networks both integrate the OT theory in the adversarial training (see the detailed derivations in Section 2.5.2.1 and Appendix B.3 to understand the relationship between adversarial training and divergence measurements), but their designs are different, where RAAN utilizes the dual formulation of OT but JAAN utilizes its entropy regularized primal formulation. In addition, RAAN implicitly adapts the classifier by matching the label distribution, however, JAAN explicitly matches the joint label and feature distribution via novel network design. Limited by the unavailable public datasets in radar μ-DS classification for evaluations, we first evaluate the two proposed adaptation networks using the benchmark datasets in computer vision, including three well-known hand-written digit datasets and the cross-modality object classification dataset between RGB and RGB-D. Finally, we apply these two adaptation networks in the radar μ-DS to classify armed and unarmed target personnels. These two networks are proved to achieve the state-of-the-art results in UDA tasks and using these two networks increases the μ-DS classification performance by a large margin when handling μ-DS sampled from the unseen scenarios.

RAAN has the following two important features which make novel contributions to the field:

1. To match feature distributions when domains discrepancies are disparate, we train a domain discriminator network together with the conventional DCNN in an adversarial manner to minimize the OT based earth-mover (EM) distance. Compared with other methods adopting geometry-oblivious measures, RAAN can better reduce large feature distribution divergence.

2. To help adapt the classifier in UDA, we propose to match the label distribution by estimating a re-weighted source domain label distribution so that it can be similar

to the unknown target label distribution. In addition, we embed it into the procedure of minimizing the earth-mover distance during the end-to-end adversarial training procedure. This not only adapts the classifier but also helps match the marginal feature distribution.

The contributions of JAAN are summarized as follows:

1. First, to reduce the large discrepancy of the joint distribution, we embed the entropy regularized OT method, i.e., the Sinkhorn algorithm [166, 167] into the DCNN and adopt the adversarial training strategy to better match the joint distributions. More specifically, we designed an adaptive distance function using a presenter network and the cosine distance for the Sinkhorn algorithm. The presenter network, the feature extractor and classifier play an adversarial game where the presenter network aims to increase the divergence (Sinkhorn loss) of joint distribution, while the feature extractor and classifier aim to decrease it.

2. Second, we design a reconstruction network by projecting the joint representation of feature and label to either the reconstructed data or certain fixed features (when fine-tuning technique in DTN is used). Note that we share the same reconstruction network for both source and target domain reconstruction. This design can be regarded as another supervisory signal for constraining the target domain joint distribution so that it contains the complete information to reconstruct certain level information of the original images. Finally, our proposed JAAN is evaluated by conducting a series of experiments using datasets with different domain distribution discrepancies and we demonstrate that our technique can potentially address large distribution discrepancies.

5.2 Related Works

To the best of authors' knowledge, no research have focused on eliminating the variations of µ-DS caused by target personnels and the aspect angles in radar communities. This is the first developments of designing and applying state-of-the-art UDA methods for radar µ-DS classification. Therefore, we compare our RAAN and JAAN with the most recent UDA methods in computer vision literature.

5.2.1 Distribution Matching using Adversarial Training

There are two strategies to reduce domain discrepancies including *feature* and *pixel* distribution matching. For feature distribution matching methods, JS-divergence based methods are the best-performing techniques for measuring the divergence of feature distributions in DANs [168, 169, 170]. Although it is not a new statistical measure, the JS divergence loss is implemented by a mini-batch approach in the DCNN in an adversarial manner [171, 77]. Domain adversarial neural networks (DANN) [172] may be the first to add a domain classifier, with the aim of extracting not only discriminative features for the main classification task, but also indistinguishable ones for the domain classifiers. The adversarial loss of DANN is implemented by directly maximizing the domain classification loss and reversing the gradient in the back-propagation. Ghifary et al. designed the deep reconstruction-classification networks (DRCN) [173] using the same approach but added another loss function to minimize the reconstruction error of the data samples between domains. More recently, Tzeng et al. proposed Adversarial Discriminative Domain Adaptation (ADDA) [169] composed of two separate discriminative networks (one for each domain) to extract useful features for the main classification task. The domain discriminator network is added so that the target network and the domain discriminator network can compete with each other until the target and source domain features cannot be distinguished. However, the JS is only effective when the two distributions have common supports, which may not be a practical assumption.

For matching pixel distributions, inspired by the good performance of adversarial training in generative models, some methods generate new images that are transferable in both domains. Liu et al. proposed Co-GAN [168] including two GANs to generate diverse images for both source and target domain. Although Co-GAN achieved good performance in adapting domains having a small discrepancy, it cannot work well when the domain shifts are disparate [169]. In contrast to Co-GAN, the pixel-level domain adaptation network (pixelDA) proposed in [170] uses one generative network to generate images indistinguishable from source and target domains. In addition, constraints on pixel level similarity between the generated and source images are utilized. In fact, the ability of generative model based methods for UDA having large discrepancy is still under investigation. In addition, matching pixel-wise distribution of large scale images may not be feasible, because of the large discrepancy and the difficult task to generate the large-scale images with complex background and foreground objects.

5.2.2 Matching Feature Distribution using OT

The most closely related approach to ours for reducing the distribution divergence is through solving the OT directly, as described in [79, 78]. However, their implementation has not been embedded into the end-to-end learning framework and only the stand-alone De-Caffe features [162] from the DANN network are used. Instead, the methods proposed in this Chapter, namely RAAN and JAAN are among the first to incorporate OT in deep adaptation network in an end-to-end manner.

More specifically, RAAN utilizes the domain discriminator network with the objective of minimizing the dual formulation of the OT based EM distance. From this perspective, the Wasserstein GAN [174, 175] is a special case to minimize the dual of EM distance, however, their ultimate goal is to generate the images. To the best of author's knowledge, RAAN may be the first to learn domain invariant features for UDA utilizing the OT based EM distance in a DNN.

On the other hand, adopting the primal regularized OT, JAAN may be the first to learn domain invariant joint feature and label representation by directly reducing the Sinkhorn divergence loss [176]. To make the contributions more obvious, JAAN also designs the adaptive distance function which further improve the performances. Note that the concurrent work [177] also adopted the EM distance as the divergence measurement in UDA, however, RAAN is handling the more generalized scenario with unbalanced datasets.

5.3 Problem Formulation

In this section, we introduce the notations used in this chapter and formulate our problem. Suppose we are given a n_{cls}-class source domain (training) set $\mathbf{D}_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$ including n_s data x_i^s labeled by y_i^s and an unlabelled n_{cls}-class target domain(testing) set $\mathbf{D}_t = \{(x_j^t)\}_{j=1}^{n_t}$ composed of n_t images x_j^t. Let us assume they share the same label space. The random variables representing the data and label in general are denoted as X

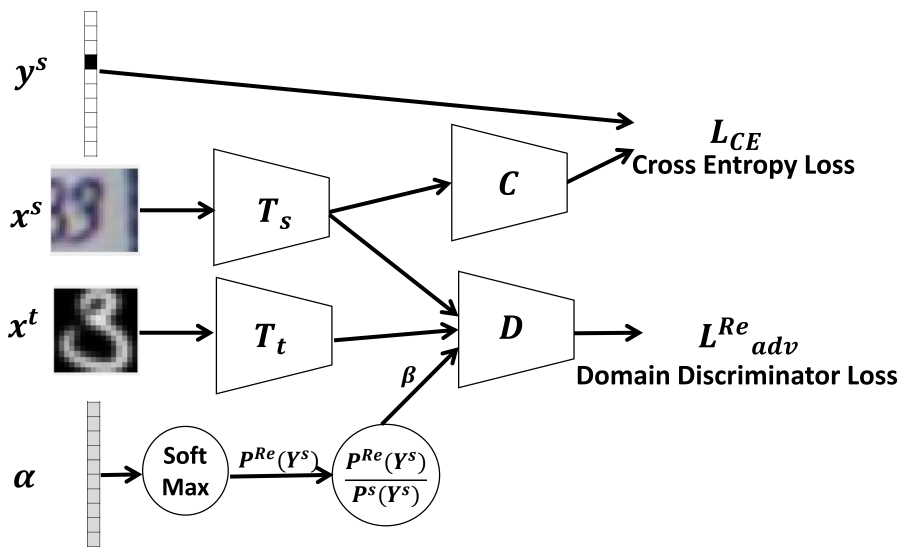


Figure 5.1: Architecture of RAAN.

and Y and the DCNN transformations for source and target domains are represented as T_s and T_t. We use $P^s(\cdot)$ and $P^t(\cdot)$ to denote the probability distributions of source and target domains.

As introduced in Section 2.5.2, to successfully conduct adaptation between domains in UDA, two essential problems are required to be addressed:

- Matching the feature distributions between $P^s(T_s(X^s))$ and $P^t(T^t(X^t))$;
- Adapting the classifier from source to target domains so that $P^s(Y^s|T_s(X^s)) = P^s(Y^t|T_t(X^t))$ or more explicitly $P^s(Y^s, T_s(X^s)) = P^t(Y^t, T_t(X^t))$.

5.4 Re-weighted Adversarial Adaptation Network

As illustrated in Figure 5.1, RAAN is composed of four networks, specifically two conventional L-layer DCNNs T_s and T_t, a domain discriminator network D and the classifier C. First, in the source domain, the DCNN T_s and the classifier C are trained to extract discriminative features from images x^s labeled by y^s by minimizing the cross entropy loss L_{CE}. Second, to adapt the classifier by matching the label distribution between domains, the re-weighted source domain label distribution $P^{Re}(Y^s)$ is computed by transforming a variable α using the soft-max function. Then it is straightforward to obtain the ratio vector as follows: $\beta = \frac{P^{Re}(Y^s)}{P^s(Y^s)}$. To extract transferable features for target domain images

x^t, the target domain DCNN T_t, domain discriminator D and the estimated density ratio vector β play the following adversarial game: β and D tries to discriminate whether features is from the target or source domain, while T_t tries to confuse D and β.

The first objective of RAAN is to adapt the classifier, which is difficult without the target domain labels. However, the low dimentionality and discrete nature of the label vector reduces the complexity to match between domains and we argue that this can assist with the adaptation of classifiers (see reasons in Section 5.4.2). With this knowledge, a re-weighted source domain label distribution $P^{Re}(Y^s)$ is obtained by mapping a variable $\alpha \in \mathbb{R}^{n_{cls}}$ by the soft-max function. Note that in the implementation, since β is a fixed function of α, learning α is equivalent to estimate the target label distribution. Estimation of α aims to ensure that $P^{Re}(Y^s)$ is similar to the unknown target one $P^t(Y^t)$. Consequently, the density ratio vector can be denoted as $\beta \in \mathbb{R}^{n_{cls}}$, with its $(y^s)^{th}$ element $\beta(y^s)$ calculated by $\beta(y^s) = \frac{P^{Re}(Y^s=y^s)}{P^s(Y^s=y^s)}$. As β can be directly computed based on α, in the following paper, we regard β as the variable under estimate.

The second objective of RAAN is to learn the domain invariant transformations T_s and T_t so that the disparate divergence between marginal feature distributions $P^t(T_t^{l=L}(X^t))$ and $P^s(T_s^{l=L}(X^s))$ is reduced. For brevity, we denote T_t and T_s to replace $T_t^{l=L}$ and $T_s^{l=L}$ respectively in the following. Given the images and labels in the source domain $\{x^s, y^s\} \in D_s$, with the aim of extracting discriminative features $T_s(x^s)$ for the classification, it is straight-forward to train the classifier C and T_s by minimizing the cross-entropy loss L_{CE} as follows:

$$\min_{T_s, C} L_{CE}. \tag{5.1}$$

To obtain transferable features $T_t(x^t)$ without labels, T_t is trained by playing an adversarial game with the domain discriminator network D and the ratio vector β so that the divergence between the re-weighted feature distribution in the source domain $\beta(y^s)P^s(T_s(x^s))$ and the target domain $P^t(T_t(x^t))$ is reduced. Additionally, to better reduce the divergence between disparate domains, the OT-based EM distance is reformulated in the adversarial manner, with more details shown in Section 5.4.1. Specifically, RAAN is trained in the following adversarial manner, where D with the help of β aims to discriminate whether features are from source or target domain, while T_t tries to confuse them. Based on the discriminator loss L_{adv}^{Re}, the following objective function can be obtained:

$$\min_{T_t} \max_{D,\beta} L_{adv}^{Re}. \tag{5.2}$$

Moreover, in addition to assuming with the adaptation of the classifier, matching the label distributions also eases the difficulty of matching the marginal feature distribution. The possible reason may be: if we assume that the feature generation processes are the same between domains, that is $P^s(T^s(X^s)|Y^s) = P^t(T^t(X^t)|Y^t)$, then $P^{Re}(Y^s) = P^t(Y^t)$ helps match the marginal feature distributions $P^s(T^s(X^s)) = P^t(T^t(X^t))$.

In Section 5.4.1, to match the marginal feature distributions between domains, an OT based EM distance is introduced and implemented in an adversarial manner in RAAN. Then in Section 5.4.2, we propose to match label distributions between domains and embed it in the adversarial training. We also explain why this helps to adapt the classifier and meanwhile to match marginal feature distributions. Finally in Section 5.4.3, we formulate the final objective function of RAAN.

5.4.1 Optimal Transport in Adversarial Training

Suppose that the empirical distributions of source and target domain features are denoted as μ^s and μ^t respectively as follows:

$$\mu^s = \sum_i^{n_s} p_i^s \delta_{T_s(x_i^s)},\ \mu^t = \sum_j^{n_t} p_j^t \delta_{T_t(x_j^t)} \tag{5.3}$$

where $\delta_{T_s(x_i^s)}$ and $\delta_{T_t(x_j^t)}$ are the Dirac functions at location $T_s(x_i^s)$ and $T_t(x_j^t)$ and p_i^s and p_j^t are their probability masses. Then, the joint probabilistic coupling, or the transportation plan between feature distributions in source and target domains can be defined as γ with the marginals μ^s and μ^t. In the discrete version, the set of probabilistic couplings $\mathbf{B}$ can be defined by Eq.(5.4) [79]:

$$\mathbf{B} = \left\{\gamma \in (\mathbb{R}^+)^{n_s \times n_t} | \gamma 1_{n_t} = \mu^s, \gamma^T 1_{n_s} = \mu^t \right\}. \tag{5.4}$$

In general, to reduce feature distribution divergence, OT based methods first estimate the optimal transportation plan between two distributions and then learn the feature transformation to minimize the cost of such a plan. Therefore, we first define the metric $J(\mu^s, \mu^t)$ in Eq.(5.5) to measure the total cost of transporting probability masses from target to source domains, where M is the distance matrix whose $(i, j)^{th}$ element is defined by the distance cost function $c(T_s(x_i^s), T_t(x_j^t))$ between features. The $(i, j)^{th}$ element $\gamma(i, j)$ indicates how much mass is moved from $T_t(x_j^t)$ to $T_s(x_i^s)$, and F is the Frobenius dot product. Subsequently for brevity, we drop the index i, j to represent x_i^s, x_j^t as x^s, x^t. After that, the OT γ_0 can be estimated by minimizing the cost $J(\mu^s, \mu^t)$ in Eq.(5.6), with the optimal transportation cost or the well-known EM distance defined by $W(\mu^s, \mu^t)$ in Eq.(5.7)[178]. Finally, assuming the ideal source domain features $T_s(x^s)$ are available, to learn the transferable features in target domains, it is intuitive to train the DCNN transformation T_t under the objective of minimizing the EM distance $W(\mu^s, \mu^t)$, as shown in Eq.(5.8).

$$J(\mu^s, \mu^t) = \langle\, \gamma, M \rangle_F,\ with\ \gamma \in \mathbf{B}, \tag{5.5}$$

$$\gamma_0 = \underset{\gamma \in \mathbf{B}}{\operatorname{argmin}} J(\mu^s, \mu^t)[79] \tag{5.6}$$

$$W(\mu^s, \mu^t) = \min_{\gamma \in \mathbf{B}} J(\mu^s, \mu^t) \tag{5.7}$$

$$\min_{T_t} W(\mu^s, \mu^t) \tag{5.8}$$

To avoid using linear programs or iterative algorithms (since this is difficult to implement in DCNN gradient descent based framework) to compute the constraint of γ in Eq.(5.4), the dual formulation of $W(\mu^s, \mu^t)$ is utilized in Eq.(5.9) (following Eq.(5.3) in [179]), considering the capability of batch-wise back-propagation in DNN. More specifically, we use the domain discriminator network D and its variant $\hat{D}$ as two dual functions in the following:

$$\begin{gathered} W(\mu^s, \mu^t) = \max_{D\, \hat{D}} L_{adv},\ where \\ L_{adv} = \mathop{\mathbb{E}}_{x^s \sim P^s(X^s)} D(T_s(x^s)) + \mathop{\mathbb{E}}_{x^t \sim P^t(X^t)} \hat{D}(T_t(x^t)) \\ s.t. D(T_s(x^s)) + \hat{D}(T_t(x^t)) \leq c(T_s(x^s), T_t(x^t)). \end{gathered} \tag{5.9}$$

For RAAN, we choose the following distance cost function $c(T_s(x^s), T_t(x^t)) = ||T_s(x^s) -$

$T_t(x^t)\|$ for reasons of computational efficiency and permitting gradient measurements, however, this does not infer that it is the only function that could be selected. According to the constraint Eq.(5.9), the best function that $\hat{D}$ has to be is $-D$, as $c(T_s(x^s), T_t(x^t))$ is defined to be non-negative. In this way, the constraint in Eq.(5.9) is equivalent to ensuring that D is a 1-Lipschitz function, or alternatively its gradient norm is smaller than 1. Therefore, if we use Eq.(5.9) to replace the EM distance Eq.(5.8), the DCNN transformation T_t and the domain discriminator network D can be trained based on the mini-max objective function in Eq.(5.10),

$$\min_{T_t} W(\mu^s, \mu^t) = \min_{T_t} \max_{D} L_{adv}, \quad where$$

$$L_{adv} = \sum_{(x^s,y^s)\sim P^s(X,Y^s)} D(T_s(x^s))P^s(T_s(x^s)|y^s)P^s(y^s) - \mathop{\mathbb{E}}_{x^t\sim P^t(X^t)} D(T_t(x^t)) \tag{5.10}$$

$$s.t. \| \bigtriangledown_{T_t(x^t)} D(T_t(x^t))\|_2 \leq 1, \ \| \bigtriangledown_{T_s(x^s)} D(T_s(x^s))\|_2 \leq 1.$$

5.4.2 Adapting the Classifier by Label Distribution Matching

Although OT based EM distance is utilized to match feature distributions $P^s(T_s(X^s))$ and $P^t(T_t(X^t))$, we argue that it is not sufficient to successfully adapt the classifier from source to target domain, since $P^s(T_s(X^s)) = P^t(T_t(X^t))$ does not infer $P^s(Y^s|T_s(X^s)) = P^t(Y^t|T_t(X^t))$. However, according to Bayes rule in (5.11), instead of matching $P^t(Y^t|T_t(X^t))$ and $P^s(Y^s|T_s(X^s))$ directly, we can learn T_t under the objective of matching $P^s(T_s(X^s)|Y^s)P^s(Y^s)$ and $P^t(T_t(X^t)|Y^t)P^t(Y^t)$.

$$P(T(X)|Y)P(Y) \propto P(Y|T(X)). \tag{5.11}$$

In fact, as no label information in the target domain $P^t(Y^t)$ is available, it is non-trivial to directly match $P^t(T_t(X^t)|Y^t)P^t(Y^t)$ and $P^s(T_s(X^s)|Y^s)P^s(Y^s)$. However, as the label is a low-dimensional and discrete variable whose distribution is well-defined, it is more straightforward to match label distributions between domains compared with its conditional variant. Therefore, we take a step back and propose to estimate the re-weighted source domain label distribution $P^{Re}(Y^s)$ so that it is similar to the unknown $P^t(Y^t)$ in the target domain. In fact, we argue that matching the label distributions between two domains can also help the adaptation of the classifier, because at least part of $P^t(T_t(X^t)|Y^t)P^t(Y^t)$ and $P^s(T_s(X^s)|Y^s)P^{Re}(Y^s)$ is matched. Based on such an assumption, since the EM-distance has been adopted to match $P^t(T_t(X^t)$ and $P^s(T_s(X^s)$, we propose to embed the re-weighted label distribution $P^{Re}(Y^s)$ into the procedure of matching the marginal feature distribution $P^s(T_s(X^s))$ and $P^t(T_t(X^t))$ in the adversarial training. To estimate the re-weighted label distribution $P^{Re}(Y^s)$, the constraint in Eq.(5.12) should be considered, where, y_i indicates the label of the i^{th} class. However, this constraint has already been considered in the implementation using the softmax function.

