DOMAIN ADAPTATION USING SPARSE REPRESENTATION LEARNING TECHNIQUES

By KRITI KUMAR[†,⋆]

(PhD20116)

Under the supervision of

Prof. Angshul Majumdar [†],

Dr. M Girish Chandra[⋆]

[†] ELECTRONICS AND COMMUNICATION ENGINEERING, IIIT DELHI

[⋆] TCS RESEARCH, BANGALORE

INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI

NEW DELHI– 110020

May, 2025

Domain Adaptation Using Sparse Representation Learning Techniques

By

Kriti Kumar[†,⋆]

(PhD20116)

A Thesis

submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

[†] Electronics and Communication Engineering, IIIT Delhi

[⋆] TCS Research, Bangalore

Indraprastha Institute of Information Technology Delhi

New Delhi– 110020

May, 2025

Certificate

This is to certify that the thesis titled *Domain Adaptation Using Sparse Representation Learning Techniques* being submitted by *Kriti Kumar*[†,⋆] to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of Doctor of Philosophy, is an original research work carried out by her under our supervision. In our opinion, the thesis has reached the standard fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree or diploma.

May, 2025

Prof. Angshul Majumdar [†],

Dr. M Girish Chandra[⋆]

[†] Electronics and Communication Engineering, IIIT Delhi

[⋆] TCS Research, Bangalore

Indraprastha Institute of Information Technology Delhi

New Delhi 110020

Acknowledgements

My doctoral research has been a journey of continuous learning, and I am deeply grateful to many people who have provided their invaluable support in various ways. I take this opportunity to express my sincere appreciation for their assistance and encouragement.

First and foremost, I would like to thank my advisors, Prof. Angshul Majumdar and Dr. M Girish Chandra, whose guidance has been instrumental in the completion of this work. I am grateful for their unwavering support, motivation, vast knowledge, patience, and consistent encouragement. Their technical expertise and dedicated involvement in my research have played a significant role in shaping not only this thesis but also my personal and academic growth over the years. I am truly fortunate to have them as my advisors.

I am also thankful to Dr. Ranjitha Prasad, Dr. Abhijit Mitra, Dr. Tanaya Guha, Dr. Lino Coria, Dr. Ananda Shankar Chowdhury, and Dr. Santi P. Maity for their insightful comments and encouragement on my research work. I sincerely appreciate the Indraprastha Institute of Information Technology Delhi for providing a conducive research environment.

I am deeply grateful to my employer, Tata Consultancy Services (TCS), for providing financial support throughout my PhD journey. I am thankful to my supervisors, Dr. Arpan Pal and Dr. Balamuralidhar P. for believing in me and encouraging me throughout this research endeavour. Further, I extend my gratitude to my TCS colleagues, particularly Dr. A. Anil Kumar, for engaging in constructive technical discussions and providing invaluable feedback on my work. I also thank Dr. Tapas Chakravarty and Manjula MS for their consistent support during this period. Additionally, I am thankful to my team members Andrew Gigie, Rokkam Krishnakanth, Naibedya Pattnaik, Saurabh Sahu, and Smriti Rani for their invaluable assistance with my research.

Further, this thesis would not be complete without acknowledging the support I received from the SALSA lab@IIITD. I am grateful to all my lab friends,

including Shalini Sharma, Pooja Gupta, Anurag Goel, and Jyoti Maggu, for their motivation and support during challenging times.

I want to thank my husband, Chirag Pujara, and son, Aarav, for their continuous support, encouragement, understanding and care during this research work. Additionally, I extend my heartfelt appreciation to my family, particularly my mother (Malini Jain), brother (Siddhant Kumar), and sister-in-law (Ria Gupta), who have always believed in me and provided love and support throughout my life.

Finally, I want to extend my gratitude to everyone who supported me directly or indirectly during the completion of my Ph.D.

Kriti Kumar

Abstract

Domain Adaptation (DA) techniques facilitate knowledge transfer from labeled data of a source domain to improve model performance on partially labeled or unlabeled target domain data, where source and target data have different underlying distributions. These methods include supervised, semi-supervised, and unsupervised approaches and find applications in diverse fields like computer vision, medical image analysis, machine fault diagnosis, etc. Typically, deep learning methods outperform other approaches but require abundant data and computational resources for satisfactory results, leading to overfitting in scenarios with limited data. In many practical application scenarios, access to data is often restricted. Hence, there is a need for techniques capable of operating effectively with limited training data for both analysis and inverse problems. In contrast to deep learning, sparse representation learning-based methods do not suffer from these drawbacks and offer enhanced performance in such cases.

In this thesis, we investigate the use of sparse representation learning, employing Dictionary Learning(DL) and Transform Learning(TL), to address Unsupervised Domain Adaptation (UDA) and Supervised Domain Adapta- tion (SDA) for analysis and inverse problems with limited data. DL is a synthesis approach well-suited for subspace modeling for data/signal reconstruction. On the other hand, TL is an analysis approach that is shown to provide improved accuracy with reduced complexity and enhanced convergence compared to its DL counterparts. Thus, we employ both DL and TL frameworks to address two problems of significant relevance in industrial settings, outlined below, and provide a comparative analysis between the two frameworks.

The first problem addresses UDA for analysis tasks, with an application focus on machine inspection. Unlike existing techniques that require massive training data and consider adaptation between different working conditions of the *same machine*, our approach addresses adaptation between *different but related machines* using limited data. This is crucial for practical applications,

such as transferring the knowledge gained from labeled data of one machine (source domain) (e.g., lab setup or simulator) to a *different but related machine* (target domain) (e.g., industrial machine) for reliable diagnosis, as a significant difference exists in the data distribution of the two domains. We propose deep DL and shallow/deep TL methods to achieve UDA via subspace interpolation for generating domain invariant features along a virtual path that connects the source and target domains for cross-domain classification. We introduce novel joint optimization formulations and necessary closed-form updates for learning the source to target mapping in an unsupervised setting. Experimental results on different bearing fault datasets demonstrate the superior performance of the proposed methods, considering the challenging adaptation between *different but related machines*, even with limited data.

The second problem addresses SDA for inverse problems, with an application focus on Multi-modal Image Super-Resolution (MISR). MISR techniques aim to produce High Resolution (HR) (target domain) versions of Low Resolution (LR) (source domain) images by utilizing information from other imaging modalities serving as guidance, which share common features like boundaries, textures, edges, etc. Traditional MISR methods typically employ Convolutional Neural Networks (CNNs) with an encoder-decoder architecture that are susceptible to overfitting in scenarios with limited data. Unlike the former methods, in this work, we propose a fusion framework employing coupled TL and DL formulations that eliminates the need for a decoder network. This reduces the trainable parameters, making them suitable for data-limited scenarios. Different methods utilizing both standard and convolutional variants of DL and TL are introduced to capture the cross-modal dependencies between the two domains. Novel joint optimization formulations, solution steps, and closed-form updates are presented. Experimental results on two publicly available datasets show improved reconstruction performance of the proposed methods, both in Peak Signal to Noise Ratio (PSNR) and Structural SIMilarity (SSIM) index on most images compared to state-of-the-art techniques, even with limited training data.

Contents

Acknowledgements **i**

Abstract **iii**

List of Tables **x**

List of Figures **xii**

List of Abbreviations **xv**

List of Symbols **xx**

1 Introduction **1**

1.1 Motivation and Background 1

1.1.1 Unsupervised Domain Adaptation for Classification Problem - Machine Inspection 1

1.1.2 Supervised Domain Adaptation for Inverse Problem - MISR 3

1.2 Research Contributions 4

1.2.1 Unsupervised Domain Adaptation for Classification Problem - Machine Inspection 4

1.2.2 Supervised Domain Adaptation for Inverse Problems - MISR 6

1.3 Dissertation Organization 10

2 Unsupervised Domain Adaptation Via Subspace Interpolating Deep Dictionary Learning for Classification 13

2.1 Motivation 13

2.2 Related Work 18

2.3 Background on DL and DDL 21

2.3.1 Dictionary Learning (DL) 21

2.3.2 Deep Dictionary Learning (DDL) 22

2.4 Unsupervised Domain Adaptation Via Subspace Interpolating Deep Dictionary Learning (DDL-UDA) 23

2.4.1 Problem Definition 23

2.4.2 Proposed UDA method using Deep Dictionary Learning (DDL-UDA) 23

2.5 Results 29

2.5.1 Dataset Description 30

2.5.2 Benchmark Methods 32

2.5.3 Experimental Details 32

2.5.4 Results Discussion 34

2.6 Summary 38

3 Unsupervised Domain Adaptation Via Subspace Interpolating Transform Learning for Classification 39

3.1 Motivation 39

3.2 Background on TL and DTL 41

3.2.1 Transform Learning (TL) 42

3.2.2 Deep Transform Learning (DTL) 43

3.3 Unsupervised Domain Adaptation Via Subspace Interpolating Transform Learning 44
3.3.1 Proposed UDA method using Transform Learning (TL-UDA) 45
3.3.2 Proposed UDA method using Deep Transform Learning (DTL-UDA) 48
3.4 Results Discussion 55
3.4.1 Optimal Parameter Settings 58
3.5 Summary 63

4 Supervised Domain Adaptation Via Joint Coupled Transform Learning for Multi-Modal Image Super-Resolution 64
4.1 Motivation 64
4.2 Related Work 68
4.3 Joint Coupled Transform Learning for Multi-Modal Image Super-Resolution 70
4.3.1 Problem Definition 71
4.3.2 Proposed MISR method using Joint Coupled Transform Learning (JCTL-MISR) 71
4.3.3 Proposed MISR method using Joint Coupled Deep Transform Learning (JCDTL-MISR) 75
4.4 Results 82
4.4.1 Data Description 82
4.4.2 Benchmark Methods 82
4.4.3 Experimental Details 83
4.4.4 Results Discussion 83
4.5 Summary 88

5 Supervised Domain Adaptation Via Joint Coupled Convolutional

Dictionary Learning for Multi-Modal Image Super-Resolution 90

5.1 Motivation 90

5.2 Background on Convolutional Dictionary Learning (CDL) . . . 93

5.3 Proposed MISR method using Joint Coupled Convolutional Dictionary Learning (JCCDL-MISR) 94

5.3.1 Training Phase 94

5.3.2 Test Phase 98

5.4 Results Discussion 99

5.4.1 Optimal Parameter Settings 103

5.5 Summary 106

6 Supervised Domain Adaptation Via Convolutional Transform Learning for Multi-Modal Image Super-Resolution 107

6.1 Motivation 107

6.2 Background on CTL and DCTL 109

6.2.1 Convolutional Transform Learning (CTL) 110

6.2.2 Deep Convolutional Transform Learning (DCTL) 111

6.3 Convolutional Transform Learning for Multi-Modal Image Super-Resolution 111

6.3.1 Proposed MISR method using Convolutional Transform Learning (CTL-MISR) 111

6.3.2 Proposed MISR method using Deep Convolutional Transform Learning (DCTL-MISR) 114

6.4 Results Discussion 116

6.4.1 Optimal Parameter Settings 121

6.5 Summary 125

7 Conclusion 127

7.1 Summary of Contribution 127
7.1.1 Unsupervised Domain Adaptation Via Subspace Interpolating Deep Dictionary Learning for Classification 127
7.1.2 Unsupervised Domain Adaptation Via Subspace Interpolating Transform Learning for Classification 128
7.1.3 Supervised Domain Adaptation Via Joint Coupled Transform Learning for Multi-Modal Image Super-Resolution 129
7.1.4 Supervised Domain Adaptation Via Joint Coupled Convolutional Dictionary Learning for Multi-Modal Image Super-Resolution 129
7.1.5 Supervised Domain Adaptation Via Convolutional Transform Learning for Multi-Modal Image Super-Resolution 130
7.2 Future Work 132

References **137**

Appendices **152**

A Appendix **153**
A.1 Derivation for the closed-form updates for Chapter 2 153

List of Tables

2.1 Details of the Bearing Fault Datasets 31

2.2 Bearing Fault Classification Results (in %) - Set I 35

2.3 Bearing Fault Classification Results (in %) - Set II 36

3.1 Bearing Fault Classification Results (in %) - Set I 58

3.2 Bearing Fault Classification Results (in %) - Set II 58

4.1 MISR Results with RGB-NIR for 4× 84

4.2 MISR Results with RGB-MS for 4× 84

4.3 Reconstruction Performance with Noisy LR Images for 'Indoor 4' 87

5.1 MISR Results with RGB-NIR for 4× 100

5.2 MISR Results with RGB-MS for 4× 101

5.3 Effect of Number of Filters (M) and Atom size (k) on JCCDL-MISR 105

6.1 MISR Results with RGB-NIR for 4× 118

6.2 MISR Results with RGB-MS for 4× 118

6.3 Effect of Number of Filters (M) and Kernel Size on CTL-MISR 122

6.4 Reconstruction Performance with Noisy LR Images for 'Indoor 4' 123

6.5 Effect of Number of Layers (N) on DCTL-MISR 124

6.6 Effect of Number of Filters (M) on DCTL-MISR 125

6.7 Effect of Kernel Size on DCTL-MISR 125

List of Figures

2.1 Unsupervised Domain Adaptation via Interpolation over M Subspaces using N layer DDL (DDL-UDA) 24

2.2 CWRU Data Setup 30

2.3 PU Data Setup 31

2.4 CRB Data Setup 31

2.5 Residue on Target Data Vs. Number of Subspaces for different configurations of DDL-UDA 37

3.1 Unsupervised Domain Adaptation via Interpolation over M Subspaces using TL (TL-UDA) 45

3.2 Unsupervised Domain Adaptation using N layer DTL via Interpolation over M Subspaces (DTL-UDA) 49

3.3 Illustration of Feature Augmentation for Subspace Interpolation-based UDA Methods using TL (TL-UDA and DTL-UDA) . . . 49

3.4 Confusion Matrix for $PU \rightarrow CRB$. 59

3.5 Performance on Target Test Data Vs. DTL-UDA configurations employing different number of layers, for all adaptation scenarios. (a) $CWRU \rightarrow CRB$ (b) $CRB \rightarrow CWRU$ (c) $CWRU \rightarrow PU$ (d) $PU \rightarrow CWRU$ (e) $CRB \rightarrow PU$ (f) $PU \rightarrow CRB$. 60

3.6 Residue on Target Data Vs. Number of Subspaces for different configurations of DTL-UDA. (a) $CWRU \rightarrow PU$ (b) $CRB \rightarrow PU$. 61

3.7 Target Test Data Accuracy Vs. Number of Subspaces of DTL-UDA (N = 3). (a) $PU \rightarrow CWRU$, $CWRU \rightarrow PU$ and $PU \rightarrow CRB$. (b) $CRB \rightarrow PU$, $CRB \rightarrow CRWU$ and $CWRU \rightarrow CRB$. 62

4.1 Block Diagram of the proposed JCTL-MISR Method 72

4.2 Block Diagram of the proposed JCDTL-MISR Method 77

4.3 Visual comparison for 'Indoor 16' image of RGB-NIR dataset. The top row presents the reconstruction error map and the bottom row shows the reconstructed HR image with different methods. . 85

4.4 Visual comparison for 'Imge7' image of RGB-MS dataset. The top row presents the reconstruction error map and the bottom row shows the reconstructed HR image with different methods. . 85

4.5 Convergence Plot of JCTL-MISR Method 86

4.6 Effect of Number of Layers on JCDTL-MISR. (a) Average PSNR Vs. Number of Layers. (b) Average SSIM Vs. Number of Layers. 86

4.7 Results for 'Indoor 4' image of RGB-NIR dataset with the noisy LR image at the top and reconstructed HR image at the bottom. . 87

4.8 Performance Comparison with Noisy Test Images for JCTL-MISR and JCDTL-MISR 88

5.1 Block Diagram of the proposed JCCDL-MISR Method 96

5.2 MISR results for 'Imge7' image of RGB-MS data with the error map at the top and the reconstructed images at the bottom. . . . 102

5.3 Convergence of JCCDL-MISR for different number of filters (M) 104

5.4 Visualization of different convolutional dictionary filters of atom size (k)=8×8 learned with RGB-MS data for different number of filters (M) 105

6.1 Block Diagram of the proposed CTL-MISR Method 113

6.2 Block Diagram of the proposed DCTL-MISR Method 116

6.3 Error Maps and Reconstructed images for 'Imge6' image of RGB-MS dataset 120
6.4 Convergence Plot of the CTL-MISR Method 121
6.5 Results for 'Indoor 4' image of RGB-NIR dataset with the noisy LR image at the top and reconstructed HR image at the bottom. . 123
6.6 Performance Comparison with Noisy Test Images 123

7.1 UDA via Interpolation using Domain-adaptive TL 133
7.2 Block Diagram of the deep JCCDL-MISR Method 134
7.3 Block Diagram of the TL-based Semi-Supervised DA 134

List of Abbreviations

UDA Unsupervised Domain Adaptation 1

SDA Supervised Domain Adaptation 1

MISR Multi-modal Image Super-Resolution 1

DA Domain Adaptation 2

MS Multispectral 3

NIR Near Infrared 3

PAN Panchromatic 3

HS Hyperspectral 3

HR High Resolution 3

LR Low Resolution 3

DL Dictionary Learning 4

TL Transform Learning 4

DDL Deep Dictionary Learning 4

DDL-UDA DDL for UDA 4

RMS Root Mean Square 33

TL-UDA TL for UDA 5

DTL Deep Transform Learning 5

DTL-UDA DTL for UDA 5

JCTL-MISR Joint Coupled Transform Learning for MISR 6

JCDTL-MISR Joint Coupled Deep Transform Learning for MISR . . . 6

PSNR Peak Signal to Noise Ratio 7

SSIM Structural SIMilarity 7

CDL Convolutional Dictionary Learning 8

CNNs Convolutional Neural Networks 8

JCCDL-MISR Joint Coupled Convolutional Dictionary Learning for MISR 8

CTL-MISR Convolutional Transform learning for MISR 8

DCTL-MISR Deep Convolutional Transform learning for MISR 8

CTL Convolutional Transform Learning 8

PCA Principle Component Analysis 15

MOD Method of Optimal Directions 21

OMP Orthogonal Matching Pursuit 25

AM Alternating Minimization 26

SVM Support Vector Machine 28

CWRU Case Western Reserve University 29

PU Paderborn University 29

CRB Cylindrical Roller Bearing 29

IF Inner-race Fault 30

OF Outer-race Fault 30

BF Bearing-race Fault 30

EDM Electro-Discharge Machining 30

H Healthy 30

RF Roller Fault 31

MK-MMD Multi Kernel Maximum Mean Discrepancy 32

JMMD Joint Maximum Mean Discrepancy 32

CORAL CORrelation ALignment 32

DANN Domain Adversarial Neural Network 32

CDAN Conditional Domain Adversarial Network 32

DCNN Deep Convolutional Neural Network 32

Acc Accuracy 34

P Precision 34

R Recall 34

F1 F1-score 34

ADMM Alternating Direction Method of Multipliers 44

ReLU Rectified Linear Unit 50

ISR Image Super-Resolution 64

JMDL Joint Multi-modal Dictionary Learning 66

Coupled DL Coupled Dictionary Learning 69

ISTA Iterative Shrinkage and Thresholding Algorithm 70

JBF Joint Bilateral Filtering 82

GF Guided image Filtering 82

JR Joint image Restoration 82

DJF Deep Joint image Filtering 82

MCDL Multi-modal Convolutional Dictionary Learning 90

DFT Discrete Fourier Transform 93

CCMOD Convolutional Constrained Method of Optimal Direction . . . 93

SGNet Structure Guided Network 99

Adam Adaptive Moment Estimation 113

Pix to Pix Guided Pixel to Pixel Transformation 117

List of Symbols

$\boldsymbol{D}$ Dictionary

$\boldsymbol{T}$ Transform

$\boldsymbol{Z}$ Dictionary/transform coefficients

$\mathcal{S}$ Source domain

$\mathcal{T}$ Target domain

$\boldsymbol{X}_s$ Source data

$\boldsymbol{X}_t$ Target data

$\boldsymbol{Y}_s$ Labels of source data

$\boldsymbol{Y}_t$ Labels of target data

d Features

k Number of dictionary/transform atoms

n_s Number of source measurements

n_t Number of target measurements

$\boldsymbol{D}_0$ Source dictionary

$\boldsymbol{T}_0$ Source transform

$\boldsymbol{Z}_0$ Source dictionary/transform coefficients

$\boldsymbol{D}_m$ m^{th} subspace dictionary

$\boldsymbol{T}_m$ m^{th} subspace transform

$\boldsymbol{\Delta D}_m$ Change in m^{th} subspace dictionary

$\boldsymbol{\Delta T}_m$ Change in m^{th} subspace transform

$\boldsymbol{Z}_m$ m^{th} subspace dictionary/transform coefficients

$\boldsymbol{D}_M$ Target dictionary

$\boldsymbol{T}_M$ Target transform

$\boldsymbol{Z}_M$ Target dictionary/transform coefficients

M Number of subspaces

N Number of layers

$\boldsymbol{J}_m$ Residue on target data

$\boldsymbol{X}$ LR images of target modality

$\boldsymbol{Y}$ HR images of guidance modality

$\boldsymbol{Z}$ HR images of target modality

$\boldsymbol{T}_X$ Transform for $\boldsymbol{X}$ modality

$\boldsymbol{T}_Y$ Transform for $\boldsymbol{Y}$ modality

$\boldsymbol{T}_Z$ Transform for $\boldsymbol{Z}$ modality

$\boldsymbol{H}_X$ Coefficient for $\boldsymbol{X}$ modality

$\boldsymbol{H}_Y$ Coefficient for $\boldsymbol{Y}$ modality

$\boldsymbol{H}_Z$ Coefficient for $\boldsymbol{Z}$ modality

$\boldsymbol{W}_X$ Weight matrix for $\boldsymbol{X}$ modality

$\boldsymbol{W}_Y$ Weight matrix for $\boldsymbol{Y}$ modality

$\boldsymbol{S}$ Convolutional dictionary/transform for $\boldsymbol{X}$ modality in matrix form

$\boldsymbol{G}$ Convolutional dictionary/transform for $\boldsymbol{Y}$ modality in matrix form

$\boldsymbol{A}$ Convolutional dictionary/transform coefficients for $\boldsymbol{X}$ modality in matrix form

$\boldsymbol{B}$ Convolutional dictionary/transform coefficients for $\boldsymbol{Y}$ modality in matrix form

$\boldsymbol{W}$ Convolutional coupled dictionary for $\boldsymbol{X}$ modality in matrix form

$\boldsymbol{V}$ Convolutional coupled dictionary for $\boldsymbol{Y}$ modality in matrix form

$\boldsymbol{T}_f$ Fusing transform

Chapter 1

Introduction

1.1 Motivation and Background

In this thesis, we focus on two problems that are of prime relevance in industry settings, including their importance to our organization. These problems provide ample scope for systematic analytical examination and experimentation with real-life data, culminating in useful results. These are: (i) Unsupervised Domain Adaptation (UDA), specifically addressing machine inspection application scenario; (ii) Supervised Domain Adaptation (SDA), specifically addressing Multi-modal Image Super-Resolution (MISR). A brief description of the problem statements is presented below.

1.1.1 Unsupervised Domain Adaptation for Classification Problem - Machine Inspection

With the advent of Industry 4.0, there has been a lot of interest in prognostics and health monitoring of industrial machines. Health monitoring of machines ensures their reliable operation and helps maximize throughput by outage prevention. In most practical machine inspection scenarios, the train and test data do not necessarily follow a similar distribution. Various factors like changes in speed, torque, sensor placement, bearing/gearbox specifications, working

environment, etc., can introduce a domain shift or discrepancy between the train (source domain) and test (target domain) data. Additionally, access to data is limited; moreover, labeled data is difficult to collect as faults are rare events, and introducing faults is laborious and economically challenging (not a viable solution) in practical industrial scenarios. Further, manual labeling is costly and requires domain expertise. These factors render the traditional methods unsuitable for real-life practical application scenarios as they fail to address the domain discrepancies in the data.

To address these scenarios, different Domain Adaptation (DA) techniques exist for UDA that learn more general diagnosis by utilizing labeled samples from multiple source domains to ensure reliable performance on the target domain. However, the existing UDA methods consider adaptation between different working conditions of the same machine. They do not consider the scenario of adaptation between different but related machines, which is often desired in practice due to the unavailability of labeled data for every machine. This is crucial for practical applications, such as transferring the knowledge gained from labeled data of one machine (e.g., lab setup or simulator) to a different but related machine (e.g., industrial machine) for reliable diagnosis. This scenario is more challenging as a significant difference exists in the data distribution of the two domains. Moreover, the existing techniques require massive training data and compute resources for optimal performance that cannot be ensured in practical application scenarios. Hence, there is a need for methods that can address UDA between different but related machines using limited data.

1.1.2 Supervised Domain Adaptation for Inverse Problem - MISR

With multi-modal imaging systems in place, many real-world applications often involve processing data from diverse imaging modalities like Multispectral (MS), Near Infrared (NIR), and RGB, each capturing different aspects of the same scene. Although such systems capture enriched sources of information for the scene of interest, factors such as cost, design, complexity, and data storage pose significant limitations. For example, in remote sensing, satellite imaging systems capture information from different modalities, such as Panchromatic (PAN) and MS/Hyperspectral (HS) bands, but at different spatial and spectral resolutions. Usually, the RGB images will have a high spatial and low spectral resolution and vice-versa for the hyperspectral imaging system. This is done by taking into account the memory constraints, design complexity, communication, and processing challenges. Thus, there is a need to use the information from multiple modalities to overcome the resolution limitation of the targeted modality for improved performance on several downstream applications. Hence, MISR techniques are required to enhance the spatial/spectral resolution of the images of the target modality, taking help from High Resolution (HR) images of guidance modality that share common features like textures, edges, and other structures. Here, a mapping is learned between the Low Resolution (LR) images of the target modality (source domain) and HR images of the target modality (target domain), guided by the HR images of the guidance modality in a supervised setting and hence falls under the category of SDA.

Despite various techniques, fusing images from different modalities is not

trivial as the correlation among images varies significantly for each multi-modal pair, making it an ill-posed problem. The problem becomes more challenging when the data is limited, as obtaining the HR target and guidance images for training is difficult in many practical application scenarios, particularly in remote sensing. Existing learning-based methods employing deep learning offer superior performance compared to other methods but usually require abundant training data and substantial computational resources to achieve satisfactory reconstruction. They are prone to overfitting in scenarios with limited data. Hence, there is a need for methods that work with limited training data for MISR.

1.2 Research Contributions

In this work, we employ sparse representation learning techniques using Dictionary Learning (DL) and Transform Learning (TL) to address the DA problems mentioned above with limited data. Different formulations are proposed employing shallow (single-layer)/deep variants and non-convolutional/convolutional variants of DL and TL. Brief details on the contributions are presented below.

1.2.1 Unsupervised Domain Adaptation for Classification Problem - Machine Inspection

We propose UDA via subspace interpolation using Deep Dictionary Learning (DDL), referred to as DDL for UDA (DDL-UDA). The source and target data are modeled using deep dictionaries, and subspace interpolation is employed to learn intermediate domains along a virtual path connecting the source and target domains that capture the domain shift. The source-to-target mapping thus learned is used to generate a shared feature space (domain-invariant features)

along the source, intermediate, and target domains for cross-domain analysis. Using the knowledge of data labels of the source domain, a separate classifier is trained on the domain-invariant features and later used to predict the unknown target labels. It is a deep extension of [1], using deep dictionaries that learn rich representations from the data and hence are used for learning the source-to-target mapping to capture the domain shift in an unsupervised setting. The proposed method is evaluated for the challenging adaptation between different but related machines. The application focuses on bearing fault classification since bearings are critical elements for all rotating equipment in machines. They are often used under extreme loads, making them vulnerable to damage and hence require continuous monitoring to avoid breakdown of the machine [2]. Experimental results obtained with three publicly available bearing fault datasets are promising; the proposed method significantly outperforms all the state-of-the-art techniques.

Next, we propose TL and Deep Transform Learning (DTL) based subspace interpolation for UDA. These methods are referred to as TL for UDA (TL-UDA) and DTL for UDA (DTL-UDA), utilizing single-layer and deep transforms, respectively. By learning single-layer/deep transforms to model the source and target domains, and interpolating intermediate domains; domain-invariant features are generated for cross-domain classification. The classifier learning is similar to the previous case. Unlike dictionary-based subspace interpolation methods, the proposed TL methods provide improved performance with reduced complexity due to the inherent benefits of TL. Experimental results on the same bearing fault datasets demonstrate the superior performance of this method,

even with limited data. An accuracy improvement of $\approx 5\%$ (or more in some cases) is reported over the best-performing competing techniques for most adaptation cases, considering the challenging adaptation between different but related machines, which is essential in real industrial applications.

Following are the publications related to the contributions towards the first research problem:

1. K. Kumar, A. Majumdar, A. Anil Kumar, M. G. Chandra, 'Unsupervised Domain Adaptation Via Subspace Interpolating Deep Dictionary Learning: A Case Study in Machine Inspection', *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Rhodes Island, Greece, 2023, pp. 1-5.

2. K. Kumar, A. Majumdar, A. Anil Kumar, M. G. Chandra, 'Transform Based Subspace Interpolation for Unsupervised Domain Adaptation Applied to Machine Inspection', *31st European Signal Processing Conference (EUSIPCO)*, Helsinki, Finland, 2023, pp. 1708-1712.

3. K. Kumar, A. Majumdar, A. Anil Kumar, and M. G. Chandra, Unsupervised Domain Adaptation for Machine Fault Diagnosis via Subspace Interpolation using Deep Transforms', in *IEEE Sensors Journal*, vol. 24, no. 13, pp. 20959-20969, July, 2024.

1.2.2 Supervised Domain Adaptation for Inverse Problems - MISR

We introduce two novel joint optimization frameworks based on TL and DTL, referred to as Joint Coupled Transform Learning for MISR (JCTL-MISR) and Joint Coupled Deep Transform Learning for MISR (JCDTL-MISR), to combine the

information from multiple modalities for generating the HR image of the target modality. Transforms are learned for the individual modalities, i.e., LR images of target modality, HR images of guidance modality and HR images of target modality. Since the different modalities image the same scene of interest, the sparse transform coefficients of the target HR image are modeled as a weighted superposition of the sparse transform coefficients of the LR image and the HR image of the target and guidance, respectively. This captures the cross-modal relationship between the different modalities. Given the HR image of the target modality in the training phase, JCTL-MISR/JCDTL-MISR methods learn the modality-specific single-layer/deep transforms, corresponding coefficients, and the appropriate weights using a novel joint optimization formulation presented in this work. In the testing phase, with the help of the learned transforms and weights, the HR image is reconstructed by feeding its corresponding LR image of the target modality and the HR image of the guidance modality. All the necessary intermediate steps and the corresponding closed-form solution updates are provided. The performance of the proposed methods is benchmarked against the state-of-the-art MISR approaches on publicly available RGB-NIR and RGB-MS datasets. Here, RGB images are used as a guidance modality to enhance the resolution of the NIR/MS images, which serve as the target modality. Results show better performance with the proposed JCTL-MISR approach compared to other state-of-the-art techniques and the dictionary variants in Peak Signal to Noise Ratio (PSNR) and Structural SIMilarity (SSIM) metrics.

Next, we address the MISR problem using convolutional dictionaries to

explore the performance enhancement with Convolutional Dictionary Learning (CDL) over the non-convolutional DL variant. Convolutional dictionaries are translation-invariant and overcome the limitations of traditional patch-based dictionaries and provide superior performance. A joint optimization is presented that learns a convolutional dictionary for each modality, i.e., LR images of target modality and HR images of guidance modality with the respective sparse coefficients constrained to reconstruct the HR images of target modality, using two coupling convolutional dictionaries that model the cross-modal dependencies between the two modalities. The effectiveness of the proposed Joint Coupled Convolutional Dictionary Learning for MISR (JCCDL-MISR) is demonstrated using the same RGB-NIR and RGB-MS datasets. Experimental results show improved reconstruction performance of the proposed method compared to the JCTL-MISR and JCDTL-MISR methods. Additionally, PSNR (> 1 dB) and SSIM index (> 1%) is observed on most images compared to state-of-the-art techniques, even with limited training data.

Next, we propose two novel Convolutional Transform Learning (CTL)-based formulations for MISR, utilizing single-layer and deep convolutional transforms, referred to as Convolutional Transform learning for MISR (CTL-MISR) and Deep Convolutional Transform learning for MISR (DCTL-MISR), respectively. Traditional MISR approaches using Convolutional Neural Networks (CNNs) typically employ an encoder-decoder architecture, which is prone to overfit in data-limited scenarios. A fusion framework is proposed in this work that eliminates the need for a decoder network, thereby reducing the trainable param-

eters and enhancing the suitability for data-limited application scenarios. Two joint optimization frameworks are introduced that learn single-layer/deep convolutional transforms for the LR images of the target modality and HR images of the guidance modality, along with a fusion transform that combines these transform features to reconstruct HR images of the target modality. In contrast to dictionary-based synthesis sparse coding methods for MISR, the proposed methods offer improved performance with reduced complexity, leveraging the inherent advantages of transform learning. The efficacy of the proposed method is demonstrated using RGB-NIR and RGB-MS datasets, showing superior reconstruction performance compared to state-of-the-art techniques, including the DL and convolutional DL variant, without introducing additional artifacts from the guidance image.

Following are the publications related to the contributions towards the second research problem:

1. A. Gigie, A. Anil Kumar, A. Majumdar, K. Kumar, M. G. Chandra, 'Joint Coupled Transform Learning Framework for Multimodal Image Super-Resolution', *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Toronto, Canada, 2021, pp. 1640-1644.

2. R. Krishna Kanth, A. Gigie, K. Kumar, A. Anil Kumar, A. Majumdar, Balamuralidhar P., 'Multi-modal Image Super-resolution with Joint Coupled Deep Transform Learning', *30th European Signal Processing Conference (EUSIPCO)*, Belgrade, Serbia, 2022, pp. 474-478.

3. K. Kumar, A. Majumdar, S. Sahu, A. Anil Kumar, M. G. Chandra, 'Joint

Coupled Convolutional Dictionary Learning for Multi-modal Image Super-Resolution', in *IEEE Sensors Letters*, vol. 9, no. 2, pp. 1-4, Feb. 2025.

4. K. Kumar, A. Majumdar, A. Anil Kumar, M. G. Chandra, 'Convolutional Analysis Sparse Coding for Multi-modal Image Super-Resolution', in *IEEE Sensors Letters*, vol. 8, no. 6, pp. 1-4, June 2024.

5. K. Kumar, A. Majumdar, A. Anil Kumar, M. G. Chandra, 'Multi-modal Image Super-Resolution via Deep Convolutional Transform Learning', *32nd European Signal Processing Conference (EUSIPCO)*, Lyon, France, 2024, pp. 671-675.

1.3 Dissertation Organization

Towards providing the details of the research contributions to the above-mentioned problems, the rest of the dissertation is structured as follows:

- Contributions towards the first research problem are presented in Chapter 2 and Chapter 3, discussing the proposed UDA formulations using DL and TL frameworks, respectively. Since both methods focus on the same problem, the related literature on UDA for machine inspection, problem definition, details of the datasets, and benchmark methods used for performance evaluation are common for both chapters. These details are covered in Chapter 2, Section 2.2, 2.4.1, 2.5.1 and 2.5.2, respectively. The proposed DDL-UDA method is discussed in Chapter 2, Section 2.4.2 with the relevant background on DL in Section 2.3, experimental results in Section 2.5 and work summary in Section 2.6.

  The details of the proposed TL-UDA and DTL-UDA methods are presented

in Chapter 3, Section 3.3.1 and 3.3.2, respectively, with the relevant background on TL in Section 3.2. The experimental results obtained with the proposed methods and work summary are presented in Section 3.4 and 3.5.

- Contributions towards the second research problem are presented in Chapter 4, Chapter 5 and Chapter 6, discussing the proposed SDA formulations using coupled TL (JCTL-MISR and JCDTL-MISR), coupled convolutional DL (JCCDL-MISR) and convolutional TL (CTL-MISR and DCTL-MISR) frameworks, respectively. Since all the methods focus on the same problem, the related literature on MISR, problem definition, details of the datasets, and benchmark methods used for performance evaluation are common for all the chapters. These details are covered in Chapter 4, Section 4.2, 4.3.1, 4.4.1 and 4.4.2, respectively. Some additional relevant benchmarks are considered in the respective chapters. The proposed JCTL-MISR method is discussed in Chapter 4, Section 4.3.2 with the details of its deep variant (JCDTL-MISR) in Section 4.3.3, experimental results in Section 4.4 and work summary in 4.5.

  The details of the proposed JCCDL-MISR are presented in Chapter 5, Section 5.3 with the relevant background on CDL in Section 5.2, experimental results in Section 5.4 and summary in Section 5.5.

  The details associated with the proposed CTL-MISR and DCTL-MISR methods are presented in Chapter 6, Section 6.3.1 and 6.3.2, respectively with the relevant background on CTL in Section 6.2. The experimental results obtained with the proposed methods and work summary are presented

in Section 6.4 and Section 6.5.

- A summary of all the research contributions are presented in Chapter 7 Section 7.1 with some ideas for future work in Section 7.2.

Chapter 2

Unsupervised Domain Adaptation Via Subspace Interpolating Deep Dictionary Learning for Classification

2.1 Motivation

With the advent of Industry 4.0, there has been a lot of interest in prognostics and health monitoring of industrial machines. Health monitoring of machines ensures their reliable operation and helps maximize throughput by outage prevention [3, 4]. Of late, many deep learning techniques have been developed that use deep representation learning to extract rich information from machine data, both for reliable fault diagnostics [5] and prognostics by scheduling dynamic predictive maintenance [6]. However, these techniques assume training and test data to follow similar distribution and need massive amount of labeled data for training. In most practical machine inspection scenarios, access to data is limited; moreover, labeled data is difficult to collect as faults are rare events. Additionally, the train and test data do not necessarily follow a similar distribution. Various factors like changes in speed, torque, sensor placement, bearing/gearbox specifications, working environment, etc., can introduce a domain shift or discrepancy between

the train (source) and test (target) data. These factors render the traditional methods unsuitable for real-life practical application scenarios and necessitate DA techniques to ensure reliable performance.

DA techniques come under the broad category of transfer learning techniques that help in transferring knowledge learned from sufficiently labeled data of the source domain (training) to the target domain (test) data, where source and target data have different underlying distributions but cater to the same application (or task) [7]. These techniques are categorized into supervised, semi-supervised and unsupervised, based on the availability of target domain labels [8]. DA have been successfully employed for various applications in different domains like computer vision [9], medical image analysis [10], and recently machine fault diagnosis [11–18]. This work focuses on UDA applied to machine inspection.

Quite recently, different algorithms have been developed based on divergence [11, 13, 14, 19], adversarial learning [15, 16, 20] and subspace-based methods [1, 21–26], for adapting the source and target domain data in an unsupervised manner for different application domains. The divergence and adversarial learning based methods align the source and target data distribution by minimizing the divergence and adversarial learning objectives, respectively. However, they require massive data for training, making them unsuitable for real-life applications with limited data. In contrast, the subspace-based methods seem to work with limited data [27]. These techniques exploit the fact that high-dimensional data often resides in a low-dimensional subspace. Subspace-based methods have been successfully applied for feature augmentation in computer vision applications

[1, 21–26]. They focus on generating discriminative features that are common (or invariant) across both the source and target domains, enabling cross-domain classification. They essentially generate intermediate feature representations along a virtual path connecting the source and target domains. As opposed to the domain subspaces obtained using Principle Component Analysis (PCA) that may not represent the original data well and result in information loss, data-driven dictionaries with non-orthogonal atoms (columns) have been proposed that provide more flexibility to model and adapt the domain data [1, 23–26].

Motivated by the advantages of DL, this work examines the performance of deep DL for UDA. Although the proposed method is generic and can be applied to different domains, the application focus of the current work is on machine inspection, specifically bearing fault diagnosis since bearings are critical elements for all rotating equipment in machines. They are often used under extreme loads, making them vulnerable to damage [2]. Unlike existing UDA techniques on bearing fault classification described in Section 2.2 that require massive training data and consider adaptation between different working conditions of the *same machine*, our approach addresses adaptation between *different but related machines* using limited data. This is crucial for practical applications, such as transferring the knowledge gained from labeled data of one machine (e.g., lab setup or simulator) to a different but related machine (e.g., industrial machine) for reliable diagnosis, as a significant change exists in the data distribution of the two domains [28].

This work presents a novel deep extension of the DL work in [1] for subspace

interpolation for UDA task, referred to as DDL-UDA. In this work, source and target data are modeled using deep dictionaries, and subspace interpolation is employed to learn intermediate domains along a virtual path connecting the source and target domains that capture the domain shift. Subsequently, domain-invariant features are generated along the source, intermediate, and target domains for classification. Unlike the prior work in [1], where subspaces are modeled using single-layer dictionaries for face recognition tasks, here, multi-layer dictionaries are employed for subspace modeling for data-limited UDA to specifically adapt the data between *different but related machines* for fault diagnosis. Deep dictionaries are able to learn robust domain-invariant representations from source and target domains resulting in improved cross-domain analysis. Kindly note that the problem formulation and solution steps differ from the shallow version [1]. The requisite problem formulation for the deep version, the solution steps, and closed-form updates are systematically derived and presented. Experimental results obtained with the publicly available bearing fault datasets [29–31] for the challenging adaptation between *different but related machines* are promising. The results show that the proposed DDL-UDA outperforms all state-of-the-art methods, demonstrating its applicability and potential for adaptation tasks.

The contributions of the work are summarized below:

- Unlike, the work in [1] that considers single-level dictionaries, we use deep dictionaries for subspace modeling, and interpolation is carried out to

connect the source and target domain that enables us to learn more effective representations for cross-domain analysis.

- Novel DDL-UDA formulation is presented with the systematically derived closed-form solutions for the associated multi-layer dictionaries and coefficients.
- This method is applied to time series data for machine inspection scenarios, unlike the work in [1], which considers images for face/object recognition applications.
- The challenging adaptation between *different but related machines* is addressed for inspection scenarios. Extensive comparative analysis against the benchmark techniques using publicly available bearing datasets demonstrates the potential of the proposed method for UDA tasks.

Towards providing the necessary details, the rest of the chapter is organized as follows. Section 2.2 covers related works on UDA for machine inspection and subspace-based UDA methods. Section 2.3 provides a brief background on DL and its deep variant DDL, which forms the basis of the proposed method. This is followed by Section 2.4, which presents the problem definition and the details of the proposed DDL-UDA method. Description of the datasets, benchmark methods used for comparison and experimental results are presented in Section 2.5. Subsequently, Section 2.6 presents the summary and future work.

2.2 Related Work

Most of the existing UDA techniques for bearing fault diagnosis are based on divergence and adversarial learning. Divergence-based methods align the source and target data by minimizing the divergence criteria to extract domain-invariant features. The divergence between the source and target data is specified by the respective methods like Joint Maximum Mean Discrepancy [11], Multi Kernels Maximum Mean Discrepancy [13, 19] and Correlation Alignment [14]. On the other hand, adversarial learning-based methods achieve the alignment between the source and target data through an adversarial objective with respect to a domain discriminator to obtain domain-invariant features. Methods like Domain Adversarial Neural Network [15] and Conditional Domain Adversarial Network [16, 20] fall under this category. A comprehensive study of the above-mentioned methods on issues like transferability of features, the influence of backbones, negative transfer, and physical prior is presented in [18]. Another recent review on deep learning-based DA techniques is given in [32], which provides guidelines for selecting the source domain data, data transformation, and the transfer learning model based on the problem encountered in bearing fault diagnosis. However, all the above methods need enormous training data for good performance. Moreover, they focus on adaptation between different working conditions of the *same machine*. They do not address adaptation between *different but related machines*, which is required in practice, given the unavailability of labeled data for each machine in most practical application scenarios. The work presented in [33] tries to address this problem by developing a high-fidelity digital twin and transfer-

ring the knowledge to real physical structures, such as gearboxes. However, to build such a digital twin, in addition to domain expertise, details of the physical structure are required, which is difficult to obtain in practice. To overcome the above-mentioned limitations, we propose a subspace-based method that adapts between the data collected from two physically distinct but related machines for reliable machine diagnosis without needing explicit details of the machine's physical structure and can work with limited data.

Unlike most of the former methods, instead of minimizing the divergence and adversarial objective, subspace-based methods align the two domains by learning a representation that can be used to classify the labeled source domain data well and can also be used to reconstruct both the source and target domain data. Subspace-based methods belong to the category of feature augmentation approaches for DA. They generate discriminative features which are invariant across both the source and target domains for cross-domain classification. They have been successfully applied in computer vision for UDA tasks [1, 21–26]. Different techniques have been proposed that focus on generating features (representations) along the virtual path connecting the two domains. The work in [21] proposed a method that generates intermediate subspaces by sampling the geodesic path connecting the source and target domain on the Grassmann manifold. This was improved by [22], which proposed a geodesic flow kernel to model the domain shift by integrating an infinite number of subspaces, characterizing the changes in geometric and statistical properties from the source to the target domain. However, as indicated earlier, the domain subspaces obtained

using PCA may result in information loss. To overcome this limitation, several works [1, 23–26] proposed DL-based techniques for subspace modeling as they offer more flexibility for modeling subspaces and offer improved adaptation results. The work in [23] focused on learning a parametric dictionary by aligning dictionaries from both domains. Another work [24] utilized joint learning of the data projections of the two domains along with a latent dictionary capable of representing both domains in the projected low-dimensional space. The work in [1] focused on generating a set of intermediate domains modeled using dictionaries that smoothly adapt the source domain to the target domain via subspace interpolation. This work was extended in [25] to generate domain-invariant features by learning a common dictionary and domain-specific dictionaries for subspace modeling to capture the domain shift. It was shown that separating the common and domain-specific dictionaries resulted in more compact and reconstructive dictionaries for domain adaptation. Another work [26] proposed a domain-shared group-sparse dictionary learning method for joint alignment of conditional and marginal distributions for domain adaptation. However, as mentioned earlier, all these methods focus on images and have not been tried for time series data which is the focus of our work.

In recent times, deep networks have gained popularity in many applications owing to their better representation learning capability. DDL models with multiple layers of dictionaries have the potential to learn rich data representation and have been shown to outperform the shallow (single-layer) DL methods [34, 35]. Motivated by the performance of deep networks, the proposed work consid-

ers a subspace interpolation-based method for UDA similar to the approach considered in [1], where subspaces are modeled using shallow (single-layer) dictionaries for face recognition tasks. In this work, source and target data are modeled using deep dictionaries, and subspace interpolation is employed to learn intermediate domains along a virtual path connecting the source and target domains that capture the domain shift for machine inspection scenarios. Subsequently, domain-invariant features are generated along the source, intermediate, and target domains for classification.

2.3 Background on DL and DDL

This section presents a brief background on shallow (single-layer) and deep variants of DL that forms the backbone of our proposed method.

2.3.1 Dictionary Learning (DL)

Given the data $\boldsymbol{X} \in \mathbb{R}^{d \times n}$ with d features of length n, $\boldsymbol{X}$ can be represented as: $\boldsymbol{X} = \boldsymbol{D}\boldsymbol{Z}$ where $\boldsymbol{D} \in \mathbb{R}^{d \times k}$ is the learnt dictionary or basis that contains k atoms as its columns and $\boldsymbol{Z} \in \mathbb{R}^{k \times n}$ are the associated coefficients. This is an unconstrained DL problem that is similar to matrix factorization and can be solved using Method of Optimal Directions (MOD) [36] by alternately solving for the two variables:

$$\min_{\boldsymbol{D},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 \tag{2.1}$$

For learning sparse representations from the data, $\boldsymbol{Z}$ is constrained to be sparse using the l_0-norm constraint resulting in the following:

$$\min_{\boldsymbol{D},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 \;\; s.t. \; \|\boldsymbol{Z}\|_0 \leq \mu \tag{2.2}$$

This ensures a sparse representation of data is learnt with maximum μ non-zero entries of $\boldsymbol{Z}$. K-SVD [37] is a popular method used for learning dictionary with sparse coefficients.

2.3.2 Deep Dictionary Learning (DDL)

A deeper variant of DL was proposed in [34, 35] which involves learning multiple layers of dictionaries that are cascaded together to learn rich representations from the data. For a 3 layer deep architecture, the DDL formulation is given as [34]:

$$\boldsymbol{X} = \boldsymbol{D}^1\phi(\boldsymbol{D}^2\phi(\boldsymbol{D}^3\boldsymbol{Z})) \tag{2.3}$$

where ϕ is the non-linear activation function between different layers. A greedy approach to solve for the dictionary and coefficients is presented in [34]. The greedy approach learns one layer of dictionary at a time. The first layer of dictionary learns from the training data. The coefficients learned from the first layer of dictionary are passed as input in the second layer of dictionary to learn the second layer dictionary atoms and the coefficients. Following the same concept, the coefficients learned from the second layer of dictionary are passed as input to the third layer of dictionary to learn the third layer dictionary atoms and the coefficients. The optimization problem for learning the multi-layer dictionaries is expressed as:

$$\min_{\boldsymbol{D}^1,\boldsymbol{D}^2,\boldsymbol{D}^3,\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{D}^1\phi(\boldsymbol{D}^2\phi(\boldsymbol{D}^3\boldsymbol{Z}))\|_F^2 \;\; s.t. \; \|\boldsymbol{Z}\|_0 \leq \mu \tag{2.4}$$

with the sparsity enforced on $\boldsymbol{Z}$ using the l_0-norm constraint. It is important to note that due to the presence of ϕ, collapsing the multiple layers of dictionary into a single dictionary in (2.4) is not equivalent to the single-layer dictionary in (2.2). Both the formulations (2.2) and (2.4) are different and result in different

coefficients. The deep formulation in (2.4) can be generalized to any N layers depending on the application use case. This DDL formulation is employed in our proposed method for adaptation tasks via the subspace interpolation presented in the next section.

2.4 Unsupervised Domain Adaptation Via Subspace Interpolating Deep Dictionary Learning (DDL-UDA)

This section presents the problem formulation and the details of the proposed subspace interpolation method using Deep Dictionary Learning for Unsupervised Domain Adaptation (DDL-UDA).

2.4.1 Problem Definition

The problem is an unsupervised adaptation between the source domain $\mathcal{S}$ and target domain $\mathcal{T}$, where the source data $\boldsymbol{X}_s$ and target data $\boldsymbol{X}_t$ have different underlying distributions $P(X_s) \neq P(X_t)$. Let $\boldsymbol{X}_s \in \mathbb{R}^{d \times n_s}$ represent the source domain data with d features of n_s measurements and $\boldsymbol{X}_t \in \mathbb{R}^{d \times n_t}$ represent the target domain data with d features of n_t measurements. Given the labels $\boldsymbol{Y}_s$ of the source domain data $\boldsymbol{X}_s$, the objective is to estimate the unknown labels $\boldsymbol{Y}_t$ of target domain data $\boldsymbol{X}_t$, assuming the label space to be the same for both the domains.

2.4.2 Proposed UDA method using Deep Dictionary Learning (DDL-UDA)

As mentioned earlier, the proposed method is a deep extension of [1], using deep dictionaries to form a smooth transition path between $\mathcal{S}$ and $\mathcal{T}$ domains by learning the intermediate domains via subspace interpolation. Starting with

the source dictionary $\boldsymbol{D}_0 \in \mathbb{R}^{d\times k}$ trained on source domain data $\boldsymbol{X}_s$ with k atoms, intermediate dictionaries (intermediate subspaces) $\boldsymbol{D}_m, m \in [1, M-1]$ are computed by decomposing the target data $\boldsymbol{X}_t$ iteratively ($m = 0, 1, .., M$), in the direction to reduce the reconstruction residue on the target data, till we reach $\boldsymbol{D}_M$ that best represents the target domain data $\boldsymbol{X}_t$. Fig. 2.1 presents the block diagram of the proposed DDL-UDA that shows the M different subspaces modeled by N layer deep dictionaries obtained by interpolation on the target data. The intermediate dictionaries capture the domain shift between the two domains. In order to learn rich representations from the data of the two domains, all the dictionaries are made deep following the DDL formulation.

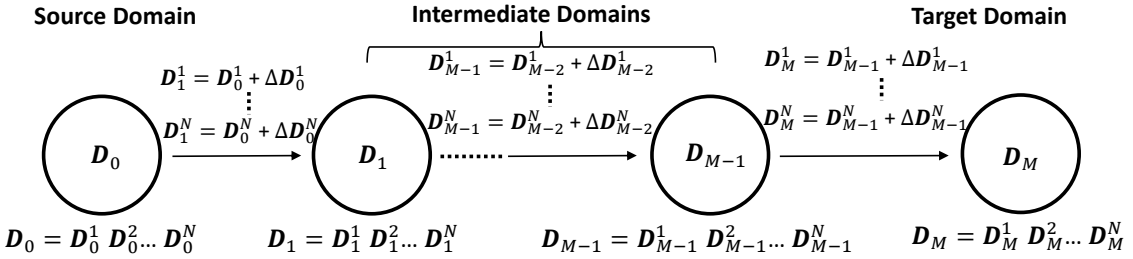


Figure 2.1: Unsupervised Domain Adaptation via Interpolation over M Subspaces using N layer DDL (DDL-UDA)

This method employs a training phase for learning the virtual path that connects $\mathcal{S}$ and $\mathcal{T}$ domains in terms of intermediate dictionaries that capture the domain shift between the two domains. Note that learning this transition path does *not* need label information and is completely unsupervised. Later, this path is used to generate domain-invariant features from the source data for learning the classifier with the knowledge of source data labels. Subsequently, in the test phase, domain-invariant features are generated from the target data using along the same virtual path and target labels are estimated with the help of the learned

classifier. More details on the two phases are presented below.

2.4.2.1 Training Phase

Here, for brevity, we provide the detailed formulation assuming $N = 3$; however, it can be generalized to any N layers. We start with the source domain dictionary $\boldsymbol{D}_0$ (m = 0), considering a 3 layer deep architecture, $\boldsymbol{D}_0 = \boldsymbol{D}_0^1\boldsymbol{D}_0^2\boldsymbol{D}_0^3$ *. The deep dictionaries for source data $\boldsymbol{X}_s$ are learnt by solving:

$$\min_{\boldsymbol{D}_0^1,\boldsymbol{D}_0^2,\boldsymbol{D}_0^3,\boldsymbol{Z}_0} \|\boldsymbol{X}_s - \boldsymbol{D}_0^1\boldsymbol{D}_0^2\boldsymbol{D}_0^3\boldsymbol{Z}_0\|_F^2, \;\; s.t. \; \|\boldsymbol{Z}_0\|_0 \leq \mu \tag{2.5}$$

Here, the coefficients $\boldsymbol{Z}_0$ can be obtained using Orthogonal Matching Pursuit (OMP) [38] and the dictionaries $\boldsymbol{D}_0^1$, $\boldsymbol{D}_0^2$, and $\boldsymbol{D}_0^3$ are updated using the standard pseudo-inverse operation. Additionally, we consider normalization for all the dictionaries in (2.5), where each of the columns in all the multi-layer dictionaries are normalized to unit norm. Subsequently, the m^{th} domain dictionary $\boldsymbol{D}_m, m \in [0, M]$ is applied on the target data $\boldsymbol{X}_t$ to generate the coefficients $\boldsymbol{Z}_m$ using OMP and the residue $\boldsymbol{J}_m$ is computed using the following:

$$\boldsymbol{Z}_m \leftarrow \min_{\boldsymbol{Z}_m} \|\boldsymbol{X}_t - \boldsymbol{D}_m^1\boldsymbol{D}_m^2\boldsymbol{D}_m^3\boldsymbol{Z}_m\|_F^2, \;\; s.t. \; \|\boldsymbol{Z}_m\|_0 \leq \mu \tag{2.6}$$

$$\boldsymbol{J}_m = \boldsymbol{X}_t - \boldsymbol{D}_m^1\boldsymbol{D}_m^2\boldsymbol{D}_m^3\boldsymbol{Z}_m \tag{2.7}$$

The new dictionary $\boldsymbol{D}_{m+1}$ is computed by estimating $\boldsymbol{\Delta D}_{m's}$ that represent the adjustment in dictionary atoms between the deep dictionaries of $\boldsymbol{D}_{m+1}$ and $\boldsymbol{D}_m$ to reduce the residue $\boldsymbol{J}_m$. The $\boldsymbol{\Delta D}_{m's}$ for 3 layer deep dictionaries are learnt using the following:

*It is important to note that non-linearity in the multiple layers of the deep dictionary is introduced via the regularization imposed on the dictionary updates through $\boldsymbol{\Delta D}_{m's}$ discussed later in (2.9), (2.11), and (2.17)

$$\min_{\boldsymbol{\Delta D}_m^1, \boldsymbol{\Delta D}_m^2, \boldsymbol{\Delta D}_m^3} \|\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m\|_F^2 + \lambda(\|\boldsymbol{\Delta D}_m^1\|_F^2 + \|\boldsymbol{\Delta D}_m^2\|_F^2 + \|\boldsymbol{\Delta D}_m^3\|_F^2) \tag{2.8}$$

where the first term reduces the residue and the second term discourages abrupt changes, thereby ensuring a smooth transition between dictionaries of adjacent domains with the help of parameter λ.

We employ Alternating Minimization (AM) approach [39] to compute the updates for $\boldsymbol{\Delta D}_{m's}$ for the multiple dictionary layers. The sub-problem for the update for $\boldsymbol{\Delta D}_m^1$ is given as:

$$\boldsymbol{\Delta D}_m^1 \leftarrow \min_{\boldsymbol{\Delta D}_m^1} \|\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m\|_F^2 + \lambda \|\boldsymbol{\Delta D}_m^1\|_F^2 \tag{2.9}$$

Expanding the above problem (2.9) in terms of trace and equating the derivative with respect to $\boldsymbol{\Delta D}_m^1$ to 0, results in the following closed-form update:

$$\boldsymbol{\Delta D}_m^1 = \boldsymbol{J}_m (\boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m)^T [(\boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m)(\boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m)^T + \lambda \boldsymbol{I}]^{-1} \tag{2.10}$$

The sub-problem for the update of $\boldsymbol{\Delta D}_m^2$ is given as:

$$\boldsymbol{\Delta D}_m^2 \leftarrow \min_{\boldsymbol{\Delta D}_m^2} \|\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m\|_F^2 + \lambda \|\boldsymbol{\Delta D}_m^2\|_F^2 \tag{2.11}$$

Using variable splitting [40] with $\boldsymbol{U} = \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m$ results in following:

$$\min_{\boldsymbol{\Delta D}_m^2, \boldsymbol{U}} \|\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{U}\|_F^2 + \gamma \|\boldsymbol{U} - \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m\|_F^2 + \lambda \|\boldsymbol{\Delta D}_m^2\|_F^2 \tag{2.12}$$

The update for $\boldsymbol{U}$ requires the solution of:

$$\boldsymbol{U} \leftarrow \min_{\boldsymbol{U}} \|\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{U}\|_F^2 + \gamma \|\boldsymbol{U} - \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m\|_F^2 \tag{2.13}$$

The closed-form update is obtained as:

$$\boldsymbol{U} = (\boldsymbol{\Delta D}_m^{1\,T} \boldsymbol{\Delta D}_m^1 + \gamma \boldsymbol{I})^{-1} (\boldsymbol{\Delta D}_m^{1\,T} \boldsymbol{J}_m + \gamma \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m) \tag{2.14}$$

The sub-problem and the associated closed-form solution of $\boldsymbol{\Delta D}_m^2$ is given as:

$$\boldsymbol{\Delta D}_m^2 \leftarrow \min_{\boldsymbol{\Delta D}_m^2} \gamma \|\boldsymbol{U} - \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m\|_F^2 + \lambda \|\boldsymbol{\Delta D}_m^2\|_F^2 \tag{2.15}$$

$$\boldsymbol{\Delta D}_m^2 = \gamma \boldsymbol{U} (\boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m)^T [\gamma (\boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m)(\boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m)^T + \lambda \boldsymbol{I}]^{-1} \tag{2.16}$$

Similarly, for $\boldsymbol{\Delta D}_m^3$, the sub-problem is given as:

$$\boldsymbol{\Delta D}_m^3 \leftarrow \min_{\boldsymbol{\Delta D}_m^3} \|\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m\|_F^2 + \lambda \|\boldsymbol{\Delta D}_m^3\|_F^2 \tag{2.17}$$

Following the same variable splitting approach as for (2.11), taking $\boldsymbol{V} = \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m$ results in the following:

$$\min_{\boldsymbol{\Delta D}_m^3, \boldsymbol{V}} \|\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2 \boldsymbol{V}\|_F^2 + \gamma \|\boldsymbol{V} - \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m\|_F^2 + \lambda \|\boldsymbol{\Delta D}_m^3\|_F^2 \tag{2.18}$$

Solving separately for $\boldsymbol{\Delta D}_m^3$ and $\boldsymbol{V}$ results in the following closed-form updates:

$$\boldsymbol{\Delta D}_m^3 = \gamma \boldsymbol{V} {\boldsymbol{Z}_m}^T (\gamma \boldsymbol{Z}_m {\boldsymbol{Z}_m}^T + \lambda \boldsymbol{I})^{-1} \tag{2.19}$$

$$\boldsymbol{V} = [(\boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2)^T (\boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2) + \gamma \boldsymbol{I}]^{-1} . ((\boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2)^T \boldsymbol{J}_m + \gamma \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m) \tag{2.20}$$

Subsequently, the deep dictionaries are updated and the new dictionary $\boldsymbol{D}_{m+1}$ is computed as:

$$\begin{aligned} \boldsymbol{D}_{m+1}^1 &= \boldsymbol{D}_m^1 + \boldsymbol{\Delta D}_m^1 \\ \boldsymbol{D}_{m+1}^2 &= \boldsymbol{D}_m^2 + \boldsymbol{\Delta D}_m^2 \\ \boldsymbol{D}_{m+1}^3 &= \boldsymbol{D}_m^3 + \boldsymbol{\Delta D}_m^3 \\ \boldsymbol{D}_{m+1} &= \boldsymbol{D}_{m+1}^1 \boldsymbol{D}_{m+1}^2 \boldsymbol{D}_{m+1}^3 \end{aligned} \tag{2.21}$$

This process is repeated iteratively till $\|\boldsymbol{\Delta D}_{m's}\|_F$ (for the multiple layers) $\leq \tau$ (empirically calculated threshold), suggesting the learnt intermediate domain dictionaries fully absorb the domain shift between the two domains. Note in each step, the residue $\boldsymbol{J}_m$ is non-increasing with respect to the current intermediate domain dictionary and the associated coefficients (proof shown in [1]). The last obtained dictionary $\boldsymbol{D}_M$ is considered as the target dictionary as it provides a good representation of the target data. The non-increasing property of the residue

$\boldsymbol{J}_m$ ensures that $\boldsymbol{D}_0$ gradually adapts to the $\boldsymbol{D}_M$ through a set of intermediate dictionaries $\boldsymbol{D}_m$, $m \in [1, M-1]$.

It is important to note that non-linearity in the multiple layers of the deep dictionary is introduced via the regularization imposed on the dictionary updates through $\boldsymbol{\Delta D}_{m's}$ as shown in (2.9), (2.11), and (2.17). Once the transition path between the two domains are learnt, invariant sparse codes are applied across the source, intermediate, and target dictionaries ($\{\boldsymbol{D}_m\}_{m=0}^{M}$) to form the new features for classification. The new feature space is obtained as: $[(\boldsymbol{D}_0\boldsymbol{Z})^T, (\boldsymbol{D}_1\boldsymbol{Z})^T, .., (\boldsymbol{D}_M\boldsymbol{Z})^T]$ where $\boldsymbol{Z} \in \mathbb{R}^k$ are the sparse codes generated either by decomposing the source data $\boldsymbol{X}_s$ with $\boldsymbol{D}_0$ (i.e., $\boldsymbol{Z}_0$) or by decomposing the target data $\boldsymbol{X}_t$ with $\boldsymbol{D}_M$ (i.e., $\boldsymbol{Z}_M$). The intermediate dictionaries recovered along the virtual path form a smooth transition in the signal space and brings the two domains into a shared feature space where the distribution shift between them is minimized.

Given the source data labels $\boldsymbol{Y}_s$, features are computed with $\boldsymbol{Z} = \boldsymbol{Z}_0$ (obtained by decomposing source data with $\boldsymbol{D}_0$) to learn a Support Vector Machine (SVM) classifier for classification task.

2.4.2.2 Test Phase

To estimate the labels $\boldsymbol{Y}_t^{test}$ associated with the target test data $\boldsymbol{X}_t^{test}$, first $\boldsymbol{Z}_M^{test}$ is obtained by decomposing target test data with $\boldsymbol{D}_M$. Subsequently, features are computed by applying $\boldsymbol{Z} = \boldsymbol{Z}_M^{test}$ across the source, intermediate and target deep dictionaries. These test features are fed to the classifier learned in the training

phase to estimate the test target labels.