Finally, to estimate the re-weighted label distribution, if we directly replace the $P^s(Y^s)$ by $P^{Re}(Y^s)$ in the mini-max objective function L_{adv} in Eq.(5.10), a new loss function L^{Re}_{adv} is obtained in Eq.(5.13), where the network D, T_t and the ratio vector β are trained in the following manner: D and β are trained in a cooperative way to estimate the EM-distance, while T_t is trained to minimize the EM-distance. From the perspective of implementation, β can be regarded as assigning different significance to data x^s in the source domain, so that the mini-batches in the two domains are sampled from similar distributions, which helps D and T_t to focus on matching $P^s(T_s(x^s))$ and $P^t(T_t(x^t))$.

$$\sum_{i=1}^{n_{cls}} P^{Re}(Y^s = y_i) = 1, \tag{5.12}$$

$$\begin{aligned}
&\min_{T_t}\max_{D,\beta} L^{Re}_{adv}, \quad where \\
L^{Re}_{adv} &= \sum_{(x^s,y^s)\sim P^s(X^s,Y^s)} D(T_s(x^s))P^s(T_s(x^s)|y^s)P^{Re}(y^s) - \mathop{\mathbb{E}}_{x^t\sim P^t(X^t)} D(T_t(x^t)) \\
&= \sum_{(x^s,y^s)\sim P^s(X^s,Y^s)} D(T_s(x^s))P^s(T_s(x^s)|y^s)\beta(y^s)P^s(y^s) - \mathop{\mathbb{E}}_{x^t\sim P^t(X^t)} D(T_t(x^t)) \\
&= \mathop{\mathbb{E}}_{(x^s,y^s)\sim P^s(X^s,Y^s)} \beta(y^s)D(T_s(x^s)) - \mathop{\mathbb{E}}_{x^t\sim P^t(X^t)} D(T_t(x^t)) \\
&s.t.\| \nabla_{T_t(x^t)} D(T_t(x^t))\|_2 \leq 1, \| \nabla_{T_s(x^s)} D(T_s(x^s))\|_2 \leq 1.
\end{aligned} \tag{5.13}$$

5.4.3 Optimization in RAAN

As shown in Figure 5.1, RAAN is proposed to jointly minimize the cross-entropy loss of the source domain samples and to reduce the divergence of the extracted feature distributions. First, we define the empirical estimate of the loss function L_{adv}^{Re} as follows:

$$L_{adv}^{Re} = \frac{1}{n_s}\sum_{i=1}^{n_s} D(\beta(y_i^s)T_s(x_i^s)) - \frac{1}{n_t}\sum_{j=1}^{n_t} D(T_t(x_j^t)). \tag{5.14}$$

Following on from the idea of controlling the 1-Lipschitz function of the domain discriminator network D [175], we explicitly constrain the gradient norm penalty term as follows:

$$L_{gp} = \| \bigtriangledown_{\hat{T}(\hat{x})} L_{adv}^{Re} - 1\|_2, \tag{5.15}$$

where $\hat{T}(\hat{x})$ is the weighted interpolation samples of $T_t(x^t)$ and $T_s(x^s)$. In summary, the total objective function in RAAN is formulated in the following adversarial manners:

$$\min_{D,\beta} - L_{adv}^{Re} + \lambda_{gp} L_{gp} + \lambda_{reg} \|\beta\|_2, \tag{5.16}$$

$$\min_{T_t} - \frac{1}{n_t}\sum_{j=1}^{n_t} D(T_t(x_j^t)), \tag{5.17}$$

$$\min_{T_s,C} \frac{1}{n_s}\sum_{i=1}^{n_s} L_{CE}(C(T_s(x_i^s)),y_i^s), \tag{5.18}$$

where $L_{CE}(C(T_s(x_i^s)),y_i^s)$ indicates the CE function with classifier C, feature vector $T_s(x_i^s)$ and its ground-truth one-hot label vector y_i^s. Note that to train the networks stably, the source domain DCNN T_s is trained first while T_t and D are trained to match the feature distributions between $T_t(x^t)$ and $T_s(x^s)$ in an adversarial manner. In addition, to stably learn the ratio vector, we add the ℓ_2-norm of β as the regularization term in Eq.(5.16). λ_{gp} and λ_{reg} indicate the regularization weights of the gradient penalty term and the ℓ_2-norm of the ratio vector respectively.

5.5 Joint Adversarial Adaptation Network(JAAN)

In the following, we introduce the notations used in JAAN. As illustrated in Figure 5.2, JAAN is composed of six networks, specifically two conventional L-layer DCNNs T_s and T_t, classifier networks C_s and C_t, a presenter network P and a reconstruction network R. In addition, we concatenate the feature and label vector as $J_i^s = concat(x_i^s, C_s(T_s^{l=L}(x_i^s)))$

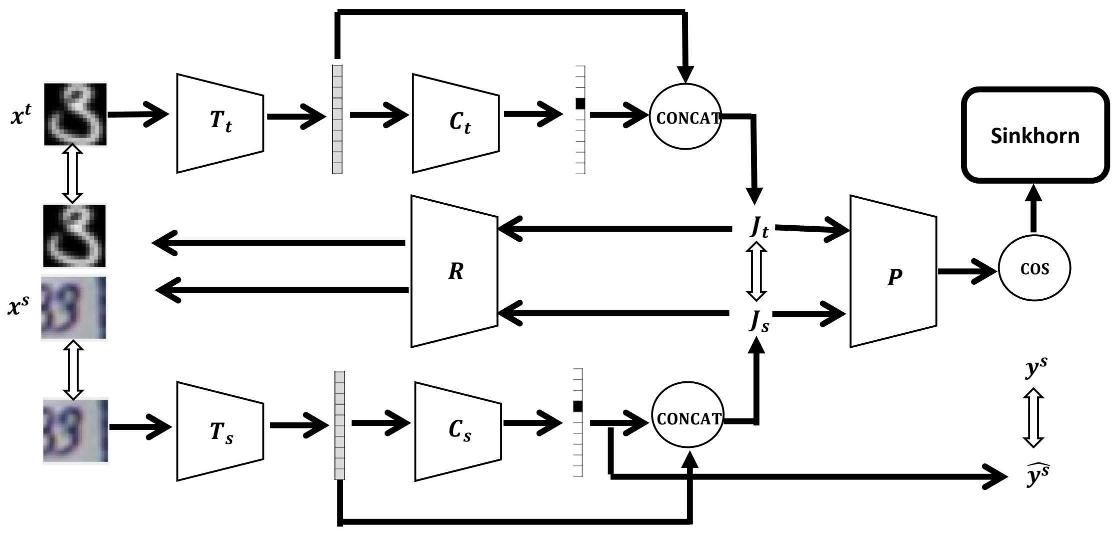


Figure 5.2: Architecture of JAAN.

to represent the joint representation and so as J^t_j for the one in the target domain. In general, JAAN's architecture and functions are illustrated in the following: first, in the source domain, the DCNN T_s and the classifier C_S are trained to extract discriminative features from images x^s labeled by y^s by minimizing the cross entropy loss $\widehat{L_{CE}}$. Second, to jointly match feature and label distribution, we concatenate feature and predicted label in two domains as J^s and J^t. Learning the target domain feature transformer and the classifier is first supervised by the error between raw images (or certain fixed feature in fine-tuning operation) and the reconstructed one using the shared reconstruction network R. Next, to extract domain-invariant joint representations J^s and J^t, the T_t and C_t are updated based on minimizing the optimal transportation efforts of transporting masses from the target domain distribution to the source one, which is calculated by the Sinkhorn algorithm having input as their distance between distribution masses. JAAN learns an adaptive distance function represented by the cosine distance between outputs of the presenter network P. Note that the adaptive distance function is updated adversarially with the update of feature transformer and classifier.

The first objective of JAAN is to learn the discriminative feature transformations T_s and classifier C_s for the main classification task, by minimizing the following empirical cross entropy loss $\widehat{L_{CE}}$ in Eq.(5.19).

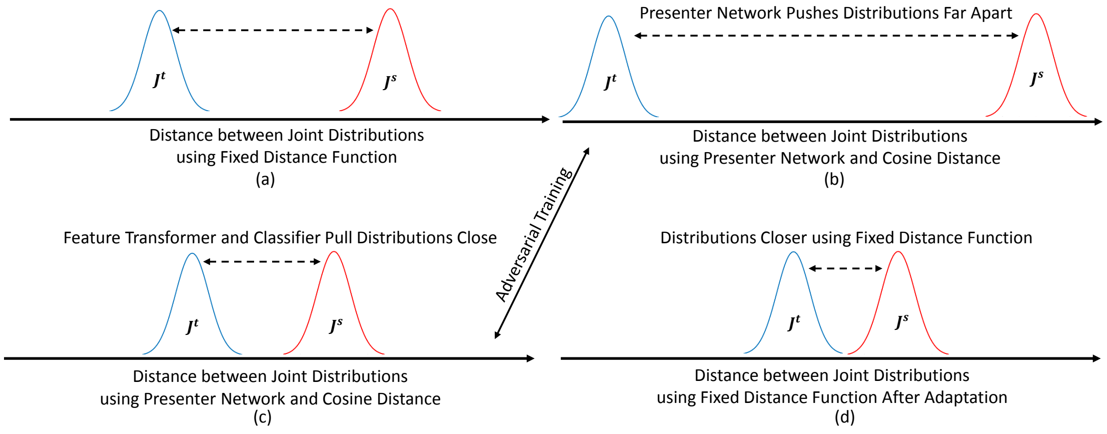
$$\min_{T_s,C_s} \widehat{L_{CE}}. \tag{5.19}$$

The second objective of JAAN is to learn domain invariant feature transformers T_s, T_t and classifiers C_s, C_t so that discrepancies between joint distributions of feature and label $P^t(T_t^{l=L}(x^t), C_t(T_t^{l=L}(x^t)))$ and $P^s(T_s^{l=L}(x^s), C_s(T_s^{l=L}(x^s)))$ are minimized. This requires the following two operations:

1. First, it is required that the joint representation of feature and predicted label from the two domains must reconstruct their raw data (or low-level features if fine-tuning technique is adopted) using the shared reconstruction network R. This is performed by minimizing the following two reconstruction losses L^s_{recon} and L^t_{recon} respectively. This operation may implicitly guarantee that the joint representation of feature and predicted label are at least scale invariant and preserve the complete information of the data, as shown in Eq.(5.20).

$$\min_{T_s,C_s,T_t,C_t,R} L^s_{recon} + L^t_{recon}. \tag{5.20}$$

2. Second, to further explicitly reduce distribution discrepancies between domains, we wish to learn the domain-invariant feature transformers and classifiers T_s, C_s, T_t, C_t by minimizing the OT based transportation cost, $L_{Sinkhorn}$ as shown in Eq.(5.21). In JAAN, we calculate the $L_{Sinkhorn}$ using the conventional entropy regularized OT method called the Sinkhorn algorithm, with joint representations J^s_i, J^t_j and the distance function $c(\cdot,\cdot)$ as inputs. To ensure that $c(\cdot,\cdot)$ largely discriminates the J^s_i and J^t_j, JAAN chooses to design and learn the adaptive distance function $c(J^s_i, J^t_j)$ as the cosine distance between the presenter network's outputs $P(J^s_i)$ and $P(J^t_j)$, indicated by Eq.(5.22) and (5.26). A more illustrative figure to explain this adversarial training process is shown in Figure 5.3. To sum up, the presenter network P plays an adversarial game with the feature transformers T_s, T_t and the classifiers C_s, C_t so that the joint distribution divergence between domains can be reduced.



J^t
J^s
Distance between Joint Distributions using Fixed Distance Function
(a)
Presenter Network Pushes Distributions Far Apart
Distance between Joint Distributions using Presenter Network and Cosine Distance
(b)
Adversarial Training
Feature Transformer and Classifier Pull Distributions Close
Distance between Joint Distributions using Presenter Network and Cosine Distance
(c)
Distributions Closer using Fixed Distance Function
Distance between Joint Distributions using Fixed Distance Function After Adaptation
(d)

Figure 5.3: Adversarial training in JAAN: (a) Distance between joint distributions described by a fixed distance function (b) Distance between joint distributions using the adaptive distance function, defined by the cosine distance of presenter network output. Presenter network P pushes the two distributions to increase the distance between them. (c) Although P pushes distributions, T_t, C_t,T_s and C_s pull the distribution closer even than before, so that joint distributions under the adaptive distance function are matched closer. (d) Distance between joint distributions under the fixed distance function after adaptation is closer so that the distributions are matched between domains.

$$\min_{T_s,C_s,T_t,C_t} L_{Sinkhorn}(c(J_i^s,J_j^t)) \tag{5.21}$$

$$\max_{P} L_{Sinkhorn}(c(J_i^s,J_j^t)) \tag{5.22}$$

In Section 5.5.1, we introduce the entropy-regularized OT method and the Sinkhorn loss for reducing the joint distribution divergence. Section 5.5.2 focuses on particularly the way to learn the adaptive and discriminative distance function adversarially from the feature transformers and classifiers. In Section 5.5.3, we formulate the final objective function and the optimization of JAAN.

5.5.1 Distribution Matching using Entropy-Regularized OT

Suppose that $\delta_{T_s(J_i^s)}$ and $\delta_{T_t(J_j^t)}$ are the Dirac functions at location $T_s(J_i^s)$ and $T_t(J_j^t)$, with p_i^s and p_j^t their probability masses, we can define the empirical joint distributions of feature and label, μ^s and μ^t in source and target domain in Eq.(5.3). In addition, as we are working in the discrete settings, $\mathbf{B}$ is defined as a set of probabilistic couplings γ in Eq.(5.4).

In general, to reduce joint distribution divergence between domains in UDA, JAAN first estimates the optimal transportation plan (the coupling γ) between two domains' distributions and secondly learns the feature transformation and classifier to minimize the transportation efforts of such a plan. For the consideration of computational efficiency, we utilized the entropy-regularized Sinkhorn loss $L_{sinkhorn}(\mu^s,\mu^t)$ [176] to represent such transportation efforts in Eq.(5.23), where M is the distance matrix whose $(i,j)^{th}$ element is defined by the distance function $c(T_s(J_i^s),T_t(J_j^t))$ between joint representations. The $(i,j)^{th}$ element of $\gamma(i,j)$ indicates how much mass is moved from $T_t(J_j^t)$ to $T_s(J_i^s)$, and F is the Frobenius dot product. The regularization term $\sum_{i,j=1}^{n_s,n_t} log(\gamma(i,j))$ indicates the entropy operation of the γ, ε represents the weight of this smooth regularization term and n_s, n_t are the number of samples in source and target domain when defining the discrete distributions. Subsequently for brevity, we drop the index i,j to represent J_i^s,J_j^t as J^s,J^t. It is then straight-forwarded to update the feature transformer and classifiers so that the Sinkhorn loss is minimized. Note that details of the iterative Sinkhorn algorithm can be found in [176] and we provide detailed pseudo-code of our implementation in Appendix B.4.

$$L_{sinkhorn}(\mu^s,\mu^t) = \langle\gamma,M\rangle_F + \varepsilon \sum_{i,j=1}^{n_s,n_t} log(n_i n_j \gamma(i,j)),\ with\ \gamma \in \mathbf{B}, \tag{5.23}$$

$$\min_{T_s,C_s,T_t,C_t} L_{Sinkhorn} \tag{5.24}$$

5.5.2 Adaptive Distance Function Learning in OT

The large domain discrepancy between joint distributions of feature and label implies that it may be difficult to find out a proper distance function c that is discriminative for comparing joint representations J^s and J^t. Therefore, JAAN selects to learn the distance function to increase the transportation efforts $L_{Sinkhorn}$ even for very similar representations J^s and J^t. Guided by this principle, learning the distance function c and the feature transformers and classifiers are trained in an adversarial way, as shown in Eq.(5.25). Rather than directly learning a distance function, JAAN designs and learns a presenter network P for projecting the joint representations J^s and J^t to a discriminative latent space and defines the cosine distance of these outputs as the distance function c, as shown in Eq.(5.26). In this way, the final loss function is given in Eq.(5.27).

$$\min_{T_s,C_s,T_t,C_t} \max_{P} L_{Sinkhorn} \tag{5.25}$$

$$c(J_i^s, J_j^t)) = 1 - \frac{P(J_i^s)^T P(J_j^t)}{\|P(J_i^s)\|_2 \|P(J_i^s)\|_2} \tag{5.26}$$

$$\min_{T_s,C_s,T_t,C_t} \max_{P} L_{Sinkhorn} \tag{5.27}$$

5.5.3 Optimization

As shown in Figure 5.2, JAAN is proposed to jointly minimize the cross entropy loss $\widehat{L_{CE}}$, the image-level (or the feature-level) reconstruction loss from two domains $L_{recon}^s + L_{recon}^t$ and the adaptive Sinkhorn loss $L_{Sinkhorn}$ to reduce the domain discrepancies. Specifically, the empirical cross entropy loss $\widehat{L_{CE}}$ and $L_{recon}^s + L_{recon}^t$ are shown in Eq.(5.28) and (5.29) respectively, where L_{CE} indicates the cross entropy operation and we use the ℓ_1 norm as the reconstruction loss.

$$\min_{T_s,C_s} \widehat{L_{CE}} = \min_{T_s,C_s} \frac{1}{n_s} \sum_{i=1}^{n_s} L_{CE}(C_s(T_s(x_i^s)), y_i^s), \tag{5.28}$$

$$\min_{T_s,C_s,T_t,C_t.R} \frac{1}{n_s} \sum_{i=1}^{n_s} \|x_i^s - R(J_i^s)\|_1 + \frac{1}{n_t} \sum_{i=1}^{n_t} \|x_j^t - R(J_j^t)\|_1 \tag{5.29}$$

In summary, the total objective function in JAAN is formulated in the following adversarial manner in Eq.(5.30), where $\lambda_{Sinkhorn}$ and λ_{recon} indicate the regularization weights of the adaptive Sinkhorn loss and the pixel-wise reconstruction loss respectively.

$$\min_{T_s,C_s,T_t,C_t,R} \max_{P} \lambda_{Sinkhorn} L_{Sinkhorn} + \lambda_{recon}(L_{recon}^s + L_{recon}^t) + \widehat{L_{CE}} \tag{5.30}$$

5.6 Implementation

The experiments in this chapter involve three adaptation tasks in hand-written digit datasets, cross-modality datasets and the µ-DS datasets, with the details illustrated in Section 5.7. These experiments are conducted on a GPU cluster, with two NVIDIA K20 GPU cards integrated in the Dell T620 server. For all experiments,

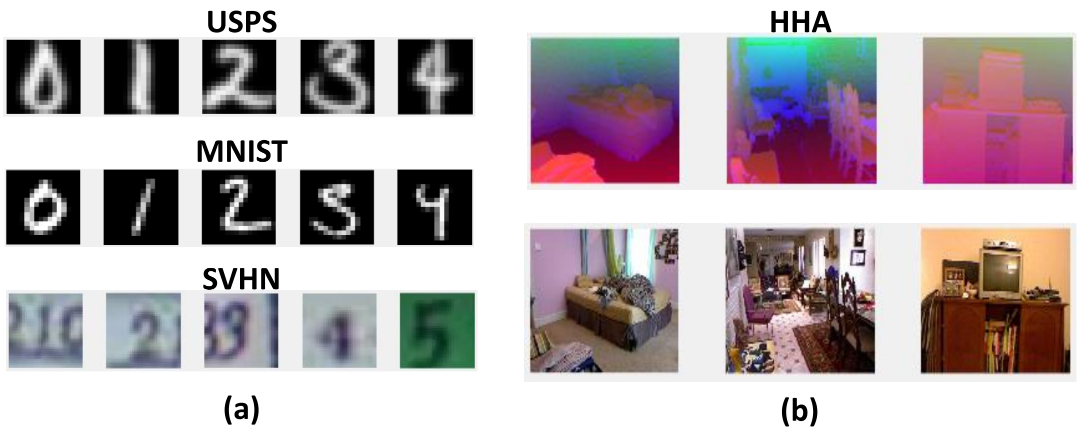


Figure 5.4: UDA datasets: (a) Three hand-written digit datasets; (b) cross-modality dataset including RGB and RGB-depth images.

we utilize the Adam optimizer, with the learning rate selected from the following set:$\{2e^{-5}, 5e^{-5}, 1e^{-4}, 2e^{-4}, 5e^{-4}, 1e^{-3}\}$. For the regularization weights, λ_{gp} and λ_{reg} are chosen from the following sets $\{1, 10, 50, 100\}$ and $\{0.1, 1, 10, 50, 100, 500\}$ respectively. We used the exponential decay, with decay factor of 0.99 for every 1000 iterations. All experiments are run 10 times, each for 100000 iterations and we report the average results. For all adaptation tasks, the batch size used is 128. For pre-processing of μ-DS and hand-written dataset, we normalize data in the range of [0,1]. For cross-modality datasets, we utilized the pre-processing scheme including detecting objects in the two modalities, cropping and resizing in ADDA [169] and VGG-16 network [154].

5.7 Experiments and Results

In this section, we introduce three UDA tasks including adaptation among the hand written digit datasets, cross-modality datasets and the μ-DS datasets. RAAN and JAAN are evaluated using them and we compare them with the most recent UDA methods. Additionally, we also describe and analyze the results in this section.

5.7.1 Adaptation between Hand-Written Digit Datasets

The first UDA task adapts between three hand-written digit datasets including MNIST [180], USPS [181] and SVHN [182]. As shown in Figure 5.4, adaptation between these three datasets are of varying difficulty. MNIST and USPS are both composed of greyscale images in a fairly well-controlled environment. To evaluate RAAN and JAAN in reducing large domain discrepancies, SVHN is also explored which is composed of RGB images in more complicated real-world scenarios, e.g, misalignment of images and different light conditions. In addition, note that the sub-class instances of SVHN are largely unbalanced (with different data numbers in each class category). Two useful measures are adopted to quantify the domain discrepancies, including the recognition results before any adaptation (which is also called the source-only recognition rates) and the well-known A-distance which calculates the generalization error of classifying the source and target domain features as a binary classification task [183, 184]. These results are shown in Table 5.1 and 5.2. Specifically in this thesis, we adopt the SVM classifier to calculate the generalization error θ of classifying the source and target domain features as a binary classification task. Then the A-distance d can be calculated as follows: $d = 2(1-2\theta)$. From observing Table 5.1, the A-distance of pair (SVHN,MNIST) is the highest, larger than the (USPS,MNIST) by 0.049 and (MNIST,USPS) by 0.079. This corresponds to the source-only recognition rates in Table 5.2 where the larger the A-distance is, the lower recognition rates achieved in the source-only adaptation tasks. These all suggest that the domain discrepancies are: $(SVHN, MNIST) > (USPS, MNIST) > (MNIST, USPS)$.

Table 5.1: A-distance between hand-written digit datasets; M, U and S refer to MNIST, USPS and SVHN respectively.

Adaptation	S to M	M to S	S to U	U to S	M to U	U to M
A-Distance	1.993	1.997	1.979	1.984	1.914	1.944

For the task of adapting between hand written digit datasets, the following three adaptation directions are chosen for the evaluation: from MNIST to USPS, from USPS to MNIST and from SVHN to MNIST. We adopt a variant of LeNet as network T_s, T_t and the domain discriminator network D is composed of three fully-connected layers activated by the ReLU with output activation numbers of 512, 512, 1 respectively. As for the dataset and experimental protocols, we utilize the one in [185] for adapting between MNIST and USPS, while for adaptation from SVHN to MNIST, we choose that in [169] for fair comparisons.

To assess the reasons underlying RAAN's performance, we denote RAAN(+) and RAAN(-) as RAAN with and without the re-weighting scheme respectively. Note that we

Table 5.2: Recognition rates (%) of adapting hand-written digit dataset.

Methods	MNIST to USPS	USPS to MNIST	SVHN to MNIST
Source Only	72.5	61.2	59.3

Table 5.3: Recognition rates (%) of adapting hand-written digit dataset.

Methods	MNIST to USPS	USPS to MNIST	SVHN to MNIST
Source Only	72.5	61.2	59.3
W-MMD [186]	72.6	65.4	67.4
Gradient Reversal[172]	77.1	73.0	73.9
Domain Confusion [187]	79.1	66.5	68.1
Co-GAN[168]	91.2	89.1	No Converge
ADDA [169]	89.4	90.1	76.0
RAAN(-)(Ours)	88.3	91.5	80.7
RAAN(+)(Ours)	89.0	92.1	89.2
JAAN (Ours)	**92.2**	**93.7**	**91.8**

are not integrating the re-weighting scheme in JAAN from the following two reasons: i) JAAN has explicitly consider to embed the label in the joint representation; ii) we tried to implement the re-weighting scheme in Sinkhorn loss but this does not converges finally. As shown in Table 5.3, when adapting between MNIST and USPS, compared with ADDA and Co-GAN, the proposed RAAN(-), RAAN(+) and JAAN achieve very competitive results and JAAN and RAAN(+) slightly outperform RAAN(-). In the most difficult task, i.e., adapting from SVHN to MNIST, RAAN(-), RAAN(+) and JAAN achieve 80.7%, 89.2% and 91.8% respectively, outperforming the state-of-the-art ADDA by 4.7%, 13.2% and 15.8%, while Co-GAN does not converge in this experiment. Based on this, it seems that the weight-sharing approach utilized in Co-GAN is not capable of generating transferable images between disparate domains such as MNIST and SVHN [169]. As RAAN(-) and JAAN utilize the same DCNN architecture to ADDA's and Co-GAN's, RAAN(-)'s and JAAN's superior performances are mainly owing to the OT based objective function and the joint feature and label representations. We hypothesize that the OT based objective function is able to better reduce the feature distribution divergence when the domains are disparate, e.g., SVHN and MNIST. With the fact that JAAN outperforms RAAN(+), we hypothesize that compared with RAAN (utilizing the dual formualtion of OT), reducing the Sinkhorn loss (primal formulation of OT) with the adaptive distance function is able to better reduce joint distribution divergence of feature and label than JS and MMD based divergence. In addition, based on the fact that RAAN(+) achieves superior performances to both ADDA and RAAN(-), we hypothesize that matching the label distribution helps adapt the classifiers, and embedding it into minimizing the EM distance of feature distributions can be regarded as two cooperative tasks.

5.7.2 Adaptation between Cross-Modality Datasets

To continue evaluating RAAN and JAAN in reducing large domain shifts, the second adaptation task is designed using the NYU-D dataset [188], adapting from the indoor object images in RGB format to the depth variants encoded by the HHA[1] format[189]. The 19-class dataset is extracted following the scheme in [169]. As shown in Figure 5.4, the domain shifts between images of RGB and HHA format are fairly large, mainly due to the low image resolutions and potential mis-alignments caused by the coarse cropping box. In addition, as shown in the instance number in Table 5.4, this dataset has unbalanced sub-class instances. Furthermore, it is challenging as the images from the target domain are in a completely different format from those in the source domain.

In this section, RAAN and JAAN are evaluated in the presence of large domain shifts that confront the adaptation from RGB images to RGB-depth images. To enable a fair comparison, we follow ADDA's experimental set-up [169] and utilized the VGG-16 architecture [154] for DCNNs T_s and T_t for RAAN and JAAN. The domain discriminator network D used for RAAN is composed of three fully-connected layers activated by the ReLU, with 1024,2048,1 outputs respectively. For JAAN, the presenter network P is composed of three fully-connected layers activated by the ReLU, with 1024, 2048, 64 outputs respectively. For the reconstruction network R, we use the fully convolution network settings to reconstruct not the raw data, but only the output of the last convolution layer results of VGG-16 network. The reason for this is that training the de-convolution networks using the adaptation datasets are difficult, since the whole VGG-16 network is trained using the ImageNet dataset. The classifiers C_s and C_t in JAAN and the one in

[1]HHA refers to an encoding technique to encode the depth image with three channels at each pixel: horizontal disparity, height above ground, and the angle the pixels local surface normal makes with the inferred gravity direction.

RAAN use one FC layer, with the the predicted label with the size of 19.

As shown in Table 5.4, we report the sub-class classification accuracy achieved by RAAN(-), RAAN(+) and JAAN, along with the re-weighted label distribution $P^{Re}(Y^s)$ yielded by RAAN(+) and the target one $P^t(Y^t)$. It can be observed from the overall recognition rates that RAAN(+) achieves an average of 34.3%, outperforming ADDA by 6.7%, RAAN(-) by 3.5% and JAAN by 1.1%. In addition, RAAN(-) and JAAN outperform ADDA by 3.2% and 5.6% respectively. The reason of RAAN(+)'s improvement over JAAN may be because for handling *extremely unbalanced* class distributions, re-weighting the label distribution in RAAN(+) is more effective than integrating the label in the joint representation as JAAN does. This is reasonable as RAAN(+)'s re-weighting scheme is explicitly integrated in the domain discriminator loss but we have to implicitly constrain to match the label distribution via the joint representation in JAAN. In fact, ADDA only achieves better performance in classes "desk", "dresser", "garbage bins" and "sofa" whose instance numbers are small. It can also be seen that RAAN(+) outperforms RAAN(-) not only from the overall recognition accuracy but also from how many classes the classifier can recognize (classes with the recognition rates more than 0%). This is potentially due to the fact that the re-weighting scheme increases the significance of instances from the sub-classes with a lower number of instances. This can be verified by comparing the number of sub-class instances with the estimated ratio vector β in Table 5.4. Although RAAN(-) and JAAN utilize the objective functions inspired by the same OT theory, JAAN outperforms RAAN(-) which implies that the prime OT formulation and the objective of regularized OT performs more robust to reduce large domain discrepancies.

5.7.3 Adaptation between Radar μ-DS Datasets

In this section, we evaluate adaptation results of RAAN and JAAN using the mono-static μ-DS from the same dataset in Chapter 4 and the details of the dataset and pre-processing methods are described in Section 4.6. We consider the domain discrepancies of μ-DS classification caused by two practical factors of variation: (i) different aspect angles and (ii) different target personnels. These two factors may give rise to potentially large statistical gaps (domain discrepancies) between training and testing datasets. Note that we

Table 5.4: Recognition rates (%) of adapting cross-modality dataset.

class	bathtub	bed	bookshelf	box	chair	counter	desk	door	dresser	garbage bin	lamp	monitor	night stand	pillow	sink	sofa	table	television	toilet	Overall
NO.instances	19	96	87	210	611	103	122	129	25	55	144	37	51	276	47	129	210	33	17	2401
Source Only	0.0	76.0	0.0	10.5	13.1	2.9	9.0	45.7	0.0	3.6	12.5	0.0	0.0	54.9	0.0	2.3	14.8	3.0	0.0	18.9
ADDA	0.0	46.9	0.0	0.5	76.2	19.4	1.6	51.9	**4.0**	**1.8**	0.7	0.0	0.0	8.3	0.0	**6.2**	13.8	0.0	0.0	27.6
RAAN(-)	0.0	**50.0**	**29.9**	0.0	62.9	1.9	0.0	27.1	0.0	0.0	0.0	0.0	0.0	**69.2**	0.0	0.0	4.3	0.0	0.0	30.8
RAAN(+)	0.0	10.4	0.0	**14.8**	70.3	**48.5**	0.0	**61.2**	0.0	0.0	0.7	0.0	0.0	63.8	**10.6**	0.0	**20.5**	0.0	0.0	**34.3**
JAAN	0.0	30.2	0.0	1.4	**92.3**	28.2	0.0	53.5	0.0	0.0	**16.0**	0.0	0.0	17.1	0.0	2.3	14.3	0.0	0.0	33.2
$Est.\beta$	2.0	1.4	1.6	0.9	0.5	1.4	1.3	1.2	1.6	1.6	1.1	1.9	1.7	0.8	1.5	1.1	0.7	1.8	1.9	−−

formulate our UDA problem in μ-DS classification scenario as the following: the source domain is composed of μ-DS from certain aspect angles and target personnels, while the target domain data is composed of μ-DS from other angles and target personnels.

Table 5.5: Experiment details for adapting two factors of variations including aspect angle and target personnel.

Node Index	Node 1 in Figure 4.7
Mono-Static μ-DS (5-second) Number	90
Train Percentage	20%
Dwell Time	1s
Augmented Train mono-static μ-DS Number	918
Augmented Test mono-static μ-DS Number	648
FFT Time Period	0.3s
FFT Overlap Ratio	0.9
Aspect Angles (Degree)	-30, 0, 30
Target Personnels	Target 1,2,3

We follow the experiment setting in Section 4.6 and the summary of the utilized μ-DS dataset is in Table 5.5. In this chapter, we mainly evaluate the effect of the proposed adaptation method using mono-static μ-DS and specifically, we design two evaluations composed of 12 adaptation tasks listed in Table 5.6. In these experiments, our interest factor is the motion while the distraction factors are target personnel and the aspect angle. Evaluation 1 (Exp 1-6) focuses on the variations caused by target personnels while the Evaluation 2 (Exp 7-12) mainly investigates the variations of the aspect angle. In addition, we also design two scenarios for each experiment including the “Factor Control” and the “No Control”. Specifically, “Factor Control” scenario controls one distraction factor when analyzing the variations of the other distraction factor, while "No Control" scenario does not consider this. Taking Exp1-6 as an example, in the "Factor Control" scenario, we use the data from *target 1 only* if we want to eliminate the variations caused by *aspect angles* but in the "No Control" scenario, we use the data from *all targets*.

To evaluate the RAAN and JAAN in μ-DS adaptation tasks, we utilize the SC-DopNet in Section 4.4 as the base network, including T_s and T_t and the classifiers. The discriminator network D and the presenter network P adopt similar architectures, where RAAN uses two fully connection layer with the input size of 128 and activation units of 64 and 1 respectively. JAAN uses two fully connection layer as well, with the input size of 130 (the size of joint label and feature representations) and activation units of 128 and 64 respectively. The classifiers used in both JAAN and RAAN adopt the single FC layer. The de-convolution network in JAAN to reconstruct the raw input data is shown in Table 5.7.

Table 5.6: Experiments to evaluate the factors of variation; the main interest factor is the motion recognition while two distraction factors include aspect angle and target personnel. We conduct these experiments in both "Factor Control" and "No Control" scenario.

Experiment Number	Variation Factor	Source Domain (Training Set)	Target Domain (Testing Set)
Evaluation 1			
Exp1	Target Personnel	Target 1,2	Target 3
Exp2	Target Personnel	Target 1,3	Target 2
Exp3	Target Personnel	Target 2,3	Target 1
Exp4	Target Personnel	Target 1	Target 2,3
Exp5	Target Personnel	Target 2	Target 1,3
Exp6	Target Personnel	Target 3	Target 1,2
Evaluation 2			
Exp7	Aspect Angle	Angle 1,2	Angle 3
Exp8	Aspect Angle	Angle 1,3	Angle 2
Exp9	Aspect Angle	Angle 2,3	Angle 1
Exp10	Aspect Angle	Angle 1	Angle 2,3
Exp11	Aspect Angle	Angle 2	Angle 1,3
Exp12	Aspect Angle	Angle 3	Angle 1,2

Table 5.7: Deconvolution network for JAAN with SC-DopNet.

Layers	Kernel Number	Kernel Size	Stride	Activation
dConv1	512	1x1	1	Relu
dConv2	128	3x5	1	Relu
dConv3	64	2x2	2	Relu
dConv4	32	5x5	2	Relu
dConv5	1	7x7	5	Relu

Table 5.8: A-distance for adaptation tasks Exp1-6. AVG represents the averaged result of the six experiments.

	Exp1	Exp2	Exp3	Exp4	Exp5	Exp6	AVG
A-distance	1.667	1.798	1.698	1.776	1.809	1.420	1.695

5.7.3.1 Adaptation Results between Factor of Target Personnels

Before describing the recognition results, we first report the A-distances of Exp1-6 in Table 5.8 to measure the domain discrepancy and the difficulty of the adaptation task. Next, adaptation results of Exp1-6 in the "Factor Control" scenario are shown in Table 5.9, Figure 5.6 and 5.7.

For base network results, JAAN outperforms RAAN in average by 1.0%. However, as highlighted, for handling variations of target personnels, JAAN outperforms RAAN in average by 3.6%. It may be concluded that JAAN outperforms RAAN when eliminating variations of target personnels. In addition, due to the similar source-only results of base networks, we hypothesize that the main reason of JAAN's improvements is the regularized OT formulation adopted in JAAN, which is more robust and suitable than the dual formulation used in RAAN to handle target personnel discrepancies.

In addition, we plot the A-distances with the base network results of all adaptation tasks of Table 5.9 together in Figure 5.5. The general trend may be summarized that the recognition rates of an experiment decrease with the increasing of its A-distance, and the vice versa with decreased A-distance.

Finally, to evaluate the effects of OT based divergence measurement, we also compare RAAN and JAAN with the widely used MMD and JS divergence described in Section 2.5.2.1. For fair comparisons, we only replace the OT based module with the MMD and JS module and keep the same base networks. It can be observed in Table 5.9, Figure

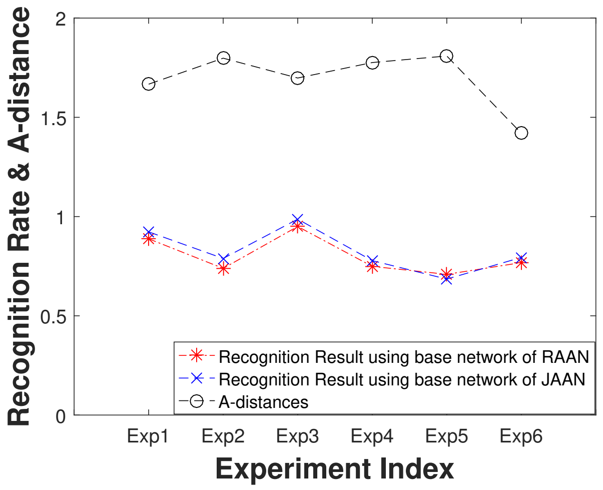


Figure 5.5: A-distance and recognition results in Exp1-6

Table 5.9: Recognition rates (%) in evaluation 1, adapting between the factors of target personnel; AVG-M refers to average results of the 6 experiments using a specific method. Base-Net refers to results using base network of RAAN and JAAN.

	Exp1	Exp2	Exp3	Exp4	Exp5	Exp6	AVG-M
Base-Net (RAAN)	84.0	71.0	95.6	68.3	67.2	71.1	76.3
MMD	91.1	74.4	95.6	75.0	67.8	76.7	80.1
JS	88.9	73.3	91.1	64.4	73.3	81.6	78.8
RAAN	91.1	76.7	97.8	91.7	75.1	77.8	85.0
Base-Net (JAAN)	85.6	72.3	96.9	72.7	63.3	73.2	77.3
MMD	95.6	82.2	98.9	70.6	65.6	81.1	82.3
JS	90.0	76.7	98.9	71.1	69.4	78.9	80.8
JAAN	**97.8**	**84.4**	**100.0**	**96.7**	**75.7**	**84.4**	**88.6**

5.6 and 5.7 that RAAN outperforms MMD and JS divergence measurement by 4.9% and 6.2%, while JAAN outperforms MMD and JS by 6.3% and 7.8% in average.

Adaptation results in the "No Control" scenario are shown in Table 5.10 and the following findings can be observed:

- *First* the base network results in the"No Control" scenario outperforms the "Factor Control" result by 4.4% and 3.7% using RAAN's and JAAN's respectively. This finding is interesting as it seems that increasing the data diversity in the factor of aspect angle helps the network to generalize better to the unseen factor of variations.

- *Second*, the improvements of RAAN and JAAN are 8.7% and 11.3% respectively

Table 5.10: Averaged recognition rates (%) of evaluation 1 in "No Control" scenarios. Base-Net refers to results using base network of RAAN and JAAN.

	Eval 1
RAAN(Base-Net, Factor Control)	76.3
RAAN(Adaptation, Factor Control)	85.0
RAAN(Base-Net, No Control)	80.7
RAAN(Adaptation, No Control)	84.5
JAAN(Base-Net, Factor Control)	77.3
JAAN(Adaptation, Factor Control)	88.6
JAAN(Base-Net, No Control)	81.0
JAAN(Adaptation, No Control)	88.2

in the "Factor Control" scenario, however the result improvements degrade to 3.8% and 7.2% in the "No Control" scenario. This proves that JAAN outperforms RAAN in both scenarios and therefore is robust to different domain discrepancies.

- *Finally*, to handle the more complex and difficult adaptation task in the "No Control" scenario, JAAN outperforms RAAN by 3.5%.

To sum up, in either the "No Control" or "Factor Control" scenarios, JAAN performs better than RAAN.

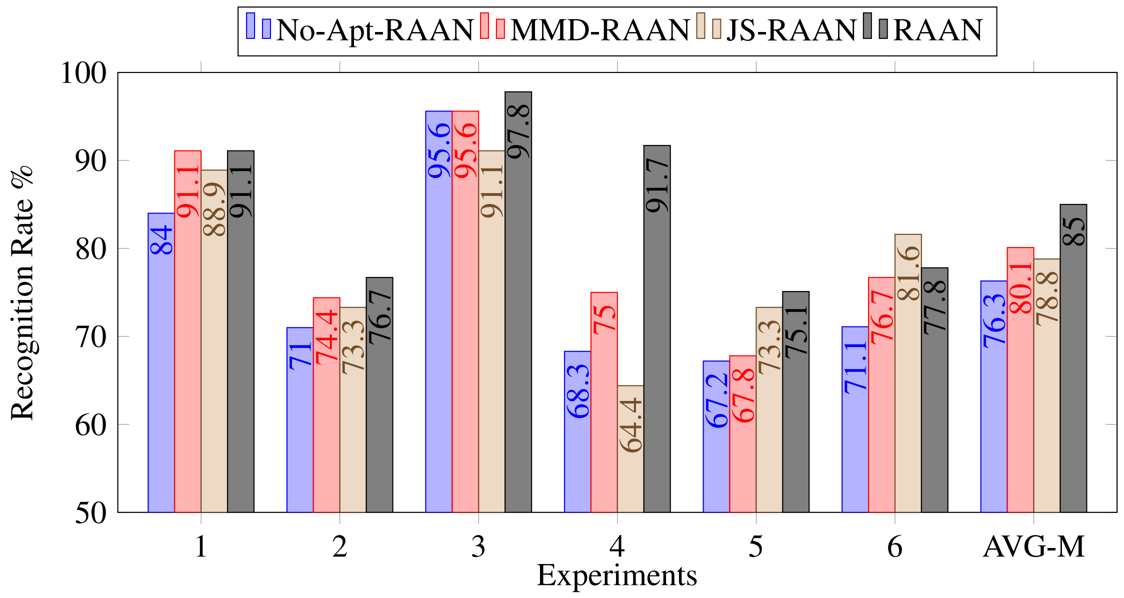


Figure 5.6: Recognition results of Exp1-6 using RAAN's base network;AVG-M refers to average results of the 6 experiments using a specific method.

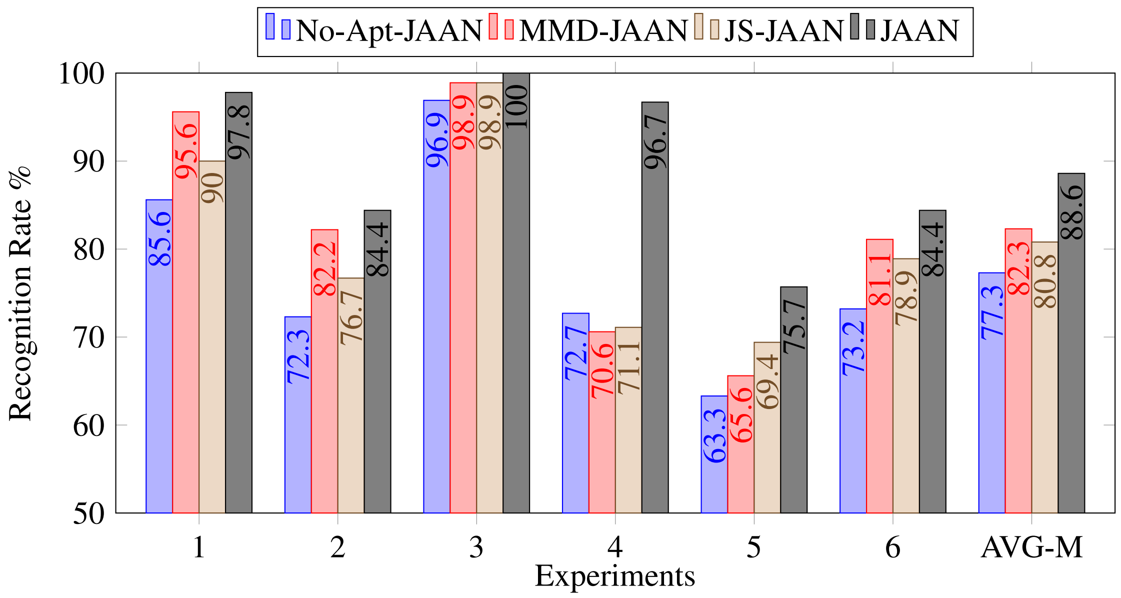


Figure 5.7: Recognition results of Exp1-6 using JAAN's base network; AVG-M refers to average results of the 6 experiments using a specific method.

Table 5.11: A-distance for adaptation tasks Exp7-12.

	Exp7	Exp8	Exp9	Exp10	Exp11	Exp12	AVG
A-distance	1.798	0.869	1.056	1.580	1.510	1.446	1.377

5.7.3.2 Adaptation Results between Factor of Aspect Angle

First, we report the A-distances of Exp7-12 in Table 5.11 and the results of Exp7-12 in "Factor Control" scenario are shown in Table 5.12, Figure 5.9 and 5.10. Compared with the average A-distance of Exp1-6 (1.695) in Table 5.8, the average result of Exp7-12 (1.377) is lower. In addition, compared with the average source-only results 76.8% in Exp1-6, the one caused by aspect angle in Exp7-12 is 88.7%. The higher source-only results and the lower A-distance of Exp7-12 both suggest the smaller domain discrepancies caused by aspect angle than the target personnels. In addition, we plot the averaged recognition results of Exp7-12 and the related A-distances in Figure 5.8. The same trend is found out as the one in Figure 5.5 where increasing the A-distance usually decreases the recognition rates.

In Table 5.12, it seems that the base network of JAAN outperform RAAN in average by 0.2% and JAAN outperforms RAAN by 0.1% in average. In Exp8-9 and Exp11-12, RAAN achieves very similar results with JAAN and we hypothesize that this is because

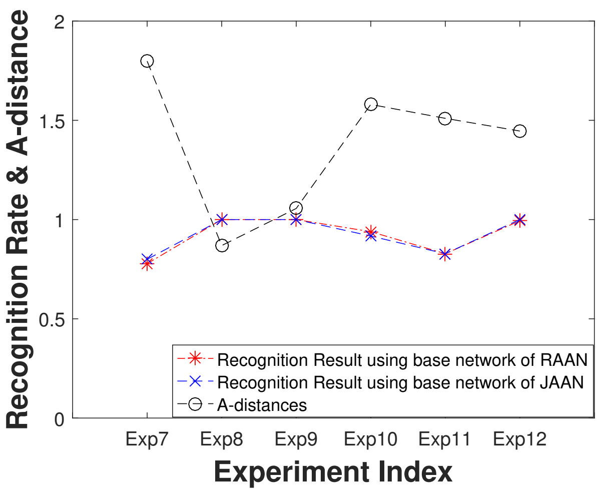


Figure 5.8: A-distance and recognition results in Exp7-12. AVG represents the averaged result of the six experiments.