Please note that for $N = 1$ the formulation becomes similar to [1]. Also, note that while we have considered SVM classifier, in general, any other suitable classifier can be used. The pseudocode of the proposed method is summarized in Algorithm 1.

Algorithm 1: Subspace Interpolation using 3 layer DDL for UDA (DDL-UDA)

Input: $\boldsymbol{X}_s$, $\boldsymbol{X}_t$
Parameters: $\lambda, \mu, \gamma, \tau, k$ (number of dictionary atoms; considered same for all the multi-layer dictionaries in this work)
Initialization: Set multi-layer dictionaries ($\boldsymbol{D}_0^1$, $\boldsymbol{D}_0^2$, $\boldsymbol{D}_0^3$) to random matrix with real numbers between 0 and 1 drawn from a uniform distribution, $m = 0$
Compute multi-layer source dictionary ($\boldsymbol{D}_0^1$, $\boldsymbol{D}_0^2$, $\boldsymbol{D}_0^3$) and $\boldsymbol{Z}_0$ with $\boldsymbol{X}_s$ using (2.5).
do
    Decompose $\boldsymbol{X}_t$ with $\boldsymbol{D}_m^1$, $\boldsymbol{D}_m^2$, and $\boldsymbol{D}_m^3$ using (2.6).
    Compute the residue $\boldsymbol{J}_m$ using (2.7).
    Estimate the adjustment in deep dictionary atoms ($\boldsymbol{\Delta D}_m^1$, $\boldsymbol{\Delta D}_m^2$, and $\boldsymbol{\Delta D}_m^3$) using (2.9), (2.11) and (2.17).
    Update the multi-layer deep dictionaries ($\boldsymbol{D}_{m+1}^1$, $\boldsymbol{D}_{m+1}^2$, $\boldsymbol{D}_{m+1}^3$, and $\boldsymbol{D}_{m+1}$) using (2.21). Normalize each column in all multi-layer dictionaries to unit norm.
    $m = m + 1$
while ($\|\boldsymbol{\Delta D}_{m's}\|_F$ (for the multiple layers) $> \tau$)
Output: $\{\boldsymbol{D}_m\}_{m=0}^{M}$ (source, intermediate and target dictionaries)

2.5 Results

The proposed interpolation-based method for machine fault diagnosis is evaluated using three publicly available bearing datasets: Case Western Reserve University (CWRU) [29], Paderborn University (PU) [30], and Cylindrical Roller Bearing (CRB) [31]. More details on the datasets, state-of-the-art methods used for benchmarking and performance evaluation metrics, along with the experimental results, are presented below.

2.5.1 Dataset Description

2.5.1.1 CWRU Dataset

This dataset consists of vibration measurements obtained from the drive end and fan end of the machine (Fig. 2.2), with a sampling frequency of 12 kHz [29]. Here, typical bearing faults like Inner-race Fault (IF), Outer-race Fault (OF), and Bearing-race Fault (BF) of three different sizes (0.007, 0.014, 0.021 inches) are created using Electro-Discharge Machining (EDM). The dataset contains Healthy (H) and faulty bearing data collected under four working conditions of loading torques 0, 1, 2, and 3 hp with speeds of 1797, 1772, 1750, and 1730 rpm, respectively.

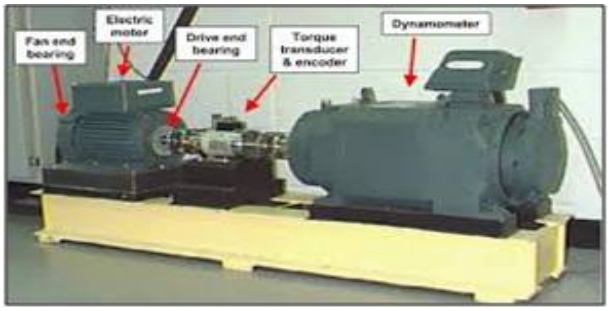


Figure 2.2: CWRU Data Setup

2.5.1.2 PU Dataset

This bearing dataset contains current and vibration data acquired at a sampling frequency of 64 kHz from a test rig that consists of a drive motor, torque measurement shaft, test modules, and a load motor [30] (Fig. 2.3). It contains data for two rotating speeds (900 and 1500 rpm) and loading torques (0.7 and 0.1 Nm). Data is collected for both H and faulty bearings with faults like IF and OF and labeled with different bearing codes. This study uses data corresponding to bearing codes K005, KI04, and KA04 for H, IF, and OF, respectively.

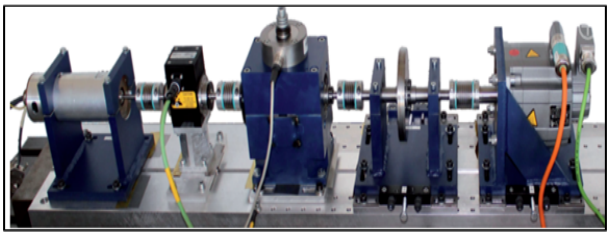


Figure 2.3: PU Data Setup

2.5.1.3 CRB Dataset

It contains vibration and acoustic data of cylindrical roller bearing (NBC: NU205E) acquired from a test rig at a sampling frequency of 70 kHz [31] (Fig. 2.4). Here, three faults, namely IF, OF and Roller Fault (RF), are created using EDM. The experiments are conducted with a shaft speed of 2050 rpm and a vertical load of 200 N. Four sets containing H and faulty bearing data with different defect widths are collected for investigation.

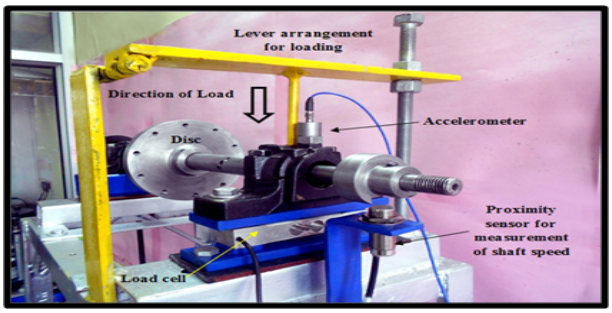


Figure 2.4: CRB Data Setup

Table 2.1: Details of the Bearing Fault Datasets

Parameters	**CWRU**	**CRB**	**PU**
Sampling Frequency (kHz)	12	70	64
Speed (rpm)	1797	2050	1500
Load	0 Hp	200 N	1000 N
Classes	H, IF, OF	H, IF, OF	H, IF, OF
Bearing Specifications			
Inner Diameter (mm)	25	25	24
Outer Diameter (mm)	52	52	33.1
Pitch Diameter (mm)	39	38.9	28.55
Rolling Element Diameter (mm)	7.94	7.5	6.75
No. of Rolling Elements	9	13	8
Defect Size (mm)	0.1778	1.01 (IF), 0.86 (OF)	–

2.5.2 Benchmark Methods

Different competing UDA methods for bearing fault diagnosis are considered for evaluating the performance of the proposed method. They are broadly classified into (i) divergence-based methods: Multi Kernel Maximum Mean Discrepancy (MK-MMD), Joint Maximum Mean Discrepancy (JMMD), CORrelation ALignment (CORAL), (ii) adversarial learning-based methods: Domain Adversarial Neural Network (DANN), Conditional Domain Adversarial Network (CDAN). These methods are implemented using a Deep Convolutional Neural Network (DCNN) backbone and bottleneck architecture as described in [18]. Given the knowledge of source labels, these methods jointly optimize the classification loss and the divergence/adversarial loss for aligning the $\mathcal{S}$ and $\mathcal{T}$ data distribution, as specified by the respective methods. Additionally, comparisons with the shallow (single-layer) DL variant (DL-UDA) [1] are also presented to demonstrate the improvement gained with the deep version. It is well demonstrated in [1] that a significant improvement is obtained with adaptation compared to no adaptation. Hence, here, we provide the results only for the adaptation case, comparing our proposed method with the aforementioned state-of-the-art techniques.

2.5.3 Experimental Details

As mentioned earlier, this work considers adaptation between *different but related machines* for *data-limited* scenario using only vibration data. Different datasets form the $\mathcal{S}$ and $\mathcal{T}$ data for this case. Table 2.1 presents the working conditions and bearing specifications of the different datasets considered in this work for

adaptation. It is to be noted that all three datasets have different operating conditions, bearing specifications, and sampling frequencies; making it a challenging adaptation task. This work considers label-consistent UDA, where the labels of $\mathcal{S}$ and $\mathcal{T}$ are consistent or the same. Hence, the labels that are common to all the datasets are only employed for adaptation and classification. With these three datasets, six combinations of $\mathcal{S} \rightarrow \mathcal{T}$ are considered for experimentation.

Experiments were carried out using both raw data and domain-specific features extracted from vibration signals for bearing fault diagnostics. However, for the data-limited case considered here, features were observed to be more effective than using raw data directly. Domain-specific features combined with the representation learning capability of the different methods resulted in effective adaptation. Hence, in this work, the results are presented only with domain-specific features as input to all the methods.

Due to the unequal sample size of different datasets, the vibration data is pre-processed by splitting into windows of 1 second for feature extraction. Overlapping windows with different % of overlap for CWRU, CRB and PU, respectively, are considered to obtain $\approx$ 1250 windows from each dataset forming a class-balanced set. This results in 416, 426, and 422 windows for each class for CWRU, CRB and PU, respectively. Five time domain features relevant to bearing fault detection, namely Root Mean Square (RMS), variance, maximum value, kurtosis and peak-to-peak value [41], are extracted from each window. Data is normalized and randomly split into train-test sets with 50% samples of each dataset are taken for training and the rest for testing. The hyperparameters for the

benchmark methods are set to the values specified in their respective papers, and through grid search, we confirmed that these values yield optimal performance. For the dictionary-based methods, the optimal number of dictionary atoms and other hyperparameters values λ, μ, γ and τ are obtained using grid search for both the shallow (DL-UDA) and deep DL (DDL-UDA) methods. The proposed DDL-UDA method was implemented in MATLAB and executed on a system equipped with an AMD Ryzen 5 4500U CPU@2.3GHz and 16 GB RAM. Learning the source-to-target mapping using 50% of the data (600 samples with 5 features) took $\approx$ 0.8 seconds. The classification of the remaining 50% of the data, using the source-trained classifier, took about 0.08 seconds. More discussion on the results is presented in the next section.

2.5.4 Results Discussion

The classification performance of the different UDA methods on the target data is evaluated using Accuracy (Acc) and weighted metrics of Precision (P), Recall (R) and F1-score (F1). Tables 2.2 and 2.3 present the classification results averaged over five random train-test sets of the $\mathcal{S}$ and $\mathcal{T}$ data for all six adaptation scenarios with the best-performing method (across most of the metrics) for each scenario highlighted in bold. The notation $\mathcal{S} \rightarrow \mathcal{T}$ denotes the adaptation from source (labeled) to target (unlabeled). The dictionary-based methods utilized k = 20 atoms and converged with $M = 5$ subspaces for all the datasets. Please note in this work, we considered the same number of atoms for all the multi-layer dictionaries in DDL-UDA. Other hyperparameter values used in the experimentation, obtained with grid search, are λ = 0.5, sparsity

μ = 5 and γ = 0.8. From both the tables, one can observe that the proposed DDL-UDA method with N = 3 layers outperforms all the other benchmark methods for all the adaptation scenarios. Except for $CRB \rightarrow CWRU$, the best-performing adversarial-based and divergence-based techniques for each of the other scenarios show comparable performance. It is to be noted that these benchmark methods were developed to address adaptation between different working conditions of the same machine and have *not* been applied for adaptation between different machines. Since domain features are fed as input to these methods, they are able to work for the challenging data-limited scenario of adaptation between *different but related machines*. Also, the benchmark methods with a DCNN backbone are designed to work with large data sets and hence perform poorly compared to our method, which works well even with limited data.

Table 2.2: Bearing Fault Classification Results (in %) - Set I

Methods	$CWRU \rightarrow CRB$				$CRB \rightarrow CWRU$				$CWRU \rightarrow PU$			
	P	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
MK-MMD [18]	88.44	82.94	81.21	82.94	83.77	83.49	79.55	83.49	84.03	72.86	73.21	72.86
JMMD [18]	87.23	79.34	76.27	79.34	81.54	80.41	76.92	80.41	84.10	73.04	73.40	73.04
CORAL [18]	84.71	71.73	65.48	71.73	86.66	77.59	74.53	77.59	83.34	72.35	72.22	72.35
DANN [18]	88.49	82.53	81.17	82.53	85.13	73.07	67.63	73.07	83.37	75.35	75.81	75.35
CDAN [18]	87.86	81.78	80.26	81.78	50.00	66.67	55.56	66.67	83.27	74.60	75.22	74.60
DL-UDA [1]	81.35	81.08	80.53	81.08	85.03	84.64	80.48	84.64	79.85	80.36	79.78	80.36
Proposed DDL-UDA (N = 2)	89.96	87.19	86.89	87.19	83.34	86.38	80.18	86.38	81.64	81.60	79.63	81.60
Proposed DDL-UDA (N = 3)	**87.01**	**89.90**	**87.57**	**89.90**	**91.97**	**87.88**	**85.07**	**87.88**	**86.80**	**82.30**	**80.84**	**82.30**
Proposed DDL-UDA (N = 4)	92.19	88.04	86.41	88.04	92.15	87.02	85.44	87.02	85.87	82.24	80.57	82.24

While the performance of the DL-UDA method is comparable and, in most cases, better than the best-performing benchmarks for the different adaptation scenarios, overall, the proposed DDL-UDA gives the best results. The 3 layer DDL-UDA outperforms all other methods for all the adaptation cases. An

Table 2.3: Bearing Fault Classification Results (in %) - Set II

Methods	$PU \rightarrow CWRU$				$CRB \rightarrow PU$				$PU \rightarrow CRB$			
	P	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
MK-MMD [18]	75.00	80.00	73.33	80.00	79.99	75.26	68.86	75.26	72.17	73.39	65.11	73.39
JMMD [18]	69.38	76.76	71.67	76.76	89.86	82.46	78.85	82.46	66.69	74.43	66.58	74.43
CORAL [18]	79.38	75.00	69.51	75.00	48.24	65.21	54.18	65.21	59.87	68.79	57.94	68.79
DANN [18]	82.26	73.27	68.86	73.27	86.20	82.37	81.74	82.37	84.68	73.05	67.48	73.05
CDAN [18]	85.75	80.28	78.87	80.28	84.36	71.15	64.42	71.15	84.96	73.46	68.33	73.46
DL-UDA [1]	69.13	78.75	72.16	78.75	84.52	78.42	74.74	78.42	81.33	79.71	78.69	79.71
Proposed DDL-UDA (N = 2)	83.98	82.98	78.56	82.98	83.76	79.62	78.91	79.62	85.65	83.22	82.24	83.22
Proposed DDL-UDA (N = 3)	**85.05**	**85.06**	**80.61**	**85.06**	**86.85**	**84.16**	**83.23**	**84.16**	**89.51**	**86.44**	**83.74**	**86.44**
Proposed DDL-UDA (N = 4)	85.74	83.52	80.87	83.52	86.69	83.96	83.62	83.96	85.78	86.35	85.74	86.35

accuracy improvement > 5% is observed with the DDL variant compared to the single-layer variant (DL-UDA) for most cases. This improvement can be attributed to the fact that deep representations are able to learn the mapping between the two domains more effectively than their shallow counterparts. Notice that the rate of improvement decreases with an increase in the number of layers, i.e., while there is a significant improvement between the first and second layers, the margin reduces between the second and third layers. In our experiments, going further deep (N = 4) did not yield any significant performance improvement; infact, a slight drop in accuracy was observed. This behavior can be attributed to the fact that increasing the number of layers increases the number of parameters that need to be learned. So, for the limited training data scenario considered in this work, further increasing the number of learnable parameters may lead to overfitting, which results in a gradual reduction in performance as we go deep. Hence, we restricted ourselves to only N = 3 layers. Overall, the proposed method provides an accuracy improvement of $\geq$ 6% over the best-performing DCNN-based adaptation methods for most adaptation cases. These results demonstrate the potential and applicability of the proposed DDL-UDA method to transfer knowledge learned from reference datasets (lab setups) to

practical industrial applications for machine fault diagnosis.

For the sake of illustration, Fig. 2.5 shows the reconstruction residue on target data at different subspaces (domains) of N = 1, 2, and 3 layer DDL-UDA configurations for $CWRU \rightarrow PU$ scenario. It can be seen that the reconstruction residue decreases with the increase in the number of subspaces for all the DDL-UDA configurations. The trend is similar for other adaptation scenarios. This is in agreement with the algorithm since the dictionaries of the subsequent subspaces are computed to reduce the residue on the target data. It can be observed that the residue $\boldsymbol{J}_m$ is highest for M = 1 that represents the case of *no adaptation*. Here, the target data is decomposed by the source domain data ($\boldsymbol{D}_0$) and the resulting coefficients are used for classification. A saturation is observed after M = 5, showing the convergence of the algorithm. This suggests that the dictionary learned for M = 5 represents the target data well and no further interpolation is required. One can observe that the residue of the 3 layer DDL-UDA is comparatively less compared to the 1 and 2 layer configuration for $M > 1$, indicating superior modeling capability of the deep network.

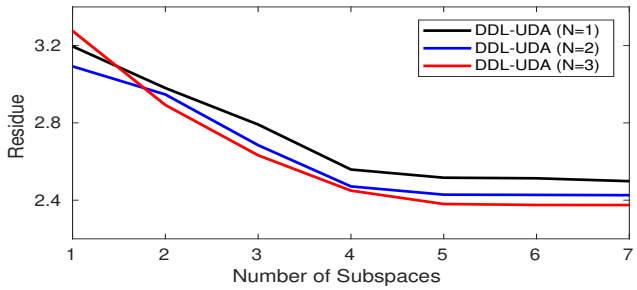


Figure 2.5: Residue on Target Data Vs. Number of Subspaces for different configurations of DDL-UDA

2.6 Summary

The chapter presents a novel deep dictionary-based subspace interpolation method to link the source and target domain data for unsupervised adaptation. Deep dictionaries can learn more rich and reliable data representations that assist in learning the mapping more effectively, thereby addressing the domain shift problem. The data from different domains is transformed to a shared feature space that is more robust and can be used for classification. The requisite formulation and solution steps are detailed. Experimental results obtained with the bearing dataset for the challenging different machine scenario are promising and demonstrate the potential of the proposed DDL-UDA method for UDA. DDL-UDA significantly outperforms all benchmark methods, achieving an accuracy improvement of $\geq 6\%$ over the best-performing DCNN-based adaptation methods for most adaptation cases, suggesting its applicability to real-life industrial applications.

To enhance the adaptation performance with reduced complexity, in the next chapter, the use of Transform Learning, an analysis variant of DL, is explored for subspace modeling. The motivation behind this approach is outlined in Chapter 3, with the corresponding formulation detailed in Section 3.3.

Chapter 3

Unsupervised Domain Adaptation Via Subspace Interpolating Transform Learning for Classification

3.1 Motivation

Although data-driven dictionaries have proved to be quite successful in different applications, the sparse coding solved repeatedly for dictionary learning is NP-hard and the approximate synthesis sparse coding algorithms can be computationally expensive [42]. Moreover, the dictionary learning problem is highly non-convex, and there is a high chance of algorithms getting stuck in local minima. To address these problems, TL-based techniques have gained more importance.

TL is an analysis approach, where the data is analyzed by learning a transform to produce the associated coefficients. Unlike dictionary which is an inverse learning problem, transform is a forward learning problem. In signal processing literature, it is well known that TL has an advantage over dictionaries in terms of application scenarios, accuracy and complexity [43], [44], [45], [42]. Especially in the image domain, TL methods produces state-of-the-art results [46], [42],

[47].

Motivated by the low complexity and performance advantages of TL, this work examines the suitability of TL for data-limited UDA via subspace interpolation. Interpolated subspaces are learned using the source, intermediate and target transforms that capture the domain shift between the source and target domain, thereby providing domain-invariant feature space for cross-domain analysis. Formulations for both, the shallow (single-layer) transform (Transform Learning for Unsupervised Domain Adaptation (TL-UDA)) and the deep version (Deep Transform Learning for Unsupervised Domain Adaptation (DTL-UDA)) are proposed here. Deep transforms facilitate more effective learning of the source-to-target domain mapping through subspace interpolation. Although the proposed methods are generic and they can be applied to different domains, the current work focuses on challenging machine inspection scenarios of unsupervised adaptation between data from *different but related machines* for fault diagnosis. Experimental results, particularly with different machine datasets, demonstrate the efficacy of the proposed method, emphasizing the superior performance against the DL version and other competing deep learning-based techniques. Results indicate that small-sized transforms perform better than the dictionary variant, thus highlighting the low computational complexity of TL-based UDA over the DL counterparts.

The contributions of the work are summarized below:

- Data-driven transforms are employed for subspace modeling, and interpola-

tion is carried out to connect the source and target domain for generating shared features for cross-domain analysis.

- Novel TL-UDA and DTL-UDA formulations are presented with the systematically derived closed-form solutions for the associated transforms and coefficients.
- The challenging adaptation between *different but related machines* is addressed for inspection scenarios. Extensive comparative analysis against the benchmark techniques using publicly available bearing datasets demonstrates the potential of the proposed method for UDA tasks.

The rest of the chapter is organized as follows: Section 3.2 offers a brief background on TL and its deep version DTL, leading to a detailed discussion of the proposed methods TL-UDA and DTL-UDA, in Section 3.3. Section 3.4 presents experimental results with public datasets, showcasing the adaptation capabilities of the proposed methods. Finally, Section 3.5 provides a summary of the work.

3.2 Background on TL and DTL

This section presents a brief overview of the shallow (single-layer) and deep variants of TL, which serve as the underlying basis for the proposed UDA formulations.

3.2.1 Transform Learning (TL)

TL is an analysis approach for learning data representation, where a transform $\boldsymbol{T}$ acts on the data $\boldsymbol{X}$ to produce the coefficients $\boldsymbol{Z}$ [44]. Mathematically, TL formulation is expressed as:

$$\boldsymbol{T}\boldsymbol{X} = \boldsymbol{Z} \tag{3.1}$$

where $\boldsymbol{X} \in \mathbb{R}^{d \times n}$ is the data matrix of d features of length n, $\boldsymbol{T} \in \mathbb{R}^{k \times d}$ is the transform of k atoms and $\boldsymbol{Z} \in \mathbb{R}^{k \times n}$ are the coefficients.

To learn sparse representations from the data, the TL formulation is formally expressed as [44]:

$$\min_{\boldsymbol{T},\boldsymbol{Z}} \|\boldsymbol{T}\boldsymbol{X} - \boldsymbol{Z}\|_F^2 + \lambda(\|\boldsymbol{T}\|_F^2 - \log\det \boldsymbol{T}) + \mu\|\boldsymbol{Z}\|_0 \tag{3.2}$$

where hyperparameters $\lambda > 0$ and $\mu > 0$. The first term in (3.2) is the data fidelity term for the TL formulation mentioned in (3.1). The second term in (3.2) is a regularizer where, the factor $-\log\det \boldsymbol{T}$ imposes a full rank on the learned transform and $\|\boldsymbol{T}\|_F^2$ balances the scale. This term is added to prevent trivial / degenerate solutions ($\boldsymbol{T}$ = 0, $\boldsymbol{Z}$ = 0 / $\boldsymbol{T} \to \infty$, $\boldsymbol{Z} \to \infty$) and control the condition number [48] of $\boldsymbol{T}$. The third term enforces sparsity on the coefficients $\boldsymbol{Z}$ to avoid overfitting. As shown in [42], the problem in (3.2) can be solved using AM [39] which offers closed-form updates for $\boldsymbol{Z}$ and $\boldsymbol{T}$. These updates are presented here for the sake of completeness. The sub-problem and the corresponding closed-form update for $\boldsymbol{Z}$ is given as:

$$\boldsymbol{Z} \leftarrow \min_{\boldsymbol{Z}} \|\boldsymbol{T}\boldsymbol{X} - \boldsymbol{Z}\|_F^2 + \mu\|\boldsymbol{Z}\|_0 \tag{3.3}$$

$$\boldsymbol{Z} = (abs(\boldsymbol{T}\boldsymbol{X}) \geq \mu).\, \boldsymbol{T}\boldsymbol{X} \tag{3.4}$$

where $\boldsymbol{Z}$ is updated via hard-thresholding against the value μ and '.' denotes the

element-wise product.

The closed-form update for $\boldsymbol{T}$ is obtained by solving the following sub-problem:

$$\boldsymbol{T} \leftarrow \min_{\boldsymbol{T}} \|\boldsymbol{T}\boldsymbol{X} - \boldsymbol{Z}\|_F^2 + \lambda(\|\boldsymbol{T}\|_F^2 - \log\det \boldsymbol{T}) \tag{3.5}$$

Following the work in [45], Cholesky decomposition and singular value decomposition is applied to obtain the update for $\boldsymbol{T}$ as:

$$\boldsymbol{X}\boldsymbol{X}^T + \lambda \boldsymbol{I} = \boldsymbol{L}\boldsymbol{L}^T \tag{3.6}$$

$$\boldsymbol{L}^{-1}\boldsymbol{X}\boldsymbol{Z}^T = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T \tag{3.7}$$

$$\hat{\boldsymbol{T}} = 0.5\boldsymbol{V}(\boldsymbol{S} + (\boldsymbol{S}^2 + 2\lambda \boldsymbol{I})^{1/2})\boldsymbol{U}^T\boldsymbol{L}^{-1} \tag{3.8}$$

In contrast to iterative optimization methods such as conjugate gradients, this solution provides improved convergence and effective computation of the transform as shown in [42]. This framework can be used to learn the data representation and carry out classification or regression tasks depending whether the output variable is discrete or continuous.

3.2.2 Deep Transform Learning (DTL)

DTL is the deep version of TL where multiple transform representations are cascaded together to generate the coefficients, where the different transforms correspond to different layers of the deep network. The N layer DTL version of the shallow (single-layer) TL formulation in (3.1) is given as:

$$\boldsymbol{T_N}(\phi...(\boldsymbol{T_2}(\phi(\boldsymbol{T_1}\boldsymbol{X})))) = \boldsymbol{Z} \tag{3.9}$$

here, ϕ denotes the activation function applied between the different layers of the deep network. Here, $\boldsymbol{T_1}$ operates on the data $\boldsymbol{X}$ to produce the first level of coefficients. $\boldsymbol{T_2}$ analyzes the first level of the coefficients to produce the

second level. Finally, $\boldsymbol{T_N}$ operates the $(N-1)$ level of coefficients to generate final coefficients $\boldsymbol{Z}$. The greedy solution for (3.9) is given in [49], where the transforms and coefficients are solved for each layer using standard TL updates and substituted for the next layer until all the coefficients are estimated. This method is sub-optimal as there is no flow of information from deep to shallow layers, thus, another method is proposed in [50] that solves for the transforms and coefficients of all layers together using a joint optimization framework expressed as:

$$\min_{\boldsymbol{T_i}'s,\boldsymbol{Z}} \|\boldsymbol{T_N}(\phi...(\boldsymbol{T_2}(\phi(\boldsymbol{T_1}\boldsymbol{X})))) - \boldsymbol{Z}\|_F^2 + \lambda \sum_{i=1}^{N} (\|\boldsymbol{T_i}\|_F^2 - \log\det \boldsymbol{T_i}) + \mu \|\boldsymbol{Z}\|_0 \quad (3.10)$$

This method uses variable splitting and Alternating Direction Method of Multipliers (ADMM) technique [51] for obtaining the requisite updates for the coefficients and transforms.

With this brief introduction, the proposed formulations for domain adaptation considering both the shallow (single-layer) and deep versions of TL are presented in the subsequent sections.

3.3 Unsupervised Domain Adaptation Via Subspace Interpolating Transform Learning

The problem definition of UDA remains the same as that of the previous work on dictionary learning (Chapter 2, Section 2.4.1). We follow the same naming convention for the source data $\boldsymbol{X}_s$ and target data $\boldsymbol{X}_t$. The proposed formulations employing TL are presented below.

3.3.1 Proposed UDA method using Transform Learning (TL-UDA)

This formulation also follows the idea of subspace interpolation for domain adaptation presented in [1], but uses data-driven transforms instead of dictionaries to model the source $\mathcal{S}$, intermediate and target $\mathcal{T}$ domains. Starting with source domain transform $\boldsymbol{T}_0 \in \mathbb{R}^{k \times d}$ associated with the source domain data $\boldsymbol{X}_s$ with k atoms, a set of intermediate transforms $\boldsymbol{T}_m, m \in [1, M-1]$ (intermediate domains) are learned by transforming the target data $\boldsymbol{X}_t$, iteratively in the direction to reduce the residue on the target data till we reach $\boldsymbol{T}_M$ that best represents the target domain data $\boldsymbol{X}_t$. Fig. 3.1 presents the block diagram of the proposed TL-UDA method that shows the different subspaces modeled by different transforms obtained by interpolation on the target data.

Similar to the DDL-UDA (Chapter 2, Section 2.4.2), this method employs a training phase for learning the virtual path that connects $\mathcal{S}$ and $\mathcal{T}$ domains but uses transforms instead of dictionaries to capture the domain shift between the two domains. Once the mapping is learned, domain-invariant features are generated to learn the classifier. Later, this $\mathcal{S}$ to $\mathcal{T}$ mapping is utilized in the test phase for generating domain-invariant features from the target data for estimating target labels. The subsequent sections present more details on the two phases.

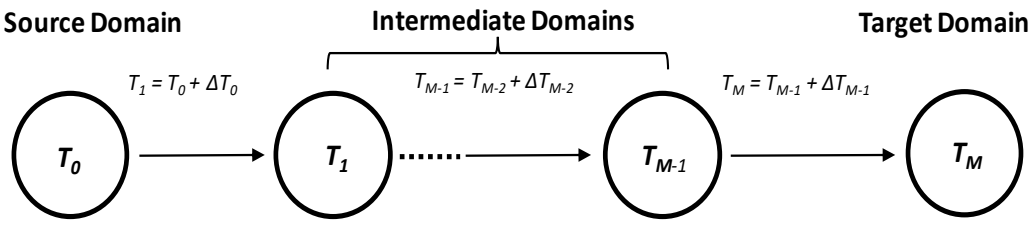


Figure 3.1: Unsupervised Domain Adaptation via Interpolation over M Subspaces using TL (TL-UDA)

3.3.1.1 Training Phase

First the source domain transform $\boldsymbol{T}_0$ for the source data $\boldsymbol{X}_s$ is learnt by solving:

$$\min_{\boldsymbol{T}_0, \boldsymbol{Z}_0} \|\boldsymbol{T}_0 \boldsymbol{X}_s - \boldsymbol{Z}_0\|_F^2 + \lambda(\|\boldsymbol{T}_0\|_F^2 - \log\det \boldsymbol{T}_0) + \mu\|\boldsymbol{Z}_0\|_0 \tag{3.11}$$

The above expression is similar to (3.2). The source transform $\boldsymbol{T}_0$ and the coefficients $\boldsymbol{Z}_0$ are obtained using the standard updates (3.3) to (3.8). Considering M subspaces with $m \in [0, M]$, for each m, the m^{th} domain transform $\boldsymbol{T}_m$ is applied on the target data X_t to generate the coefficients $\boldsymbol{Z}_m$ following the update of (3.3) and the residue $\boldsymbol{J}_m$ is computed using the following:

$$\boldsymbol{Z}_m \leftarrow \min_{\boldsymbol{Z}_m} \|\boldsymbol{T}_m \boldsymbol{X}_t - \boldsymbol{Z}_m\|_F^2 + \mu\|\boldsymbol{Z}_m\|_0 \tag{3.12}$$

$$\boldsymbol{J}_m = \boldsymbol{T}_m \boldsymbol{X}_t - \boldsymbol{Z}_m \tag{3.13}$$

The new transform $\boldsymbol{T}_{m+1}$ is computed by estimating $\boldsymbol{\Delta T}_m$ that represent the adjustment in the transform atoms between the transforms of $\boldsymbol{T}_{m+1}$ and $\boldsymbol{T}_m$ that helps in reducing the residue $\boldsymbol{J}_m$. $\boldsymbol{\Delta T}_m$ is estimated by solving:

$$\boldsymbol{\Delta T}_m \leftarrow \min_{\boldsymbol{\Delta T}_m} \|\boldsymbol{\Delta T}_m \boldsymbol{J}_m - \boldsymbol{Z}_m\|_F^2 + \lambda(\|\boldsymbol{\Delta T}_m\|_F^2 - \log\det \boldsymbol{\Delta T}_m) \tag{3.14}$$

The above sub-problem has a form similar to (3.5), hence we follow the same updates given in (3.6) to (3.8) for computing $\boldsymbol{\Delta T}_m$ by appropriately changing the input from $\boldsymbol{X}$ to $\boldsymbol{J}_m$. Note here the first term reduces the residue and the second term discourages abrupt changes in the transforms of adjacent domains. Subsequently, the new transform $\boldsymbol{T}_{m+1}$ is obtained as:

$$\boldsymbol{T}_{m+1} = \boldsymbol{T}_m + \eta \boldsymbol{\Delta T}_m \tag{3.15}$$

where η is introduced to ensure a smooth transition between the transforms of neighbouring domains.