Table 5.12: Recognition rates (%) in evaluation 2, adapting the factors of aspect angle; AVG-M refers to average results of the 6 experiments using a specific method. Base-Net refers to result of base network of RAAN and JAAN.

Node1 Target1	Exp7	Exp8	Exp9	Exp10	Exp11	Exp12	AVG-M
Base-Net(RAAN)	71.1	100.0	100.0	88.9	78.9	93.9	88.8
MMD	74.4	100.0	100.0	92.2	79.4	99.4	90.9
JS	71.1	100.0	100.0	90.4	81.1	98.3	90.2
RAAN	77.8	100.0	100.0	**93.9**	**82.8**	99.4	92.3
Base-Net (JAAN)	73.3	100.0	100.0	90.6	80.5	86.9	88.6
MMD	77.8	100.0	100.0	92.2	81.1	99.4	91.7
JS	74.4	100.0	100.0	91.1	**82.8**	91.7	90.0
JAAN	**80.0**	**100.0**	**100.0**	91.7	**82.8**	**100.0**	**92.4**

the domain discrepancies of these tasks are small, which is verified by the A-distance in Table 5.11. In Exp7, where the A-distance is the largest in Exp7-12, JAAN outperforms RAAN by 2.2%. In addition, we also compare RAAN and JAAN with other divergence measurements in Table 5.12, where both RAAN and JAAN outperform the others around 1.5% in average. It may be concluded that when domain discrepancy is relatively small, RAAN and JAAN achieve similar results while JAAN outperforms RAAN when domain discrepancies are disparate.

Finally, we report the results of Exp7-12 in "No Control" scenario in Table 5.13 and the following findings can be observed:

- *First* the base results using RAAN's and JAAN's base networks in the "Factor

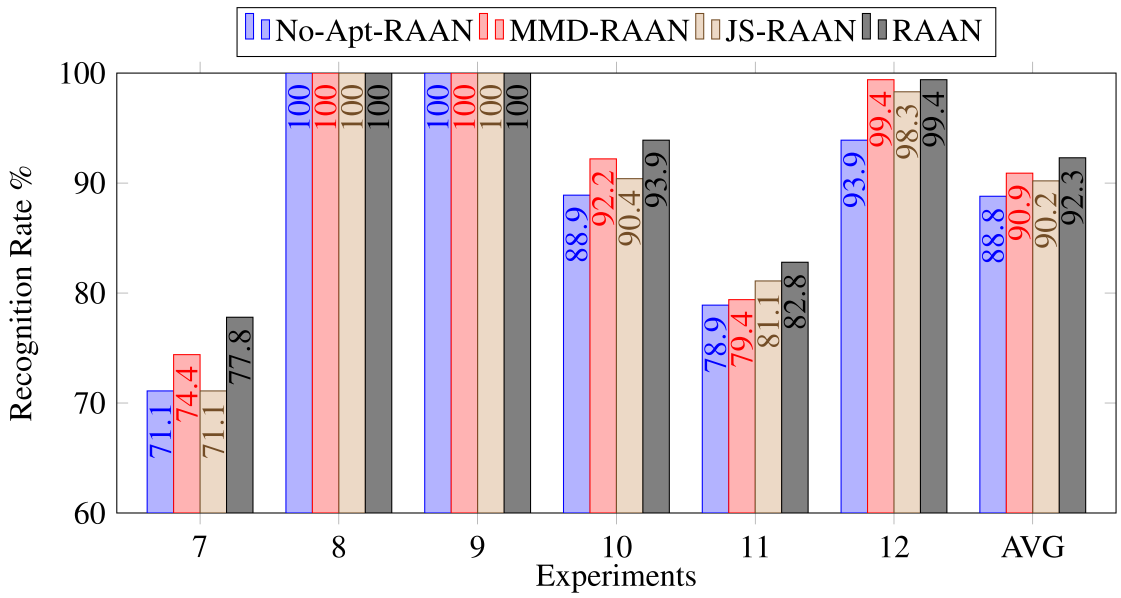


Figure 5.9: Recognition results of Exp7-12 using RAAN's base network.

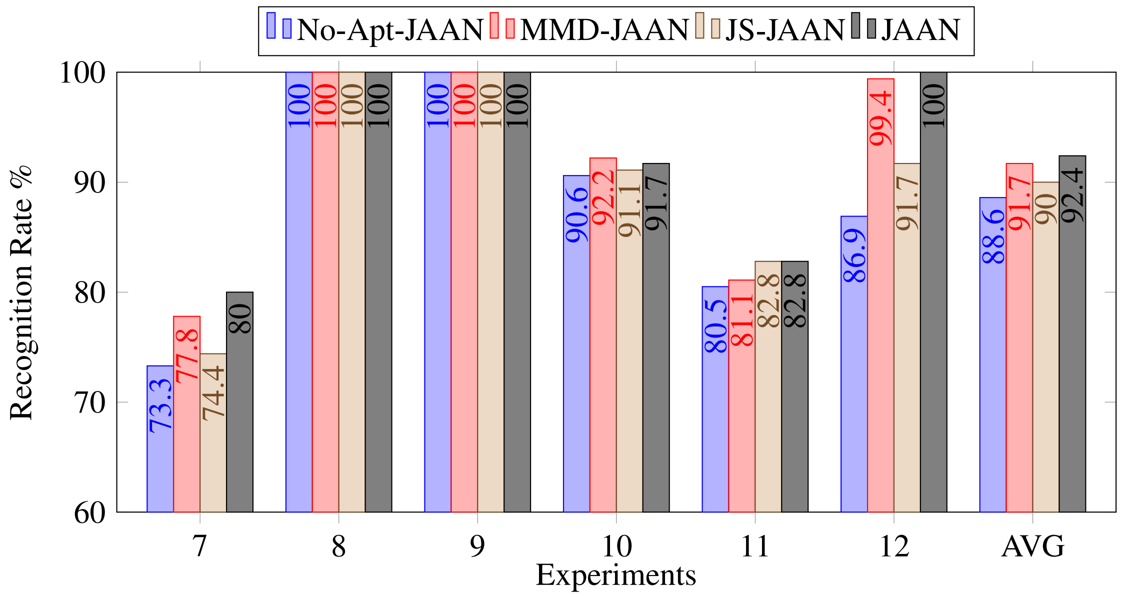


Figure 5.10: Recognition results of Exp7-12 using JAAN's base network.

Control" scenario outperforms the "No Control" scenario by 8.6% and 9.2% respectively. This may be because the aspect angle variation increases the domain discrepancies.

- *Second*, the improvements of RAAN and JAAN are 3.5% and 3.8% respectively in the "Factor Control" scenario and the result improvements increase to 4.5% and 7.4% in the "No Control" scenario. This proves that JAAN outperforms RAAN in

Table 5.13: Average recognition rates (%) of evaluation 2 in “No Control” scenarios. Base-Net refers to results using base network of RAAN and JAAN.

	Eval 2
RAAN(Base-Net, Factor Control)	88.8
RAAN(Adaptation, Factor Control)	92.3
RAAN(Base-Net, No Control)	80.2
RAAN(Adaptation, No Control)	84.7
JAAN(Base-Net, Factor Control)	88.6
JAAN(Adaptation, Factor Control)	92.4
JAAN(Base-Net, No Control)	79.4
JAAN(Adaptation, No Control)	86.8

both scenatios and achieves more robust results.

- *Third*, to handle the more complex and difficult adaptation task in the “No Control” scenario, JAAN outperforms RAAN by an average of 2.1%.

To sum up, in “Factor Control” scenario, JAAN and RAAN achieve similar results when domain discrepancies are small. However, in “No Control” scenario, JAAN outperforms RAAN when adapting aspect angle variations.

5.8 Analysis

In this section, we analyze the results including the re-weighting scheme in adversarial training, the domain distribution divergence in both quantitative and qualitative ways and finally the sensitivity study of parameter selection in Sinkhorn algorithm in JAAN. The evaluation is in the context of the most challenging benchmark (with unbalanced label distribution), which involves adapting from the SVHN to the MNIST.

5.8.1 Evaluate the Re-weighting Scheme in RAAN

In Figure 5.11, we evaluate the re-weighting scheme by comparing the ground truth label ratio vector (red) and the learned one (blue). It can be seen that some ratios are accurate while others are not. However, the relative ratio trend of the learned ratio vector β follows that of the ground truth.

As the label distributions between SVHN and MNIST are largely mismatched, it will confuse the domain discriminator and the feature distributions will be matched in a

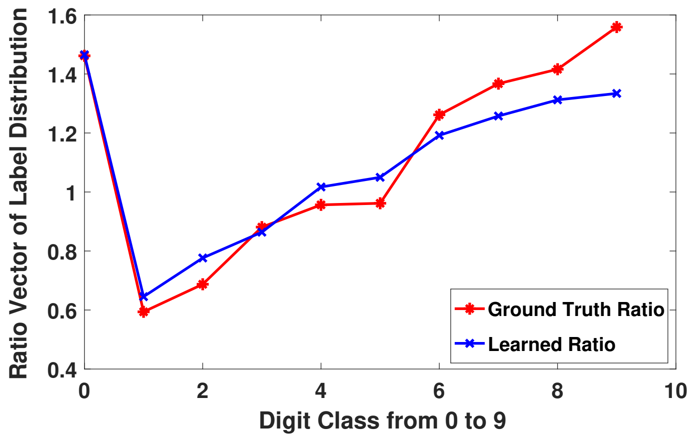


Figure 5.11: Ratio of label distribution between SVHN and MNIST; red line indicates the ground truth ratio, while blue one indicates the estimated ratio.

Table 5.14: $\mathcal{A}$-distance of adversarial training method

Metric	Source Only	ADDA	RAAN(-)	RAAN(+)	JAAN
$\mathcal{A}$-Distance	1.673	1.548	1.526	1.506	1.341

biased manner. In addition, the mismatch of label distribution will directly give rise to the mismatch of classifiers as well. However, as shown in Figure 5.11, RAAN(+) successfully matches the distribution of labels by simply learning the ratio vector embedded in the adversarial training. Therefore, this can be regarded as the main reason for the 9% improvement achieved by RAAN(+) compared to RAAN(-) shown in Table 5.3. To sum up, matching the label distribution can better adapt the classifiers.

To understand the instance re-weighting scheme intuitively, it is implemented by assigning different significance to source domain instances. For example, as shown in Figure 5.11, the learned ratio of digit “0” is around 1.5, which means that in the adversarial training, each sample from digit “0” in SVHN dataset can be regarded as 1.5 samples.

5.8.2 Ablation Study of Modules in JAAN

In this section, we evaluate JAAN’s components by comparing the recognition rates yielded by an ablation study. The components and parameters in an ablation study include: usage of reconstruction network and its loss, usage of adaptive distance function learning, batch size number, entropy regularization weights in Sinkhorn loss and the update frequencies of the feature transformer network, the classifier and the presenter network. As the domain discrepancy between SVHN and MNIST is fairly large, we choose to conduct ablation study based on this adaptation task.

For evaluating the role and contribution of reconstruction network and the adaptive distance function learning, we reported the recognition results obtained in the following: the average recognition rates obtained without the reconstruction network and loss is in average 85.8%, while the one without the adaptive distance function learning is in average 81.7%. Experimentally, we may conclude that the reconstruction loss and learning the adaptive distance function are essential in adapting the joint feature and label distributions. The potential reason for the use of reconstruction network may be: reconstruction of the original images (or fixed features) using shared network between domains provides implicit constraints for complete feature information for reconstruction at least matching the scale level of the representations. In addition, the potential explanation for the use of the adaptive distance function learning may be: with the aim of matching disparate domain distributions, it requires a more powerful distance function to measure the distribution discrepancy between domains.

We next analyze the effects of important parameters in JAAN, as shown in the following Tables. Note that when analyzing effects of a parameter, we fixed the others using the following optimal values: batch size of 64, 0.1 as the weights of regularization and 20 iterations of updating the presenter network per update of the feature transformer networks and classifiers. As shown in Table 5.15, it seems that JAAN's performance is sensitive to the batch size when it is smaller than 64 (in a 10-class classification task). We can explain this from the mechanism of the Sinkhorn algorithm, where the optimal transportation map requires calculation based on finding the most similar two samples from each domain. In this way, the larger the batch size, the more samples we can select in each operation. When the batch size is larger than 32, it seems that JAAN stably performs in the adaptation task. Based on Table 5.16, it seems that JAAN is not that sensitive to the weights of entropy regularization and similar conclusions can be drawn based on observation of Table 5.17.

Table 5.15: Recognition rates (%) using different batch sizes

Batch Size	16	32	64	128	256
RecognitionRates	85.2	89.6	91.8	90.2	90.5

Table 5.16: Recognition rates (%) using different weights of entropy regularization.

weights of Entropy Regularization	0.01	0.05	0.1	0.5	1	2
RecognitionRates	89.6	91.8	90.5	90.2	91.8	90.5

5.8.3 Evaluate Distribution Divergence of Feature Embeddings

To analyze JAAN's performance on reducing the distribution discrepancies in a quantitative way, we calculate the A distance suggested by the UDA community [183, 184]. The extracted features under comparison are: source-only features, features extracted by ADDA and the one extracted by RAAN(-) and RAAN(+), which is based on the dual formulation of JAAN.

As shown in the Table 5.14, the A distances of feature embeddings with no adaptation, adapted by ADDA, RAAN and JAAN progressively decrease. This implies that JAAN outperforms others in reducing the distribution divergence. In the experimental set up, as JAAN utilized the same feature transformer network as the others, compared with ADDA, we may conclude that better performance of JAAN is mainly due to the OT based distribution divergence compared with the geometry-oblivious JS divergence. Additionally, the improvement of JAAN over RAAN shows the superior performance of adaptive function learning in RAAN, rather than the fixed ℓ_1-norm distance function used in RAAN.

Finally, to measure the feature distribution divergence in a qualitative way, we utilized the T-SNE software package [190] to visualize the 2-D embedding of the extracted features. It can be seen in Figure 5.12 that the example points from the same class adapted by JAAN in Figure 5.12(d) are clustered closer than those by other methods. In addition, clusters from different classes adapted by JAAN are separated further apart compared with other methods. The findings are consistent with the A-distance measurements in Table 5.14.

Table 5.17: Recognition rates (%) using different iterations of updating presenter network per update of the feature transformer and classifier.

Iterations	5	10	20	30	40	50
RecognitionRates	89.6	91.2	91.8	90.2	91.8	90.5

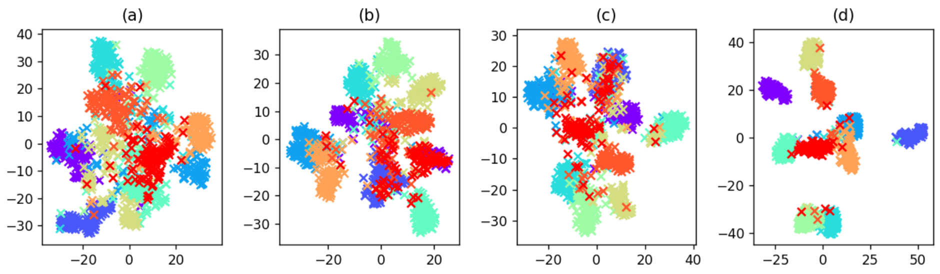


Figure 5.12: T-SNE plot of features when adapting from SVHN to MNIST; (a) No adaptation (b) Adaptation after ADDA (c) Adaptation after RAAN (d) Adaptation after JAAN. We randomly select 1000 features samples from 10 classes, with 100 samples per class.

.

5.9 Summary

μ-DS classification methods in real-world applications are limited by distribution discrepancies or variations between the training data and the unseen test data. We observe two main factors of variation for μ-DS, including target personnel and the aspect angle. We argue that in most real-world applications, collecting and labelling data from all the target personnels and aspect angles are expensive, time-consuming and inefficient. In this chapter, with the aim of increasing the generalization capability for the μ-DS classification, we introduce the UDA task to eliminate the variations between training (source domain) and test (target domain) μ-DS.

To the best of the authors' knowledge, no UDA methods have been applied for the μ-DS classification, which proves the novelty of this chapter. Comparing with the computer vision research, previous UDA methods are mainly based on the divergence measurement relying on the common support between the feature distributions, but in real-world applications, it is in general not feasible due to the uncontrollable data collection scenarios. To tackle this problem, the proposed RAAN and JAAN are inspired by OT based divergence measurement which is oblivious to the uncommon support. More specifically, the work carried out in this chapter makes specific three key contributions to the knowledge base:

1. We first propose two factors of variations in μ-DS classification, including the target personnel and the aspect angles and apply the UDA methods to eliminate these variations.

2. The RAAN and JAAN are the first adaptation networks to integrate the OT formulations in an end-to-end training manner. We implement both the dual and primal regularized loss functions in the context of adversarial feature learning.

3. We evaluate the RAAN and JAAN in computer vision UDA datasets and apply them in the μ-DS classification.

Through an extensive set of experiments using various UDA datasets, RAAN and JAAN outperform state-of-the-art methods based on MMD and JS divergence measurements in computer vision community, especially when the domain distribution divergence is large. Specifically, RAAN and JAAN outperform the current best-perfoming method by 4.8% and 6.8% in hand-written digit datasets respectively. For the most challenging cross-modality dataset with disparate domain discrepancy, RAAN and JAAN outperform the current best-perfoming method by 15.4% and 14.3% respectively. In addition, JAAN outperforms RAAN in the hand-written digit dataset by 2.47% in average however, RAAN outperforms JAAN by 1.1% in cross-modality dataset. In general, JAAN outperforms RAAN in almost all settings which may be due to the fact that primal OT formulation performs more robust than the dual formulation. However, when the data number from different classes is highly unbalanced, RAAN outperforms JAAN. We hypothesize that it is more effective to match label distribution by integrating the re-weighting scheme directly in the network loss function design (as RAAN did) rather than using a joint representation of label and features (as JAAN did).

In the context of the μ-DS classification, to the best of the author's knowledge, no research has focused on eliminating the factors of variations (distraction factor) to improve the motion recognition (interested factor). We therefore summarize our conclusions as follows:

- *First*, we propose to measure the A-distance of different factors of variation in μ-DS so that the relevant domain discrepancy can be identified. We find that *variations caused by target personnel are larger than the aspect angle*.

- *Second*, in the "Factor Control" scenario, JAAN outperforms RAAN by 3.6% in average when addressing the target personnel variations, while they achieve similar results when handling aspect angles. Compared with other widely used MMD and JS divergence measurements, RAAN and JAAN outperforms the current best-performing result by 4.9% and 6.3% for target personnel, and by 1.4% and 2.1% for aspect angle respectively.
- *Finally*, in "No Control" scenarios, JAAN outperforms RAAN by 2.1% and 3.5% respectively for adapting the target personnel and aspect angle variations.

To sum up, RAAN and JAAN have been proved to be very beneficial in reducing the distribution divergence and increasing the generalization capability of the DCNN in both computer vision and radar community.

Chapter 6

ENet: Ensembling Cooperative and Adversarial Learning for μ-DS Classification

6.1 Introduction

In Chapter 5, the UDA problem was introduced to the μ-DS classification tasks and included a discussion of the two main factors of variations that cause distribution discrepancy between training and testing μ-DS dataset. To reduce distribution discrepancy between training and test dataset in an unsupervised way, two adaptation networks RAAN and JAAN were proposed. In fact, the two adaptation networks are inspired by the adversarial learning (AL) strategy in Section 2.5.2.1, 2.5.2 and 2.5.2.3 in the sense that we want to learn features that are only useful for the motion recognition task, but insensitive and not useful for recognizing the target and aspect angles of the μ-DS. This chapter investigates another learning strategy called cooperative learning (CL) which is exactly different from the AL strategy. CL learns features that are useful for both the main task, e.g. motion recognition and the auxiliary one e.g. target or aspect angle recognition. The work is driven by the idea that the features useful for a task may exhibit underlying properties which will also be useful for alternative recognition tasks. Note that as CL strategy is very similar to the idea of IL, introduced in Section 2.5.1 and the Ensemble-Net (ENet) is very similar to the DEN introduced in Section 2.5.1.2, 2.5.1.3, we refer to these two sections for more background details. Since there are two main factors of variations (human target and aspect angle) or alternatively two auxiliary tasks besides the main motion recognition, we investigate in this chapter the relationship between the motion recognition and the following two auxiliary ones respectively: recognize the human target and the aspect angle. Note that the AL strategy used here is different from the two proposed adaptation networks in Chapter 5. This chapter assumes that the labels of three tasks in the training stage are available but the adaptation networks in Chapter 5 deals with a more challenging scenario where no labels were available in the two auxiliary tasks. Specifically, there are three main contributions of this chapter:

1. We propose the objective function for cooperative learning strategy and make comparisons of the one used for adversarial learning strategy.

2. We investigate whether the two additional tasks are cooperative or adversarial to the main motion recognition tasks.

3. We propose an ensembling architecture to handle the three tasks and evaluate the algorithm using recognition performance of the main motion recognition task.

6.2 Related Works

To the best of the author's knowledge, we have not found any μ-DS classification works about improving the motion recognition results leveraging the relationships of classifying multiple auxiliary factors, for example target personnels and aspect angles. Therefore, we have to compare with the works in multi-task learning framework in computer vision and machine learning communities.

Multi-task learning framework is widely utilized in computer vision and machine learning communities, for example, extracting useful features that are useful for both classification and the image reconstruction purposes[72], for classification and detection [73, 191] and even for the most recently proposed detecting and tracking video objects [82]. These methods ensemble different types of tasks however, ENet is only focused on classification tasks but on classifying multiple factors of the same input μ-DS. The most similar series of works in the multi-label learning framework [192, 193], but these two works can be regarded as using the CL strategy. More specifically, they have not incorporate adversarial learning strategies in feature learning, not to mention they investigate and evaluate the strategy to adopt different combinations of CL and AL for different factors. However, ENet considers the way to ensemble both CL and AL for two auxiliary tasks with the aim of improving the classification results of the main task. These prove the novel contributions made by ENet.

6.3 Cooperative Learning and Adversarial Learning

This section first formulates our problem adopting cooperative and adversarial learning strategy. Then we aim to introduce networks and objective functions adopting cooperative and adversarial learning strategies.