Now the new transform $\boldsymbol{T}_{m+1}$ associated with the next intermediate domain is used on the target data to compute the residue in the feature space. This process continues iteratively till $\|\boldsymbol{\Delta T}_m\|_F \leq \tau$ (empirically calculated threshold), suggesting the learnt intermediate domain transforms fully absorb the domain shift between $\mathcal{S}$ and $\mathcal{T}$ domains. The last obtained transform $\boldsymbol{T}_M$ is considered as the target transform that completely represents the target data. Kindly note we consider normalization for all the learned transforms, where the rows of all the transforms are normalized to unit norm. The pseudocode of the proposed TL-UDA method is summarized in Algorithm 2.

Once the transition path between the two domains is learnt, invariant sparse codes (coefficients) are applied across the source, intermediate, and target transforms ($\{\boldsymbol{T}_m\}_{m=0}^{M}$) to form new features for classification. The new feature space is given as: $[(\boldsymbol{T}_0^{-1}\boldsymbol{Z})^T, (\boldsymbol{T}_1^{-1}\boldsymbol{Z})^T, .., (\boldsymbol{T}_M^{-1}\boldsymbol{Z})^T]$ where $\boldsymbol{Z} \in \mathbb{R}^k$ are the sparse codes generated either by transforming source data $\boldsymbol{X}_s$ with $\boldsymbol{T}_0$ (i.e., $\boldsymbol{Z}_0$) or transforming target data $\boldsymbol{X}_t$ with $\boldsymbol{T}_M$ (i.e., $\boldsymbol{Z}_M$). Since the labels $\boldsymbol{Y}_s$ are known only for the source data $\boldsymbol{X}_s$, the classifier is trained using features obtained by applying $\boldsymbol{Z}_0$ across the source, intermediate and target transforms. Please note, here again, we have used an SVM classifier, but in general, any suitable classifier can be employed.

3.3.1.2 Test Phase

To estimate the labels $\boldsymbol{Y}_t^{test}$ associated with the test target data $\boldsymbol{X}_t^{test}$, first $\boldsymbol{Z}_M^{test}$ is computed by applying $\boldsymbol{T}_M$ on $\boldsymbol{X}_t^{test}$. Subsequently, features are computed by applying $\boldsymbol{Z}_M^{test}$ across the source, intermediate and target transforms. These test

Algorithm 2: Subspace Interpolation using TL for UDA (TL-UDA)

Input: $\boldsymbol{X}_s$, $\boldsymbol{X}_t$
Parameters: $\lambda, \mu, \tau, \eta, k$ (number of transform atoms)
Initialization: Set transform $\boldsymbol{T}_0$ to random matrix with real numbers between 0 and 1 drawn from a uniform distribution, $m = 0$
Compute source transform $\boldsymbol{T}_0$ and $\boldsymbol{Z}_0$ with $\boldsymbol{X}_s$ using (3.11).
do
    Transform $\boldsymbol{X}_t$ with $\boldsymbol{T}_m$ using (3.12).
    Compute the residue $\boldsymbol{J}_m$ using (3.13).
    Estimate the adjustment in transform atoms $\boldsymbol{\Delta T}_m$ using (3.14).
    Update the transform $\boldsymbol{T}_{m+1}$ using (3.15).
    Normalize each row of the transform to unit norm.
    $m = m + 1$
while ($\|\boldsymbol{\Delta T}_m\|_F > \tau$)
Output: $\{\boldsymbol{T}_m\}_{m=0}^{M}$ (source, intermediate and target transforms)

features are fed to the classifier learnt in the training phase to estimate the test target labels.

3.3.2 Proposed UDA method using Deep Transform Learning (DTL-UDA)

The proposed approach employs deep transforms for subspace modeling and uses a similar interpolation method as discussed in Section 3.3.1 for domain adaptation. The multiple layers of transforms (deep transforms) are able to learn rich representation from the data, resulting in effective modeling of the source $\mathcal{S}$, target $\mathcal{T}$, and the intermediate domains connecting $\mathcal{S}$ and $\mathcal{T}$ data. First, the source domain transform $\boldsymbol{T}_0 \in \mathbb{R}^{k \times d}$ associated with the source domain data $\boldsymbol{X}_s$ is learned. Subsequently, a set of intermediate transforms $\boldsymbol{T}_m, m \in [1, M-1]$ (intermediate domains) are learned by transforming the target data $\boldsymbol{X}_t$, iteratively along the direction that reduces the residue on the target data (described using (3.28) and (3.30), (3.31) (3.32)) till we reach $\boldsymbol{T}_M$, the best representation of the target domain data $\boldsymbol{X}_t$. Here, each of the transform $\boldsymbol{T}_0, .., \boldsymbol{T}_M$ are made deep by cascading multiple layers of transforms utilizing the DTL formulation presented

in (3.10). Fig. 3.2 shows the block diagram of the proposed N layer DTL-UDA with M subspaces to absorb the domain shift between $\mathcal{S}$ and $\mathcal{T}$.

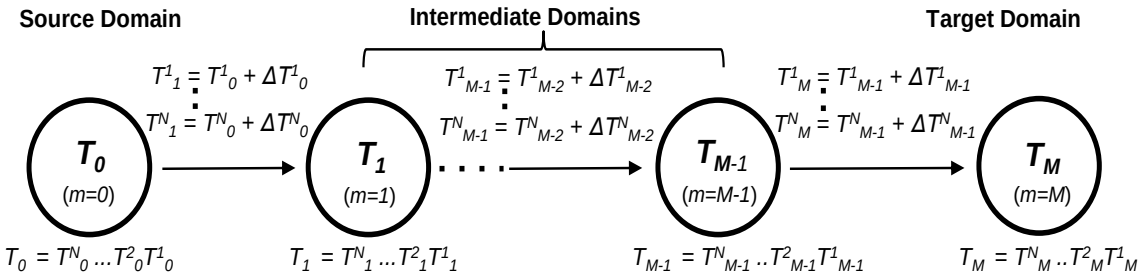


Figure 3.2: Unsupervised Domain Adaptation using N layer DTL via Interpolation over M Subspaces (DTL-UDA)

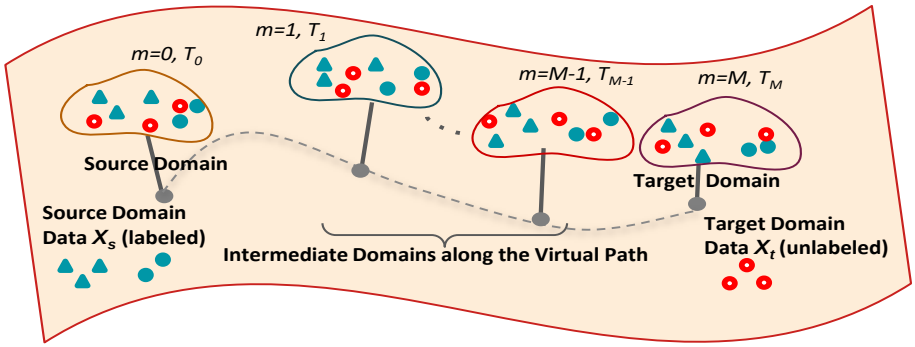


Figure 3.3: Illustration of Feature Augmentation for Subspace Interpolation-based UDA Methods using TL (TL-UDA and DTL-UDA)

Similar to TL-UDA method discussed in Section 3.3.1, this method has a training and test phase described below.

3.3.2.1 Training Phase

Here, the detailed formulation is presented considering $N = 3$ for brevity. However, the formulation can be generalized to any N layers. Let $m = 0$ denote the subspace of the source domain, then starting with the source domain transform $\boldsymbol{T}_0$, modeled using a 3 layer deep transform architecture $\boldsymbol{T}_0 = \boldsymbol{T}_0^3 \boldsymbol{T}_0^2 \boldsymbol{T}_0^1$, the deep transforms for source data $\boldsymbol{X}_s$ are learned by solving:

$$\min_{\boldsymbol{T}_0^1, \boldsymbol{T}_0^2, \boldsymbol{T}_0^3, \boldsymbol{Z}_0} \|\boldsymbol{T}_0^3(\phi(\boldsymbol{T}_0^2(\phi(\boldsymbol{T}_0^1 \boldsymbol{X}_s)))) - \boldsymbol{Z}_0\|_F^2 + \lambda \sum_{i=1}^{3} (\|\boldsymbol{T}_0^i\|_F^2 - \log\det \boldsymbol{T}_0^i) + \mu \|\boldsymbol{Z}_0\|_0 \tag{3.16}$$

where ϕ is the activation function. Here, the deep transforms and coefficients are updated using a greedy approach, considering them one layer at a time with a Rectified Linear Unit (ReLU)-type activation function. Substituting $\boldsymbol{T}_m^N \boldsymbol{Z}_m^{N-1} = \boldsymbol{Z}_m^N$, the 3 layer deep transforms and coefficients for the source domain data are updated by solving:

$$\min_{\boldsymbol{T}_0^1, \boldsymbol{Z}_0^1} \|\boldsymbol{T}_0^1 \boldsymbol{X}_s - \boldsymbol{Z}_0^1\|_F^2 + \lambda(\|\boldsymbol{T}_0^1\|_F^2 - \log\det \boldsymbol{T}_0^1) + \mu\|\boldsymbol{Z}_0^1\|_0 \tag{3.17}$$

$$\min_{\boldsymbol{T}_0^2, \boldsymbol{Z}_0^2} \|\boldsymbol{T}_0^2 \boldsymbol{Z}_0^1 - \boldsymbol{Z}_0^2\|_F^2 + \lambda(\|\boldsymbol{T}_0^2\|_F^2 - \log\det \boldsymbol{T}_0^2) + \mu\|\boldsymbol{Z}_0^2\|_0 \tag{3.18}$$

$$\min_{\boldsymbol{T}_0^3, \boldsymbol{Z}_0^3} \|\boldsymbol{T}_0^3 \boldsymbol{Z}_0^2 - \boldsymbol{Z}_0^3\|_F^2 + \lambda(\|\boldsymbol{T}_0^3\|_F^2 - \log\det \boldsymbol{T}_0^3) + \mu\|\boldsymbol{Z}_0^3\|_0 \tag{3.19}$$

Here, $\boldsymbol{Z}_0^3 = \boldsymbol{Z}_0$ in (3.16). The multi-layer transforms and coefficients are updated using an AM approach. The transforms are computed by solving:

$$\boldsymbol{T}_0^1 \leftarrow \min_{\boldsymbol{T}_0^1} \|\boldsymbol{T}_0^1 \boldsymbol{X}_s - \boldsymbol{Z}_0^1\|_F^2 + \lambda(\|\boldsymbol{T}_0^1\|_F^2 - \log\det \boldsymbol{T}_0^1) \tag{3.20}$$

$$\boldsymbol{T}_0^2 \leftarrow \min_{\boldsymbol{T}_0^2} \|\boldsymbol{T}_0^2 \boldsymbol{Z}_0^1 - \boldsymbol{Z}_0^2\|_F^2 + \lambda(\|\boldsymbol{T}_0^2\|_F^2 - \log\det \boldsymbol{T}_0^2) \tag{3.21}$$

$$\boldsymbol{T}_0^3 \leftarrow \min_{\boldsymbol{T}_0^3} \|\boldsymbol{T}_0^3 \boldsymbol{Z}_0^2 - \boldsymbol{Z}_0^3\|_F^2 + \lambda(\|\boldsymbol{T}_0^3\|_F^2 - \log\det \boldsymbol{T}_0^3) \tag{3.22}$$

The sub-problems above (3.20), (3.21) and (3.22) have a similar form of (3.5) and hence the same closed-form updates (3.6) to (3.8) can be used to compute them by appropriately changing the input. The sub-problem to solve for the coefficients is similar to (3.3), and hence, they follow the same closed-form update given in (3.4).

Considering M subspaces, for $m \in [0, M]$, the m^{th} domain deep transform $\boldsymbol{T}_m = \boldsymbol{T}_m^3 \boldsymbol{T}_m^2 \boldsymbol{T}_m^1$ is subsequently applied on the target data $\boldsymbol{X}_t$ for generating the coefficients $\boldsymbol{Z}_m^3$ (last layer) by solving:

$$\boldsymbol{Z}_m^3 \leftarrow \min_{\boldsymbol{Z}_m^3} \|\boldsymbol{T}_m^3 \boldsymbol{T}_m^2 \boldsymbol{T}_m^1 \boldsymbol{X}_t - \boldsymbol{Z}_m^3\|_F^2 + \mu \|\boldsymbol{Z}_m^3\|_0 \tag{3.23}$$

Once $\boldsymbol{Z}_m^3$ is computed, it is fed as input to the second last layer. The same process is repeated for the remaining layers. Following this greedy paradigm, the coefficients of the other layers for the m^{th} domain are computed by solving:

$$\boldsymbol{Z}_m^1 \leftarrow \min_{\boldsymbol{Z}_m^1} \|\boldsymbol{T}_m^1 \boldsymbol{X}_t - \boldsymbol{Z}_m^1\|_F^2 + \gamma \|\boldsymbol{T}_m^2 \boldsymbol{Z}_m^1 - \boldsymbol{Z}_m^2\|_F^2 \;\; s.t. \;\; \boldsymbol{Z}_m^1 \geq 0 \tag{3.24}$$

$$\boldsymbol{Z}_m^2 \leftarrow \min_{\boldsymbol{Z}_m^2} \|\boldsymbol{T}_m^2 \boldsymbol{Z}_m^1 - \boldsymbol{Z}_m^2\|_F^2 + \gamma \|\boldsymbol{T}_m^3 \boldsymbol{Z}_m^2 - \boldsymbol{Z}_m^3\|_F^2 \;\; s.t. \;\; \boldsymbol{Z}_m^2 \geq 0 \tag{3.25}$$

where the hyperparameter γ is set to 1, giving equal importance to all the layers as in [50]. Note that $\boldsymbol{Z}_m^3$ follows the update given in (3.4). As mentioned earlier, for other coefficients, a ReLU-type non-linearity is employed between the layers that essentially sets the negative values of the coefficients of each layer to 0. The terms in (3.24) and (3.25) are expanded in terms of trace and derivatives are computed with respect to $\boldsymbol{Z}_m^1$ and $\boldsymbol{Z}_m^2$, respectively. Equating the derivatives to 0, the following closed-form updates are obtained:

$$\boldsymbol{Z}_m^1 = [I + \gamma (\boldsymbol{T}_m^2)^T \boldsymbol{T}_m^2]^{-1} (\boldsymbol{T}_m^1 \boldsymbol{X}_t + \gamma (\boldsymbol{T}_m^2)^T \boldsymbol{Z}_m^2) \tag{3.26}$$

$$\boldsymbol{Z}_m^2 = [I + \gamma (\boldsymbol{T}_m^3)^T \boldsymbol{T}_m^3]^{-1} (\boldsymbol{T}_m^2 \boldsymbol{Z}_m^1 + \gamma (\boldsymbol{T}_m^3)^T \boldsymbol{Z}_m^3) \tag{3.27}$$

Subsequently, the coefficients $\boldsymbol{Z}_m^3$ are used to compute the residue $\boldsymbol{J}_m$ on the target data $\boldsymbol{X}_t$ as:

$$\boldsymbol{J}_m = \boldsymbol{T}_m \boldsymbol{X}_t - \boldsymbol{Z}_m^3 = \boldsymbol{T}_m^3 \boldsymbol{T}_m^2 \boldsymbol{T}_m^1 \boldsymbol{X}_t - \boldsymbol{Z}_m^3 \tag{3.28}$$

In the direction of reducing the residue further, the transform for the next subspace $\boldsymbol{T}_{m+1}$ is obtained by computing $\boldsymbol{\Delta T}_{m's}$ that represents the adjustment in the transform atoms of the different layers $\boldsymbol{\Delta T}_m^1$, $\boldsymbol{\Delta T}_m^2$, $\boldsymbol{\Delta T}_m^3$ between the transforms of adjacent domains $\boldsymbol{T}_{m+1}$ and $\boldsymbol{T}_m$. $\boldsymbol{\Delta T}_{m's}$ account for the residue and are obtained by solving:

$$\min_{\boldsymbol{\Delta T}^1_m, \boldsymbol{\Delta T}^2_m, \boldsymbol{\Delta T}^3_m} \|\boldsymbol{\Delta T}^3_m \boldsymbol{\Delta T}^2_m \boldsymbol{\Delta T}^1_m \boldsymbol{J}_m - \boldsymbol{Z}^3_m\|_F^2 + \lambda \sum_{i=1}^{3} (\left\|\boldsymbol{\Delta T}^i_m\right\|_F^2 - \log\det \boldsymbol{\Delta T}^i_m) \tag{3.29}$$

Note here while the first term minimizes the residue, the second term discourages abrupt changes in the multi-layer transforms of the adjacent domains.

Similar to the previous case, each of the layers is greedily solved one at a time, which results in:

$$\boldsymbol{\Delta T}^1_m \leftarrow \min_{\boldsymbol{\Delta T}^1_m} \|\boldsymbol{\Delta T}^1_m \boldsymbol{J}_m - \boldsymbol{Z}^1_m\|_F^2 + \lambda(\|\boldsymbol{\Delta T}^1_m\|_F^2 - \log\det \boldsymbol{\Delta T}^1_m) \tag{3.30}$$

$$\boldsymbol{\Delta T}^2_m \leftarrow \min_{\boldsymbol{\Delta T}^2_m} \|\boldsymbol{\Delta T}^2_m \boldsymbol{Z}^1_m - \boldsymbol{Z}^2_m\|_F^2 + \lambda(\|\boldsymbol{\Delta T}^2_m\|_F^2 - \log\det \boldsymbol{\Delta T}^2_m) \tag{3.31}$$

$$\boldsymbol{\Delta T}^3_m \leftarrow \min_{\boldsymbol{\Delta T}^3_m} \|\boldsymbol{\Delta T}^3_m \boldsymbol{Z}^2_m - \boldsymbol{Z}^3_m\|_F^2 + \lambda(\|\boldsymbol{\Delta T}^3_m\|_F^2 - \log\det \boldsymbol{\Delta T}^3_m) \tag{3.32}$$

The sub-problems in (3.30), (3.31), and (3.32) have a form similar to (3.5), hence they follow the same closed-form updates (3.6) - (3.8) by appropriately changing the variables. In the subsequent step, the deep transforms for the next subspace (intermediate domain) are updated as follows:

$$\begin{aligned} \boldsymbol{T}^1_{m+1} &= \boldsymbol{T}^1_m + \eta \boldsymbol{\Delta T}^1_m \\ \boldsymbol{T}^2_{m+1} &= \boldsymbol{T}^2_m + \eta \boldsymbol{\Delta T}^2_m \\ \boldsymbol{T}^3_{m+1} &= \boldsymbol{T}^3_m + \eta \boldsymbol{\Delta T}^3_m \\ \boldsymbol{T}_{m+1} &= \boldsymbol{T}^3_{m+1} \boldsymbol{T}^2_{m+1} \boldsymbol{T}^1_{m+1} \end{aligned} \tag{3.33}$$

where η is introduced to ensure a smooth transition between the deep transforms of the adjacent domains. The new transform $\boldsymbol{T}_{m+1}$ is subsequently applied on the target data, and the residue is computed. This process continues iteratively till $\|\boldsymbol{\Delta T}_{m's}\|_F$ (for the multiple layers) $\leq \tau$ (threshold), suggesting the domain

shift is fully absorbed by the learned intermediate domains between $\mathcal{S}$ and $\mathcal{T}$. Similar to the single-layer (TL-UDA) case, the last transform $\boldsymbol{T}_M$ is taken as the target transform since it efficiently represents the target data. It is to be noted that normalization is considered for all the learned deep transforms, where rows of all the transforms are normalized to unit norm in each iteration. Additionally, same number of atoms (k) is considered for all the multi-layer transforms. The pseudocode of the proposed DTL-UDA detailing the algorithm steps is given in Algorithm 3.

Using Algorithm 3, the virtual path between $\mathcal{S}$ and $\mathcal{T}$ is learned, and the source, intermediate, and target deep transforms ($\{\boldsymbol{T}_m\}_{m=0}^{M}$) thus obtained are used to compute domain-invariant features for classification similar to the TL-UDA method discussed in Section 3.3.1. The domain-invariant features essentially represents the concatenation of data projections through all the subspaces connecting the two domains. For the sake of illustration, Fig. 3.3 shows the features generated along the virtual path connecting the $\mathcal{S}$ and $\mathcal{T}$ domains. Here, the similarity between data samples from different domains is measured by integrating the distance of their projection along the virtual path. Since labels $\boldsymbol{Y}_s$ are known for the source domain data $\boldsymbol{X}_s$, features are computed using $\boldsymbol{Z}_0$ to train a classifier for cross-domain classification.

3.3.2.2 Test Phase

The test phase is similar to Section 3.3.1.2, with the only difference that here, each of the subspaces (source, intermediate and target) are modeled using deep transforms instead of single-layer transforms.

Algorithm 3: UDA via Subspace Interpolation using 3 layer DTL (DTL-UDA)

Input: Source domain data $\boldsymbol{X}_s$, Target domain data $\boldsymbol{X}_t$
Parameters: $\lambda, \gamma, \tau, \eta, \mu, k$ (number of transform atoms)
Initialization: Set multi-layer transforms $(\boldsymbol{T}_0^1, \boldsymbol{T}_0^2, \boldsymbol{T}_0^3)$ to random matrix with real numbers between 0 and 1 drawn from a uniform distribution, subspace $m = 0$
Compute multi-layer source transform $(\boldsymbol{T}_0^1, \boldsymbol{T}_0^2, \boldsymbol{T}_0^3)$ and $\boldsymbol{Z}_0$ with $\boldsymbol{X}_s$ using (3.16).
do
    Transform $\boldsymbol{X}_t$ with $\boldsymbol{T}_m^1$, $\boldsymbol{T}_m^2$, and $\boldsymbol{T}_m^3$ and compute $\boldsymbol{Z}_m^3$ using (3.23).
    Estimate $\boldsymbol{J}_m$ (residue) using (3.28).
    Compute $\boldsymbol{\Delta T}_m^1$, $\boldsymbol{\Delta T}_m^2$, and $\boldsymbol{\Delta T}_m^3$ (adjustment in deep transform atoms) using (3.30), (3.31) and (3.32).
    Update multi-layer deep transforms $(\boldsymbol{T}_{m+1}^1, \boldsymbol{T}_{m+1}^2, \boldsymbol{T}_{m+1}^3)$, and $\boldsymbol{T}_{m+1}$ using (3.33). Normalize each row in all the multi-layer transforms to unit norm.
    $m = m + 1$
while ($\|\boldsymbol{\Delta T}_{m's}\|_F$ (for the different layers) $\geq \tau$)
Output: $\{\boldsymbol{T}_m\}_{m=0}^M$ (source, intermediate and target transforms)

It is important to note that as discussed in [42], the total cost per iteration of the batch TL update (sparse coding and transform) is $\mathcal{O}(nd^2)$, where n is the number of training samples and d is the feature dimension ($d << n$). Here, the sparse coding update is a simple thresholding that takes $n \times d$ time. The transform update requires matrix multiplications and eigenvalue decompositions that takes $n \times d^2$ time for matrix products and d^3 time for SVD computation. Hence, overall the cost per iteration is dominated by the transform update and is given as $\mathcal{O}(nd^2 + d^3) \approx \mathcal{O}(nd^2)$. This is comparatively much lower than the cost per iteration of DL update (sparse coding and dictionary) using K-SVD [37], considering an overcomplete dictionary $\boldsymbol{D} \in \mathbb{R}^{d \times k}$(where $k > d$, k being the number of dictionary atoms). The sparse coding update in K-SVD employs orthogonal matching pursuit, that takes $n \times d \times k \times \mu$ time while the dictionary update time is $n \times d \times \mu$, where μ is the sparsity level. Overall, sparse coding update dominates, hence total cost per iteration is $\mathcal{O}(nd^3)$ (assuming $\mu \propto d$, and $k \propto d$, as stated in [42]). Since the subspace interpolation method for

DA requires M subspaces to be learned for connecting the two domains, with each subspace modeled using N layer deep configuration, the computational complexity of the proposed DTL-UDA scales to $\mathcal{O}(NMnd^2)$ as compared to $\mathcal{O}(NMnd^3)$ for the dictionary variant (DDL-UDA) (presented in Chapter 2, Section 2.4.2).

3.4 Results Discussion

The proposed TL-UDA and DTL-UDA methods are evaluated for machine fault diagnosis using the same three bearing fault datasets discussed in Chapter 2, Section 2.5.1. The data pre-processing remains the same as presented in Chapter 2, Section 2.5.3, and the same 3 classes - Healthy, Inner-race Fault (IF), and Outer-race Fault (OF) are considered for classification. Also, the same benchmark methods as mentioned in Chapter 2, Section 2.5.2 are considered for performance evaluation. Additionally, comparisons with the subspace interpolation methods employing single-layer DL (DL-UDA) [1], its deep variant (DDL-UDA) (Chapter 2, Section 2.4.2) are reported to highlight the performance enhancement achieved with the TL and DTL version (TL-UDA and DTL-UDA) for adaptation tasks. Kindly note that 3 layer DDL-UDA configuration is considered for comparison as it gave the best results for the datasets considered in this work.

Here again, Acc and weighted metrics of P, R, and F1 are used to assess the classification performance of the different UDA methods on the target data. The results summarized in Tables 3.1 and 3.2 present the performance averaged over five random train-test sets of the $\mathcal{S}$ and $\mathcal{T}$ data for all six adaptation scenarios

with the best-performing method (across most of the metrics) for each scenario highlighted in bold. The dictionary-based methods utilized the same values of the hyperparameters as discussed in Chapter 2, Section 2.5.4 with k = 20 atoms and M = 5 subspaces. The hyperparameter tuning for the transform-based methods was conducted using grid search to obtain optimal values. The transform-based methods utilized k = 5 atoms and converged with M = 5 subspaces. Other hyperparameter values for the TL-based methods used for experimentation are λ = 0.8, sparsity μ = 0.1 and γ = 1 and η = 0.04. The proposed TL-UDA and DTL-UDA methods were implemented in MATLAB and executed on a system equipped with an AMD Ryzen 5 4500U CPU@2.3GHz and 16 GB RAM. Learning the source-to-target mapping using 50% of the data (600 samples with 5 features) took $\approx$ 0.5 seconds for both methods. The classification of the remaining 50% of the data, using the source-trained classifier, was completed in about 0.04 seconds for both the methods.

From both the tables, one can observe that the proposed DTL-UDA method with N = 3 layers outperforms all the other benchmark methods for all the combinations. One can observe that the subspace interpolation-based techniques modeled using dictionaries and transforms (and their deep variants) results in improved performance for all the adaptation scenarios over the DCNN-based methods. This is because more data is required for them to learn meaningful representations. Moreover, the accuracy of these methods also depends on the backbone model used for implementation [18]. The subspace interpolation-based techniques are able to learn the mapping between $\mathcal{S}$ and $\mathcal{T}$ more efficiently,

even with limited training data. Among the two subspace-based techniques employing DL and TL, with small-sized transforms ($k = 5$, square transform), the single-layer TL-UDA performs better than the single-layer DL-UDA ($k = 20$, overcomplete dictionary) [1]. The trend is same even for the deep variants, i.e., DDL-UDA and DTL-UDA, which demonstrate the computational advantage of using transforms over dictionaries for subspace modeling tasks. One can observe the performance improvement with the deep variants compared to the single-layer variants. Here, 3 layer DTL-UDA gave the best performance for all the cases. The results with other DTL-UDA configurations utilizing different number of layers N and subspaces M are presented in detail in the optimal parameter settings section given below.

Fig. 3.4 presents the confusion matrix of one adaptation scenario ($PU \rightarrow CRB$) obtained with different methods for one test set. It can be seen that compared to other methods, the proposed DTL-UDA has the best performance with fewer false positives, and DDL-UDA is the second best. This indicates that the domain-invariant features obtained with subspace interpolation-based methods are more class-discriminative. Also, they can differentiate between healthy and faulty cases *more distinctly than others that fail due to limited training data*.

The proposed method provides an accuracy improvement of $\geq 10\%$ over the best-performing DCNN-based adaptation methods for most adaptation cases. Additionally, it displays superior performance ($\approx \geq 5\%$) over the dictionary counterparts. This hybrid approach combining domain-specific features with

representation learning using transforms provides an effective adaptation for the data-limited scenario considered in this work. The results demonstrate the applicability of the domain-specific features and the proposed subspace-based method for adapting the knowledge learned from one machine ($\mathcal{S}$) to another machine ($\mathcal{T}$) for bearing fault detection.

Table 3.1: Bearing Fault Classification Results (in %) - Set I

Methods	$CWRU \rightarrow CRB$				$CRB \rightarrow CWRU$				$CWRU \rightarrow PU$			
	P	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
MK-MMD [18]	88.44	82.94	81.21	82.94	83.77	83.49	79.55	83.49	84.03	72.86	73.21	72.86
JMMD [18]	87.23	79.34	76.27	79.34	81.54	80.41	76.92	80.41	84.10	73.04	73.40	73.04
CORAL [18]	84.71	71.73	65.48	71.73	86.66	77.59	74.53	77.59	83.34	72.35	72.22	72.35
DANN [18]	88.49	82.53	81.17	82.53	85.13	73.07	67.63	73.07	83.37	75.35	75.81	75.35
CDAN [18]	87.86	81.78	80.26	81.78	50.00	66.67	55.56	66.67	83.27	74.60	75.22	74.60
DL-UDA [1]	81.35	81.08	80.53	81.08	85.03	84.64	80.48	84.64	79.85	80.36	79.78	80.36
DDL-UDA (Chapter 2)	87.01	89.90	87.57	89.90	91.97	87.88	85.07	87.88	86.80	82.30	80.84	82.30
Proposed TL-UDA	90.30	83.10	79.88	83.10	91.93	85.70	82.81	85.70	88.76	80.20	76.64	80.20
Proposed DTL-UDA (N = 3)	**95.88**	**93.91**	**93.33**	**93.91**	**93.46**	**90.03**	**89.06**	**90.03**	**86.54**	**83.55**	**83.04**	**83.55**

Table 3.2: Bearing Fault Classification Results (in %) - Set II

Methods	$PU \rightarrow CWRU$				$CRB \rightarrow PU$				$PU \rightarrow CRB$			
	P	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**
MK-MMD [18]	75.00	80.00	73.33	80.00	79.99	75.26	68.86	75.26	72.17	73.39	65.11	73.39
JMMD [18]	69.38	76.76	71.67	76.76	89.86	82.46	78.85	82.46	66.69	74.43	66.58	74.43
CORAL [18]	79.38	75.00	69.51	75.00	48.24	65.21	54.18	65.21	59.87	68.79	57.94	68.79
DANN [18]	82.26	73.27	68.86	73.27	86.20	82.37	81.74	82.37	84.68	73.05	67.48	73.05
CDAN [18]	85.75	80.28	78.87	80.28	84.36	71.15	64.42	71.15	84.96	73.46	68.33	73.46
DL-UDA [1]	69.13	78.75	72.16	78.75	84.52	78.42	74.74	78.42	81.33	79.71	78.69	79.71
DDL-UDA (Chapter 2)	85.05	85.06	80.61	85.06	86.85	84.16	83.23	84.16	89.51	86.44	83.74	86.44
Proposed TL-UDA	90.32	82.17	77.41	82.17	89.29	81.13	76.97	81.13	87.89	84.39	82.48	84.39
Proposed DTL-UDA (N = 3)	**93.62**	**89.82**	**88.72**	**89.82**	**91.59**	**89.44**	**89.29**	**89.44**	**94.48**	**94.02**	**93.95**	**94.02**

3.4.1 Optimal Parameter Settings

To gain more insights, different experiments were conducted to study the effect of the number of layers and subspaces on the performance of the proposed DTL-UDA method.