6.3.1 Problem Formulation and ENet Architecture

In this section, we first introduce the notation in this chapter and formulate our problem. Suppose we are given n_{Task} tasks and within j^{th} task we are dealing with an n^{j}_{cls}-class recognition problem. The training set $\mathbf{D_{Tr}} = \{(x_i^{Tr}, y_{i,j}^{Tr})\}_{i=1}^{n_{Tr}}{}_{j=1}^{n_{Task}}$ including n_{Tr} data x_i^{Tr} labeled by $\{y_{i,j}^{Tr}\}_{j=1}^{n_{Task}}$ and an unlabelled n^{j}_{cls}-class testing dataset $\mathbf{D_{Te}} = \{(x_j^{Te})\}_{j=1}^{n_{Te}}$ composed of n_{Te} data x_j^{Te} and share the same multi-label space. The random variables representing the data and label in general are denoted as X and Y and the DCNN transformations are represented as T. Under the setting of µ-DS classification of armed or unarmed walking motions ($n_{Task} = 3$, including motion, target and aspect angle recognition), where the motion recognition is a binary classification task and both the target and aspect angle recognition involve distinguishing three classes.

To investigate the applicability of CL and AL strategies in µ-DS recognition, we designed the ENet shown in Figure 6.1, composed of the SC-DopNet as the base network and two sub-networks for the two auxiliary tasks respectively. First the useful and generic low-level convolution layers are shared and represented as $T_{Conv}{=}T_{Conv1} \circ T_{Conv2}$. Second, for simplicity, we represent the multiple training labels as $y_{i,H}^{Tr}, y_{i,M}^{Tr}, y_{i,A}^{Tr}$ and the cascaded FC layers as $T_{FC}^{H} = T_{FC1}^{H} \circ T_{FC2}^{H} \circ CLS^{H}$, $T_{FC}^{M} = T_{FC1}^{M} \circ T_{FC2}^{M} \circ CLS^{M}$ and $T_{FC}^{A} = T_{FC1}^{A} \circ T_{FC2}^{A} \circ CLS^{A}$ for human, motion and angle recognition task respectively. The two sub-networks T_{FC}^{H}, T_{FC}^{A} with their objective functions L_{CE}^{H}, L_{CE}^{A} are designed

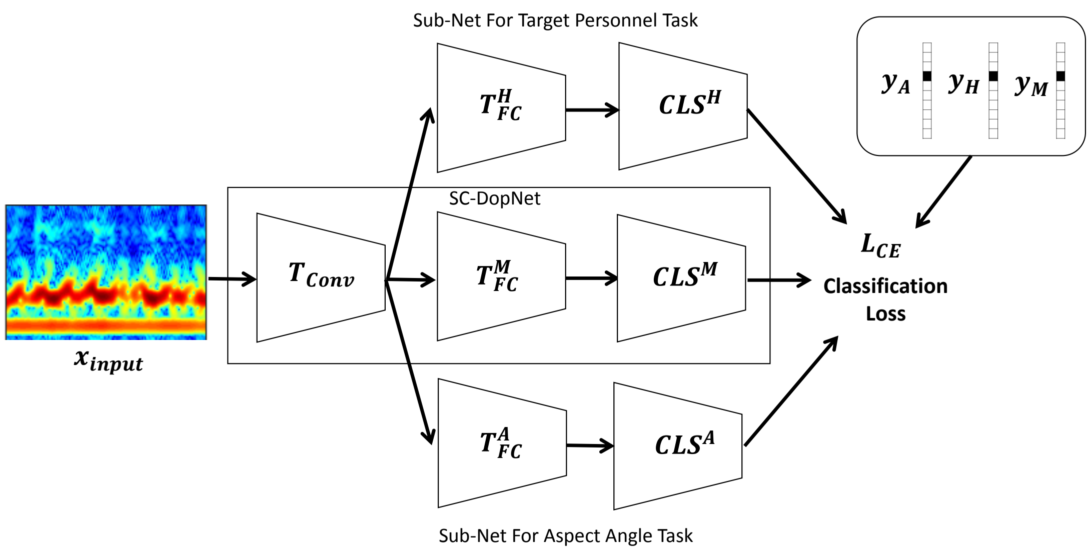


Figure 6.1: Architecture of ENet; M, H and A refer to motion, human target personnel and aspect angle respectively. Conv and FC and CLS refer to convolutional, fully connected and the classifier respectively.

with the same aim of achieving better performance in the main task. To sum up, the following sections investigate the effect of adopting CL or AL strategy for each auxiliary task, by evaluating recognition rates of the main motion recognition task.

6.3.2 Cooperative Learning and Adversarial Learning

First, we define the objective function for the main motion recognition task. With the input μ-DS x_i^{Tr}, the output features of the convolution layers are obtained as $T_{Conv}(x_i^{Tr})$, which are the low-level features useful for all the three final recognition tasks. For motion recognition, the network T_{Conv} and T_{FC}^{M} are trained based on minimizing the conventional CE loss L_{CE}^{M} given the motion label $y_{i,M}^{Tr}$ in Eq.(6.1), where CE is the CE function.

$$\min_{T_{FC}^{M},T_{Conv}} L_{CE}^{M} = \min_{T_{FC}^{M},T_{Conv}} \sum_{i=1}^{n^{Tr}} CE(\, T_{FC}^{M}(T_{Conv}(x_i^{Tr})),\, y_{i,M}^{Tr}) \tag{6.1}$$

With the aim of learning useful and discriminative features for both main and auxiliary tasks, the CL strategy is utilized and the following objective function L_{CE}^{H} for target personnel recognition is designed in Eq.(6.2), so as L_{CE}^{A} for aspect angle recognition in Eq.(6.3). The CL strategy can be interpreted as the following two perspectives:

1. From a specific auxiliary task, the feature extractor T_{Conv} and the classifiers T_{FC}^{H} or

T_{FC}^{A} are trained based on minimizing the same and cooperative objective functions.

2. From the relationship between main and auxiliary tasks, feature extractor T_{Conv} is updated to extract discriminate features which contribute to both the main and auxiliary tasks. Alternatively, the main and auxiliary tasks are cooperative and the main task is playing a cooperative game with the auxiliary tasks. The cooperative game means that the extracted features contribute to multiple recognition tasks.

$$\min_{T_{FC}^{H}, T_{Conv}} L_{CE}^{H} = \min_{T_{FC}^{H}, T_{Conv}} \sum_{i=1}^{n^{Tr}} CE(\, T_{FC}^{H}(T_{Conv}(x_i^{Tr})),\, y_{i,H}^{Tr}) \tag{6.2}$$

$$\min_{T_{FC}^{H}, T_{Conv}} L_{CE}^{A} = \min_{T_{FC}^{A}, T_{Conv}} \sum_{i=1}^{n^{Tr}} CE(\, T_{FC}^{A}(T_{Conv}(x_i^{Tr})),\, y_{i,A}^{Tr}) \tag{6.3}$$

Aiming at learning useful and discriminative features only for the main task, we may want the features $T_{Conv}(x_i^{Tr})$ to confuse auxiliary classifiers, which is the approach behind the AL strategy. If AL is utilized, the following mini-max objective function for target personnel recognition can be designed in Eq.(6.4), so as $L_{CE}^{A,Adv}$ for aspect angle recognition in Eq.(6.5). The AL strategy can be interpreted as the following, similar to the one described in previous paragraph:

1. From a specific auxiliary task, the feature extractor T_{Conv} and the classifiers T_{FC}^{H} or T_{FC}^{A} are trained based on minimizing the different and adversarial objective functions. Specifically, the auxiliary task classifiers T_{FC}^{H} and T_{FC}^{A} are designed to classify human and aspect angles but the feature extractor network T_{Conv} aims to confuse these classifiers.

2. From the perspective of task relationships, feature extractor T_{Conv} is updated to extract discriminate feature which contributes only the main task but degrades the auxiliary tasks. In other words, the main task is playing an adversarial game with the auxiliary tasks. The basic knowledge of adversarial learning or adversarial game are introduced in Section 2.5.2.1 from the perspective of divergence measurements.

$$\max_{T_{Conv}} \min_{T_{FC}^{H}} L_{CE}^{H} = \max_{T_{Conv}} \min_{T_{FC}^{H}} \sum_{i=1}^{n^{Tr}} CE(\, T_{FC}^{H}(T_{Conv}(x_i^{Tr})),\, y_{i,H}^{Tr}) \tag{6.4}$$

$$\max_{T_{Conv}} \min_{T_{FC}^A} L_{CE}^A = \max_{T_{Conv}} \min_{T_{FC}^A} \sum_{i=1}^{n^{Tr}} CE(\ T_{FC}^A(T_{Conv}(x_i^{Tr})),\ y_{i,A}^{Tr}) \tag{6.5}$$

6.3.3 ENet: Ensemble Cooperative and Adversarial Learning

To recognize armed and unarmed walking, as there are two auxiliary tasks, the exact strategy for the whole network training requires investigation. Since for each task, there are potentially two strategies or objective functions under selection to ensemble multiple loss functions, we propose the linear combination method to ensemble the task-specific loss functions in Eq.(6.7), where ω_A and ω_H are the linear combination weights of the loss functions, I_A and I_H are indicator functions, depending on choice of learning strategies.

$$\min_{T_{FC}^M, T_{FC}^A, T_{FC}^H} L_{CE}^M + \omega_A \times L_{CE}^A + \omega_H \times L_{CE}^H \tag{6.6}$$

$$\begin{gathered} \min_{T_{Conv}} L_{CE}^M + \omega_A \times I_A \times L_{CE}^A + \omega_H \times I_H \times L_{CE}^H \\ \mathrm{I}_A = \begin{cases} 1, CL\ is\ used \\ -1, AL\ is\ used \end{cases} \quad \mathrm{I}_H = \begin{cases} 1, CL\ is\ used \\ -1, AL\ is\ used \end{cases} \end{gathered} \tag{6.7}$$

6.4 Experiments and Result Analysis

In this section, we evaluate different learning strategies for ensembling three recognition tasks on the basis of recognition rates of the main task. First, we evaluate ensembling results of two tasks and investigate exact relationships between the main one and each of the auxiliary ones. Next, we evaluate the best choice of learning strategies for ensembling three tasks.

The dataset used is following the same experimental configurations and the pre-processing methods in Chapter 4 (see Section 4.6 for details), where the experimental µ-DS are utilized to recognize armed and unarmed target personnel from three aspect angles, three target personnels and three nodes. For simplicity and controlling the factor of radar nodes, we only evaluate the mono-static µ-DS in ENet, with the details in Table 6.1. Note that we use the augmentation scheme proposed in Section 4.3.2 and obtain 918 augmented mono-static µ-DS for training and 648 for testing when evaluating one radar node. Finally we conduct three separate experiments for evaluating ENet performances of the three nodes. The μ-DS in the experiments are shown in Figure 4.8 and 4.9.

Table 6.1: Experiment summary details for evaluating different task relationships and the optimal learning strategies.

Mono-Static μ-DS (5-second) No.	90
Train Percentage	20%
Dwell Time	1s
Augmented Train mono-static μ-DS No.	918
Augmented Test mono-static μ-DS No.	648
FFT Time Period	0.3s
FFT Overlap Ratio	0.9
Aspect Angles (Degree)	-30, 0, 30
target personnels	Target 1,2,3

To clarify the experiments, we summarize the following 9 experiments in Table 6.2, where the Exp1-6 evaluate relationships between two tasks for all three nodes while the Exp7-9 evaluate relationships among the three tasks for all the three nodes. Additionally, we summarize different learning strategies or task relationships in Table 6.3 using the selected weight ω_A and ω_H. More specifically in Table 6.3, as there is only single auxiliary task under evaluation in Exp1-6, the learning strategy under selection is either cooperative or adversarial; however for ensembling two auxiliary tasks in Exp7-9, there are four combination strategies under selection. Finally, for the weights ω_A and ω_H under selection in the optimization, we choose them in the same parameter set $\{0.001, 0.01, 0.1, 1, 5\}$.

To evaluate the performance of different strategies, we use the average and the best-performing recognition rates when selecting ω_A and ω_H in the parameter set. For each parameter, we run 10 times to calculate the average recognition rate. We run 5000 iterations using the Adam optimizer in Tensorflow and report the average recognition rates of the final 100 iterations.

6.4.1 Results of Ensembling Two Tasks (Exp1-6)

In this section, we evaluate the main motion recognition results by ensembling each of the two auxiliary tasks , namely the target personnel and aspect angle recognition. The SC-DopNet is adopted for T_{FC}^M, T_{Conv} and the FC layer for the two auxiliary sub-networks T_{FC}^A, T_{FC}^H. We investigate different weights ω_A and ω_H using both CL and AL strategies

Table 6.2: Experiment summary for evaluating different task relationships and the optimal learning strategies.

	Aspect Angle	Target Personnel	Motion	Node
Exp1	✓		✓	1
Exp2		✓	✓	1
Exp3	✓		✓	2
Exp4		✓	✓	2
Exp5	✓		✓	3
Exp6		✓	✓	3
Exp7	✓	✓	✓	1
Exp8	✓	✓	✓	2
Exp9	✓	✓	✓	3

Table 6.3: Experiment summary of the learning strategies under evaluation and the parameter set under selection.

	Relationship under evaluation	Weight in the optimization
Exp1-6	(Cooperative, Adversarial)	{0.001,0.01,0.1,1,5}
Exp7-9	(Angle Cooperative, Target Cooperative)	{0.001,0.01,0.1,1,5}
	(Angle Cooperative, Target Adversarial)	{0.001,0.01,0.1,1,5}
	(Angle Adversarial, Target Cooperative)	{0.001,0.01,0.1,1,5}
	(Angle Adversarial, Target Adversarial)	{0.001,0.01,0.1,1,5}

while fix the main task weight as 1. Note that we use the best recognition rate in each experiment as the main evaluation metric. Based on the extensive results presented in Table 6.4, 6.5 and 6.6, we summarize our observed relationships between main motion recognition and the aspect angle or target personnel recognition. In addition, to clearly show the trend of results, we plot the recognition rates of Exp1-6 with the parameter set in Figure 6.2.

Results of ensembling the motion and aspect angle recognition are shown in the second column of Table 6.4, 6.5 and 6.6 for node 1, 2 and 3 respectively. In addition, we report the baseline results of SC-DopNet without the ensembling from Chapter 4 and summarize the results in Figure 6.3 based on Table 6.4, 6.5 and 6.6

From the best-performing result in Figure 6.3, *firstly*, ensembling single auxiliary task in ENet improves the baseline result of SC-DopNet. In specific, adopting the best ensembling strategy outperforms the baseline by 2.2%, 19.8% and 2.3% for the three nodes respectively, which implies that the ENet architecture is the most useful for node

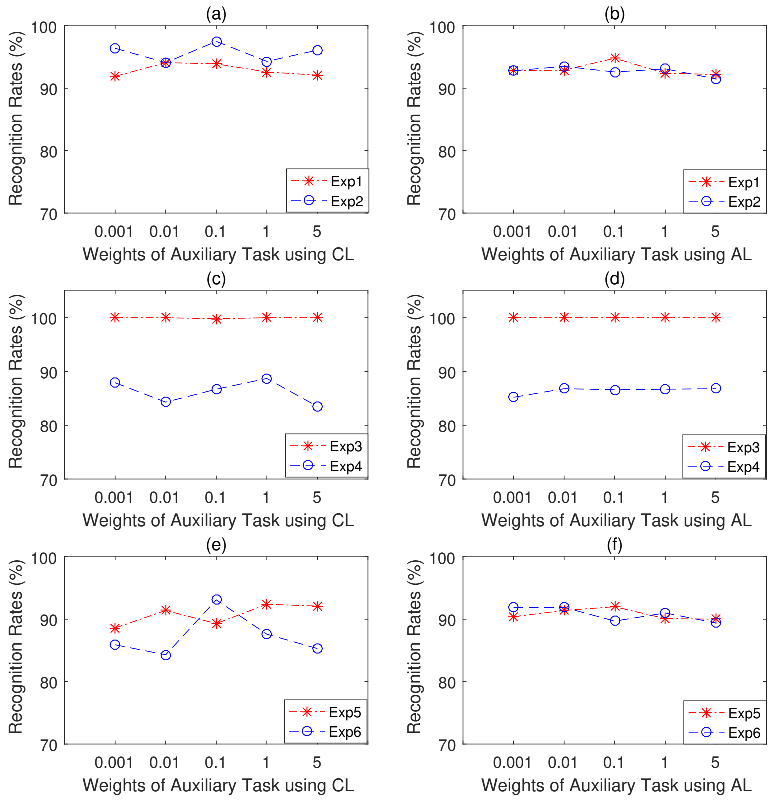


Figure 6.2: Results of CL and AL strategies in Exp1-6, summarizing the results of Table 6.4, 6.5 and 6.6; motion recognition rates (a) in Exp1 and Exp2 using CL; (b) in Exp1 and Exp2 using AL; (c) in Exp3 and Exp4 using CL; (d) in Exp3 and Exp4 using AL; (e) in Exp5 and Exp6 using CL; (f) in Exp5 and Exp6 using AL.

2. The reason may be that the aspect angle for node 2 is the largest and the μ-DS is the least discriminative, as clearly observed in Figure 4.8 and 4.9. Compared with node 2, improvements of recognition results are smaller and less obvious when the aspect angles are relatively smaller for node 1 and 3. *Secondly*, to investigate the relationships between the motion and aspect angle recognition, AL based method outperforms CL for node 1 (see Exp1 in Table 6.4), however, CL based method achieves better result for node 3 (see Exp5 in Table 6.6). Note that both CL and AL based methods for node 2 achieve 100% (see Exp3 in Table 6.5), outperforming the SC-DopNet baseline (without ensemble) by 19.8%. Therefore we conclude in Table 6.7 that ***the relationship between aspect angle and motion recognition largely depends on different geometries and there is no unified relationship for all the three nodes***.

In Figure 6.3, ensembling the motion and target personnel recognition outperforms

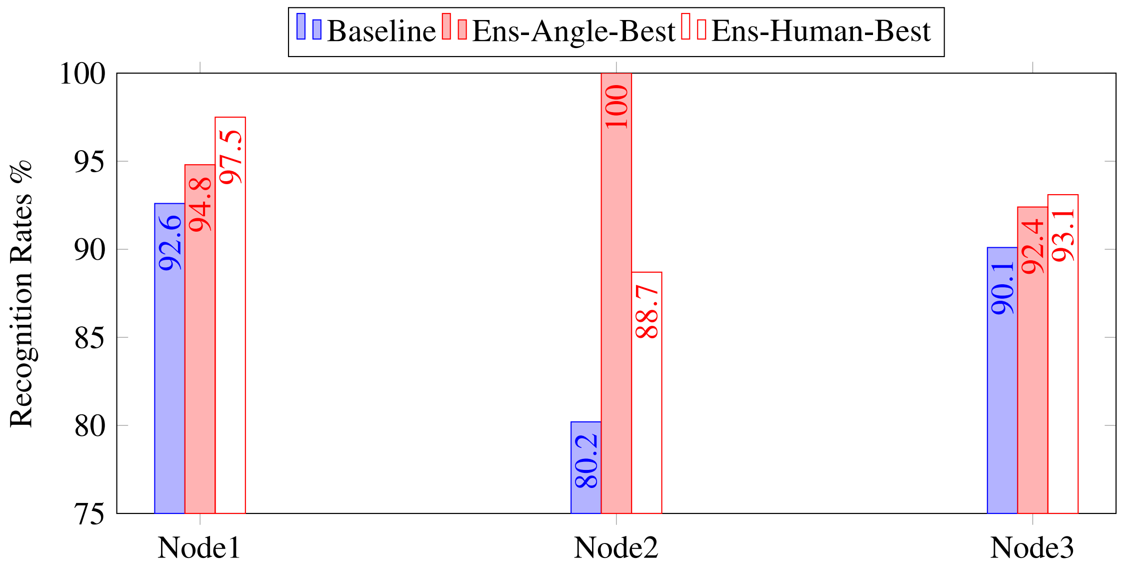


Figure 6.3: Result comparisons of Exp1-6 with the baseline; Ens-Angle-Best: ensemble angle recognition and show best result; Ens-Human-Best: ensemble target personnel and show best results.

Table 6.4: Evaluated Task Relationships using Exp 1 and 2. Recognition rates $A(B)$ indicate: $A\%$ for motion recognition rate and $B\%$ for auxiliary task recognition rate.

Tasks	Motion (Aspect Angle) Exp1	Motion (Target Personnel) Exp2
Cooperative (5)	92.1 (87.9)	96.1 (88.5)
Cooperative (1)	92.6 (84.1)	94.3 (50.0)
Cooperative (0.1)	93.9 (82.9)	**97.5 (66.9)**
Cooperative (0.01)	**94.1** (80.8)	94.1 (63.1)
Cooperative (0.001)	91.9 (80.8)	96.4 (77.4)
Variations	2.042	2.900
Adversarial (5)	92.2 (41.7)	91.5 (41.1)
Adversarial (1)	92.4 (24.4)	93.1 (44.5)
Adversarial (0.1)	**94.8 (19.2)**	92.6 (55.3)
Adversarial (0.01)	92.9 (53.7)	**93.5** (82.3)
Adversarial (0.001)	92.8 (79.5)	92.8 (86.4)
Variations	2.071	1.503
92.6% [see Chapter 4, Table 4.5]		

the baseline SC-DopNet by 4.9%, 8.5% and 3.0% respectively for the three nodes. As shown in the third column of Table 6.4, 6.5 and 6.6, the best-performing CL based methods outperform AL by 4.0%, 1.9% and 1.2% for three nodes respectively. These findings suggest that ***the motion and target personnel recognition are better modelled***

Table 6.5: Evaluated task relationships using Exp3 and 4, recognition rates $A(B)$ indicate: $A\%$ for motion recognition rate and $B\%$ for auxiliary task recognition rate.

Tasks	Motion (Aspect Angle) Exp3	Motion (Target Personnel) Exp4
CooperativeW5	100.0 (88.9)	83.5 (44.9)
CooperativeW1	100.0 (89.4)	**88.7 (43.5)**
CooperativeW0.1	100.0 (87.9)	86.7(61.7)
CooperativeW0.01	99.8 (84.4)	84.3 (68.7)
CooperativeW0.001	100.0 (85.4)	87.9 (75.8)
Variations	0.179	4.504
AdversarialW5	100.0 (13.3)	**86.8** (41.2)
AdversarialW1	100.0 (10.4)	86.7 (43.1)
AdversarialW0.1	100.0 (9.6)	86.6 (58.6)
AdversarialW0.01	100.0 (79.4)	**86.8** (73.3)
AdversarialW0.001	100.0 (80.4)	85.2(77.5)
Variations	0	1.374
80.2% [see Chapter 4, Table 4.5]		

Table 6.6: Evaluated task relationships using Exp5 and 6, recognition rates $A(B)$ indicate: $A\%$ for motion recognition rate and $B\%$ for auxiliary task recognition rate.