3.4.1.1 Effect of number of layers, N

The performance achieved with different deep configurations of DTL-UDA, considering different values of N on the target test data for all the six adaptation scenarios is shown in Fig. 3.5. As we go deep (to some extent), a significant

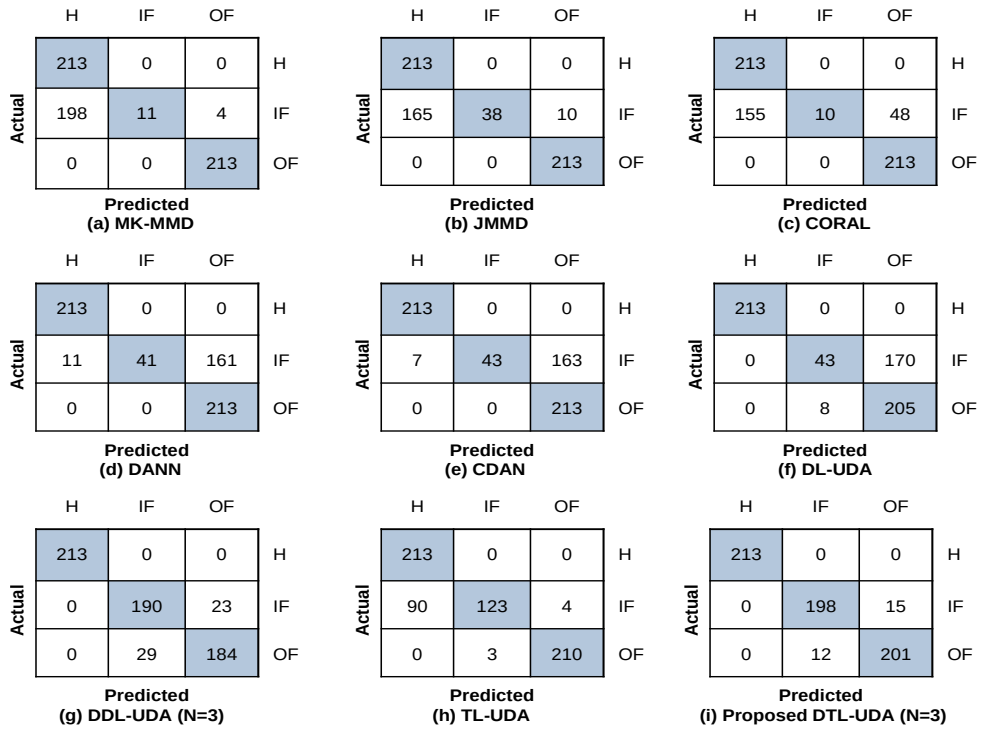


Figure 3.4: Confusion Matrix for $PU \rightarrow CRB$.

performance improvement is observed, both in terms of F1 and Acc. This can be attributed to the fact that deep models can learn more complex relationships from the data, in contrast to the shallow (single-layer) variant. It can be seen that a 3 layer deep model displays superior performance in terms of both metrics for all six scenarios. Going beyond $N = 3$ layers did not help boost the accuracy further; instead, saturation or degradation in performance was observed for the different adaptation scenarios. One possible reason could be limited data availability. As the number of layers of DTL-UDA increases, the number of trainable parameters also increases; hence, more data is required to learn an effective representation. For the data-limited scenario considered in this work, 3 layer DTL-UDA achieved the optimal performance, and therefore, the results of this configuration are presented in Tables 3.1 and 3.2.

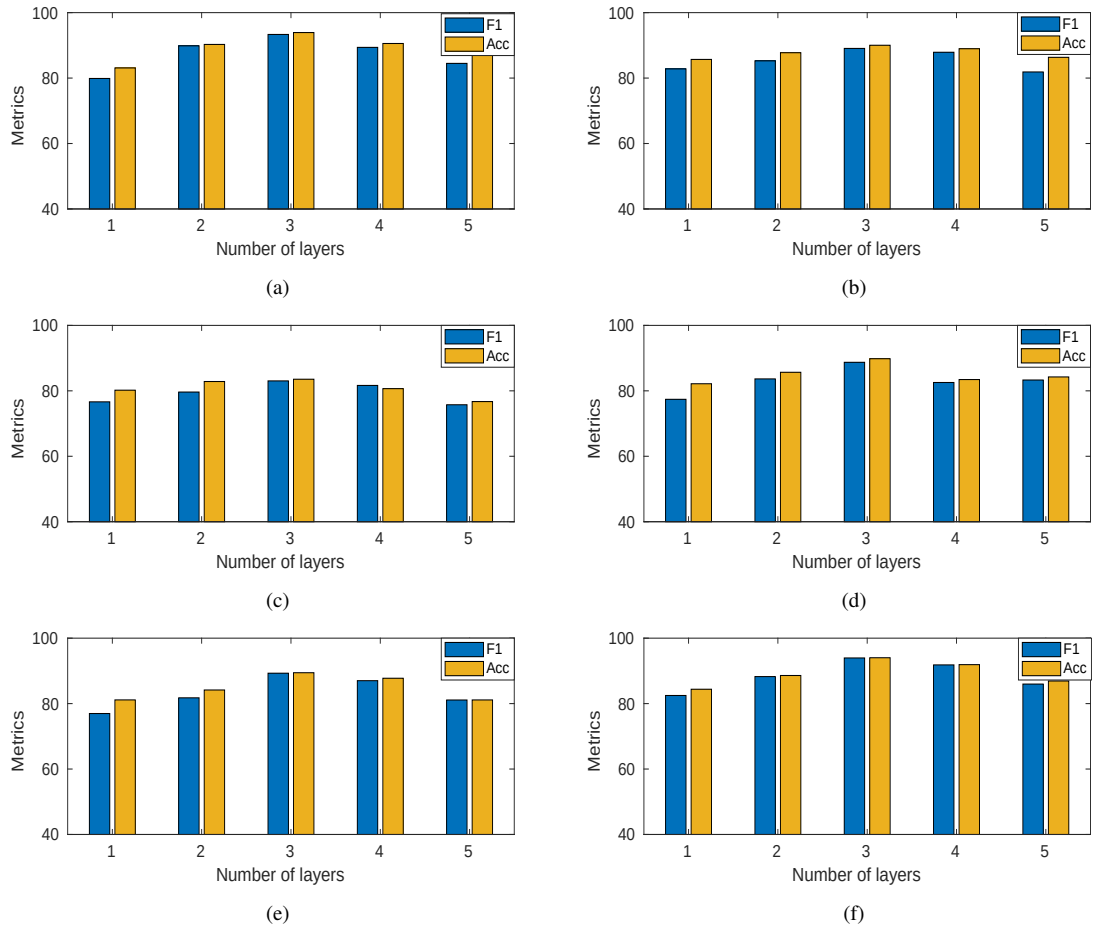


Figure 3.5: Performance on Target Test Data Vs. DTL-UDA configurations employing different number of layers, for all adaptation scenarios. (a) $CWRU \rightarrow CRB$ (b) $CRB \rightarrow CWRU$ (c) $CWRU \rightarrow PU$ (d) $PU \rightarrow CWRU$ (e) $CRB \rightarrow PU$ (f) $PU \rightarrow CRB$.

3.4.1.2 Effect of number of subspaces, M

Here, experiments were conducted with DTL-UDA considering different number of subspaces that connect the $\mathcal{S}$ and $\mathcal{T}$ domains to study the effect on the following: (a) reconstruction residue on target training data, $\boldsymbol{J}_m$, and (b) accuracy on target test data. Since $N = 3$ layer DTL-UDA was observed as the best-performing configuration, a study is carried out considering only N = 1, 2, and 3 layer DTL-UDA configurations. Fig. 3.6 shows the plot of residue at different subspaces (domains) of N = 1, 2, and 3 layer DTL-UDA configurations for two scenarios $CWRU \rightarrow PU$ and $CRB \rightarrow PU$, respectively. Note that

the reconstruction residue keeps decreasing with the increase in the number of subspaces for all the configurations. This is in conformance with the algorithm as the transforms that model each subsequent subspace are computed in such a way that they reduce the residue on the target data. Note that for all DTL-UDA configurations, the residue $\boldsymbol{J}_m$ is highest for $M = 1$ that represents the case of *no adaptation* where the transforms learned using the source domain data ($\boldsymbol{T}_0$) are directly applied on the target data. In Fig. 3.6 (a) and (b), for both scenarios, one can see that the residue saturates after $M = 5$, showing the convergence of the algorithm. This indicates that the transform learned for $M = 5$ represents the target data well, and further interpolation is not required. Additionally, one can see that the residue of the 3 layer DTL-UDA is comparatively less compared to the 1 and 2 layer configuration for $M > 1$, indicating better modeling capability of the deep network.

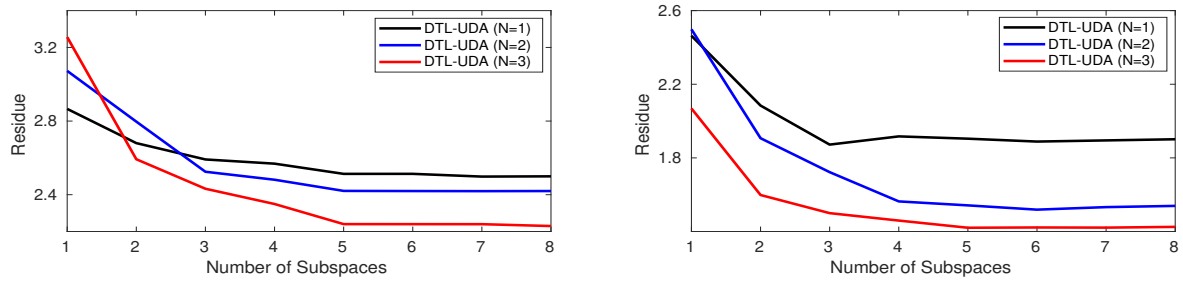
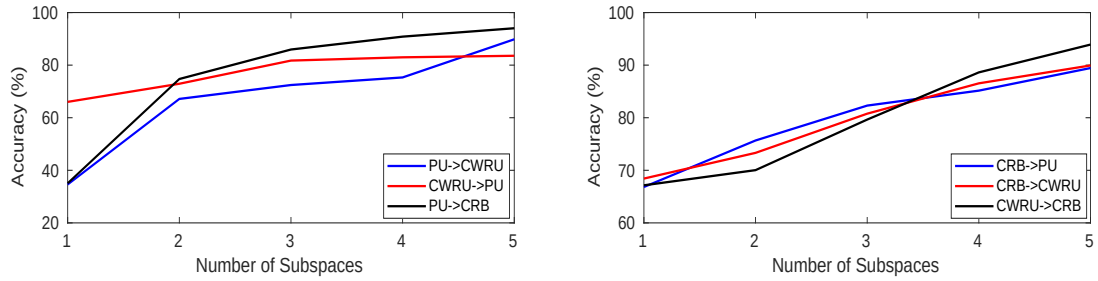


Figure 3.6: Residue on Target Data Vs. Number of Subspaces for different configurations of DTL-UDA. (a) $CWRU \rightarrow PU$ (b) $CRB \rightarrow PU$.

To study the effect of M on the test accuracy of target data, different DTL-UDA configurations are learned considering different values of M. Fig. 3.7 shows the accuracy plot obtained with different subspaces (domains) considered in DTL-UDA configuration for all six adaptation scenarios. A 3 layer DTL-UDA configuration is employed here since it performs the best for all the adaptation

scenarios. Also, results are presented only till M = 5 as no significant improvement in accuracy was observed beyond that due to convergence. In Fig. 3.7 (a) and (b), it can be seen that the accuracy is poor for DTL-UDA configuration with M = 1 (*no adaptation*) across all datasets since the domain discrepancy between $\mathcal{S}$ and $\mathcal{T}$ data is not addressed. Here, the transforms learned using the source domain data ($\boldsymbol{T}_0$) are directly employed on the target data to generate the features which, when fed to the classifier trained on source domain data, estimates the target labels. One can observe that as the value of M increases, the performance improves, with the highest accuracy achieved for the DTL-UDA configuration with M = 5, which is in conformance with the residue plot in Fig. 3.6. Low residue on target training data indicates better adaptation capability, resulting in better accuracy on target test data. Notice a significant improvement in accuracy obtained with adaptation compared to the DTL-UDA configuration without adaptation across different datasets. This demonstrates that the intermediate subspaces effectively absorb the domain shift between $\mathcal{S}$ and $\mathcal{T}$ well, resulting in robust adaptation. Kindly note that Tables 3.1 and 3.2 present the results obtained with the best-performing DTL-UDA configuration obtained through the optimal parameter setting.



(a) (b)

Figure 3.7: Target Test Data Accuracy Vs. Number of Subspaces of DTL-UDA (N = 3). (a) $PU \rightarrow CWRU$, $CWRU \rightarrow PU$ and $PU \rightarrow CRB$. (b) $CRB \rightarrow PU$, $CRB \rightarrow CRWU$ and $CWRU \rightarrow CRB$.

3.5 Summary

This chapter introduces novel transform-based subspace interpolation methods for UDA using TL and DTL frameworks. They are utilized to address the challenging adaptation between *different but related machines*. The methods model the source and target domain data as low-dimensional subspaces using shallow (single-layer) transform and deep transforms, respectively and learns intermediate domains connecting the two domains to generate domain-invariant features for cross-domain classification. The formulations employing TL and DTL, along with closed-form updates, are presented. Experiments on three publicly available bearing datasets demonstrate the effectiveness of our method, achieving an accuracy improvement of $\approx \geq 5\%$ compared to dictionary counterparts and $\geq 10\%$ against the best-performing DCNN benchmark. This highlights the potential of the proposed methods in learning reliable data representations, particularly in the limited data scenario, making them suitable for real-life industrial applications.

Chapter 4

Supervised Domain Adaptation Via Joint Coupled Transform Learning for Multi-Modal Image Super-Resolution

4.1 Motivation

Image Super-Resolution (ISR) refers to enhancing pixel-level image resolution by minimizing the visual artifacts. While different techniques exist, generating the HR version of the LR image is a complex task that involves inferring the missing pixel values, which makes ISR a challenging and ill-posed problem [52]. Recently, with the availability of multi-modal imaging systems, in many practical applications such as remote sensing [53][54], seed viability studies [55], environment monitoring [56], food processing [57], medical field [58] and forensic [59], the scene of interest is often captured by multiple modalities. Although such systems capture enriched sources of information for the scene of interest, factors such as cost, design complexity and data storage pose significant limitations. For example, in remote sensing, satellite imaging systems capture the information from different modalities, such as PAN and MS/HS bands, but at different spatial and spectral resolutions [60]. Usually, the RGB images will have

a high spatial and low spectral resolution, and vice-versa for the HS imaging system. This is done by taking into account the memory constraints, design complexity, communication and processing challenges. Thus, there is a need to use the information from multiple modalities to overcome the resolution limitation of the targeted modality, which has become essential for several downstream applications. Since certain features such as boundaries, textures and edges are common across multiple modalities, these cross-modal dependencies can be modeled to reconstruct the HR image from the LR image of target modality with the guidance of HR image from another modality.

Traditionally, resolution enhancement has been achieved using unimodal image super-resolution methods [61–64]. Recently, MISR methods that leverage information from multiple modalities have gained much interest due to their improved performance. The main objective of MISR is to enhance the resolution of images from the target modality by taking guidance from (HR) images from a different modality that share some common features like edges, textures, etc. Despite various techniques, fusing images from different modalities is not trivial as the correlation among images varies significantly for each multi-modal pair, making it an ill-posed problem.

Typically, deep learning approaches [65–67] achieve superior reconstruction than other methods but demand extensive training data and significant computational resources. However, obtaining the HR target and guidance images for training is a challenge in many practical application scenarios, particularly in remote sensing [60]. Hence, there is a need for methods that work with lim-

ited training data for MISR. In contrast to deep learning, sparse representation learning-based methods employing DL [68, 69] provide enhanced performance, particularly in data-limited scenarios. Lately, sparse representation learning methods employing TL [42] are gaining more attention. It has been shown in [43], [44] that the TL-based approaches are more generalizable and the transforms learned are well-conditioned with less sparsification error compared to poorly conditioned dictionary-based counterparts. They are shown to offer improved accuracy with reduced complexity and better convergence for different application scenarios compared to their DL counterparts [42, 44].

In this work, motivated by the advantages of TL, two novel frameworks for MISR, referred to as Joint Coupled Transform Learning (JCTL-MISR) and Joint Coupled Deep Transform Learning (JCDTL-MISR) are proposed. Since the different imaging modalities capture the same scene of interest, similar to Joint Multi-modal Dictionary Learning (JMDL) method [69], both the proposed TL formulations model the sparse *transform* coefficients of the target HR image as a weighted superposition of the sparse coefficients of the target LR image and the guidance HR image. During the training phase, using the available HR images of the target modality, JCTL-MISR and its deep variant (JCDTL-MISR) learn the individual specific transforms, corresponding coefficients and the appropriate weights using a novel joint optimization framework. In the testing phase, the learned transforms and weights are applied on the LR image of the target modality and the HR image of the guidance modality to reconstruct the HR image of target modality.

Unlike JMDL, in the case of proposed JCTL-MISR, closed-form solution can be obtained for the sparse coefficients and iterative algorithms are not necessary. In contrast to JCTL-MISR, where only single-layer of transform is employed for both the guidance and target modality, in JCDTL-MISR, deeper layers of TL, i.e., Deep Transform Learning (DTL) [49] are employed at both the guidance and target modality to learn rich representation that aids in the super-resolution task. The proposed methods have been benchmarked against state-of-the-art approaches, and results are provided on two different multi-modal imaging scenarios (RGB-NIR & RGB-MS). Results show that the proposed methods provide improved reconstruction, both qualitatively and quantitatively, compared to the state-of-the-art MISR techniques, including the dictionary-based methods. Further, JCDTL-MISR displays superior performance over the shallow (single-layer) JCTL-MISR variant. To the best of our knowledge, TL frameworks have not been investigated for MISR.

The main contributions of the work are summarized below:

- Two novel joint learning formulations for MISR, employing TL referred to as JCTL-MISR and its deep variant JCDTL-MISR are presented to effectively exploit the correlation between the different modalities for improved reconstruction.
- The solutions steps and the derived closed-form updates are provided both for the shallow and deep JCTL-MISR variants.
- Experimental results on two public datasets are presented, supplemented

with comparisons against the benchmarks that demonstrate the ability of the proposed methods to train, even with limited data.

Towards providing the necessary details, the rest of the chapter is organized as follows. Section 4.2 covers related works on MISR. Section 4.3 describes the proposed JCTL-MISR and JCDTL-MISR frameworks for MISR. Description of the datasets and experimental results with comparisons against the benchmark methods are presented in Section 4.4. Finally, the summary of the work is in Section 4.5.

4.2 Related Work

The existing MISR approaches predominantly fall into two categories: (i) Filtering-based methods and (ii) Learning-based methods. Filtering-based methods employ joint image filtering techniques like guided image filtering [70], joint bilateral filtering [71], and joint image restoration [72]. These methods construct joint filters by considering specific features such as edges and textures from the guidance image. However, as shown in [68], these techniques fail in scenarios when the local structures in the guidance and target images are not consistent, resulting in the transfer of erroneous structure details that are not originally present in the target image.

On the other hand, the learning-based techniques learn the cross-modal dependencies between the different imaging modalities from the training samples. The learning-based methods employ deep-learning [65–67] and sparse-representation learning based on DL [68, 69, 73], to learn the complex relationship between

the different modalities. The work in [65] proposed a learning-based approach for joint filtering based on CNNs that employs three sub-networks. The first two sub-networks learn the features from the LR target and HR guidance images. The features of both modalities are concatenated and fed as input to the third sub-network, which selectively transfers the salient features from guidance images to target images to generate HR images of the target modality. Another work [66] introduced a deep HS image sharpening method that learns the image priors via CNN-based residual learning and incorporates the learned deep priors into the LR-HS image and HR-MS image fusion framework to reconstruct the HR-HS image. A recent comprehensive survey on the latest deep learning-based methods in the field of HS image super-resolution is presented in [67]. It discusses various techniques for both unimodal and multi-modal image super-resolution, outlining the characteristics of different methods, their comparisons, and some typical use cases. In general, deep learning methods are known to outperform other approaches but require significant training data and computing resources. They tend to overfit in data-limited scenarios [68, 74]. Moreover, these models cannot guarantee measurement consistency between inputs and outputs during testing [74, 75]. In contrast to deep learning, sparse representation learning-based methods using dictionaries do not exhibit the above shortcomings as shown [68], [73] and provide enhanced performance, particularly in data-limited scenarios.

Sparse representation learning employing DL is a synthesis approach well suited for data/signal reconstruction. They have been explored for addressing MISR problem [68, 69, 73]. The work in [68] presents a Coupled Dictionary

Learning (Coupled DL) approach exploits joint sparse representations induced by coupled dictionaries to capture complex dependencies between different modalities. It essentially learns unique and common dictionaries for the different modalities with the assumption that the common dictionaries share the same sparse representation across modalities. It offers superior performance over deep learning-based methods, especially for data-limited scenarios with faster training speed, and demonstrates robustness to noise. The work in [69] introduces a JMDL approach, which jointly learn dictionaries for each modality (LR of target, HR of target and HR of guidance) and two transform matrices to combine the modalities. Here, the cross-modal dependencies are modeled as a weighted superposition of individual sparse *dictionary* coefficients. The JMDL model is turned into a neural network designed by a coupled unfolding of the Iterative Shrinkage and Thresholding Algorithm (ISTA). The work in [73] presents a deep extension of [69] that converts the JMDL model into a deep neural network, referred to as deep coupled ISTA network.

Motivated by the improved performance of DL methods for MISR for data-limited scenarios, this work introduces TL-based frameworks for MISR for superior reconstruction performance.

4.3 Joint Coupled Transform Learning for Multi-Modal Image Super-Resolution

This section presents the problem formulation and the details of the proposed joint coupled TL methods for MISR employing the shallow (single-layer) and

deep TL frameworks discussed in Chapter 3, Section 3.2.

4.3.1 Problem Definition

The objective of MISR is to generate HR images of target modality $\boldsymbol{Z}$ from LR images of target modality $\boldsymbol{X}$, guided by the HR images of another modality $\boldsymbol{Y}$ by learning the cross-modal relationship between the distinct modalities. Although the images from guidance and target modalities contain distinct features, since they capture the same scene, they share common features such as edges, textures, and shapes that can be exploited for super-resolution tasks.

4.3.2 Proposed MISR method using Joint Coupled Transform Learning (JCTL-MISR)

The proposed method learns dedicated transforms ($\boldsymbol{T}_X, \boldsymbol{T}_Y, \boldsymbol{T}_Z$) and their associated sparse coefficients ($\boldsymbol{H}_X, \boldsymbol{H}_Y, \boldsymbol{H}_Z$) for each of the imaging modalities $\boldsymbol{X}, \boldsymbol{Y}$ and $\boldsymbol{Z}$. Further, since the different modalities capture the same scene of interest, the sparse representations are considered to be related to each other similar to [69, 73] (discussed in Section 4.1). Mathematically, JCTL-MISR models this relationship as a weighted superposition of sparse coefficients: $\boldsymbol{H}_Z = \boldsymbol{W}_X\boldsymbol{H}_X + \boldsymbol{W}_Y\boldsymbol{H}_Y$, where $\boldsymbol{W}_X$ and $\boldsymbol{W}_Y$ are the unknown weight matrices. In the training phase, having the knowledge of $\boldsymbol{Z}$, the different transforms ($\boldsymbol{T}_X, \boldsymbol{T}_Y, \boldsymbol{T}_Z$), associated sparse coefficients ($\boldsymbol{H}_X, \boldsymbol{H}_Y, \boldsymbol{H}_Z$) and weight matrices ($\boldsymbol{W}_X$, $\boldsymbol{W}_Y$) are learnt together in a joint optimization framework. During the test phase, given the LR image of target modality ($\boldsymbol{X}_{test}$) and the HR image of guidance modality ($\boldsymbol{Y}_{test}$), the learned transforms ($\boldsymbol{T}_X, \boldsymbol{T}_Y, \boldsymbol{T}_Z$) and weight matrices ($\boldsymbol{W}_X$, $\boldsymbol{W}_Y$) are used to compute the HR image of target

modality $\boldsymbol{Z}_{test}$. Fig. 4.1 presents the block diagram of the proposed JCTL-MISR method. More details on the training and test phases are presented below.

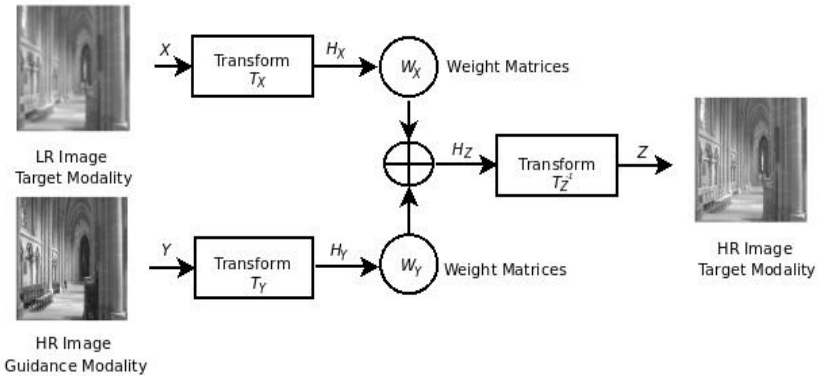


Figure 4.1: Block Diagram of the proposed JCTL-MISR Method

4.3.2.1 Training Phase

In the proposed JCTL-MISR approach, we model the transforms, the corresponding coefficients and the weights matrices using the following:

$$\min_{\substack{\boldsymbol{T}_X,\boldsymbol{T}_Y,\boldsymbol{T}_Z,\boldsymbol{H}_X \\ \boldsymbol{H}_Y,\boldsymbol{H}_Z,\boldsymbol{W}_X,\boldsymbol{W}_Y}} \|\boldsymbol{T}_X\boldsymbol{X}-\boldsymbol{H}_X\|_F^2+\|\boldsymbol{T}_Y\boldsymbol{Y}-\boldsymbol{H}_Y\|_F^2+\|\boldsymbol{T}_Z\boldsymbol{Z}-\boldsymbol{H}_Z\|_F^2+\mu(\sum_{K\in X,Y,Z}\|\boldsymbol{H}_K\|_1)$$
$$+\eta\|\boldsymbol{H}_Z-\boldsymbol{W}_X\boldsymbol{H}_X-\boldsymbol{W}_Y\boldsymbol{H}_Y\|_F^2+\lambda(\sum_{K\in X,Y,Z}(\|\boldsymbol{T}_K\|_F^2-\log\det\boldsymbol{T}_K)) \tag{4.1}$$

where $\{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z}\}\in\mathbb{R}^{d\times n}$ represents the vectorized 2D image patches of d dimension and n denotes the number of patches. Here, $\{\boldsymbol{H}_X,\boldsymbol{H}_Y,\boldsymbol{H}_Z\}\in\mathbb{R}^{d\times n}$, $\{\boldsymbol{T}_X,\boldsymbol{T}_Y,\boldsymbol{T}_Z\}\in\mathbb{R}^{d\times d}$ and $\{\boldsymbol{W}_X,\boldsymbol{W}_Y\}\in\mathbb{R}^{d\times d}$. Since the problem in (4.1) is jointly non-convex, we use an ADMM [76] based variable splitting approach to determine the transforms, corresponding coefficients and weight matrices. All the parameters in the updating algorithm are initialized with random matrices having real numbers between 0 and 1 drawn from a uniform distribution based on the size of the matrices.

The transforms $\boldsymbol{T}_X,\boldsymbol{T}_Y,\boldsymbol{T}_Z$ are learned by solving the following:

$$\begin{aligned}
&\min_{\boldsymbol{T}_X} \|\boldsymbol{T}_X\boldsymbol{X} - \boldsymbol{H}_X\|_F^2 + \lambda(\|\boldsymbol{T}_X\|_F^2 - \log\det \boldsymbol{T}_X) \\
&\min_{\boldsymbol{T}_Y} \|\boldsymbol{T}_Y\boldsymbol{Y} - \boldsymbol{H}_Y\|_F^2 + \lambda(\|\boldsymbol{T}_Y\|_F^2 - \log\det \boldsymbol{T}_Y) \\
&\min_{\boldsymbol{T}_Z} \|\boldsymbol{T}_Z\boldsymbol{Z} - \boldsymbol{H}_Z\|_F^2 + \lambda(\|\boldsymbol{T}_Z\|_F^2 - \log\det \boldsymbol{T}_Z)
\end{aligned} \tag{4.2}$$

The above-listed problems are similar to (3.5). The closed-form updates for $\boldsymbol{T}_X$, $\boldsymbol{T}_Y$ and $\boldsymbol{T}_Z$ are obtained using the standard updates (3.6) to (3.8) discussed in Chapter 3, Section 3.2, by appropriately changing the inputs. The transform coefficients $\boldsymbol{H}_X, \boldsymbol{H}_Y, \boldsymbol{H}_Z$ are learnt using:

$$\begin{aligned}
&\min_{\boldsymbol{H}_X} \|\boldsymbol{T}_X\boldsymbol{X} - \boldsymbol{H}_X\|_F^2 + \eta\|\boldsymbol{H}_Z - \boldsymbol{W}_X\boldsymbol{H}_X - \boldsymbol{W}_Y\boldsymbol{H}_Y\|_F^2 + \mu\|\boldsymbol{H}_X\|_1 \\
&\min_{\boldsymbol{H}_Y} \|\boldsymbol{T}_Y\boldsymbol{Y} - \boldsymbol{H}_Y\|_F^2 + \eta\|\boldsymbol{H}_Z - \boldsymbol{W}_X\boldsymbol{H}_X - \boldsymbol{W}_Y\boldsymbol{H}_Y\|_F^2 + \mu\|\boldsymbol{H}_Y\|_1 \\
&\min_{\boldsymbol{H}_Z} \|\boldsymbol{T}_Z\boldsymbol{Z} - \boldsymbol{H}_Z\|_F^2 + \eta\|\boldsymbol{H}_Z - \boldsymbol{W}_X\boldsymbol{H}_X - \boldsymbol{W}_Y\boldsymbol{H}_Y\|_F^2 + \mu\|\boldsymbol{H}_Z\|_1
\end{aligned} \tag{4.3}$$

Here the closed-form updates for transform coefficients are derived by differentiating the equations with respect to the corresponding variables and then equating to zero. Since the problem involves coupling, the standard approach of iterative soft-thresholding [77] is used due to the presence of l_1-norm. Using basic matrix manipulation and soft thresholding the closed-form expressions can be written as:

$$\begin{aligned}
\boldsymbol{H}_X &= \text{sign}(\boldsymbol{A}_X).\max(0, |\boldsymbol{A}_X| - \boldsymbol{B}_X) \\
\boldsymbol{H}_Y &= \text{sign}(\boldsymbol{A}_Y).\max(0, |\boldsymbol{A}_Y| - \boldsymbol{B}_Y) \\
\boldsymbol{H}_Z &= \text{sign}(\boldsymbol{A}_Z).\max(0, |\boldsymbol{A}_Z| - \boldsymbol{B}_Z)
\end{aligned} \tag{4.4}$$

where $\boldsymbol{A}_X = (\boldsymbol{I} + \eta\boldsymbol{W}_X{}^T\boldsymbol{W}_X)^{-1}(\eta\boldsymbol{W}_X{}^T(\boldsymbol{H}_Z - \boldsymbol{W}_Y\boldsymbol{H}_Y) + \boldsymbol{T}_X\boldsymbol{X})$, $\boldsymbol{B}_X = (\boldsymbol{I} + \eta\boldsymbol{W}_X{}^T\boldsymbol{W}_X)^{-1}(\frac{\mu}{2}\boldsymbol{J})$, $\boldsymbol{A}_Y = (\boldsymbol{I} + \eta\boldsymbol{W}_Y{}^T\boldsymbol{W}_Y)^{-1}(\eta\boldsymbol{W}_Y{}^T(\boldsymbol{H}_Z - \boldsymbol{W}_X\boldsymbol{H}_X) + \boldsymbol{T}_Y\boldsymbol{Y})$, $\boldsymbol{B}_Y = (\boldsymbol{I} + \eta\boldsymbol{W}_Y{}^T\boldsymbol{W}_Y)^{-1}(\frac{\mu}{2}\boldsymbol{J})$, $\boldsymbol{A}_Z = \frac{1}{1+\eta}.(\eta(\boldsymbol{W}_X\boldsymbol{H}_X + \boldsymbol{W}_Y\boldsymbol{H}_Y) + \boldsymbol{T}_Z\boldsymbol{Z})$, and $\boldsymbol{B}_Z = \frac{\mu}{2(1+\eta)}\boldsymbol{J}$, here $\boldsymbol{J}$ denotes a matrix of all ones of size $d \times n$.