Tasks	Motion (Aspect Angle) Exp5	Motion (Target Personnel) Exp6
CooperativeW5	92.1 (98.2)	85.3 (88.7)
CooperativeW1	**92.4 (98.8)**	87.6 (35.0)
CooperativeW0.1	89.3 (95.7)	**93.1 (56.4)**
CooperativeW0.01	91.4 (96.9)	84.3 (61.9)
CooperativeW0.001	88.6 (96.4)	85.9 (80.2)
Variations	3.419	6.977
AdversarialW5	90.0 (29.6)	89.4 (44.1)
AdversarialW1	90.1 (18.9)	91.0 (43.2)
AdversarialW0.1	92.0 (28.2)	89.7 (52.9)
AdversarialW0.01	91.4 (81.2)	**91.9** (66.3)
AdversarialW0.001	90.4 (93.2)	**91.9** (77.5)
Variations	1.757	2.372
90.1% [see Chapter 4, Table 4.5]		

as CL tasks.

6.4.2 Results of Ensembling Three Tasks (Exp7-9)

In this section, we evaluate the motion recognition results by ensembling two auxiliary tasks, namely target personnel and the aspect angle recognition. Since two strategies

Table 6.7: Summarized optimal relationships when ensembling single auxiliary task.

Tasks	Relationship
Motion and Aspect Angle Recognition, Node 1	Adversarial
Motion and target personnel recognition, Node 1	Cooperative
Motion and Aspect Angle Recognition, Node 2	Unclear
Motion and target personnel recognition, Node 2	Cooperative
Motion and Aspect Angle Recognition, Node 3	Cooperative
Motion and target personnel recognition, Node 3	Cooperative

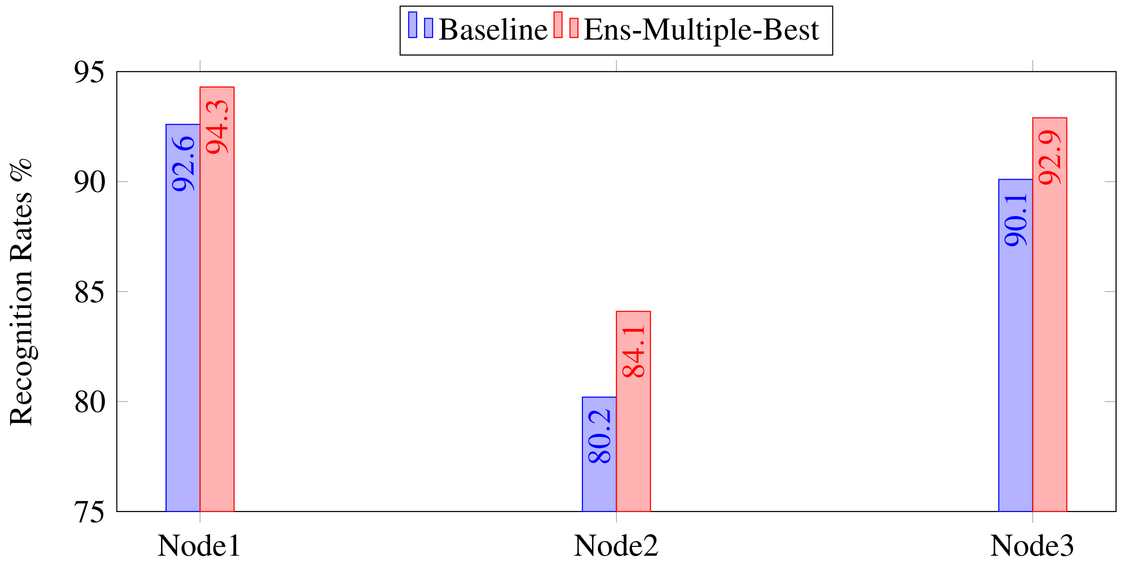


Figure 6.4: Result comparison of Exp7-9 with the baseline; Ens-Multiple-Avg: average result when ensembling both angle and target personnel recognition; Ens-Angle-Best: best result when ensembling both angle and target personnel recognition.

can be selected to ensemble single auxiliary task, there are four potential strategy combinations. For the purpose of clear representation of the results, we summarize the main results in Figure 6.4 and Table 6.8 and we leave the detailed results for every combination and parameter selection in Table C.1, C.2 and C.3 in Appendix C.

Table 6.8: Evaluated Relationships among Three Tasks.

Node	Relationship
1	Adversarial for Angle Recognition, Cooperative for target personnel recognition.
2	Adversarial for Angle Recognition, Cooperative for target personnel recognition.
3	Cooperative for Angle Recognition, Cooperative for target personnel recognition.

In Figure 6.4, *first*, the best-performing ensembling strategy in ENet outperforms the baseline by 1.7%, 3.9% and 2.8% respectively for the three nodes. Among the three nodes, the result improvement for node 2 is the largest which suggests that ***ensembling strategy in ENet is more useful for radar nodes with larger aspect angles***. Compared with the results when ensembling single task, these result improvements degrade and we hypothesize the following two reasons: i) the first may be that ensembling two tasks increases the parameter searching space and handles a more complex task which makes it more difficult to train the network; ii) this may be potentially because the gradients of the shared feature extractor compensate when learning multiple tasks. More detailed research works are required for further investigation.

As shown in Table C.1, C.2 and C.3, adopting CL for target personnel recognition achieves the best performing results for all nodes, no matter which strategy to ensemble for aspect angle recognition. This suggests that ***the relationship between motion and arget personnel recognition are cooperative when ensembling two auxiliary tasks.*** For strategies to ensemble aspect angle recognition in this more complex task, AL based methods outperform CL for node 1 but CL outperforms AL for node 3. These basically match with the findings when ensembling single task in Table 6.7. The only difference is the result of node 2 where AL based methods outperforms CL to ensemble aspect angle recognition. These observations suggest that ***the relationship between the aspect angle and motion recognition depends on different geometries.*** All the above findings are summarized in Table 6.8.

6.5 Summary

μ-DS classification methods in real-world applications are limited by two factors of variations, including the target personnel and aspect angle. In this chapter, to learn useful features for the motion recognition task only, we design ENet and ensemble the two auxiliary tasks including aspect angle and target personnel recognition. More specifically, the works carried out in this chapter generate the following contributions:

1. To the best of our knowledge, ENet may be the first development to utilize the label information of two auxiliary tasks, namely the aspect angle and target per-

sonnel recognition in a multi-task learning framework to boost the main motion recognition performance.

2. For ensembling multiple tasks, we propose two learning strategies, namely the CL and AL for incorporating more constraints on the feature extractor network of ENet.

3. Finally, we evaluate the ENet when ensembling single auxiliary and both auxiliary tasks for three nodes in the context of mono-static radar system. We also investigate and summarize the observed relationship and the relevant optimal ensembling strategy between the auxiliary and the main tasks via extensive experiments.

To sum up, ENet has been proved very beneficial for mono-static μ-DS classifcation by ensembling two auxiliary tasks using different proposed strategies. To the best of the author's knowledge, no research has focused on this in the context of μ-DS classification.

Through careful evaluation of the relationships between the auxiliary and the main task, we obtain the best-performing ensembling strategies among motion, angle and target personnel recognition. The following main findings are summarized:

- Ensembling either single or two auxiliary tasks outperforms the baseline SC-DopNet method, but the improvement to ensemble two tasks is not as good as the single auxiliary task. Specifically, ensembling single auxiliary task (best-perfoming strategy) outperforms the baseline SC-DopNet method in average by 3.6%, 14.2% and 2.7% from node 1 to 3 respectively. However, Ensembling two tasks outperforms the baseline by 1.7%, 3.9% and 2.2% from node 1 to 3 respectively.

- Based on the previous point, no matter which task to ensemble, the result improvement of ENet is larger when the aspect angle is larger.

- For all geometries and nodes, the target personnel recognition can be modelled as cooperative with the main motion recognition.

- The relationship between the aspect angle and the motion recognition largely depends on the radar geometries.

Chapter 7

Conclusions

This thesis investigates advanced radar Doppler techniques, including the signs-of-life detection using PWR and various μ-DS feature learning and classification methods using PWR and pulse multi-static radar for activity recognition. This chapter summarizes the contributions and outlines the potential future works.

7.1 Contribution Summary

The works carried out in this thesis explore the capability of radar Doppler information in activity monitoring, including the signs-of-life detection and the activity recognition. The detailed contributions are summarized in the following:

1. In *Chapter 3*, to the best of our knowledge, this is the first real-time phase-sensitive CAF processing using PWR to extract instantaneous Doppler information representing the chest-wall motion. The proposed method has been evaluated via both LoS and through-wall experiments to successfully detect the chest-wall motion of human targets. In addition, we design different bi-static radar geometries in LoS experiments and link the geometries with relevant detection results. Finally, we also analyze the potential problems met in the real-time implementation, explore relevant solutions and discuss the assumptions of the success of phase-sensitive CAF processing. This method has been verified to be robust to DSI from clutters and not limited to long integration time in the real-time Doppler processing. It is worth noting that the success of the proposed phase extraction method depends on the background phase stability of the PWR system. Before any trial and usage

of the phase-sensitive CAF method, we may carefully check the phase stability test of the PWR system because the large variation of the background phase may dominate in the phase variation of both the background and the one caused by the chest movement. In this way, the hampel filtering cannot be successfully applied to eliminate the large variations of the background noise (they are sometime at low frequencies as well). More detailed discussions can be seen in sections 3.4.1 and 3.7.1.

2. Also in *Chapter 3*, the PWR μ-DS dataset of six daily motions are collected and we focus on integrating the sparsity prior in designing the feature learning and classification methods. Specifically, this is the first application of SRC for the PWR μ-DS classification and SRC has been verified to outperform the well-known SVM by 33.7% in average on the PWR μ-DS dataset. In addition, facing a relatively small-scale dataset, we design a novel DCNN layer namely the DLL for the DTN and replace it with the FC layers in order to ensure the sparsity of the extracted features. All these designs and proposed methods have been evaluated using the PWR μ-DS dataset and DLL based DTN method has been verified to outperform the conventional DTN methods by 2% in average. The improved recognition results prove the following benefits of sparsity prior:

   - it is robust to the small perturbations and therefore suitable for small-scale dataset by increasing the generalization capability.
   - it implicitly allows variable-size feature representations, which controls the effective dimensions of the features. This is suitable for small-scale dataset from the perspectives of over-fitting prevention and the outlier elimination.

   Finally, we conduct the ablation study to analyze the effects of the sub-modules.

3. In *Chapter 4*, we propose the SC-DopNet and MC-DopNet for fine-grained μ-DS classification caused by very similar human motions. The novelties are summarized as follows:

   - The SC-DopNet is novel since we propose two effective schemes to train the networks using the small-scale μ-DS dataset, including suitable data augmentation schemes and proper regularizer of balancing the E-dist and M-dist. In addition, the effects of the schemes with the conventional DCNN training methods are analyzed using a detailed ablation study in the context of μ-DS classification, which has never been done before. By balancing the E and M-dist, recognition results of SC-DopNet on node 1 has been verified to outperform the one without the regularization by 7.6% in average. In addition, compared with the best empirical feature selection and classifier design, SC-DopNet outperforms the best-performing method by 12.5% in average.

- The MC-DopNet is proposed for integrating feature representations of multi-static radar in the final decision making. This is novel since we propose two effective fusion schemes based on two different strategies, namely (1) *"win by sacrificing worst case"* and (2) *"win by sacrificing best case"*. All the proposed methods have been evaluated via extensive experiments and comparisons of the proosed GIR and ℓ_{12}-norm methods largely depend on different geometries of the multi-static radar. However, no matter which propsoed strategy to use, MC-DopNet results outperform the best-performing non-deep feature selection with fusion methods by at least 9.8% using data from all the aspect angles. Finally, we also argue and discuss how to utilize the statistics of SC-DopNet results to infer the selection of fusion strategies for MC-DopNet based on different experimental scenarios, however, this finding requires more experimental data to evaluate.

4. In *Chapter 5*, we propose two deep adaptation networks, namely the RAAN and JAAN to eliminate the variational factors of μ-DS and to generalize the model to the unlabeled test samples with unseen factors. The novelties and contributions are summarized as follows:

    - μ-DS classifications in real-world applications are limited by distribution discrepancies or variations between the training data and the unseen test data. We may be the first to propose the two main factors of variation of μ-DS, including human personnel and the aspect angles.

- To the best of our knowledge, the work in this chapter may be the first application of UDA methods for μ-DS classification in order to eliminate the variations of the two factors.
- RAAN and JAAN are both based on the OT theory, which are novel both in the μ-DS classification and computer vision and machine learning communities. We evaluate these two networks using both computer vision datasets and the μ-DS datasets. RAAN and JAAN have been verified to outperform the best-performing UDA methods in computer vision by average of 4.8% and 6.8% in hand-written digit datasets and by average of 15.4% and 14.3% in the cross-modality dataset. For eliminating the two factors of variation in μ-DS in the "Factor Control" scenario, RAAN and JAAN outperforms the current best-performing result by 4.9% and 6.3% for target personnel, and by 1.4% and 2.1% for aspect angle respectively. For eliminating the two factors in "No Control" scenario, JAAN outperforms RAAN by 2.1% and 3.5% respectively for adapting the target personnel and aspect angle variations.
- We measure the variations of factors among different adaptation tasks quantitatively using the A-distance and various experiments are conducted to verify the proposed methods under different level of variations.

RAAN and JAAN have been proved to be very beneficial in reducing the variational factors, increasing the generalization capability of the DCNN and improving the recognition performance both in computer vision and radar community.

5. In *Chapter 6*, we propose ENet to improve the main motion recognition results by leveraging the labels of multiple auxiliary tasks, including the aspect angle and the target personnel recognition. The summarized contributions are as follows:

   - To the best of our knowledge, ENet may be the first developments to utilize labels of multiple factors by ensembling the main and auxiliary tasks.
   - We propose two learning strategies for ensembling multiple tasks, including the CL and AL.

- We investigate the potential relationships between each auxiliary task and the main ones using experiments.

We evaluate the proposed ENet using multi-static radar μ-DS and conclude the followings:

- Ensembling either single or two auxiliary tasks outperforms the baseline SC-DopNet method, but the improvement to ensemble two tasks is not as good as the single auxiliary task. Specifically, ensembling single auxiliary task (best-perfoming strategy) outperforms the baseline SC-DopNet method in average by 3.6%, 14.2% and 2.7% from node 1 to 3 respectively. However, Ensembling two tasks outperforms the baseline by 1.7%, 3.9% and 2.2% from node 1 to 3 respectively.
- No matter which task to ensemble, the improvement of ENet is larger when the aspect angle is larger.
- For all geometries and nodes, the target personnel recognition can be modelled as cooperative with the main motion recognition.
- The relationship between the aspect angle and the motion recognition largely depends on the radar geometries.

7.2 Future Work

The future work comprises the following three perspectives, including i) to estimate the chest-wall movements using phase information from the multi-static radar; ii) to extend JAAN by embedding the re-weighting scheme and matching the label distribution explicitly iii) to embed the sequential modelling in μ-DS classification. The more details are illustrated in the following:

- Signs-of-life detection using multi-static radar: the proposed phase-sensitive method is based on mono-static PWR and the phase output is limited by the bi-static radar geometries. In practical scenario, we cannot ensure that the target is sitting in the place and breathing. However, if multiple surveillance signals can be obtained from the multi-static PWR, the phase-sensitive CAF processing for each node can estimate the phase variations simultaneously. Leveraging the phase information from multiple nodes, we may estimate the chest movement more robustly, since the set-up of the multi-static radar is more robust to the geometries, especially the target position. In addition, based on the bi-static geometries and the measured phase information, we may estimate the rough geo-locations and the angle of the target by phase information extraction.

- Extend JAAN by integrating the re-weighting scheme: as introduced in Chapter 5, in the most challenging cross-modality dataset, RAAN outperforms JAAN and we argue that the re-weighting scheme is more effective to reduce domain discrepancies with a largely unbalanced dataset. In the real-world applications, unbalanced dataset is very common and we wish to improve JAAN by explicitly integrating the re-weighting scheme to estimate the target domain label distribution. This is also feasible since we only need to adopt the re-weighting formulation of RAAN in the JAAN formulation.

- Embed temporal sequential modelling: In this thesis, we have not explicitly considered to extract the the temporal features of the μ-DS. Since μ-DS is a sequence data, we may leverage the Long-Short-Temporal Model (LSTM) based Recurrent Neural Network (RNN) to model the sequence data and take the output feature of RNN as the input of all the proposed models in this thesis.

Appendix A

Materials of sparse coding and dictionary learning

A.1 Restricted Isometry Properties

In general, the Restricted Isometry Properties proposed the following two conditions so that the equivalence is guaranteed and we can obtain the sparsest solution s for the sparse coding problem [32, 33]:

- **Atom Incoherence:** this condition requires that the dictionary atom, or columns in D are incoherent with each other.
- **Sparsity of the Sparse Codes:** this condition requires the exact sparse code s is sufficiently sparse.

A.2 Pseudo-Codes of MP, OMP and Subspace Pursuit

In this section, we provide the pseudo-codes of MP, OMP and Subspace Pursuit algorithms in Algorithm 1, 2 and 3 respectively.

Algorithm 1: Matching Pursuit Algorithm

```
Input: Input signal, $y \in \mathbb{R}^M$; Dictionary, $D \in \mathbb{R}^{M \times N}$; Maximum active support
       number $N_s$; Reconstruction error bound $threshold$; Maximum iteration
       number $K$.
Output: Active support set $\mathbf{S_{act}} = \{s_i\}_{i=1}^{N_s}$; Index set $\Omega \subset [1,2,...,N]$
        corresponding to $\mathbf{S_{act}}$.
Initialising the residual $r_1 = y$;
for $k = 1 : K$ do
    i) Find the index $i$ by selecting the maximum inner product value $\|D^T r_l\|$.
    ii) Calculate the coefficients $s_i \leftarrow \frac{\|D^T r_l\|}{\|D_i\|^2}$, where $D_i$ is the $i^{th}$ atom of the
      dictionary $D$.
    iii) Add $s_i$ and $i$ in $\mathbf{S_{act}}$ and $\Omega$ respectively.
    iv) Calculate residual $r_{l+1} = r_l - s_i d_i$
    $l = l + 1$.
    Break if $l \geq N_s$ or $\|r_l\| \leq threahold$.
return $\mathbf{S_{act}}$ and $\Omega$
```

Algorithm 2: Orthogonal Matching Pursuit Algorithm

```
Input: Input signal, $y \in \mathbb{R}^M$; Dictionary, $D \in \mathbb{R}^{M \times N}$; Maximum active support
       number $N_s$; Reconstruction error bound $threshold$; Maximum iteration
       number $K$.
Output: Active support set $\mathbf{S_{act}} = \{s_i\}_{i=1}^{N_s}$; Index set $\Omega \subset [1,2,...,N]$
        corresponding to $\mathbf{S_{act}}$.
Initialising the residual $r_1 = y$;
for $k = 1 : K$ do
    i) Find the index $i$ by selecting the maximum inner product value $\|D^T r_l\|$.
    ii) Calculate the coefficients $s_i \leftarrow \frac{\|D^T r_l\|}{\|d_i\|^2}$, where $d_i$ is the $i^{th}$ atom of the
      dictionary $D$.
    iii) Add $s_i$ and $i$ in $\mathbf{S_{act,k}}$ and $\Omega_k$ respectively.
    iv) Calculate the projection matrix $P_k = D_{\Omega_k}(D_{\Omega_k}^T D_{\Omega_k})^{-1} D_{\Omega_k}^T$ to the
      subspace spanned by $D_{\Omega_k}$.
    v) Calculate residual $r_{k+1} = (I - P_k)y$
    $k = k + 1$.
    Break if $k \geq N_s$ or $\|r_k\| \leq threahold$.
return $\mathbf{S_{act,k}}$ and $\Omega_k$
```

Algorithm 3: Subspace Pursuit Algorithm

```
Input: Input signal, $y \in \mathbb{R}^M$; Dictionary, $D \in \mathbb{R}^{M \times N}$; Maximum active support
       number $N_s$; Reconstruction error bound $threshold$; Maximum iteration
       number $L$.
Output: Active support set $\mathbf{S_{act}} = \{s_i\}_{i=1}^{N_s}$; Index set $\Omega \subset [1,2,...,N]$
        corresponding to $\mathbf{S_{act}}$.
Initialising the residual $r_1 = y$;
for $k = 1 : K$ do
    i) Find the index set $\Omega_{supp1,k}$ by selecting the largest $2N_s$ inner product
      values $\|D^T r_l\|$.
    ii) Find another index support set $\Omega_{supp2,k}$ by selecting the largest $N_s$
      pseudo-inverse formulation: $(D^T_{\Omega_{supp1,k}} D_{\Omega_{supp1,k}})^{-1} D^T_{\Omega_{supp1,k}} r_k$
    iii) Calculate the coefficients $S_{act,k}$ based on $(D^T_{\Omega_{supp2,k}} D_{\Omega_{supp2,k}})^{-1} D^T_{\Omega_{supp2,k}} r_k$.
    iv) Set $\Omega_k = \Omega_{supp2,k}$.
    v) Calculate residual $r_{k+1} = r_k - D_{\Omega_k} S_{act,k}$
    $k = k + 1$.
    Break if $\|r_k\| \leq threahold$.
return $\mathbf{S_{act,k}}$ and $\Omega_k$
```

Appendix B

Materials of DCNN and Deep Adaptation Network

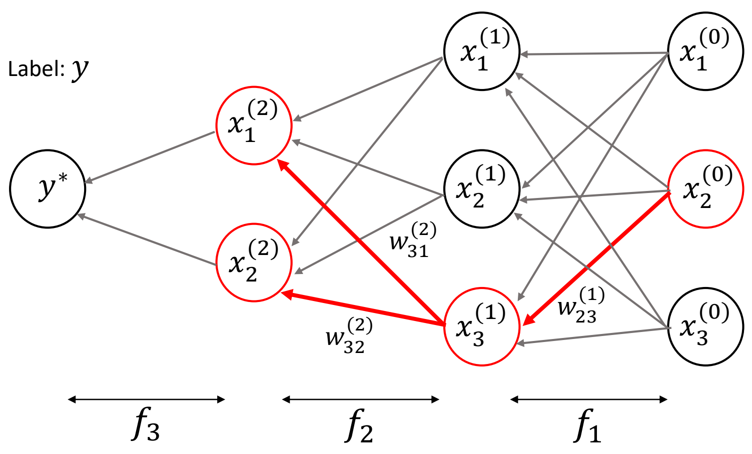
B.1 Optimization of DCNN

The whole optimization procedure is composed of the following three steps, including feed-forward step, back-prop step and weight update step. Specifically in FigureB.1, we use a simple Neural Network composed of three FC layers as an example to illustrate the steps to update the parameter $w_{23}^{(1)}$ which indicates the weight connecting the second element of the input $x_2^{(0)}$ to the third element of the first layer's output $x_3^{(1)}$.