The weight matrices $\boldsymbol{W}_X$ and $\boldsymbol{W}_Y$ are learned using the following expressions:

$$\begin{aligned} \min_{\boldsymbol{W}_X} \|\boldsymbol{H}_Z - \boldsymbol{W}_X \boldsymbol{H}_X - \boldsymbol{W}_Y \boldsymbol{H}_Y\|_F^2 \\ \min_{\boldsymbol{W}_Y} \|\boldsymbol{H}_Z - \boldsymbol{W}_X \boldsymbol{H}_X - \boldsymbol{W}_Y \boldsymbol{H}_Y\|_F^2 \end{aligned} \tag{4.5}$$

Following the standard least square method, the closed-form updates for (4.5) can be expressed as:

$$\begin{aligned} \boldsymbol{W}_X &= (\boldsymbol{H}_Z - \boldsymbol{W}_Y \boldsymbol{H}_Y)\boldsymbol{H}_X^{\dagger} \\ \boldsymbol{W}_Y &= (\boldsymbol{H}_Z - \boldsymbol{W}_X \boldsymbol{H}_X)\boldsymbol{H}_Y^{\dagger} \end{aligned} \tag{4.6}$$

where † denotes the pseudo-inverse. Note that the transforms, their coefficients and weight matrices are updated iteratively until the objective function in (4.1) converges, i.e., the loss of (4.1) over the subsequent iterations does not change significantly, with the absolute value being less than an empirically calculated threshold τ. This concludes the training phase, where the transforms of the respective modalities and weight matrices (coupling terms) are learned. The pseudo-code of the training phase of the proposed JCTL-MISR method is summarized in Algorithm 4.

Algorithm 4: JCTL for MISR (JCTL-MISR)

Input: $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$
Parameters: $\lambda, \mu, \eta, \tau, k$ (number of transform atoms)
Initialization: Set transforms $\boldsymbol{T}_X$, $\boldsymbol{T}_Y$, $\boldsymbol{T}_Z$, weight matrices $\boldsymbol{W}_X$, $\boldsymbol{W}_Y$, and coefficients $\boldsymbol{H}_X$, $\boldsymbol{H}_Y$, $\boldsymbol{H}_Z$, to random matrix with real numbers between 0 and 1 drawn from a uniform distribution, iteration $m = 0$
do
    Update the transforms $\boldsymbol{T}_X$, $\boldsymbol{T}_Y$ and $\boldsymbol{T}_Z$ from $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$ using (4.2).
    Compute the transform coefficients $\boldsymbol{H}_X$, $\boldsymbol{H}_Y$ and $\boldsymbol{H}_Z$ using (4.4).
    Update the weight matrices $\boldsymbol{W}_X$, and $\boldsymbol{W}_Y$ using (4.6).
    $m = m + 1$
while (JCTL-MISR loss (4.1) $< \tau$)
Output: $\boldsymbol{T}_X, \boldsymbol{T}_Y, \boldsymbol{T}_Z, \boldsymbol{W}_X, \boldsymbol{W}_Y$ (modality-specific transforms and weight matrices)

4.3.2.2 Test Phase

Given the test LR image of target modality $\boldsymbol{X}_{test}$ and the HR image of guidance modality $\boldsymbol{Y}_{test}$, the sparse representations $\boldsymbol{H}_{X_{test}}$ and $\boldsymbol{H}_{Y_{test}}$ are computed using:

$$\begin{aligned} \min_{\boldsymbol{H}_{X_{test}}} \|\boldsymbol{T}_X \boldsymbol{X}_{test} - \boldsymbol{H}_{X_{test}}\|_F^2 + \mu \|\boldsymbol{H}_{X_{test}}\|_1 \\ \min_{\boldsymbol{H}_{Y_{test}}} \|\boldsymbol{T}_Y \boldsymbol{Y}_{test} - \boldsymbol{H}_{Y_{test}}\|_F^2 + \mu \|\boldsymbol{H}_{Y_{test}}\|_1 \end{aligned} \tag{4.7}$$

The closed-form update of $\boldsymbol{H}_{X_{test}}$ is the standard expression for LASSO based optimization problems [78] and the equation is given below:

$$\boldsymbol{H}_{\boldsymbol{X_{test}}} = \text{sign}(\boldsymbol{T}_X \boldsymbol{X}_{test}).\max\left(0, |\boldsymbol{T}_X \boldsymbol{X}_{test}| - \frac{\mu}{2}\right) \tag{4.8}$$

Similar equations can be derived for $\boldsymbol{H}_{Y_{test}}$ by using $\boldsymbol{T}_Y$ and $\boldsymbol{Y}_{test}$. Using the cross-modality model, the sparse representation $\boldsymbol{H}_{Z_{test}}$ can be computed from $\boldsymbol{H}_{X_{test}}$ and $\boldsymbol{H}_{Y_{test}}$ as: $\boldsymbol{H}_{Z_{test}} = \boldsymbol{W}_X \boldsymbol{H}_{X_{test}} + \boldsymbol{W}_Y \boldsymbol{H}_{Y_{test}}$.

With the help of learnt $\boldsymbol{T}_Z$, the reconstructed HR image of target modality $\boldsymbol{Z}_{test}$ can then be obtained as: $\boldsymbol{Z}_{test} = \boldsymbol{T}_Z^{\dagger} \boldsymbol{H}_{Z_{test}}$.

4.3.3 Proposed MISR method using Joint Coupled Deep Transform Learning (JCDTL-MISR)

The proposed approach is a deep version of the JCTL-MISR method discussed in Section 4.3.2. It employs deep transforms for learning rich representations from the different imaging modalities for improved MISR performance. The block diagram of the proposed JCDTL-MISR method is presented in Fig. 4.2. Here, N layer deep transforms are learnt from $\boldsymbol{X}$ and $\boldsymbol{Y}$ utilizing the DTL formulation (3.10) (discussed in Chapter 3, Section 3.2), while a single-layer transform is learnt from $\boldsymbol{Z}$. Similar to JCTL-MISR method discussed in Section 4.3.2, the

transform coefficients are related to each other since the different modalities capture the same scene of interest. This relationship is mathematically expressed as: $\boldsymbol{H}_Z = \boldsymbol{W}_X \boldsymbol{H}_{XN} + \boldsymbol{W}_Y \boldsymbol{H}_{YN}$ where $\boldsymbol{W}_X$, $\boldsymbol{W}_Y$ are the unknown weight matrices, $\boldsymbol{H}_Z$ is the representation (or coefficients) for $\boldsymbol{Z}$ and $\boldsymbol{H}_{XN}$, $\boldsymbol{H}_{YN}$ are the N^{th} layer DTL coefficients learnt from $\boldsymbol{X}$ and $\boldsymbol{Y}$ respectively. The JCDTL-MISR optimization formulation is given as:

$$\begin{aligned}
&\min_{\substack{\boldsymbol{T}_{Xi}'s, \boldsymbol{T}_{Yi}'s, \boldsymbol{T}_Z, \boldsymbol{W}_X, \\ \boldsymbol{W}_Y, \boldsymbol{H}_{Xi}'s, \boldsymbol{H}_{Yi}'s, \boldsymbol{H}_Z}} \left\|\boldsymbol{T}_{XN}(\boldsymbol{T}_{X(N-1)}(..(\boldsymbol{T}_{X1}\boldsymbol{X}))) - \boldsymbol{H}_{XN}\right\|_F^2 \\
&+ \left\|\boldsymbol{T}_{YN}(\boldsymbol{T}_{Y(N-1)}(..(\boldsymbol{T}_{Y1}Y))) - \boldsymbol{H}_{YN}\right\|_F^2 + \|\boldsymbol{T}_Z\boldsymbol{Z} - \boldsymbol{H}_Z\|_F^2 \\
&+ \lambda\Big(\sum_{K\in X,Y}\sum_{i=1}^{N}(\|\boldsymbol{T}_{Ki}\|_F^2 - \log\det\boldsymbol{T}_{Ki})\Big) + \lambda(\|\boldsymbol{T}_Z\|_F^2 - \log\det\boldsymbol{T}_Z) \\
&+ \eta\|\boldsymbol{H}_Z - \boldsymbol{W}_X\boldsymbol{H}_{XN} - \boldsymbol{W}_Y\boldsymbol{H}_{YN}\|_F^2 + \mu\Big(\sum_{K\in X,Y}\|\boldsymbol{H}_{KN}\|_1 + \|\boldsymbol{H}_Z\|_1\Big) \\
&\text{s.t. } \boldsymbol{T}_{X(N-1)}(..(\boldsymbol{T}_{X1}\boldsymbol{X})) \geq 0,.., \boldsymbol{T}_{X1}\boldsymbol{X} \geq 0, \boldsymbol{T}_{Y(N-1)}(..(\boldsymbol{T}_{Y1}\boldsymbol{Y})) \geq 0,.., \\
&\boldsymbol{T}_{Y1}\boldsymbol{Y} \geq 0.
\end{aligned} \tag{4.9}$$

Here for $i = 1, .., N$, $\{\boldsymbol{T}_{Xi}, \boldsymbol{T}_{Yi}\}$ are the deep transforms and $\{\boldsymbol{H}_{Xi}, \boldsymbol{H}_{Yi}\}$ are their associated coefficients that are learnt from $\boldsymbol{X}$ and $\boldsymbol{Y}$. $\boldsymbol{T}_Z$ is the single-layer transform that is learnt from $\boldsymbol{Z}$ to generate the $\boldsymbol{H}_Z$. Sparsity is enforced on the coefficients $\boldsymbol{H}_{XN}$, $\boldsymbol{H}_{YN}$ and $\boldsymbol{H}_Z$ using the l_1-norm constraint. For the coefficients associated with other layers of the deep transforms for $\boldsymbol{X}$ and $\boldsymbol{Y}$ modalities, a ReLU type non-linearity is considered between the deep layers by forcing the negative values of the coefficients to 0. Here, λ, μ and η are the tunable hyperparameters whose optimal values are obtained using grid search.

In the rest of the section, without loss of generality, we provide the solution

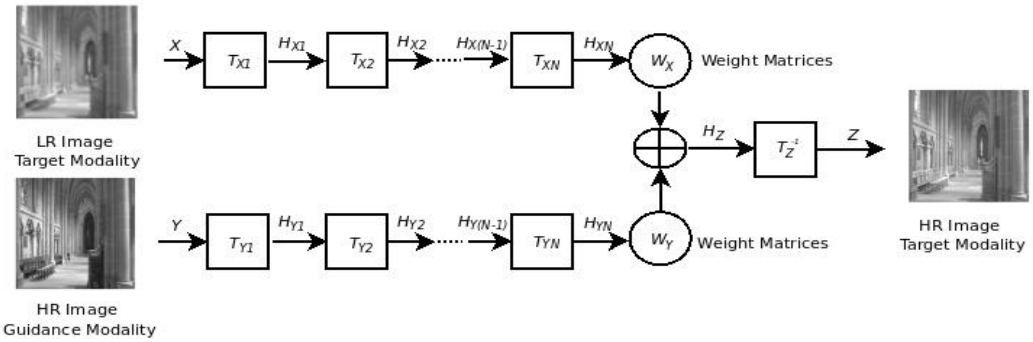


Figure 4.2: Block Diagram of the proposed JCDTL-MISR Method

steps of the above optimization formulation by assuming $N = 3$. One can derive the solution by following similar steps for any general N. The joint formulation of (4.9) for a 3 layer deep network can be expressed as:

$$\begin{aligned}
&\min_{\substack{\boldsymbol{T}_{X1},\boldsymbol{T}_{X2},\boldsymbol{T}_{X3},\boldsymbol{T}_{Y1},\boldsymbol{T}_{Y2},\boldsymbol{T}_{Y3},\boldsymbol{T}_{Z},\boldsymbol{W}_{X},\\ \boldsymbol{W}_{Y},\boldsymbol{H}_{X1},\boldsymbol{H}_{X2},\boldsymbol{H}_{X3},\boldsymbol{H}_{Y1},\boldsymbol{H}_{Y2},\boldsymbol{H}_{Y3},\boldsymbol{H}_{Z}}} (\|\boldsymbol{T}_{X3}\boldsymbol{H}_{X2}-\boldsymbol{H}_{X3}\|_F^2+\|\boldsymbol{T}_{X2}\boldsymbol{H}_{X1}-\boldsymbol{H}_{X2}\|_F^2 \\
&+\|\boldsymbol{T}_{X1}\boldsymbol{X}-\boldsymbol{H}_{X1}\|_F^2)+(\|\boldsymbol{T}_{Y3}\boldsymbol{H}_{Y2}-\boldsymbol{H}_{Y3}\|_F^2+\|\boldsymbol{T}_{Y2}\boldsymbol{H}_{Y1}-\boldsymbol{H}_{Y2}\|_F^2 \\
&+\|\boldsymbol{T}_{Y1}\boldsymbol{Y}-\boldsymbol{H}_{Y1}\|_F^2)+\|\boldsymbol{T}_{Z}\boldsymbol{Z}-\boldsymbol{H}_{Z}\|_F^2+\lambda(\sum_{K\in X,Y}\sum_{i=1}^{3}(\|\boldsymbol{T}_{Ki}\|_F^2-\log\det\boldsymbol{T}_{Ki})) \\
&+\lambda\|\boldsymbol{T}_{Z}\|_F^2-\log\det\boldsymbol{T}_{Z})+\eta\|\boldsymbol{H}_{Z}-\boldsymbol{W}_{X}\boldsymbol{H}_{X3}-\boldsymbol{W}_{Y}\boldsymbol{H}_{Y3}\|_F^2 \\
&+\mu(\sum_{K\in X,Y}\|\boldsymbol{H}_{K3}\|_1+\|\boldsymbol{H}_{Z}\|_1) \\
&\text{s.t. } \boldsymbol{T}_{X2}(\boldsymbol{T}_{X1}\boldsymbol{X})\geq 0,\ \boldsymbol{T}_{X1}\boldsymbol{X}\geq 0,\ \text{similarly, } \boldsymbol{T}_{Y2}(\boldsymbol{T}_{Y1}\boldsymbol{Y})\geq 0,\ \boldsymbol{T}_{Y1}\boldsymbol{Y}\geq 0.
\end{aligned} \tag{4.10}$$

Here, the first six terms are the data fidelity terms corresponding to 3 layer deep transforms for $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively. The seventh term is for learning the single-layer transform for $\boldsymbol{Z}$. The eighth and ninth terms are regularization penalties on the transforms. The tenth term is the coupling term, and the rest are the additional constraints on the transform coefficients that enforce sparse solutions. Here, $\{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z}\}\in\mathbb{R}^{d\times n}$ represent the vectorized 2D image patches of dimension d features (pixels) with n measurements. In our case, the transforms and the

weight matrices, $\{\boldsymbol{T}_{X1}, \boldsymbol{T}_{X2}, \boldsymbol{T}_{X3}, \boldsymbol{T}_{Y1}, \boldsymbol{T}_{Y2}, \boldsymbol{T}_{Y3}, \boldsymbol{T}_{Z}, \boldsymbol{W}_{X}, \boldsymbol{W}_{Y}\} \in \mathbb{R}^{d\times d}$ and coefficients $\{\boldsymbol{H}_{X1}, \boldsymbol{H}_{X2}, \boldsymbol{H}_{X3}, \boldsymbol{H}_{Y1}, \boldsymbol{H}_{Y2}, \boldsymbol{H}_{Y3}, \boldsymbol{H}_{Z}\} \in \mathbb{R}^{d\times n}$.

Similar to JCTL-MISR method discussed in Section 4.3.2, this method has a training and test phase. In the training phase, having the knowledge of $\boldsymbol{Z}$, the different deep transforms for $\boldsymbol{X}$ and $\boldsymbol{Y}$, shallow transform for $\boldsymbol{Z}$, and their associated sparse coefficients are learned along with the weight matrices ($\boldsymbol{W}_X$, $\boldsymbol{W}_Y$) by solving (4.10). These learned transforms and weight matrices are utilized in the test phase to compute the HR image of target modality $\boldsymbol{Z}_{test}$ from LR image of target modality $\boldsymbol{X}_{test}$ and HR image of guidance modality $\boldsymbol{Y}_{test}$. More details on these two phases are presented below.

4.3.3.1 Training Phase

In this phase, we employ ADMM based variable splitting approach to compute the closed-form updates for the transforms, corresponding coefficients and weight matrices for the optimization problem presented in (4.10). The sub-problems to solve for updating the deep transforms for the modality $\boldsymbol{X}$ are given as:

$$\min_{\boldsymbol{T}_{X1}} \| \boldsymbol{T}_{X1}\boldsymbol{X} - \boldsymbol{H}_{X1} \|_F^2 + \lambda(\|\boldsymbol{T}_{X1}\|_F^2 - \log\det \boldsymbol{T}_{X1}) \tag{4.11}$$

$$\min_{\boldsymbol{T}_{X2}} \|\boldsymbol{T}_{X2}\boldsymbol{H}_{X1} - \boldsymbol{H}_{X2}\|_F^2 + \lambda(\|\boldsymbol{T}_{X2}\|_F^2 - \log\det \boldsymbol{T}_{X2}) \tag{4.12}$$

$$\min_{\boldsymbol{T}_{X3}} \|\boldsymbol{T}_{X3}\boldsymbol{H}_{X2} - \boldsymbol{H}_{X3}\|_F^2 + \lambda(\|\boldsymbol{T}_{X3}\|_F^2 - \log\det \boldsymbol{T}_{X3}) \tag{4.13}$$

In the similar way, the sub-problems to solve for updating the deep transforms for modality $\boldsymbol{Y}$ can be obtained by replacing $\boldsymbol{X}$ with $\boldsymbol{Y}$ in the above (4.11)-(4.13). Now, the sub-problem to solve for the transform $\boldsymbol{T}_Z$ is given as:

$$\min_{\boldsymbol{T}_{Z}} \|\boldsymbol{T}_{Z}\boldsymbol{Z} - \boldsymbol{H}_{Z}\|_F^2 + \lambda(\|\boldsymbol{T}_{Z}\|_F^2 - \log\det \boldsymbol{T}_{Z}) \tag{4.14}$$

Notice that (4.11)-(4.14) resemble (3.5) and hence the standard closed-form updates (3.6) to (3.8) discussed in Chapter 3, Section 3.2 can be used for updating the transforms by appropriately changing the inputs.

The coefficients $\boldsymbol{H}_{X1}, \boldsymbol{H}_{X2}$ for modality $\boldsymbol{X}$ are computed by solving the sub-problems given below.

$$\min_{\boldsymbol{H}_{X1}} \|\boldsymbol{T}_{X2}\boldsymbol{H}_{X1} - \boldsymbol{H}_{X2}\|_F^2 + \|\boldsymbol{T}_{X1}\boldsymbol{X} - \boldsymbol{H}_{X1}\|_F^2, \quad s.t.\ \boldsymbol{H}_{X_1} \geq 0. \tag{4.15}$$

$$\min_{\boldsymbol{H}_{X2}} \|\boldsymbol{T}_{X3}\boldsymbol{H}_{X2} - \boldsymbol{H}_{X3}\|_F^2 + \|\boldsymbol{T}_{X2}\boldsymbol{H}_{X1} - \boldsymbol{H}_{X2}\|_F^2, \quad s.t.\ \boldsymbol{H}_{X_2} \geq 0. \tag{4.16}$$

The closed-form solutions for $\boldsymbol{H}_{X1}, \boldsymbol{H}_{X2}$ are obtained by taking a derivative of the sub-problems with respect to the argument variable and equating it to 0. This results in the following updates:

$$\boldsymbol{H}_{X1} = \max(0, (\boldsymbol{I} + \boldsymbol{T}_{X2}{}^T\boldsymbol{T}_{X2})^\dagger.(\boldsymbol{T}_{X2}{}^T\boldsymbol{H}_{X2} + \boldsymbol{T}_{X1}\boldsymbol{X})) \tag{4.17}$$

$$\boldsymbol{H}_{X2} = \max(0, (\boldsymbol{I} + \boldsymbol{T}_{X3}{}^T\boldsymbol{T}_{X3})^\dagger.(\boldsymbol{T}_{X3}{}^T\boldsymbol{H}_{X3} + \boldsymbol{T}_{X2}\boldsymbol{H}_{X1})) \tag{4.18}$$

where the above $\max(.)$ denotes a ReLU type non-linearity considered between the deep layers. Similarly, the updates for the coefficient $\boldsymbol{H}_{Y1}, \boldsymbol{H}_{Y2}$ for modality $\boldsymbol{Y}$ can be obtained by replacing $\boldsymbol{X}$ with $\boldsymbol{Y}$ in (4.17) and (4.18). The coefficients of the last layer i.e., N = 3 in this case is estimated by solving the following:

$$\min_{\boldsymbol{H}_{X3}} \|\boldsymbol{T}_{X3}\boldsymbol{H}_{X2} - \boldsymbol{H}_{X3}\|_F^2 + \mu \|\boldsymbol{H}_{X3}\|_1 + \eta(\|\boldsymbol{H}_Z - \boldsymbol{W}_X\boldsymbol{H}_{X3} - \boldsymbol{W}_Y\boldsymbol{H}_{Y3}\|_F^2) \tag{4.19}$$

Due to the l_1-norm constraint on $\boldsymbol{H}_{X3}$, basic matrix manipulation and soft thresholding is used similar to the previous JCTL-MISR method (discussed in Section 4.3.2) to obtain the closed-form update:

$$\boldsymbol{H}_{X3} = \text{sign}(\boldsymbol{A}_{\boldsymbol{X}}).\max(0, |\boldsymbol{A}_{\boldsymbol{X}}| - \boldsymbol{B}_{\boldsymbol{X}}) \tag{4.20}$$

where $\boldsymbol{A}_{\boldsymbol{X}} = \boldsymbol{D}^\dagger.(\eta\boldsymbol{W}_X{}^T(\boldsymbol{H}_Z - \boldsymbol{W}_Y\boldsymbol{H}_{Y3}) + \boldsymbol{T}_{X3}\boldsymbol{H}_{X2})$, $\boldsymbol{B}_{\boldsymbol{X}} = \boldsymbol{D}^\dagger.\ (\frac{\mu}{2}\boldsymbol{J})$ and

$\boldsymbol{D} = \boldsymbol{I} + \eta \boldsymbol{W}_X{}^T \boldsymbol{W}_X$, $\boldsymbol{J}$ is an all ones matrix. In the similar way, the coefficients $\boldsymbol{H}_{Y3}$ associated with modality $\boldsymbol{Y}$ are updated with the following closed-form update:

$$\boldsymbol{H}_{Y3} = \text{sign}(\boldsymbol{A_Y}).\max(0, |\boldsymbol{A_Y}| - \boldsymbol{B_Y}) \tag{4.21}$$

where $\boldsymbol{A_Y} = \boldsymbol{E}^{\dagger}.\ (\eta \boldsymbol{W}_Y{}^T(\boldsymbol{H}_Z - \boldsymbol{W_X}\boldsymbol{H}_{X3}) + \boldsymbol{T}_{Y3}\boldsymbol{H}_{Y2})$, $\boldsymbol{B_Y} = \boldsymbol{E}^{\dagger}.\ (\frac{\mu}{2}\boldsymbol{J})$ and $\boldsymbol{E} = \boldsymbol{I} + \eta \boldsymbol{W}_Y{}^T \boldsymbol{W}_Y$. The sub-problem and the associated closed-form solution for $\boldsymbol{H}_Z$ is given as:

$$\min_{\boldsymbol{H}_Z} \|\boldsymbol{T}_Z\boldsymbol{Z} - \boldsymbol{H}_Z\|_F^2 + \mu \|\boldsymbol{H}_Z\|_1 + \eta(\|\boldsymbol{H}_Z - \boldsymbol{W}_X\boldsymbol{H}_{X3} - \boldsymbol{W}_Y\boldsymbol{H}_{Y3}\|_F^2) \tag{4.22}$$

$$\boldsymbol{H}_Z = \text{sign}(\boldsymbol{A_Z}).\max(0, |\boldsymbol{A_Z}| - \boldsymbol{B_Z}) \tag{4.23}$$

where $\boldsymbol{A_Z} = \frac{1}{(1+\eta)}.(\eta(\boldsymbol{W}_X\boldsymbol{H}_{X3} + \boldsymbol{W}_Y\boldsymbol{H}_{Y3}) + \boldsymbol{T_Z}\boldsymbol{Z})$ and $\boldsymbol{B_Z} = \frac{\mu}{2(1+\eta)}$. The weight matrices, $\boldsymbol{W}_X$ and $\boldsymbol{W}_Y$ are updated by solving the following sub-problems:

$$\min_{\boldsymbol{W}_X} \|\boldsymbol{H}_Z - \boldsymbol{W}_X\boldsymbol{H}_{X3} - \boldsymbol{W}_Y\boldsymbol{H}_{Y3}\|_F^2 \tag{4.24}$$

$$\min_{\boldsymbol{W}_Y} \|\boldsymbol{H}_Z - \boldsymbol{W}_X\boldsymbol{H}_{X3} - \boldsymbol{W}_Y\boldsymbol{H}_{Y3}\|_F^2 \tag{4.25}$$

Kindly note the above problems are similar to (4.5) of the JCTL-MISR method. Hence, the closed-form updates given in (4.6) are used updating for $\boldsymbol{W}_X$ and $\boldsymbol{W}_Y$ by changing the inputs appropriately.

Similar to JCTL-MISR method discussed in Section 4.3.2, the transforms and coefficients update go through many iterations until convergence is met. This completes the training phase and the pseudo-code of the same is summarized in Algorithm 5.

4.3.3.2 Test Phase

In this phase, the learnt transforms for the different modalities and the weight matrices are utilized to compute the coefficients for the test data $\boldsymbol{X}_{test}$ and $\boldsymbol{Y}_{test}$

Algorithm 5: JCDTL for MISR (JCDTL-MISR)

Input: $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$
Parameters: $\lambda, \mu, \eta, \tau, k$ (number of transform atoms), $N = 3$ (number of layers)
Initialization: Set transforms $\{\boldsymbol{T}_{Xi}\}_{i=1}^{N}$, $\{\boldsymbol{T}_{Yi}\}_{i=1}^{N}$, $\boldsymbol{T}_Z$, weight matrices $\boldsymbol{W}_X$, $\boldsymbol{W}_Y$, and coefficients $\{\boldsymbol{H}_{Xi}\}_{i=1}^{N}$, $\{\boldsymbol{H}_{Yi}\}_{i=1}^{N}$, $\boldsymbol{H}_Z$, to random matrix with real numbers between 0 and 1 drawn from a uniform distribution, iteration $m = 0$
do
    Update the deep transforms $\boldsymbol{T}_{X1}$, $\boldsymbol{T}_{X2}$ and $\boldsymbol{T}_{X3}$ for modality $\boldsymbol{X}$ using (4.11), (4.12) and (4.13).
    Update the deep transforms $\boldsymbol{T}_{Y1}$, $\boldsymbol{T}_{Y2}$ and $\boldsymbol{T}_{Y3}$ for modality $\boldsymbol{Y}$ by replacing $\boldsymbol{X}$ with $\boldsymbol{Y}$ in (4.11), (4.12) and (4.13).
    Update the transform $\boldsymbol{T}_Z$ using (4.14).
    Update the deep transform coefficients $\boldsymbol{H}_{X1}$, $\boldsymbol{H}_{X2}$ and $\boldsymbol{H}_{X3}$ for modality $\boldsymbol{X}$ using (4.17), (4.18) and (4.20).
    Update the deep transform coefficients $\boldsymbol{H}_{Y1}$, $\boldsymbol{H}_{Y2}$ for modality $\boldsymbol{Y}$ by replacing $\boldsymbol{X}$ with $\boldsymbol{Y}$ in (4.17), (4.18) and update $\boldsymbol{H}_{Y3}$ using (4.21).
    Compute the transform coefficients $\boldsymbol{H}_Z$ for modality $\boldsymbol{Z}$ using (4.23).
    Update the weight matrices $\boldsymbol{W}_X$, and $\boldsymbol{W}_Y$ using (4.24) and (4.25), respectively.
    $m = m + 1$
while (JCDTL-MISR loss (4.10) $< \tau$)
Output: $\{\boldsymbol{T}_{Xi}\}_{i=1}^{N}$, $\{\boldsymbol{T}_{Yi}\}_{i=1}^{N}$, $\boldsymbol{T}_Z$, $\boldsymbol{W}_X$, $\boldsymbol{W}_Y$ (modality-specific transforms and weight matrices)

later estimate $\boldsymbol{Z}_{test}$. The test coefficients of the first two layers for modality $\boldsymbol{X}$ are computed as: $\boldsymbol{H}_{X1}^{test} = \boldsymbol{T}_{X1}\boldsymbol{X}_{test}$ and $\boldsymbol{H}_{X2}^{test} = \boldsymbol{T}_{X2}\,\boldsymbol{H}_{X1}^{test}$. Similarly, the test coefficients of the first two layers for modality $\boldsymbol{Y}$ are given as: $\boldsymbol{H}_{Y1}^{test} = \boldsymbol{T}_{Y1}\,\boldsymbol{Y}_{test}$ and $\boldsymbol{H}_{Y2}^{test} = \boldsymbol{T}_{Y2}\,\boldsymbol{H}_{Y1}^{test}$. For the update of $\boldsymbol{H}_{X3}^{test}$ and $\boldsymbol{H}_{Y3}^{test}$ the following sub-problems need to be solved:

$$\min_{\boldsymbol{H}_{X3}^{test}} \|\boldsymbol{T}_{X3}\boldsymbol{H}_{X2}^{test} - \boldsymbol{H}_{X3}^{test}\|_F^2 + \mu\|\boldsymbol{H}_{X3}^{test}\|_1 \tag{4.26}$$

$$\min_{\boldsymbol{H}_{Y3}^{test}} \|\boldsymbol{T}_{Y3}\boldsymbol{H}_{Y2}^{test} - \boldsymbol{H}_{Y3}^{test}\|_F^2 + \mu\|\boldsymbol{H}_{Y3}^{test}\|_1 \tag{4.27}$$

These are standard expression for LASSO based optimization problems [78] for which the closed-form update is given as:

$$\boldsymbol{H}_{X3}^{test} = \text{sign}(\boldsymbol{T}_{X3}\boldsymbol{H}_{X2}^{test}).\max(0, |\boldsymbol{T}_{X3}\boldsymbol{H}_{X2}^{test}| - \frac{\mu}{2}) \tag{4.28}$$

$$\boldsymbol{H}_{Y3}^{test} = \text{sign}(\boldsymbol{T}_{Y3}\boldsymbol{H}_{Y2}^{test}).\max(0, |\boldsymbol{T}_{Y3}\boldsymbol{H}_{Y2}^{test}| - \frac{\mu}{2}) \tag{4.29}$$

Subsequently, $\boldsymbol{H}_Z^{test}$ and reconstructed image $\boldsymbol{Z}_{test}$ is computed as: $\boldsymbol{H}_Z^{test}$= $\boldsymbol{W}_X \boldsymbol{H}_{X3}^{test} + \boldsymbol{W}_Y \boldsymbol{H}_{Y3}^{test}$ and $\boldsymbol{Z}_{test}$= $\boldsymbol{T}_Z^{\dagger}\ \boldsymbol{H}_Z^{test}$.

4.4 Results

In this work, we have considered two different datasets, RGB-MS [79] and RGB-NIR [80] for evaluating the performance of the proposed method. A brief description of the datasets and the benchmark methods used for comparison is presented below, along with the results and discussion.

4.4.1 Data Description

In both datasets, the RGB image is considered the guidance modality, while the MS and NIR of the respective datasets form the target modality. Both the datasets include the HR images for both the guidance ($\boldsymbol{Y}$) and target modalities ($\boldsymbol{Z}$), and they are of same resolution. In our experiments, similar to [68], the LR image of target modality is generated by downsizing the HR image by a factor and then applying bicubic interpolation on the downsized image to upscale by the same factor. We have considered a downsampled factor of 4 for both RGB-MS and RGB-NIR to ensure a fair comparison with benchmark techniques.

4.4.2 Benchmark Methods

Five different competing techniques are considered for comparison. Out of these, three are based on joint image filtering methods (Joint Bilateral Filtering (JBF) [71]), Guided image Filtering (GF) [70]) and Joint image Restoration (JR) [72])). The other two methods are based on learning methods employing deep learning (Deep Joint image Filtering (DJF) [65]) and dictionary learning (Coupled DL

[68]). All the above methods for comparison have been implemented using the software code made available in their repositories for both the multi-modal datasets.