- **Feed-Forward Step:** in the feed-forward step of FigureB.1, we first compute and store the relevant activations of intermediate layers $x_3^{(1)}$, $x_1^{(2)}$, $x_2^{(2)}$, the predicted label y^* and the loss function L w.r.t the input x.
- **Back-Prob Step:** Next, we calculate the gradient $\frac{\partial L}{\partial w_{23}^{(1)}}$ based on the equations in the second step of Figure B.1 using the feed-forwarded results in previous step.
- **Weight Update Step:** Finally, we update the weight $w_{23}^{(1)}$ by gradient descent, as shown in the third step in FigureB.1.

B.2 Batch Normalization in DCNN

In SGD, there exists large variations between different batch data which can increase the distribution oscillations of both network weights and the output activations. As the



i) Feed-Forward Step:
Compute and store intermediate outputs;
$x^{(1)}$ $x^{(2)}$ y^*

ii) Back-Prop Step:
Compute gradients w.r.t. network weights

$$\frac{\partial L}{\partial w_{23}^{(1)}} = \frac{\partial L}{\partial x_3^{(1)}} \frac{\partial x_3^{(1)}}{\partial w_{23}^{(1)}}$$

$$= \left(\frac{\partial L}{\partial x_2^{(2)}} \frac{\partial x_2^{(2)}}{\partial x_3^{(1)}} + \frac{\partial L}{\partial x_1^{(2)}} \frac{\partial x_1^{(2)}}{\partial x_3^{(1)}}\right) \frac{\partial x_3^{(1)}}{\partial w_{23}^{(1)}}$$

$$\frac{\partial L}{\partial x_2^{(2)}} = \frac{\partial L}{\partial y^*} \frac{\partial y^*}{\partial x_2^{(2)}}$$

$$\frac{\partial L}{\partial x_1^{(2)}} = \frac{\partial L}{\partial y^*} \frac{\partial y^*}{\partial x_1^{(2)}}$$

iii) Weight Update Step:
Update the weight via gradient descent.

$$w_{23}^{(1)} \leftarrow w_{23}^{(1)} - \gamma \frac{\partial L}{\partial w_{23}^{(1)}}$$

Figure B.1: Back-propagation in Neural Networks.

.

oscillations potentially can lead to training instability, we have to use smaller learning rates, which is prone to the gradient vanishing problem. Besides adopting the proper weight initialization techniques in Section 2.4.2.1, a more direct method is the Batch Normalization (BN) [194]. The basic idea behind is: instead of normalizing the input of the network as we usually do, we can also normalize inputs of all layers within the network. Note that BN has been integrated before activation function of each layer so that the input statistics of activation function is with zero mean and variance of one. This potentially allows the usage of larger learning rates, non-ideal initialization and alleviates the gradient vanishing problem caused by some activation functions like sigmoid and tanh.

As shown in the Eq.(B.1), the BN operation computes the whitened sample $\hat{x}^i$ (with zero mean and unit variance) using the mini-batch samples $x^i \in \mathbb{R}^d, i \in [1, 2, ..., M]$. Note that ε in Eq.(B.1) is used for preventing the intractable computation. Another common practice is to use two learnable scaling scalars for controlling the output of BN $\hat{y}^i$ which is also the input of the activation function in Eq.(B.2), as the zero mean and unit variance samples may not be optimal for activation functions. Note that both γ and the β in

Eq.(B.2) are with the same dimension as x^i.

$$\mu = \frac{1}{M}\sum_{i=1}^{M} x^i,$$

$$\sigma^2 = \frac{1}{M}\sum_{i=1}^{M}(x^i - \mu)^2 \tag{B.1}$$

$$\hat{x}^i = \frac{x^i - \mu}{\sqrt{\sigma^2 + \varepsilon}},$$

$$\hat{y}^i = \gamma \hat{x}^i + \beta \tag{B.2}$$

B.3 Relationship between JS divergence and Adversarial Training Optimization

First assuming that the network T is fixed, the optimal discriminator network D^* can be solved by the Eq.B.3 and the tractable solution is $D^*(T(x)) = \frac{P_s(T(x))}{P_s(T(x)) + P_t(T(x))}$. Next, we can derive the optimization function $L_{Disc}(T)$ by replacing the discriminator network with the optimal one $D^*(T(x))$. More specifically, the optimization function w.r.t the network T is $L_{Disc}(T)$ in Eq.B.3.1. Note that since the JS divergence is at least zero so the minimum of $L_{Disc}(T)$ is $-log4$ as long as the distributions from two domains are matched. Therefore, the network T is optimized based on the Eq.B.4.

$$L_{Disc}(T) = \max_{D} \int_{T(x)} P_s(T(x))log(D(T(x))) + P_t(T(x))log(1 - D(T(x)))dT(x) \tag{B.3}$$

$$\min_{T} L_{Disc}(T) \tag{B.4}$$

$$\begin{aligned}
L_{Disc}(T) = \int_{T(x)} & P_s(T(x))log(\frac{P_s(T(x))}{P_s(T(x))+P_t(T(x))}) \\
& +P_t(T(x))log(\frac{P_t(T(x))}{P_s(T(x))+P_t(T(x))})dT(x) \\
& = \mathbb{E}_{T(x)\sim P_s(T(x))}[log(\frac{P_s(T(x))}{P_s(T(x))+P_t(T(x))})] \\
& +\mathbb{E}_{T(x)\sim P_t(T(x))}[log(\frac{P_t(T(x))}{P_s(T(x))+P_t(T(x))})] \\
& = \mathbb{E}_{T(x)\sim P_s(T(x))}[log(\frac{1}{2})+log(\frac{P_s(T(x))}{\frac{1}{2}(P_s(T(x))+P_t(T(x)))})] \\
& +\mathbb{E}_{T(x)\sim P_t(T(x))}[log(\frac{1}{2})+log(\frac{P_t(T(x))}{\frac{1}{2}(P_s(T(x))+P_t(T(x)))})] \\
& = -log(4)+KL(P_t(T(x))||\frac{P_s(T(x)}{2(P_s(T(x))+P_t(T(x)))}) \\
& +KL(P_s(T(x))||\frac{P_t(T(x)}{2(P_s(T(x))+P_t(T(x)))}) \\
& = -log(4)+JS(P_s(T(x))||P_t(T(x)))
\end{aligned} \tag{B.5}$$

B.4 Pseudo-code of Sinkhorn Algorithm

In this section, we illustrate the methods JAAN utilized for calculating the Sinkhorn loss, as shown in the following Algorithm 4. Note that in the last line, *diag* indicates the function to transform a vector to a diagonal matrix whose diagonal elements remains the vector.

Algorithm 4: Sinkhorn loss

```
Input: Source Domain Joint Representation $(J_i^s)_{i=1}^m$, Target Domain Joint
       Representation $(J_j^t)_{j=1}^m$, Presenter Network $P$, weight of entropy
       regularization $\varepsilon$, total iteration number $L$
Output: Transportation Map $\gamma$, Sinkhorn Loss $L_{sinkhorn}$
.
Initialising $b = \mathbf{1}_m$
$\forall(i, j),\ calculate\ c(J_i^s, J_j^t) = cos(P(J_i^s), P(J_j^t)),\ M_{i,j} = M(J_i^s, J_j^t)$
$\mathbf{K}_{i,j} = exp^{-\frac{M(J_i^s, J_j^t)}{\varepsilon}}$
for $l = 1 : L$ do
    Update $a$, given $b$ and $\mathbf{K}$:
        $a = \frac{\mathbf{1}_m}{\mathbf{K}b}$    (B.6)
    Update $b$, given $a$ and $\mathbf{K}$:
        $b = \frac{\mathbf{1}_m}{\mathbf{K}^T a}$    (B.7)
    Increment $l$.
return $\gamma = diag(a)\mathbf{K}diag(b),\ L_{sinkhorn} = \langle\gamma, M\rangle_F$
```

Appendix C

Results of Ensembling Two Tasks in Exp7-9 in Chapter 6

In this appendix, we provide the detailed results of ensembling two tasks in Exp7-9 of Chapter 6.

Table C.1: Result of ensembling two tasks, node 1, percentage in (%).

(ω_A,ω_H)	Motion Coop Angle Coop Human	Motion Coop Angle Adv Human	Motion Adv Angle Coop Human	Motion Adv Angle Adv Human
(5,5)	92.7,83.4,90.0	92.7,82.3,26.3	92,5,22.2, 87.8	93.5,37.5,29.3
(5,1)	92.6,83.7,90.0	92.6,83.3,30.3	92.3,26.9,88.4	93.2,27.7,34.1
(5,0.1)	92.8, 82.1, 88.8	92.8,83.2,43.8	93.4,24.6,86.9	92.5,27.3,61.4
(5,0.01)	92.6, 83.0, 87.5	93.2,83.4,73.2	92.7,24.8,85.6	92.8,24.0,81.9
(5,0.001)	92.4,82.3,85.8	93.3,83.0, 86.0	93.2,30.6,84.7	93.2,27.1,83.6
(1,5)	93.0, 84.9, 88.5	92.3,82.9,24.9	92.7, 30.0, 89.3	93.0,33.6,26.5
(1,1)	92.9,84.0,89.0	93.6,83.4,29.6	92.6, 21.5, 87.7	93.4,28.6,29.1
(1,0.1)	92.9,84.1,88.9	93.2, 82.4,36.2	**94.3, 27.1, 88.8**	92.8,23.4,57.5
(1,0.01)	93.8, 83.9, 87.9	94.0,83.5,53.0	93.6, 30.0, 86.7	93.6,25.1,81.9
(1,0.001)	93.4,83.3,87.8	93.3,82.5, 82.1	93.2, 27.3, 85.6	93.3,23.3,84.6
(0.1,5)	92.9,81.1,88.2	93.5,80.9, 26.5	93.2, 52.7, 89.0	92.6, 68.6, 26.8
(0.1,1)	93.2,84.6,89.0	93.3,82.3,28.1	93.1, 46.0, 89.7	93.8,39.1,27.3
(0.1,0.1)	93.5, 84.3,89.9	93.5,83.5,34.5	93.7, 34.9, 89.5	92.7,27.3,30.3
(0.1,0.01)	93.4,81.6,88.3	92.9,82.1,43.2	93.3, 23.6, 86.7	93.6, 29.8, 57.0
(0.1,0.001)	93.5,82.8, 87.6	93.6,82.7,65.0	93.7,33.6, 85.2	93.6,70.9,23.6
(0.01,5)	93.3,82.1,90.9	92.6,79.2,27.1	92.6, 66.6,88.3	92.9,76.8,23.6
(0.01,1)	93.9,81.4,88.5	93.8,82.2,24.2	93.2, 63.9, 89.4	93.2,65.9,26.8
(0.01,0.1)	93.5,81.5,87.9	93.2,81.9,30.8	92.8,41.4,88.1	93.3,55.6,32.2
(0.01,0.01)	93.8,81.7,88.8	92.8,79.8,47.1	93.5,54.1,86.9	94.0, 50.8, 54.4
(0.01,0.001)	93.9,81.7,89.8	94.2,82.4,83.9	93.8,42.2,86.0	93.0,48.9, 33.6
(0.001,5)	93.1,80.5,89.5	93.1,77.8,26.4	93.6,80.7,90.0	92.9, 78.3, 26.0
(0.001,1)	93.2,81.7,88.2	93.3,78.8,27.1	93.0, 74.5, 88.3	93.1,78.1,24.8
(0.001,0.1)	93.7,82.2,89.1	93.7,80.9,34.1	94.0 ,72.1, 88.5	93.2,63.1,31.5
(0.001,0.01)	93.6,82.3,88.8	93.8,78.0,61.5	93.5,71.7,87.9	93.4,61.5,38.5
(0.001,0.001)	93.5,81.4,88.7	92.3,82.0,80.3	93.3,70.7,83.9	93.8,74.7,81.4
Baseline	Motion recognition rate without multi-task learning framework 92.6 [see Chapter 4, Table 4.5]			

Table C.2: Result of ensembling two tasks, node 2, percentage in (%).

(ω_A,ω_H)	Motion Coop Angle Coop Human	Motion Coop Angle Adv Human	Motion Adv Angle Coop Human	Motion Adv Angle Adv Human
(5,5)	80.4, 75.7, 83.2	77.8,78.4,26.1	81.3, 29.1, 81.9	81.2, 32.0, 31.7
(5,1)	81.4,76.8, 80.3	79.4,76.7,27.5	82.8, 28.5, 80.5	82.1, 25.0, 35.8
(5,0.1)	80.1, 74.4, 79.8	81.0,76.9,26.7	81.3, 33.3, 78.1	79.2,26.9,77.4
(5,0.01)	80.9, 74.8, 79.6	80.3,77.8,78.9	79.6, 28.0, 79.2	83.8,25.8,79.2
(5,0.001)	81.9,76.3,79.1	78.4,76.2,78.3	80.4, 26.3, 80.0	81.8, 28.1,79.8
(1,5)	81.1, 76.9,81.5	79.6,72.3,25.2	81.8,30.5,80.0	81.2,42.4,31.9
(1,1)	81.3,74.4,82.5	81.5,77.2,27.5	81.9,29.4,78.3	81.9,27.3,28.5
(1,0.1)	80.5, 75.9, 81.1	78.8,76.0,49.4	79.6,24.0,81.2	83.2,32.9,54.2
(1,0.01)	80.8,74.4,77.3	81.9,75.3,76.5	80.8,31.2,79.0	81.6,26.9,78.1
(1,0.001)	81.1,78.2,78.9	82.4,74.4,77.7	81.8,32.3,76.9	81.2,25.0,76.0
(0.1,5)	81.6,75.4,82.2	81.3,73.8,26.9	79.5,56.0,80.0	82.8,70.7,25.5
(0.1,1)	80.3,74.6,81.7	80.9,71.7,24.2	80.8,36.9,80.0	81.2,59.4,22.8
(0.1,0.1)	80.7,73.6,81.0	81.7,68.5,32.8	80.6,29.7,81.7	80.0,29.4,27.9
(0.1,0.01)	81.0,73.4,81.9	80.8,71.5,59.5	83.1,29.8,79.4	81.1,30.1,64.3
(0.1,0.001)	81.7,75.2,76.7	81.1,73.2,78.0	80.5,25.6,75.7	81.7,28.4,75.8
(0.01,5)	80.3,71.9,82.3	80.5,71.0,20.2	82.3,70.9,81.8	79.8,70.7,30.2
(0.01,1)	82.8,72.7,82.8	80.9,71.0,20.5	78.8,62.4,81.5	79.6,68.2,23.5
(0.01,0.1)	81.2,70.3,79.0	80.6,69.3,27.8	80.2,65.5,79.4	82.1,61.9,28.4
(0.01,0.01)	81.7,72.7,78.5	80.7,72.8,71.9	**84.1**,67.6,78.5	81.0,64.1,72.8
(0.01,0.001)	82.5,73.2,79.7	80.5,71.0,76.2	80.6,53.8,75.6	81.3,63.8,72.9
(0.001,5)	79.2,71.2,81.8	82.1,69.5,27.5	80.4,73.2,81.6	80.0,69.0,27.2
(0.001,1)	80.8,72.2,82.1	80.8,70.1,26.4	78.6,70.5,82.3	82.1,70.6,25.5
(0.001,0.1)	81.9,74.1,79.4	81.5,66.5,31.2	82.7,66.8,80.4	81.5,66.9,33.6
(0.001,0.01)	82.0,71.4,81.1	79.7,69.4,78.9	81.5,70.4,74.6	79.4,64.2,67.4
(0.001,0.001)	82.4,71.4,76.7	82.4,68.1,75.2	80.2,67.9,76.1	81.6,69.5,76.9
Baseline	Motion recognition rate without multi-task learning framework 80.2 [see Chapter 4, Table 4.5]			

Table C.3: Result of ensembling two tasks, node 3, percentage in (%).

(ω_A,ω_H)	Motion Coop Angle Coop Human	Motion Coop Angle Adv Human	Motion Adv Angle Coop Human	Motion Adv Angle Adv Human
(5,5)	91.0,99.0,81.6	91.5,98.8,26.2	91.2,13.5,81.9	89.7,27.2,33.1
(5,1)	90.0,98.5,82.4	90.3,99.2,27.7	90.2,23.2,80.0	91.1,16.9,33.3
(5,0.1)	92.2,98.6,83.7	91.2,97.8,39.8	91.9,15.0,78.0	91.3,13.7,54.2
(5,0.01)	90.5, 98.3, 75.4	91.8,98.6,66.7	91.2,20.4,74.0	90.9,25.3,70.9
(5,0.001)	89.2, 97.7, 78.1	91.6,99.0,79.2	90.0.24.6,71.7	91.1,22.5,74.1
(1,5)	90.6, 97.5, 84.2	91.1,98.1,29.2	90.1,31.5,80.7	**92.5,27.0,30.3**
(1,1)	91.0, 98.4, 84.9	91.9,98.7,32.1	91.9,12.3,76.5	90.1,24.2,34.6
(1,0.1)	91.7, 98.0,78.5	90.2,97.9,42.4	91.0,18.5,79.4	91.0,10.9,42.7
(1,0.01)	91.0, 98.1, 78.9	90.6,98.5,56.9	92.1,24.8,78.5	89.6,18.1,67.4
(1,0.001)	91.7, 97.7, 78.6	91.7,98.1,72.7	89.8,13.6,70.6	91.0,6.0,68.1
(0.1,5)	89.4, 97.4, 81.1	91.4,97.3,24.8	89.2,79.0,84.3	91.9,93.8,33.5
(0.1,1)	89.3, 97.8, 83.7	91.2,97.3,30.0	91.2,67.8,80.4	90.8,63.7,30.9
(0.1,0.1)	91.5, 98.6, 80.4	92.0,98.6,33.7	91.3,78.2,80.3	89.8,62.7,37.8
(0.1,0.01)	91.3, 98.5, 77.9	91.8,97.8,55.5	90.3,38.3,72.2	87.7,51.5,69.4
(0.1,0.001)	89.3, 97.8, 74.4	90.5,97.4,69.8	90.7,70.1,74.0	89.0,50.9,63.0
(0.01,5)	91.9, 98.0, 81.4	92.4,96.1,33.0	90.2,98.0,85.3	91.7,97.3,27.2
(0.01,1)	89.7, 96.1, 78.7	91.0,97.6,27.5	91.4,96.3,82.8	**92.5,95.7,27.2**
(0.01,0.1)	91.0, 96.9, 81.7	90.9,97.5,35.4	87.7,87.9,76.8	91.2,86.7,31.7
(0.01,0.01)	90.9, 96.8,76.4	90.1,94.6,50.8	90.0,94.6,77.5	92.4,92.5,71.0
(0.01,0.001)	91.8, 97.2, 78.4	90.0,97.4,75.9	92.3,81.0,76.5	87.6,88.6,67.4
(0.001,5)	90.7, 98.4, 80.5	92.3,96.7,28.6	91.1,97.5,83.9	89.6,93.8,18.9
(0.001,1)	**92.9**, 95.0, 82.5	90.1,95.1,21.2	90.1,94.8,83.2	91.8,97.2,26.4
(0.001,0.1)	90.2, 93.3, 78.9	90.3,97.1,34.8	88.7,94.6,79.0	91.0,97.1,36.7
(0.001,0.01)	91.2, 94.2, 78.1	87.0,86.2,58.5	92.3,94.0,78.4	90.9,93.6,50.8
(0.001,0.001)	91.4, 94.2, 74.0	90.0,94.0,71.1	90.6,96.1,76.8	90.9,93.5,70.9
Baseline	Motion recognition rate without multi-task learning framework 90.1 [see Chapter 4, Table 4.5]			

Bibliography

[1] Bassem R Mahafza and Atef Elsherbeni. *MATLAB simulations for radar systems design*. Chapman and Hall/CRC, 2003.

[2] Shaun Raymond Doughty. *Development and performance evaluation of a multi-static radar system*. PhD thesis, University of London, 2008.

[3] Francesco Fioranelli, Matthew Ritchie, and Hugh Griffiths. Multistatic human micro-doppler classification of armed/unarmed personnel. *IET Radar, Sonar & Navigation*, 9(7):857–865, 2015.

[4] Francesco Fioranelli, Matthew Ritchie, and Hugh Griffiths. Aspect angle dependence and multistatic data fusion for micro-doppler classification of armed/unarmed personnel. *IET Radar, Sonar & Navigation*, 9(9):1231–1239, 2015.

[5] Victor C Chen. Doppler signatures of radar backscattering from objects with micro-motions. *IET Signal Processing*, 2(3):291–300, 2008.

[6] V Chen, D Tahmoush, and W Miceli. Micro-doppler signatures-review, challenges, and perspectives. *Radar–Micro Doppler Signatures: Processing and Applications*, pages 1–25, 2014.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[9] Merrill Ivan Skolnik. Radar handbook. 1970.

[10] Merrill I Skolnik. Introduction to radar. *Radar handbook*, 2, 1962.

[11] Kevin Chetty, Graeme E Smith, and Karl Woodbridge. Through-the-wall sensing of personnel using passive bistatic wifi radar at standoff distances. *IEEE Transactions on Geoscience and Remote Sensing*, 50(4):1218–1226, 2012.

[12] Qingchao Chen, Bo Tan, Karl Woodbridge, and Kevin Chetty. Indoor target tracking using high doppler resolution passive wi-fi radar. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5565–5569. IEEE, 2015.

[13] Philip Mayne Woodward. *Probability and Information Theory, with Applications to Radar: International Series of Monographs on Electronics and Instrumentation*, volume 3. Elsevier, 2014.

[14] Victor C Chen. *The micro-Doppler effect in radar*. Artech House, 2011.

[15] Hermann Rohling. Radar cfar thresholding in clutter and multiple target situations. *IEEE transactions on aerospace and electronic systems*, (4):608–621, 1983.

[16] François Auger, Patrick Flandrin, Paulo Gonçalvès, and Olivier Lemoine. Time-frequency toolbox. *CNRS France-Rice University*, 46, 1996.

[17] Shie Qian and Dapang Chen. Decomposition of the wigner-ville distribution and time-frequency distribution series. *IEEE Transactions on Signal Processing*, 42(10):2836–2842, 1994.

[18] Patrick Flandrin, Gabriel Rilling, and Paulo Goncalves. Empirical mode decomposition as a filter bank. *IEEE signal processing letters*, 11(2):112–114, 2004.