4.4.3 Experimental Details

The reconstruction performance of the HR image of target modality is evaluated using PSNR (dB) and SSIM metrics. The RGB image used as guidance modality is converted to grayscale for both datasets. Additionally, the MS image at 640 nm is considered for experimentation for the RGB-MS dataset. The learning-based benchmark methods are trained on randomly selected 31 image pairs for RGB-NIR and 35 image pairs for RGB-MS, respectively. Each 512×512 image of all modalities is normalized and divided into non-overlapping patches of size 16×16. Later it is converted into a vectorized patch size (d) of 256. The hyperparameters for the benchmark methods are set to the values specified in their respective papers, and through grid search, we confirmed that these values yield optimal performance. For the TL-based methods, the hyperparameters values μ, λ and η are obtained using grid search for both the shallow (JCTL-MISR) and deep TL (JCDTL-MISR) methods.

4.4.4 Results Discussion

The reconstruction results of 5 random test images obtained with RGB-NIR and RGB-MS datasets are summarized in Tables 4.1 and 4.2, respectively, where the column headings represent the name of the test images considered for evaluation. The entries in bold and underline denote the best and second best performance, respectively, for each of the test images. It can be observed that the proposed

JCDTL-MISR method with 3 layers performs well for most of the images compared to other benchmarks for both the datasets. Also, compared to the DL counterpart (Coupled DL), the proposed shallow JCTL-MISR achieves better reconstruction results for all images. Notice an improvement in performance with the proposed deep variant (JCDTL-MISR) compared to the proposed shallow variant (JCTL-MISR) on both datasets. This demonstrates that deep layers can learn rich representation that helps in effectively modeling the cross-modal dependencies between the different modalities, resulting in improved super-resolution results.

Table 4.1: MISR Results with RGB-NIR for 4×

RGB-NIR Dataset										
	Indoor 4		Indoor 5		Indoor 11		Indoor 16		Indoor21	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Proposed JCDTL-MISR (N = 3)	**30.629**	0.941	30.580	0.951	**30.384**	0.903	**33.070**	**0.950**	**29.668**	**0.915**
Proposed JCTL-MISR	30.341	0.915	30.414	0.937	28.879	0.896	32.018	0.925	29.554	0.895
Coupled DL [68]	30.021	0.905	29.865	0.915	27.808	0.893	31.438	0.916	29.442	0.893
DJF [65]	26.958	0.898	27.804	0.899	27.015	0.817	30.149	0.890	27.619	0.853
JR [72]	26.271	0.841	25.076	0.939	22.864	0.815	23.502	0.867	21.626	0.794
GF [70]	29.854	**0.946**	**32.052**	**0.971**	27.589	0.901	31.916	0.938	27.133	0.909
JBF [71]	26.354	0.919	31.283	0.968	26.480	**0.906**	30.431	0.929	25.746	0.902

Table 4.2: MISR Results with RGB-MS for 4×

RGB-MS Dataset										
	Imge6		Imge7		Imgf5		Imgf7		Imgh3	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Proposed JCDTL-MISR (N = 3)	**31.441**	**0.841**	**35.496**	0.899	**37.522**	**0.947**	**33.339**	0.888	**39.403**	**0.948**
Proposed JCTL-MISR	31.049	0.835	33.222	0.889	36.277	0.939	31.964	0.864	37.140	0.941
Coupled DL [68]	28.793	0.814	32.669	0.877	34.239	0.906	31.401	0.878	36.107	0.920
DJF [65]	20.968	0.828	26.732	**0.938**	32.588	0.824	23.851	**0.902**	30.788	0.924
JR [72]	26.519	0.814	32.781	0.889	33.933	0.890	29.295	0.804	33.999	0.922
GF [70]	25.332	0.774	29.709	0.869	31.706	0.901	28.045	0.880	33.518	0.807
JBF [71]	25.535	0.746	29.655	0.799	32.411	0.886	28.874	0.839	34.461	0.902

For visual comparison, Figs. 4.3 and 4.4 present the reconstructed image obtained with different techniques for a single test image from RGB-NIR and RGB-MS datasets, respectively. The bottom row shows the reconstructed HR image for different methods, and the top row shows the corresponding error

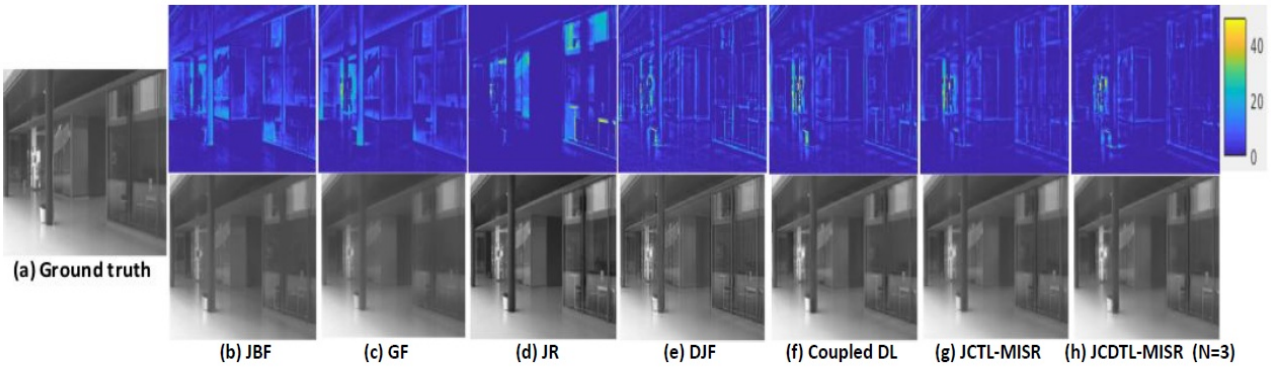


Figure 4.3: Visual comparison for 'Indoor 16' image of RGB-NIR dataset. The top row presents the reconstruction error map and the bottom row shows the reconstructed HR image with different methods.

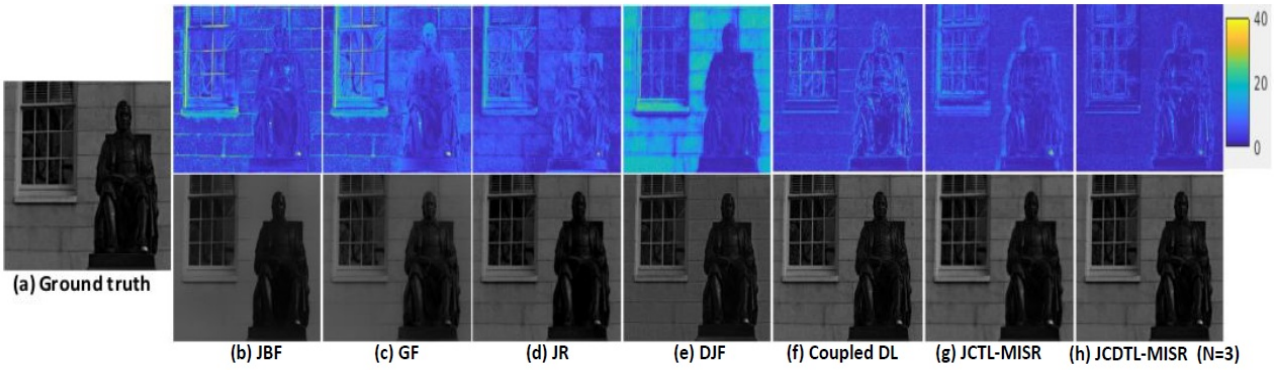


Figure 4.4: Visual comparison for 'Imge7' image of RGB-MS dataset. The top row presents the reconstruction error map and the bottom row shows the reconstructed HR image with different methods.

map. It can be observed that the reconstructed error is minimal for the proposed methods compared to other techniques, indicating better visual quality. In this work, the JCTL-MISR used the following values for the hyperparameters: λ = 10, μ = 0.001 and η = 100 for both the datasets. The JCDTL-MISR method employed the following values of the hyperparameters: λ = 2.8, η = 0.00021, and μ = 0.004 for both the datasets. Fig. 4.5 presents the convergence plot of the JCTL-MISR method obtained with the RGB-MS dataset. It can be seen that the algorithm converges within a few iterations. Similar convergence is obtained with the other dataset and the deep variant (JCDTL-MISR) method. The proposed JCTL-MISR and JCDTL-MISR methods were implemented in MATLAB on

an AMD Ryzen 5 4500U CPU@2.3GHz and 16 GB RAM. Training over 50 iterations using 30 image pairs took $\approx$ 5 minutes for JCTL-MISR and $\approx$ 8 minutes for JCDTL-MISR. The testing process for a single image pair required $\approx$ 20 milliseconds.

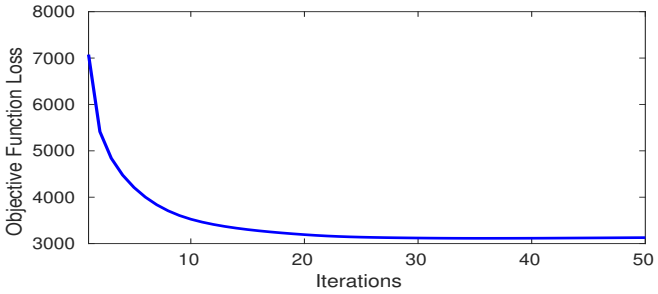
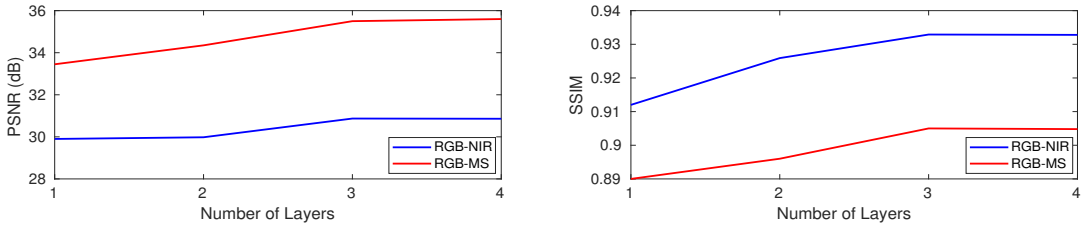


Figure 4.5: Convergence Plot of JCTL-MISR Method



(a) (b)

Figure 4.6: Effect of Number of Layers on JCDTL-MISR. (a) Average PSNR Vs. Number of Layers. (b) Average SSIM Vs. Number of Layers.

To illustrate the effect of number of layers in the JCDTL-MISR configuration, Fig. 4.6 presents the plots of PSNR and SSIM Vs. the number of layers of JCDTL-MISR for both the image datasets. A significant improvement is observed as we increase the layers from 1 to 3. However, going deep beyond 3 layers did not significantly improve the reconstruction results. This plot reflects the law of diminishing returns, and hence, in our work, we considered a 3 layer JCDTL-MISR configuration for the MISR task. One can notice an average improvement $>$ 2 dB in PSNR and $>$ 1% in SSIM with the 3 layer JCDTL-MISR over the shallow (single-layer) JCTL-MISR variant for RGB-MS dataset. Similarly, an average improvement of $\approx$ 1 dB in PSNR and $>$ 2% in SSIM is

observed with the 3-layer JCDTL-MISR over JCTL-MISR for RGB-NIR dataset.

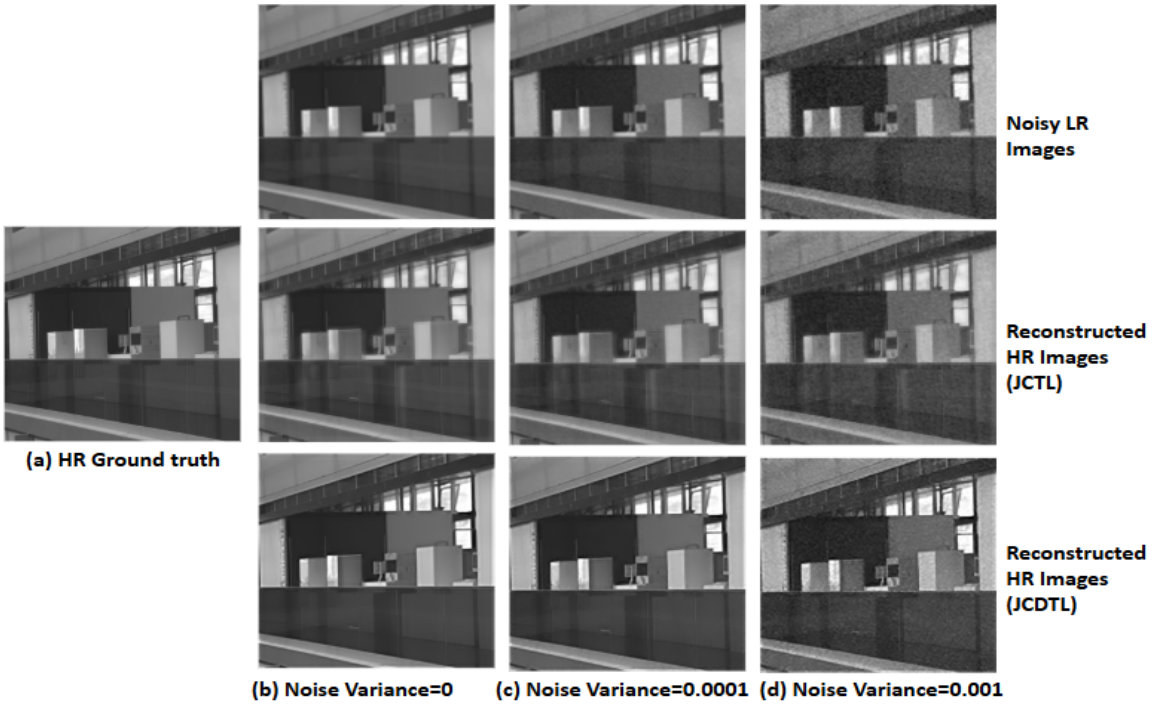


Figure 4.7: Results for 'Indoor 4' image of RGB-NIR dataset with the noisy LR image at the top and reconstructed HR image at the bottom.

Table 4.3: Reconstruction Performance with Noisy LR Images for 'Indoor 4'

	Noise Variance=0		Noise Variance=0.0001		Noise Variance=0.001	
Image	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
LR	29.87	0.89	27.84	0.86	26.91	0.64
JCTL-MISR	30.34	0.92	28.67	0.92	27.24	0.73
JCDTL-MISR	30.63	0.94	29.58	0.94	27.52	0.71

To study the effect of noise on the reconstruction capability of the proposed methods, similar to [68], Gaussian noise with 0 mean and certain variance (0, 0.001 and 0.0001) is added to the LR test image. The model trained with noise-free LR images is directly tested on the noisy test images. Fig. 4.7 shows the reconstruction results obtained using JCTL-MISR and JCDTL-MISR for one LR test image affected by noise. The corresponding PSNR and SSIM values for both the LR and reconstructed images are summarized in Table 4.3. These results demonstrate that the proposed methods effectively enhance the resolution

of the noisy LR image. Furthermore, for comparative analysis, Fig. 4.8 presents the average PSNR and SSIM plots for the RGB-NIR dataset, comparing the proposed approaches with a prior dictionary-based MISR method (Coupled DL). It is observed that both JCTL and JCDTL show improved PSNR compared to Coupled DL across all noise levels, although the SSIM values are comparatively lower than Coupled DL at high noise levels.

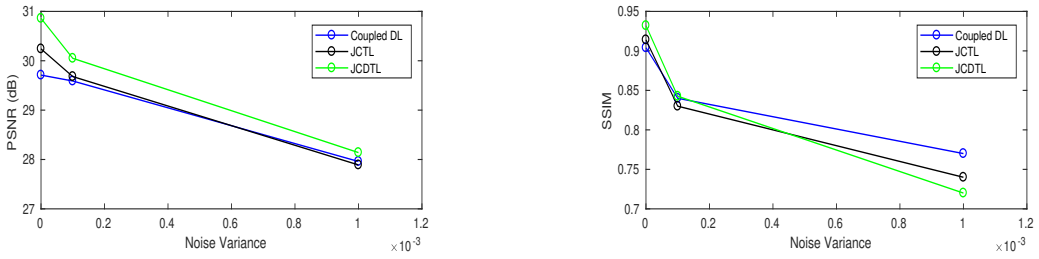


Figure 4.8: Performance Comparison with Noisy Test Images for JCTL-MISR and JCDTL-MISR

4.5 Summary

In this chapter, we focus on MISR, where an LR image of a target modality is improved with the guidance of an HR image from another modality. To model the cross-modal dependencies, two novel joint optimization formulations based on TL framework, referred to as JCTL-MISR and its deep variant JCDTL-MISR, have been proposed. All the solution steps and requisite closed-form updates are provided. Results obtained with two publicly available RGB-NIR and RGB-MS datasets demonstrate the improved reconstruction performance of the proposed techniques compared to state-of-the-art methods. Furthermore, an average improvement $>$ 2 dB in PSNR and $>$ 1% in SSIM is observed with the 3 layer JCDTL-MISR over the shallow (single-layer) JCTL-MISR variant for RGB-MS dataset. Similarly, an average improvement of $\approx$ 1 dB in PSNR and $>$

2% in SSIM is observed with the 3-layer JCDTL-MISR over JCTL-MISR for RGB-NIR dataset.

To further enhance the performance, in the subsequent chapter, a convolutional variant of dictionary learning is investigated that overcomes the limitations of patch-based methods and enables rich representations to be learned across different modalities. The motivation behind this approach is discussed in Chapter 5, with the proposed formulation presented in Section 5.3.

Chapter 5

Supervised Domain Adaptation Via Joint Coupled Convolutional Dictionary Learning for Multi-Modal Image Super-Resolution

5.1 Motivation

While DL based sparse representation learning has been popular for addressing inverse problems, including MISR [68, 73], it does come with certain constraints. One notable limitation is that the traditional patch-based dictionaries are inherently not translation invariant, i.e., the basis elements often appear as shifted versions of one another. Hence, they require more training data for generalization. Additionally, as these dictionaries operate on individual patches rather than the entire image, they reconstruct and sparsify patches independently, due to which the underlying structure of the signal may get lost. CDL, on the other hand, employs translation-invariant dictionaries [81], which can effectively mitigate these limitations. They have been recently utilized for MISR and are shown to provide improved image reconstruction over the non-convolutional DL variants [82]. The Multi-modal Convolutional Dictionary Learning (MCDL) work in [82] is a

two-stage approach, wherein the first stage, common and unique convolutional dictionaries, and their associated common and unique sparse coefficients are learned from both the LR images of target and HR images of guidance modality. Subsequently, in the second stage, the learned common coefficients and unique coefficients of the target modality are employed to learn two more convolutional dictionaries to reconstruct the HR images of the target modality. Although it offers superior performance compared to the non-convolutional DL variants, the CDL-based method requires learning many parameters (6 convolutional dictionaries and 3 coefficients). Moreover, it is computationally intensive, making it unsuitable for real-life applications with limited data.

Owing to the inherent advantages of CDL over DL for image processing, this work presents a novel CDL-based method for improved MISR, specially catering to data-limited scenarios. It exploits the potential of convolutional dictionaries for learning representation from the LR target and HR guidance modalities and modeling the cross-modal dependencies to reconstruct the HR image of the target modality. A joint optimization is presented that learns a convolutional dictionary for each modality, i.e., LR images of target modality and HR images of guidance modality with the respective sparse coefficients. These sparse coefficients are constrained to reconstruct the HR images of the target modality using two coupling convolutional dictionaries that model the cross-modal dependencies between the two modalities. Unlike the prior CDL-based two stage approach [82], the proposed method uses a joint optimization formulation. Moreover, it does not require learning a dedicated dictionary/transform for the HR image of

the target modality (like other works [68, 82], JCTL-MISR and JCDTL-MISR (discussed in Chapter 4, Section 4.3.2 and 4.3.3)). This results in a more efficient formulation requiring fewer learning parameters, making it suitable for limited-data scenarios. It is to be noted that the solution to the proposed formulation requires variable splitting and ADMM [76] that are systematically derived. To the best of our knowledge, the formulation and the new algorithmic steps worked out from the first principles for the problem at hand does not exist elsewhere.

The main contributions of the work are summarized below:

- A novel joint learning formulation employing CDL referred to as JCCDL-MISR, is presented to effectively exploit the correlation between the different modalities for MISR.
- The solution steps and the derived closed-form updates are provided. Additionally, we present comparisons against the benchmarks, demonstrating the ability of the proposed method to learn, even with limited training data.

To furnish the necessary details of the proposed work, the rest of the chapter is organized as follows. Section 5.2 presents a brief background on CDL that forms the basis of the proposed method. Section 5.3 presents a detailed explanation of the proposed JCCDL-MISR method. Experimental results with public datasets and comparisons with benchmark methods are presented in Section 5.4. Finally, Section 5.5 summarizes the work.

5.2 Background on Convolutional Dictionary Learning (CDL)

In CDL, the signal $\{\boldsymbol{x}_i\}_{i=1}^{N}$ with N measurements each of n dimension, is reconstructed using a set of M dictionary filters $\{\boldsymbol{d}_m\}_{m=1}^{M}$, and the set of coefficients $\{\boldsymbol{a}_{m,i}\}_{m=1}^{M}$ by solving the following optimization [81, 83]:

$$\min_{\{\boldsymbol{d}_m\},\{\boldsymbol{a}_{m,i}\}} \frac{1}{2}\sum_{i=1}^{N}\left\|\boldsymbol{x}_i - \sum_{m=1}^{M}\boldsymbol{d}_m * \boldsymbol{a}_{m,i}\right\|_2^2 + \lambda \sum_{i,m} \|\boldsymbol{a}_{m,i}\|_1 \text{ s.t. } \quad \|\boldsymbol{d}_m\|_2^2 = 1 \ \ \forall m \quad (5.1)$$

where '$*$' denotes the convolution operation and $\lambda > 0$ controls the sparsity of the coefficients $\boldsymbol{a}_{m,i}$ using l_1-norm constraint. The l_2-norm on $\boldsymbol{d}_m$ is employed to compensate for the scaling ambiguity between the dictionary atoms and the coefficients. Defining $\boldsymbol{D}_m$ as a linear operator such that $\boldsymbol{D}_m\boldsymbol{a}_{m,i} = \boldsymbol{d}_m * \boldsymbol{a}_{m,i}$ and taking $\boldsymbol{D} = \begin{pmatrix}\boldsymbol{D}_1 & \dots & \boldsymbol{D}_M\end{pmatrix}$, $\boldsymbol{X} = \begin{pmatrix}\boldsymbol{x}_1 & \dots & \boldsymbol{x}_N\end{pmatrix}$ and $\boldsymbol{A} = \begin{pmatrix}\boldsymbol{a}_{1,1} & .. & \boldsymbol{a}_{1,N} \\ : & : & : \\ \boldsymbol{a}_{M,1} & .. & \boldsymbol{a}_{M,N}\end{pmatrix}$, (5.1) can be re-written as:

$$\min_{\boldsymbol{D},\boldsymbol{A}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{A}\|_F^2 + \lambda \|\boldsymbol{A}\|_1 \quad (5.2)$$

The problem in (5.2) is not jointly convex in both the dictionary filters and coefficients; hence, AM [84] is employed to estimate them. Here, the convolutional sparse coefficients can be solved by ADMM in the Discrete Fourier Transform (DFT) domain as shown in [81], and the convolutional dictionaries can be solved using the Convolutional Constrained Method of Optimal Direction (CCMOD) [85]. These concepts have been utilized to work out the proposed CDL-based method for MISR.

5.3 Proposed MISR method using Joint Coupled Convolutional Dictionary Learning (JCCDL-MISR)

The problem definition of MISR remains the same as that of the previous work on JCTL-MISR (Chapter 4, Section 4.3.1). We follow the same naming convention for the different imaging modalities for uniformity. The proposed formulation employing CDL is presented below.

Given the training images of LR target $\boldsymbol{x}_i$, HR guidance $\boldsymbol{y}_i$ and HR target $\boldsymbol{z}_i$ modalities ($i = 1, 2, .., N$), convolutional dictionaries $\{\boldsymbol{s}_m\}_{m=1}^{M}$ and $\{\boldsymbol{g}_m\}_{m=1}^{M}$ of M filters and the associated sparse coefficients $\{\boldsymbol{a}_{m,i}\}_{m=1}^{M}$ and $\{\boldsymbol{b}_{m,i}\}_{m=1}^{M}$, respectively are learned for the different image modalities $\boldsymbol{x}$ and $\boldsymbol{y}$. As mentioned earlier, the different modalities capture the same scene of interest; hence, their coefficients are considered to be related to one another. In JCCDL-MISR, the relationship between the coefficients is modeled using two coupled convolutional dictionaries $\{\boldsymbol{w}_m\}_{m=1}^{M}$ and $\{\boldsymbol{v}_m\}_{m=1}^{M}$ expressed as $\boldsymbol{z}_i = \sum_{m=1}^{M} \boldsymbol{w}_m * \boldsymbol{a}_{m,i} + \sum_{m=1}^{M} \boldsymbol{v}_m * \boldsymbol{b}_{m,i}$. This enables the coupled dictionaries to learn and effectively combine the high-frequency information from HR guidance image and low-frequency information from LR target image, to synthesize the HR target image. Similar to the previous supervised learning methods for MISR, discussed in Section 4.3.2, this method has a training and test phase that is described below.

5.3.1 Training Phase

During training, having the knowledge of $\boldsymbol{z}_i$, a joint optimization is carried out to learn the dictionaries and coefficients together using the following formulation:

$$\min_{\substack{\{\boldsymbol{s}_m\},\{\boldsymbol{g}_m\},\{\boldsymbol{a}_{m,i}\},\\ \{\boldsymbol{b}_{m,i}\},\{\boldsymbol{w}_m\},\{\boldsymbol{v}_m\}}} \frac{1}{2}\sum_i \left\| \boldsymbol{x}_i - \sum_m \boldsymbol{s}_m * \boldsymbol{a}_{m,i} \right\|_2^2 + \frac{1}{2}\sum_i \left\| \boldsymbol{y}_i - \sum_m \boldsymbol{g}_m * \boldsymbol{b}_{m,i} \right\|_2^2$$
$$+\lambda_a \sum_{i,m} \|\boldsymbol{a}_{m,i}\|_1 + \lambda_b \sum_{i,m} \|\boldsymbol{b}_{m,i}\|_1 + \frac{\mu}{2}\sum_i \left\| \boldsymbol{z}_i - \sum_m \boldsymbol{w}_m * \boldsymbol{a}_{m,i} - \sum_m \boldsymbol{v}_m * \boldsymbol{b}_{m,i} \right\|_2^2 \quad (5.3)$$
$$\text{s.t. } \|\boldsymbol{s}_m\|_2^2 = 1, \|\boldsymbol{g}_m\|_2^2 = 1, \|\boldsymbol{w}_m\|_2^2 = 1, \|\boldsymbol{v}_m\|_2^2 = 1$$

Here, the first two terms ensure that each of the dictionary filters and coefficients are learned such that they reconstruct the images of the respective modality well. The fifth term defines the coupling between the coefficients of the different image modalities to reconstruct the HR image of target modality, controlled by $\mu > 0$. The remaining terms constrain the learned coefficients to be sparse using λ_a and λ_b. Defining $\boldsymbol{S}_m$, $\boldsymbol{G}_m$, $\boldsymbol{W}_m$ and $\boldsymbol{V}_m$ as linear operators such that $\boldsymbol{S}_m \boldsymbol{a}_{m,i} = \boldsymbol{s}_m * \boldsymbol{a}_{m,i}$, $\boldsymbol{G}_m \boldsymbol{b}_{m,i} = \boldsymbol{g}_m * \boldsymbol{b}_{m,i}$, $\boldsymbol{W}_m \boldsymbol{a}_{m,i} = \boldsymbol{w}_m * \boldsymbol{a}_{m,i}$ and $\boldsymbol{V}_m \boldsymbol{b}_{m,i} = \boldsymbol{v}_m * \boldsymbol{b}_{m,i}$. Using the block-structured matrix notations for dictionary, coefficients, and the input as defined in (5.2), one can re-write (5.3) as:

$$\min_{\substack{\boldsymbol{S},\boldsymbol{G},\boldsymbol{A}\\ \boldsymbol{B},\boldsymbol{W},\boldsymbol{V}}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{S}\boldsymbol{A}\|_F^2 + \frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{G}\boldsymbol{B}\|_F^2 + \frac{\mu}{2}\|\boldsymbol{Z} - \boldsymbol{W}\boldsymbol{A} - \boldsymbol{V}\boldsymbol{B}\|_F^2 + \lambda_a\|\boldsymbol{A}\|_1 + \lambda_b\|\boldsymbol{B}\|_1 \quad (5.4)$$

Fig. 5.1 presents the block diagram of the proposed JCCDL-MISR method in matrix form. Here, AM is employed to solve for the dictionary filters and the coefficients. In the first step, coefficients or sparse codes are updated, keeping the dictionary filters fixed. In the second step, dictionary filters are updated, keeping the coefficients fixed. These steps are described below in detail.

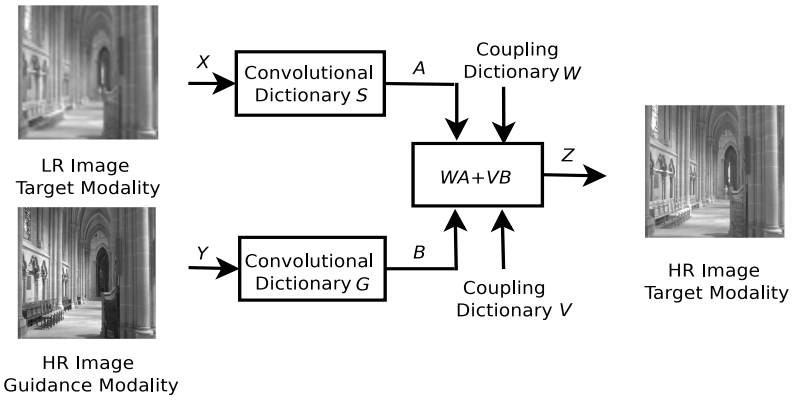


Figure 5.1: Block Diagram of the proposed JCCDL-MISR Method

5.3.1.1 Sparse Coding Update

The coefficients $\boldsymbol{A}$ and $\boldsymbol{B}$ are updated in an alternating way by solving for one while keeping the other fixed. The update for $\boldsymbol{A}$ is obtained by solving:

$$\min_{\boldsymbol{A}} \frac{1}{2} \left\| \boldsymbol{X} - \boldsymbol{S}\boldsymbol{A} \right\|_F^2 + \frac{\mu}{2} \left\| \boldsymbol{X'} - \boldsymbol{W}\boldsymbol{A} \right\|_F^2 + \lambda_a \left\| \boldsymbol{A} \right\|_1 \tag{5.5}$$

where $\boldsymbol{X'} = \boldsymbol{Z} - \boldsymbol{V}\boldsymbol{B}$. Applying ADMM to (5.5) using variable splitting by introducing an auxiliary variable $\boldsymbol{P}_a$ constrained to be equal to the primary variable $\boldsymbol{A}$ results in:

$$\min_{\boldsymbol{A},\boldsymbol{P}_a} \frac{1}{2} \left\| \boldsymbol{X} - \boldsymbol{S}\boldsymbol{A} \right\|_F^2 + \frac{\mu}{2} \left\| \boldsymbol{X'} - \boldsymbol{W}\boldsymbol{A} \right\|_F^2 + \lambda_a \left\| \boldsymbol{P}_a \right\|_1 \quad \text{s.t.} \quad \boldsymbol{A} = \boldsymbol{P}_a \tag{5.6}$$

With $\boldsymbol{U}_a$ as the dual variable, the ADMM iterations are expressed as:

$$\boldsymbol{A}^{t+1} = \min_{\boldsymbol{A}} \frac{1}{2} \left\| \boldsymbol{X} - \boldsymbol{S}\boldsymbol{A} \right\|_F^2 + \frac{\mu}{2} \left\| \boldsymbol{X'} - \boldsymbol{W}\boldsymbol{A} \right\|_F^2 + \frac{\rho}{2} \left\| \boldsymbol{A} - \boldsymbol{P}_a^t + \boldsymbol{U}_a^t \right\|_F^2, \tag{5.7}$$

$$\boldsymbol{P}_a^{t+1} = \min_{\boldsymbol{P}_a} \frac{\rho}{2} \left\| \boldsymbol{A}^{t+1} - \boldsymbol{P}_a + \boldsymbol{U}_a^t \right\|_F^2 + \lambda_a \left\| \boldsymbol{P}_a \right\|_1, \tag{5.8}$$

$$\boldsymbol{U}_a^{t+1} = \boldsymbol{U}_a^t + \boldsymbol{A}^{t+1} - \boldsymbol{P}_a^{t+1}, \tag{5.9}$$

where ρ controls the convergence rate of an algorithm. Taking $\boldsymbol{Q}_a = \boldsymbol{P}_a^t - \boldsymbol{U}_a^t$, the solution of $\boldsymbol{A}$ can be obtained by taking derivative of (5.7) with respect to $\boldsymbol{A}$ and equating it to 0, which results in the following:

$$(\boldsymbol{S}^H\boldsymbol{S} + \mu\boldsymbol{W}^H\boldsymbol{W} + \rho\boldsymbol{I})\boldsymbol{A} = \boldsymbol{S}^H\boldsymbol{X} + \mu\boldsymbol{W}^H\boldsymbol{X'} + \rho\boldsymbol{Q}_a \tag{5.10}$$

The above equation can be efficiently solved in the DFT domain. Similar to [82], applying DFT to the above equation results in the following:

$$(\hat{\boldsymbol{S}}^H\hat{\boldsymbol{S}} + \mu\hat{\boldsymbol{W}}^H\hat{\boldsymbol{W}} + \rho\boldsymbol{I})\hat{\boldsymbol{A}} = \hat{\boldsymbol{S}}^H\hat{\boldsymbol{X}} + \mu\hat{\boldsymbol{W}}^H\hat{\boldsymbol{X}}' + \rho\hat{\boldsymbol{Q}}_a \tag{5.11}$$

where $\hat{}$ denotes the DFT of the respective parameters. The solution of (5.11) can be obtained using iterated Sherman-Morrison algorithm [86]. The solution of (5.8) can be easily obtained by soft-thresholding similar to the work in [82] and (5.9) is updated by simple arithmetic operations. Next, the update for $\boldsymbol{B}$ is obtained by solving:

$$\min_{\boldsymbol{B}} \frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{G}\boldsymbol{B}\|_F^2 + \frac{\mu}{2}\|\boldsymbol{Y}' - \boldsymbol{V}\boldsymbol{B}\|_F^2 + \lambda_b\|\boldsymbol{B}\|_1 \tag{5.12}$$

where $\boldsymbol{Y}' = \boldsymbol{Z} - \boldsymbol{W}\boldsymbol{A}$. Following similar updates of ADMM as in (5.5), the solution to $\boldsymbol{B}$ in DFT domain is obtained as:

$$(\hat{\boldsymbol{G}}^H\hat{\boldsymbol{G}} + \mu\hat{\boldsymbol{V}}^H\hat{\boldsymbol{V}} + \rho\boldsymbol{I})\hat{\boldsymbol{B}} = \hat{\boldsymbol{G}}^H\hat{\boldsymbol{Y}} + \mu\hat{\boldsymbol{V}}^H\hat{\boldsymbol{Y}}' + \rho\hat{\boldsymbol{Q}}_b \tag{5.13}$$

where $\boldsymbol{Q}_b = \boldsymbol{P}_b^t - \boldsymbol{U}_b^t$. It is to be noted that the auxiliary variables $\boldsymbol{P}_b$ and dual variables $\boldsymbol{U}_b$ associated with the primary variable $\boldsymbol{B}$ follow the standard updates similar to (5.8) and (5.9). The ADMM updates of both the coefficients are alternatively solved for a pre-defined number of iterations.