[19] Norden E Huang, Zheng Shen, Steven R Long, Manli C Wu, Hsing H Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Henry H Liu. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. In *Proceedings of the Royal Society of London A: mathematical, physical and engineering sciences*, volume 454, pages 903–995. The Royal Society, 1998.

[20] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[21] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

[22] John Wright, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S Huang, and Shuicheng Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031–1044, 2010.

[23] Jakob J Verbeek, Nikos Vlassis, and Ben Kröse. Efficient greedy learning of gaussian mixture models. *Neural computation*, 15(2):469–485, 2003.

[24] Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

[25] Stephen J Wright, Robert D Nowak, and Mário AT Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

[26] Meng Yang, Dengxin Dai, Linlin Shen, and Luc Van Gool. Latent dictionary learning for sparse representation based classification. In *Proceedings CVPR 2014*, pages 4138–4145, 2014.

[27] Bhaskar D Rao and Kenneth Kreutz-Delgado. An affine scaling methodology for best basis selection. *IEEE Transactions on signal processing*, 47(1):187–200, 1999.

[28] Bhaskar D Rao, Kjersti Engan, Shane F Cotter, Jason Palmer, and Kenneth Kreutz-Delgado. Subset selection in noise based on diversity measure minimization. *IEEE transactions on Signal processing*, 51(3):760–770, 2003.

[29] David Wipf and Srikantan Nagarajan. Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions. *IEEE Journal of Selected Topics in Signal Processing*, 4(2):317–329, 2010.

[30] David P Wipf, Bhaskar D Rao, and Srikantan Nagarajan. Latent variable bayesian models for promoting sparsity. *IEEE Transactions on Information Theory*, 57(9):6236–6255, 2011.

[31] Sunil L Kukreja, Johan Löfberg, and Martin J Brenner. A least absolute shrinkage and selection operator (lasso) for nonlinear system identification. *IFAC proceedings volumes*, 39(1):814–819, 2006.

[32] Emmanuel J Candès et al. Compressive sampling. In *Proceedings of the international congress of mathematicians*, volume 3, pages 1433–1452. Madrid, Spain, 2006.

[33] Emmanuel J Candes. The restricted isometry property and its implications for compressed sensing. *Comptes rendus mathematique*, 346(9-10):589–592, 2008.

[34] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11):1413–1457, 2004.

[35] José M Bioucas-Dias and Mário AT Figueiredo. A new twist: Two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Transactions on Image processing*, 16(12):2992–3004, 2007.

[36] Mário AT Figueiredo, Robert D Nowak, and Stephen J Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of selected topics in signal processing*, 1(4):586–597, 2007.

[37] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[38] Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 399–406. Omnipress, 2010.

[39] Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.

[40] David L Donoho, Yaakov Tsaig, Iddo Drori, and Jean-luc Starck. Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit, submitted to. In *IEEE Trans. on Information theory*. Citeseer, 2006.

[41] Joel A Tropp and Anna C Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on information theory*, 53(12):4655–4666, 2007.

[42] Deanna Needell and Joel A Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and computational harmonic analysis*, 26(3):301–321, 2009.

[43] Wei Dai and Olgica Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE transactions on Information Theory*, 55(5):2230–2249, 2009.

[44] Joel Tropp, Anna C Gilbert, et al. Signal recovery from partial information via orthogonal matching pursuit. *IEEE Trans. Inform. Theory*, 53(12):4655–4666, 2007.

[45] Thong T Do, Lu Gan, Nam Nguyen, and Trac D Tran. Sparsity adaptive matching pursuit algorithm for practical compressed sensing. In *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*, pages 581–587. IEEE, 2008.

[46] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Discriminative learned dictionaries for local image analysis. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[47] Ignacio Ramirez, Pablo Sprechmann, and Guillermo Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3501–3508. IEEE, 2010.

[48] Meng Yang, Lei Zhang, Xiangchu Feng, and David Zhang. Fisher discrimination dictionary learning for sparse representation. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 543–550. IEEE, 2011.

[49] Zhuolin Jiang, Zhe Lin, and Larry S Davis. Label consistent K-SVD: Learning a discriminative dictionary for recognition. volume 35, pages 2651–2664, 2013.

[50] Yang Liu, Wei Chen, Qingchao Chen, and Ian Wassell. Support discrimination dictionary learning for image classification. In *European Conference on Computer Vision*, pages 375–390. Springer, 2016.

[51] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *Advances in neural information processing systems*, pages 4898–4906, 2016.

[52] Steven B Damelin and Willard Miller Jr. *The mathematics of signal processing*, volume 48. Cambridge University Press, 2012.

[53] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[54] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[55] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[57] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[58] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[59] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[60] Douglas M Kline and Victor L Berardi. Revisiting squared-error and cross-entropy functions for training neural network classifiers. *Neural Computing & Applications*, 14(4):310–318, 2005.

[61] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[62] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[63] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[64] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[65] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[66] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[67] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[68] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[69] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[70] Vladimir Naumovich Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

[71] Grégoire Mesnil, Yann Dauphin, Xavier Glorot, Salah Rifai, Yoshua Bengio, Ian Goodfellow, Erick Lavoie, Xavier Muller, Guillaume Desjardins, David Warde-Farley, et al. Unsupervised and transfer learning challenge: a deep learning approach. In *Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning workshop-Volume 27*, pages 97–111. JMLR. org, 2011.

[72] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[73] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[74] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2014.

[75] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 372–387. IEEE, 2016.

[76] Alex Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A hilbert space embedding for distributions. In *International Conference on Algorithmic Learning Theory*, pages 13–31. Springer, 2007.

[77] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[78] Nicolas Courty, Rémi Flamary, Amaury Habrard, and Alain Rakotomamonjy. Joint distribution optimal transportation for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 3730–3739, 2017.

[79] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1853–1865, 2017.

[80] Ronald Poppe. Vision-based human motion analysis: An overview. *Computer vision and image understanding*, 108(1-2):4–18, 2007.

[81] James W Davis and Aaron F Bobick. The representation and recognition of human movement using temporal templates. In *cvpr*, page 928. IEEE, 1997.

[82] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3038–3046, 2017.

[83] Daniel Weinland and Edmond Boyer. Action recognition using exemplar-based embedding. In *CVPR 2008-IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE Computer Society, 2008.

[84] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558, 2013.

[85] Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. 2008.

[86] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72. IEEE, 2005.

[87] Pyry Matikainen, Martial Hebert, and Rahul Sukthankar. Trajectons: Action recognition through the motion analysis of tracked features. In *2009 IEEE 12th international conference on computer vision workshops, ICCV workshops*, pages 514–521. IEEE, 2009.

[88] Ross Messing, Chris Pal, and Henry Kautz. Activity recognition using the velocity histories of tracked keypoints. In *2009 IEEE 12th international conference on computer vision*, pages 104–111. IEEE, 2009.

[89] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941, 2016.

[90] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[91] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016.

[92] Shugao Ma, Leonid Sigal, and Stan Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1942–1950, 2016.

[93] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.

[94] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[95] Emmanuel Munguia Tapia, Stephen S Intille, and Kent Larson. Activity recognition in the home using simple and ubiquitous sensors. In *International conference on pervasive computing*, pages 158–175. Springer, 2004.

[96] Zhenyu He and Lianwen Jin. Activity recognition from acceleration data based on discrete consine transform and svm. In *2009 IEEE International Conference on Systems, Man and Cybernetics*, pages 5041–5044. IEEE, 2009.

[97] Matthai Philipose, Kenneth P Fishkin, Mike Perkowitz, Donald J Patterson, Dieter Fox, Henry Kautz, and Dirk Hahnel. Inferring activities from interactions with objects. *IEEE pervasive computing*, (4):50–57, 2004.

[98] Beth Logan, Jennifer Healey, Matthai Philipose, Emmanuel Munguia Tapia, and Stephen Intille. A long-term evaluation of sensing modalities for activity recog-

nition. In *International conference on Ubiquitous computing*, pages 483–500. Springer, 2007.

[99] Girija Chetty and Matthew White. Body sensor networks for human activity recognition. In *2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN)*, pages 660–665. IEEE, 2016.

[100] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *International workshop on ambient assisted living*, pages 216–223. Springer, 2012.

[101] Christian Vollmer, Horst-Michael Gross, and Julian P Eggert. Learning features for activity recognition with shift-invariant sparse coding. In *International conference on artificial neural networks*, pages 367–374. Springer, 2013.

[102] Nuria Oliver, Eric Horvitz, and Ashutosh Garg. Layered representations for human activity recognition. In *Proceedings. Fourth IEEE International Conference on Multimodal Interfaces*, pages 3–8. IEEE, 2002.

[103] Jianbo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali Krishnaswamy. Deep convolutional neural networks on multichannel time series for human activity recognition. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[104] Youngwook Kim, Sungjae Ha, and Jihoon Kwon. Human detection using doppler radar based on physical characteristics of targets. *IEEE Geoscience and Remote Sensing Letters*, 12(2):289–293, 2015.

[105] Youngwook Kim and Hao Ling. Human activity classification based on micro-doppler signatures using a support vector machine. *IEEE Transactions on Geoscience and Remote Sensing*, 47(5):1328–1337, 2009.

[106] Ram M Narayanan and Matthew Zenaldin. Radar micro-doppler signatures of various human activities. *IET Radar, Sonar & Navigation*, 9(9):1205–1215, 2015.

[107] Francesco Fioranelli, Matthew Ritchie, and Hugh Griffiths. Bistatic human micro-doppler signatures for classification of indoor activities. In *2017 IEEE Radar Conference (RadarConf)*, pages 0610–0615. IEEE, 2017.

[108] Qingchao Chen, Matthew Ritchie, Yang Liu, Kevin Chetty, and Karl Woodbridge. Joint fall and aspect angle recognition using fine-grained micro-doppler classification. In *Radar Conference (RadarConf), 2017 IEEE*, pages 0912–0916. IEEE, 2017.

[109] Qingchao Chen, Bo Tan, Kevin Chetty, and Karl Woodbridge. Activity recognition based on micro-doppler signature with in-home wi-fi. In *e-Health Networking, Applications and Services (Healthcom), 2016 IEEE 18th International Conference on*, pages 1–6. IEEE, 2016.

[110] W. Li, B. Tan, Y. Xu, and R. J. Piechocki. Log-likelihood clustering-enabled passive rf sensing for residential activity recognition. *IEEE Sensors Journal*, 18(13):5413–5421, July 2018.

[111] Thayananthan Thayaparan, Sumeet Abrol, Edwin Riseborough, LJ Stankovic, Denis Lamothe, and Grant Duff. Analysis of radar micro-doppler signatures from experimental helicopter and human data. *IET Radar, Sonar & Navigation*, 1(4):289–299, 2007.

[112] Francesco Fioranelli, Matthew Ritchie, and Hugh Griffiths. Performance analysis of centroid and svd features for personnel recognition using multistatic micro-doppler. *IEEE Geoscience and Remote Sensing Letters*, 13(5):725–729, 2016.

[113] Zhaoxi Chen, Gang Li, Francesco Fioranelli, and Hugh Griffiths. Personnel recognition and gait classification based on multistatic micro-doppler signatures using deep convolutional neural networks. *IEEE Geoscience and Remote Sensing Letters*, 2018.

[114] Cory D Kidd, Robert Orr, Gregory D Abowd, Christopher G Atkeson, Irfan A Essa, Blair MacIntyre, Elizabeth Mynatt, Thad E Starner, and Wendy Newstetter.

The aware home: A living laboratory for ubiquitous computing research. In *International Workshop on Cooperative Buildings*, pages 191–198. Springer, 1999.

[115] Mark Perry, Alan Dowdall, Lorna Lines, and Kate Hone. Multimodal and ubiquitous computing systems: supporting independent-living older users. *IEEE Transactions on information technology in biomedicine*, 8(3):258–270, 2004.

[116] AM Adami, TL Hayes, and M Pavel. Unobtrusive monitoring of sleep patterns. In *Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual International Conference of the IEEE*, volume 2, pages 1360–1363. IEEE, 2003.

[117] Sajal K Das, Diane J Cook, Amiya Battacharya, Edwin O Heierman, and Tze-Yun Lin. The role of prediction algorithms in the mavhome smart home architecture. *IEEE Wireless Communications*, 9(6):77–84, 2002.

[118] Yoshinori Isoda, Shoji Kurakake, and Hirotaka Nakano. Ubiquitous sensors based human behavior modeling and recognition using a spatio-temporal representation of user states. In *Advanced Information Networking and Applications, 2004. AINA 2004. 18th International Conference on*, volume 1, pages 512–517. IEEE, 2004.

[119] NM Barnes, NH Edwards, DAD Rose, and P Garner. Lifestyle monitoring-technology for supported independence. *Computing & Control Engineering Journal*, 9(4):169–174, 1998.

[120] Marie Chan, Cyril Hariton, Patrick Ringeard, and Eric Campo. Smart house automation system for the elderly and the disabled. In *Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on*, volume 2, pages 1586–1589. IEEE, 1995.

[121] John Krumm, Steve Harris, Brian Meyers, Barry Brumitt, Michael Hale, and Steve Shafer. Multi-camera multi-person tracking for easyliving. In *Visual Surveillance, 2000. Proceedings. Third IEEE International Workshop on*, pages 3–10. IEEE, 2000.

[122] Daniel E Riedel, Svetha Venkatesh, and Wanquan Liu. Spatial activity recognition in a smart home environment using a chemotactic model. In *Intelligent Sensors, Sensor Networks and Information Processing Conference, 2005. Proceedings of the 2005 International Conference on*, pages 301–306. IEEE, 2005.

[123] Marie Chan, Daniel Estève, Christophe Escriba, and Eric Campo. A review of smart homespresent state and future challenges. *Computer methods and programs in biomedicine*, 91(1):55–81, 2008.

[124] Tatsuya Yamazaki. The ubiquitous home. *International Journal of Smart Home*, 1(1):17–22, 2007.

[125] Kyriaki Kalimeri, Aleksandar Matic, and Alessandro Cappelletti. Rfid: Recognizing failures in dressing activity. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2010 4th International Conference*, pages 1–4. IEEE, 2010.

[126] Ni Zhu, Tom Diethe, Massimo Camplani, Lili Tao, Alison Burrows, Niall Twomey, Dritan Kaleshi, Majid Mirmehdi, Peter Flach, and Ian Craddock. Bridging e-health and the internet of things: The sphere project. *IEEE Intelligent Systems*, 30(4):39–46, 2015.

[127] Mu-Cyun Tang, Fu-Kang Wang, and Tzyy-Sheng Horng. Vital-sign detection based on a passive wifi radar. In *RF and Wireless Technologies for Biomedical and Healthcare Applications (IMWS-BIO), 2015 IEEE MTT-S 2015 International Microwave Workshop Series on*, pages 74–75. IEEE, 2015.

[128] W. Li, B. Tan, and R. J. Piechocki. Non-contact breathing detection using passive radar. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6, May 2016.

[129] Usman Mahmood Khan, Zain Kabir, and Syed Ali Hassan. Wireless health monitoring using passive wifi sensing. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2017 13th International*, pages 1771–1776. IEEE, 2017.

[130] G. Li, R. Zhang, M. Ritchie, and H. Griffiths. Sparsity-driven micro-doppler feature extraction for dynamic hand gesture recognition. *IEEE Transactions on Aerospace and Electronic Systems*, 54(2):655–665, April 2018.

[131] M. S. Seyfiolu and S. Z. Grbz. Deep neural network initialization methods for micro-doppler classification with low training sample support. *IEEE Geoscience and Remote Sensing Letters*, 14(12):2462–2466, Dec 2017.

[132] James E Palmer, H Andrew Harms, Stephen J Searle, and LindaM Davis. Dvb-t passive radar signal processing. *IEEE transactions on Signal Processing*, 61(8):2116–2126, 2013.

[133] P Falcone, F Colone, C Bongioanni, and P Lombardo. Experimental results for ofdm wifi-based passive bistatic radar. In *Radar Conference, 2010 IEEE*, pages 516–521. IEEE, 2010.

[134] Fabiola Colone, Paolo Falcone, Carlo Bongioanni, and Pierfrancesco Lombardo. Wifi-based passive bistatic radar: Data processing schemes and experimental results. *IEEE Transactions on Aerospace and Electronic Systems*, 48(2):1061–1079, 2012.

[135] Bo Tan, Karl Woodbridge, and Kevin Chetty. A real-time high resolution passive wifi doppler-radar and its applications. In *Radar Conference (Radar), 2014 International*, pages 1–6. IEEE, 2014.

[136] Ronald K Pearson. Outliers in process modeling and identification. *IEEE Transactions on control systems technology*, 10(1):55–63, 2002.

[137] John D Emerson and David C Hoaglin. *Understanding robust and exploratory data analysis*. 1983.

[138] Lien Quach, Andrew M Galica, Richard N Jones, Elizabeth Procter-Gray, Brad Manor, Marian T Hannan, and Lewis A Lipsitz. The nonlinear relationship between gait speed and falls: the maintenance of balance, independent living, intel-

lect, and zest in the elderly of boston study. *Journal of the American Geriatrics Society*, 59(6):1069–1073, 2011.

[139] Carl De Boor, Carl De Boor, Etats-Unis Mathématicien, Carl De Boor, and Carl De Boor. *A practical guide to splines*, volume 27. Springer-Verlag New York, 1978.

[140] Frederick N Fritsch and Ralph E Carlson. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2):238–246, 1980.

[141] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. *Numerische mathematik*, 14(5):403–420, 1970.

[142] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[143] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696. ACM, 2009.

[144] Ettus Research. LP0965 Antenna Specification. `https://www.ettus.com/product/details/LP0965`, 2009. [Online Materials].

[145] Simple Wi-Fi Network. Yagi Antenna Specification. `https://www.simplewifi.com/products/yagi`, 2009. [Online Materials].

[146] Edimax AP. Edimax AP Specification. `https://www.edimax.com/edimax/merchandise/merchandise_detail/data/edimax/global/home_legacy_access_points/ew-7416apn/`, 2009. [Online Materials].

[147] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[148] Francesco Fioranelli, Matthew Ritchie, Alessio Balleri, and Hugh Griffiths. Practical investigation of multiband mono-and bistatic radar signatures of wind turbines. *IET Radar, Sonar & Navigation*, 11(6):909–921, 2017.

[149] Gang Li, Rui Zhang, Matthew Ritchie, and Hugh Griffiths. Sparsity-driven micro-doppler feature extraction for dynamic hand gesture recognition. *IEEE Transactions on Aerospace and Electronic Systems*, 2017.

[150] Dustin P Fairchild and Ram M Narayanan. Classification of human motions using empirical mode decomposition of human micro-doppler signatures. *IET Radar, Sonar & Navigation*, 8(5):425–434, 2014.

[151] Amy Brewster and Alessio Balleri. Extraction and analysis of micro-doppler signatures by the empirical mode decomposition. In *Radar Conference (RadarCon), 2015 IEEE*, pages 0947–0951. IEEE, 2015.

[152] Sevgi Zübeyde Gürbüz, Barış Erol, Bahri Çağlıyan, and Bürkan Tekeli. Operational assessment and adaptive selection of micro-doppler features. *IET Radar, Sonar & Navigation*, 9(9):1196–1204, 2015.

[153] Matthew Ritchie, Matthew Ash, Qingchao Chen, and Kevin Chetty. Through wall radar classification of human micro-doppler using singular value decomposition analysis. *Sensors*, 16(9):1401, 2016.

[154] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[155] Youngwook Kim and Taesup Moon. Human detection and activity classification based on micro-doppler signatures using deep convolutional neural networks. *IEEE geoscience and remote sensing letters*, 13(1):8–12, 2016.

[156] Youngwook Kim and Brian Toomajian. Hand gesture recognition using micro-doppler signatures with convolutional neural network. *IEEE Access*, 4:7125–7130, 2016.

[157] Youngwook Kim, Jinhee Park, and Taesup Moon. Classification of micro-doppler signatures of human aquatic activity through simulation and measurement using transferred learning. In *Radar Sensor Technology XXI*, volume 10188, page 101880V. International Society for Optics and Photonics, 2017.

[158] Mehmet Saygın Seyfioğlu and Sevgi Zübeyde Gürbüz. Deep neural network initialization methods for micro-doppler classification with low training sample support. *IEEE Geoscience and Remote Sensing Letters*, 14(12):2462–2466, 2017.

[159] Branka Jokanović and Moeness Amin. Fall detection using deep learning in range-doppler radars. *IEEE Transactions on Aerospace and Electronic Systems*, 54(1):180–189, 2018.

[160] Jarez S Patel, Francesco Fioranelli, Matthew Ritchie, and Hugh Griffiths. Multistatic radar classification of armed vs unarmed personnel using neural networks. *Evolving Systems*, pages 1–10, 2017.

[161] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[162] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.

[163] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[164] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[165] Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[166] Philip A Knight. The sinkhorn–knopp algorithm: convergence and applications. *SIAM Journal on Matrix Analysis and Applications*, 30(1):261–275, 2008.

[167] Aude Genevay, Gabriel Peyré, and Marco Cuturi. Learning generative models with sinkhorn divergences. *arXiv preprint arXiv:1706.00292*, 2017.

[168] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *Advances in neural information processing systems*, pages 469–477, 2016.

[169] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464*, 2017.

[170] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *arXiv preprint arXiv:1612.05424*, 2016.

[171] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pages 271–279, 2016.

[172] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

[173] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016.

[174] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[175] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

[176] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pages 2292–2300, 2013.

[177] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Adversarial representation learning for domain adaptation. *arXiv preprint arXiv:1707.01217*, 2017.

[178] Leonid Kantorovitch. On the translocation of masses. *Management Science*, 5(1):1–4, 1958.

[179] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

[180] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[181] John S Denker, WR Gardner, Hans Peter Graf, Donnie Henderson, RE Howard, W Hubbard, Lawrence D Jackel, Henry S Baird, and Isabelle Guyon. Neural network recognizer for hand-written zip code digits. In *Advances in neural information processing systems*, pages 323–331, 1989.

[182] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011.

[183] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.

[184] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

[185] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE international conference on computer vision*, pages 2200–2207, 2013.

[186] Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. *arXiv preprint arXiv:1705.00609*, 2017.

[187] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[188] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[189] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision*, pages 345–360. Springer, 2014.

[190] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[191] Anurag Arnab, Sadeep Jayasumana, Shuai Zheng, and Philip HS Torr. Higher order conditional random fields in deep neural networks. In *European Conference on Computer Vision*, pages 524–540. Springer, 2016.

[192] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2014.

[193] Chong Liu, Peng Zhao, Sheng-Jun Huang, Yuan Jiang, and Zhi-Hua Zhou. Dual set multi-label learning. 2018.

[194] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.