5.3.1.2 Dictionary Update

Now keeping the coefficients fixed, the dictionary filters $\{\boldsymbol{s}_m\}_{m=1}^M$, $\{\boldsymbol{g}_m\}_{m=1}^M$, $\{\boldsymbol{w}_m\}_{m=1}^M$, and $\{\boldsymbol{v}_m\}_{m=1}^M$ are updated. The dictionary filters $\{\boldsymbol{s}_m\}_{m=1}^M$, $\{\boldsymbol{g}_m\}_{m=1}^M$ of the respective modalities are updated by solving the standard convolutional dictionary learning problem using the CCMOD [85]. The updates for $\{\boldsymbol{w}_m\}_{m=1}^M$ and $\{\boldsymbol{v}_m\}_{m=1}^M$ are obtained by converting them into standard convolutional dictionary learning problem as:

$$\min_{\{\boldsymbol{w}_m\}} \frac{1}{2}\sum_i \left\| \boldsymbol{x'}_i - \sum_m \boldsymbol{w}_m * \boldsymbol{a}_{m,i} \right\|_2^2 \text{ s.t. } \|\boldsymbol{w}_m\|_2^2 = 1 \forall m \tag{5.14}$$

$$\min_{\{\boldsymbol{v}_m\}} \frac{1}{2}\sum_i \left\| \boldsymbol{y'}_i - \sum_m \boldsymbol{v}_m * \boldsymbol{b}_{m,i} \right\|_2^2 \text{ s.t. } \|\boldsymbol{v}_m\|_2^2 = 1 \forall m \tag{5.15}$$

where $\boldsymbol{x'}_i = \boldsymbol{z}_i - \sum_m \boldsymbol{v}_m * \boldsymbol{b}_{m,i}$ and $\boldsymbol{y'}_i = \boldsymbol{z}_i - \sum_m \boldsymbol{w}_m * \boldsymbol{a}_{m,i}$. Both (5.14), (5.15) are solved using standard CDL updates. The dictionary filters are updated alternatively for a pre-defined number of iterations.

Note that the dictionaries and coefficients are updated in turns until the objective function in (5.3) converges, i.e., the loss of (5.3) over the subsequent iterations does not change significantly, with the absolute value being less than an empirically calculated threshold. At the end of the training phase, dictionary filters of the respective modalities and coupling terms are learned. The pseudo-code of the training phase of the proposed JCCDL-MISR method is summarized in Algorithm 6.

5.3.2 Test Phase

During testing, given the LR target $\boldsymbol{x}^{test}$ and HR guidance $\boldsymbol{y}^{test}$ images, the HR target $\boldsymbol{z}^{test}$ image is reconstructed by first computing the coefficients $\{\boldsymbol{a}_m^{test}\}$ and $\{\boldsymbol{b}_m^{test}\}$ using their respective learnt dictionaries $\{\boldsymbol{s}_m\}$ and $\{\boldsymbol{g}_m\}$ following the standard convolutional sparse coding update given in [81], by solving:

$$\min_{\{\boldsymbol{a}_m^{test}\}} \frac{1}{2} \left\| \boldsymbol{x}^{test} - \sum_{m=1}^{M} \boldsymbol{s}_m * \boldsymbol{a}_m^{test} \right\|_2^2 + \lambda_a \sum_{m=1}^{M} \left\| \boldsymbol{a}_m^{test} \right\|_1 \tag{5.16}$$

$$\min_{\{\boldsymbol{b}_m^{test}\}} \frac{1}{2} \left\| \boldsymbol{y}^{test} - \sum_{m=1}^{M} \boldsymbol{g}_m * \boldsymbol{b}_m^{test} \right\|_2^2 + \lambda_b \sum_{m=1}^{M} \left\| \boldsymbol{b}_m^{test} \right\|_1 \tag{5.17}$$

Algorithm 6: JCCDL for MISR (JCCDL-MISR)

Input: $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$
Parameters: $\lambda_a, \lambda_b, \mu, \tau, k$ (size of dictionary atoms or filters), M (Number of filters)
Initialization: Set modality-specific convolutional dictionaries $\{\boldsymbol{s}_m\}_{m=1}^{M}$, $\{\boldsymbol{g}_m\}_{m=1}^{M}$, and coupled convolutional dictionaries $\{\boldsymbol{w}_m\}_{m=1}^{M}$, $\{\boldsymbol{v}_m\}_{m=1}^{M}$ to random matrix with real numbers between 0 and 1 drawn from a uniform distribution, iteration $p = 0$
do
    Sparse Coding Update:
        Update the source modality sparse codes $\boldsymbol{A}$ in the DFT domain using ADMM iterations with the closed-form given in (5.11).
        Update the guidance modality sparse codes $\boldsymbol{B}$ in the DFT domain using ADMM iterations with the closed-form given in (5.13).
    Dictionary Update:
        Update the convolutional dictionaries for source modality $\{\boldsymbol{s}_m\}_{m=1}^{M}$ and guidance modality $\{\boldsymbol{g}_m\}_{m=1}^{M}$ using standard CDL updates [85].
        Update the coupled convolutional dictionaries $\{\boldsymbol{w}_m\}_{m=1}^{M}$ using (5.14).
        Update the coupled convolutional dictionaries $\{\boldsymbol{v}_m\}_{m=1}^{M}$ using (5.15).
    $p = p + 1$
while (JCCDL-MISR loss (5.3) $< \tau$)
Output: $\{\boldsymbol{s}_m\}_{m=1}^{M}$, $\{\boldsymbol{g}_m\}_{m=1}^{M}$, $\{\boldsymbol{w}_m\}_{m=1}^{M}$, and $\{\boldsymbol{v}_m\}_{m=1}^{M}$ (modality-specific dictionaries and coupled dictionaries)

Subsequently, using the learnt coupling dictionaries $\{\boldsymbol{w}_m\}$ and $\{\boldsymbol{v}_m\}$, the HR target $\boldsymbol{z}^{test}$ image is reconstructed as: $\boldsymbol{z}^{test} = \sum_{m=1}^{M} \boldsymbol{w}_m * \boldsymbol{a}_m^{test} + \sum_{m=1}^{M} \boldsymbol{v}_m * \boldsymbol{b}_m^{test}$.

5.4 Results Discussion

The proposed JCCDL-MISR method is evaluated for super-resolution using the same two datasets, RGB-MS [79] and RGB-NIR [80] discussed in Chapter 4, Section 4.4.1. In addition to the benchmark methods mentioned in Chapter 4, Section 4.4.2, comparisons with the shallow and deep variants of Joint Coupled Transform Learning (JCTL-MISR, JCDTL-MISR) (Chapter 4, Section 4.3.2 and 4.3.3), the recent MCDL [82] and the recent deep learning based Structure Guided Network (SGNet) [87]) are also considered for performance evaluation of the proposed method. Kindly note that 3 layer JCDTL-MISR configuration is considered for comparison as it gave the best results for the datasets considered

in this work. The data pre-processing remains the same as presented in Chapter 4, Section 4.4.3, and the HR image reconstruction quality is evaluated using the same PSNR (dB) and SSIM metrics.

As mentioned in Section 4.4.3, the learning-based methods learn the respective model parameters by training over non-overlapping 16×16 patches of the images. During testing, patches of the test image are individually reconstructed and then combined to form the full image. Unlike previous methods, techniques using CDL (JCCDL-MISR and MCDL) use non-overlapping patches of size 256×256 for training, and testing is conducted on the entire image and *not* patches. The proposed JCCDL-MISR method is trained with *only* 10 image pairs for each dataset, a substantial reduction from the 31 and 35 image pairs used for benchmark methods. Hyperparameters for all techniques are tuned using grid search. The JCCDL-MISR method is implemented using the SPORCO library in Matlab [88].

Table 5.1: MISR Results with RGB-NIR for 4×

RGB-NIR Dataset										
	Indoor 4		Indoor 5		Indoor 11		Indoor 16		Indoor21	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Proposed JCCDL-MISR	**31.644**	0.942	**32.058**	0.958	**31.711**	0.925	31.801	0.944	**31.403**	**0.924**
MCDL [82]	28.932	0.902	28.307	0.908	28.874	0.842	30.601	0.914	28.787	0.865
JCDTL-MISR (Chapter 4)	30.629	0.941	30.580	0.951	30.384	0.903	**33.070**	0.950	29.668	0.915
JCTL-MISR (Chapter 4)	30.341	0.915	30.414	0.937	28.879	0.896	32.018	0.925	29.554	0.895
Coupled DL [68]	30.021	0.905	29.865	0.915	27.808	0.893	31.438	0.916	29.442	0.893
SGNet [87]	30.551	0.928	29.477	0.938	30.178	**0.941**	28.784	**0.953**	29.551	0.921
DJF [65]	26.958	0.898	27.804	0.899	27.015	0.817	30.149	0.890	27.619	0.853
JR [72]	26.271	0.841	25.076	0.939	22.864	0.815	23.502	0.867	21.626	0.794
GF [70]	29.854	**0.946**	32.052	**0.971**	27.589	0.901	31.916	0.938	27.133	0.909
JBF [71]	26.354	0.919	31.283	0.968	26.480	0.906	30.431	0.929	25.746	0.902

Tables 5.1 and 5.2 present the MISR results for 5 test image pairs of RGB-NIR and RGB-MS, respectively. The entries in bold and underline denote the

Table 5.2: MISR Results with RGB-MS for 4×

RGB-MS Dataset										
	Imge6		Imge7		Imgf5		Imgf7		Imgh3	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Proposed JCCDL-MISR	**31.779**	**0.856**	**36.732**	0.928	**38.984**	**0.954**	**34.830**	**0.907**	**40.361**	**0.961**
MCDL [82]	25.477	0.605	28.955	0.709	33.426	0.880	27.894	0.722	35.564	0.896
JCDTL-MISR (Chapter 4)	31.441	0.841	35.496	0.899	37.522	0.947	33.339	0.888	39.403	0.948
JCTL-MISR (Chapter 4)	31.049	0.835	33.222	0.889	36.277	0.939	31.964	0.864	37.140	0.941
Coupled DL [68]	28.793	0.814	32.669	0.877	34.239	0.906	31.401	0.878	36.107	0.920
SGNet [87]	27.678	0.749	30.667	0.791	34.243	0.907	28.906	0.794	37.148	0.914
DJF [65]	20.968	0.828	26.732	**0.938**	32.588	0.824	23.851	0.902	30.788	0.924
JR [72]	26.519	0.814	32.781	0.889	33.933	0.890	29.295	0.804	33.999	0.922
GF [70]	25.332	0.774	29.709	0.869	31.706	0.901	28.045	0.880	33.518	0.807
JBF [71]	25.535	0.746	29.655	0.799	32.411	0.886	28.874	0.839	34.461	0.902

best and second best performance, respectively for each of the test images. Kindly note that JCCDL considers the configuration that gave optimal results obtained through the analysis of optimal parameter settings presented below. The optimal hyperparameter values for JCCDL are: M = 4, atom size (k) = 8×8, $\lambda_a = \lambda_b = 0.1$ and $\mu = 0.01$, for both the datasets. Notably, joint learning-based approaches (Coupled DL, JCTL-MISR, JCDTL-MISR, and JCCDL-MISR) exhibit superior performance compared to filtering (JR, GF, and JBF), deep learning (DJF, SGNet), and two-stage (MCDL) approaches for most images. Joint learning effectively captures discriminative and common features from each modality, contributing to improved HR image reconstruction for the target modality. Among the joint learning methods, the proposed JCCDL-MISR outperforms other benchmark methods for most images, despite being trained with limited data. This demonstrates the potential of using shift-invariant dictionaries, i.e., convolutional dictionaries, for robust and effective modeling for MISR. One can observe that the deep model (DJF, SGNet) renders poor performance with limited training data as it requires more data for optimal reconstruction. The proposed JCCDL-MISR method shows an improvement of > 1 dB in PSNR

and $> 1\%$ in SSIM on most images compared to the best-performing benchmark techniques, even with limited training data.

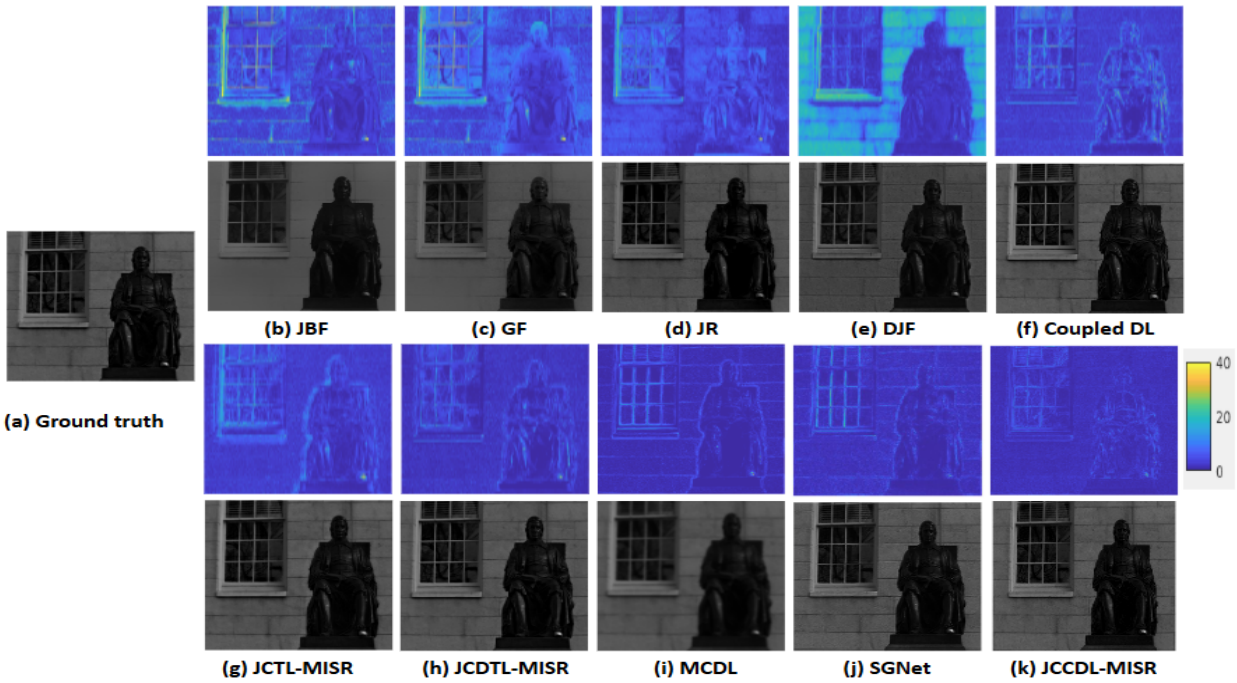


Figure 5.2: MISR results for 'Imge7' image of RGB-MS data with the error map at the top and the reconstructed images at the bottom.

It is important to note that compared to MCDL approach that requires learning 6 convolutional dictionaries (common and unique dictionaries for each LR target, HR guidance and HR target modalities) and 3 sparse coefficients (1 common and 2 unique coefficients associated with LR target and HR guidance modality), JCCDL-MISR requires learning only 4 convolutional dictionaries (one each for LR and HR modality, and 2 for coupling) and 2 sparse coefficients (associated with LR and HR modality, respectively). Thus, even with reduced complexity, JCCDL-MISR offers improved performance over MCDL. For the sake of illustration, the error maps and reconstructed images for one test image ('Imge7') from RGB-MS data obtained with different methods are given in Fig. 5.2. It shows that the visual results are in conformance with the PSNR and

SSIM values presented in Table 5.2, with JCCDL-MISR exhibiting reduced reconstruction error.

5.4.1 Optimal Parameter Settings

To gain more insights, different experiments were conducted to study the effect of the number of filters and kernel size on the performance of the proposed JCCDL-MISR method and observe its convergence.

5.4.1.1 Algorithm Convergence

The convergence of the proposed JCCDL-MISR is examined for various configurations, considering different numbers of filters (M = 2, 4, 8, 12) with the atom size of k = 8×8. In Fig. 5.3, the convergence plot of different JCCDL-MISR configurations with RGB-MS data is presented. The algorithm consistently converges within a few iterations, with M = 4 configuration converging more rapidly than others. The M = 2 configuration exhibits the slowest convergence, while M = 8, 12 configurations are little better and demonstrate similar behavior. The proposed JCCDL-MISR took ≈ 10 minutes for training through 50 iterations for updating dictionaries and coefficients over 10 image pairs on an AMD Ryzen 5 4500U CPU@2.3GHz with 16 GB RAM (without GPU). The test process took ≈ 20 milliseconds for updating the coefficients through 10 iterations for single image pair. It is worth noting that the computation time is considerably lower compared to existing deep learning methods like DJF, SGNet, which require ≈ 4× time for training for the same data size, despite using a GPU accelerator, with comparable test time.

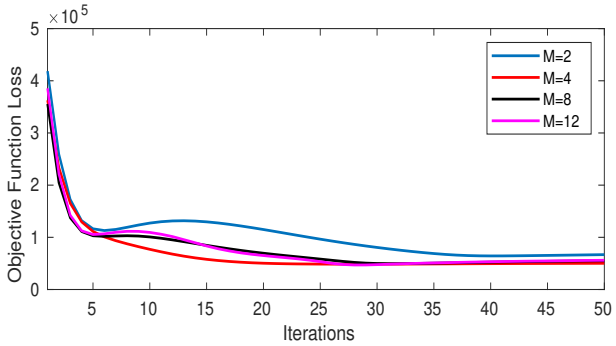


Figure 5.3: Convergence of JCCDL-MISR for different number of filters (M)

5.4.1.2 Effect of Number of Filters, M

Table 5.3 A summarizes the MISR results for two test images from RGB-MS data using atom (filter) size of 8×8. JCCDL-MISR with $M = 4$ performs the best among others (values highlighted in bold in Table 5.3 A). Fewer filters cannot learn effective representation and tend to underfit, while more filters fail to generalize and tend to overfit with limited training data, as considered in this work. For illustration, Fig. 5.4 presents the dictionaries learned using various configurations of the proposed JCCDL-MISR method employing different numbers of filters $M = 4$, 8 and 12 with $k = 8\times8$ on RGB-MS data. LR images are known to be dominated by low-frequency components, resulting in a smoothed dictionary $\boldsymbol{s}_m$ emphasizing low frequencies for the LR target modality. On the other hand, HR guidance modality images contain both high and low-frequency components, that enables the learned filters $\boldsymbol{g}_m$ to accommodate a wider frequency range. One can observe a high correlation between $\boldsymbol{g}_m$ and $\boldsymbol{w}_m$, as well as between $\boldsymbol{s}_m$ and corresponding $\boldsymbol{v}_m$. This correlation arises from the shared set of sparse codes (coefficients) $\boldsymbol{a}_m$ and $\boldsymbol{b}_m$, respectively. Also, it can be observed that the learned coupling dictionaries effectively combines the high frequency information from

HR guidance and low frequency information from LR target to synthesize the HR target image.

Table 5.3: Effect of Number of Filters (M) and Atom size (k) on JCCDL-MISR

A. Effect of M with k = 8×8

Number of Filters	Imgf5		Imgh3	
(M)	PSNR	SSIM	PSNR	SSIM
2	33.18	0.941	35.03	0.947
4	**38.98**	**0.954**	**40.36**	**0.961**
8	36.01	0.947	37.18	0.945

B. Effect of k with M = 4

Atom size	Imgf5		Imgh3	
(k)	PSNR	SSIM	PSNR	SSIM
4×4	33.91	0.915	34.17	0.944
8×8	**38.98**	**0.954**	**40.36**	**0.961**
12×12	33.87	0.931	35.29	0.960

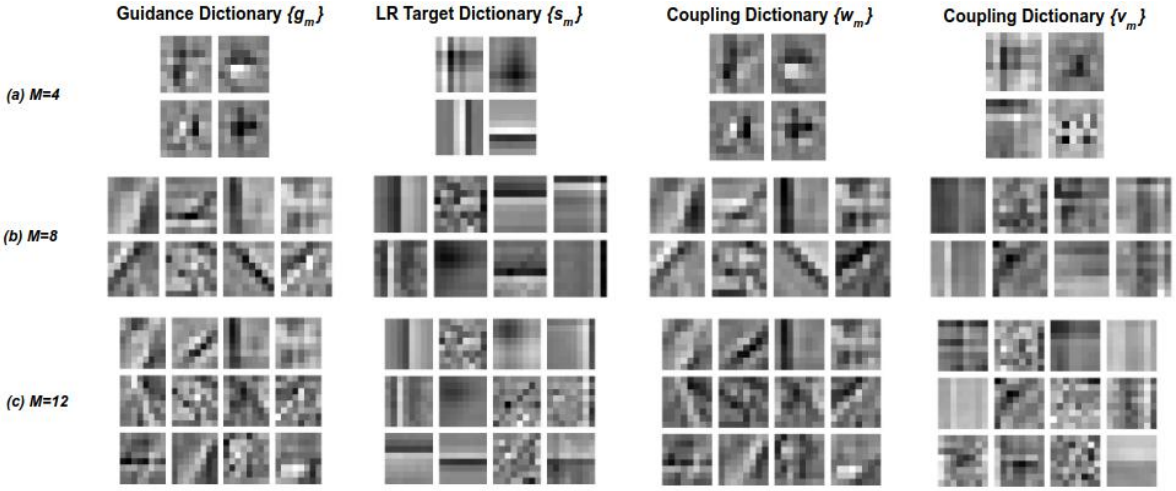


Figure 5.4: Visualization of different convolutional dictionary filters of atom size (k)=8×8 learned with RGB-MS data for different number of filters (M)

5.4.1.3 Effect of Atom size, k

Table 5.3 B displays the reconstruction performance for select test images from RGB-MS data using dictionary atom (filter) sizes of 4×4, 8×8, and 12×12, with M = 4. It is evident that the JCCDL-MISR configuration with atom size of k = 8×8 yields the best results (values highlighted in bold in Table 5.3 B). Other configurations exhibit poor results as large atom sizes tend to lose some local details while small atom sizes fail to capture global structural features, given the limited training data.

5.5 Summary

This chapter introduces JCCDL-MISR, a novel joint optimization technique utilizing CDL to enhance low-resolution images from the target modality by incorporating information from high-resolution images in the guidance modality. The solution steps and closed-form updates of the optimization problem are provided. The methodology leverages the superior representation of CDL, coupled with systematically formulated joint optimization steps, to achieve improved performance and the ability to work with limited data. The results demonstrate that the proposed method, trained with only 10 image pairs, outperforms benchmark methods trained with approximately 30-35 image pairs.

The subsequent chapter investigates the use of convolutional transform learning-based strategies, to further enhance the reconstruction performance. Convolutional transforms are the analysis variant of convolutional dictionaries that offer improved performance with reduced complexity. The motivation of the approach is discussed in Chapter 6, with the proposed formulation in Section 6.3.

Chapter 6

Supervised Domain Adaptation Via Convolutional Transform Learning for Multi-Modal Image Super-Resolution

6.1 Motivation

Conventional methods for MISR employing CNNs typically adopt an encoder-decoder architecture [89–91] that often demands massive data for optimal reconstruction and tends to overfit in data-limited scenarios. A possible solution is the use of sparse representation learning methods for MISR tasks. Both DL and TL-based frameworks have been explored for MISR, with the non-convolutional TL-based methods (JCTL-MISR and JCDTL-MISR methods discussed in Chapter 4) offering enhanced accuracy with reduced complexity over the DL variants [68]. Recently, CDL [81] employing shift-invariant dictionaries (filters) have also been applied for MISR [82] and are shown to provide improved image reconstruction over the non-convolutional DL variants.

Inspired by the enhanced data synthesis ability of TL over DL, this work presents a CTL framework for MISR that exploits the benefits of TL and the power of convolutional sparse coding to effectively model the complex depen-

dencies among the different imaging modalities for super-resolution tasks. Two novel joint learning formulations using the shallow (single-layer) and deep CTL variants are presented to learn the convolutional transforms and associated coefficients/features from the LR images of target modality and the HR images of guidance modality. Additionally, a fully connected (standard non-convolutional) transform is learned that combines (fuses) the modality-specific coefficients to effectively generate the HR images of the target modality. Unlike CNNs, which typically require an encoder-decoder architecture for data/signal synthesis, the proposed methods are fusion methods that fuse the information from the target and guidance modalities for data synthesis, thus eliminating the need for learning the decoder network. This results in fewer learning parameters that can work with limited data. Also, the learned convolutional filters (transforms) of the proposed method are constrained to be mutually different, promoting diversity in the learning paradigm which is not ensured in a CNN-based method. Also, unlike the dictionary-based synthesis sparse coding methods for MISR, the proposed methods offer enhanced performance with reduced complexity due to some of the inherent advantages of TL (discussed in Chapter 4).

It is to be noted that unlike the CTL-based fusion approach described in [92–94], which involves unsupervised/supervised learning of features from individual modalities or sensor channels for inference tasks, the proposed methods employs supervised learning for the inverse problem of data/image reconstruction that results in a different formulation.

In summary, following are the main contributions of this work:

- Two novel MISR methods employing CTL and DCTL, referred to as CTL-MISR and DCTL-MISR are presented that incorporate a supervised joint learning framework to learn the complex dependencies among the different modalities and effectively fuses them for enhancing the resolution of target modality.
- Unlike CNNs-based methods, the proposed formulations ensure that the learned filters are unique that results in few learning parameters, making them suitable for data-limited scenarios.
- Experimental validation using two datasets, accompanied by comparisons against benchmarks, optimal parameter settings and noise sensitivity analysis, are provided to demonstrate the effectiveness and applicability of the proposed method for MISR applications.

The rest of the chapter is organized as follows. Section 6.2 presents a brief background on CTL and DCTL that forms the basis of our proposed methods for MISR. Section 6.3 provides the details of the proposed MISR methods using both the shallow and deep variants of CTL. Experimental results, comparisons against benchmarks, and analysis for optimal parameter settings are given in Section 6.4. Finally, Section 6.5 presents the summary of the work.

6.2 Background on CTL and DCTL

This section presents a brief overview of the shallow (single-layer) and deep variants of CTL, which serve as the underlying basis for the proposed MISR formulations.

6.2.1 Convolutional Transform Learning (CTL)

CTL is an approach to learn convolutional filters from data/signals in an unsupervised fashion. Given the data $\{\boldsymbol{x}_i\}_{i=1}^{N}$ with N measurements each of d dimension, a set of M transform filters $\{\boldsymbol{t}_m\}_{m=1}^{M}$ are learnt to produce a set of features or coefficients $\{\boldsymbol{a}_{m,i}\}_{m=1}^{M}$ using the standard formulation [95]:

$$\min_{\boldsymbol{t}_m,\boldsymbol{a}_{m,i}} \sum_{i=1}^{N}\sum_{m=1}^{M}\Big(\|\boldsymbol{t}_m * \boldsymbol{x}_i - \boldsymbol{a}_{m,i}\|_2^2 + \phi(\boldsymbol{a}_{m,i})\Big) + \mu\|\boldsymbol{T}\|_F^2 - \lambda \log\det(\boldsymbol{T}) \tag{6.1}$$

Here, the first term is the data fidelity term where $*$ denotes the convolution operation and $\boldsymbol{a}_{m,i}$ are the coefficients, also called convolutional analysis sparse codes. The second term imposes regularization on the coefficients with the help of ϕ to avoid over-fitting. The remaining terms are associated with additional constraints on $\boldsymbol{T}$ with hyperparameters μ and λ, where $\boldsymbol{T} = [\boldsymbol{t}_1|\boldsymbol{t}_2|...|\boldsymbol{t}_M]$, a concatenation of transform filters and $\det(\boldsymbol{T})$ denotes its determinant. While the term $\|\boldsymbol{T}\|_F^2$ ensures that the values of transform filters remain bounded, $-\log\det(\boldsymbol{T})$ prevents trivial and degenerate solution, and ensures diversity in the learned filters.

The above equation (6.1) can be expressed in matrix form as:

$$\min_{\boldsymbol{T},\boldsymbol{A}} \|\boldsymbol{T}\cdot\boldsymbol{X} - \boldsymbol{A}\|_2^2 + \Phi(\boldsymbol{A}) + \mu\|\boldsymbol{T}\|_F^2 - \lambda \log\det(\boldsymbol{T}) \tag{6.2}$$

where $\boldsymbol{X} = [\boldsymbol{x}_1|\boldsymbol{x}_2|...|\boldsymbol{x}_N]$, $\boldsymbol{A} = [\boldsymbol{a}_{1,i}|\boldsymbol{a}_{2,i}|...|\boldsymbol{a}_{M,i}]_{1\leq i\leq N}$ and $\boldsymbol{T}\cdot\boldsymbol{X} = \begin{pmatrix} \boldsymbol{t}_1 * \boldsymbol{x}_1 & .. & \boldsymbol{t}_M * \boldsymbol{x}_1 \\ : & : & : \\ \boldsymbol{t}_1 * \boldsymbol{x}_N & .. & \boldsymbol{t}_M * \boldsymbol{x}_N \end{pmatrix}$. In (6.2), $\Phi(\boldsymbol{A}) = \sum_{m=1}^{M}\phi(\boldsymbol{a}_{m,i})$, which applies the penalty column-wise on $\boldsymbol{A}$. The solution to (6.2) is obtained using

AM [39] that solves for $\boldsymbol{T}$ and $\boldsymbol{A}$ in an alternative manner. More information on the closed-form updates and convergence guarantees can be found in [95].

6.2.2 Deep Convolutional Transform Learning (DCTL)

The deep version of CTL, referred to as DCTL is formulated by stacking multiple convolutional transforms one after the other to generate the features or coefficients. The formulation for N layer DCTL in matrix form is expressed as [96]:

$$\min_{\boldsymbol{T}_{j's},\boldsymbol{A}} \|\boldsymbol{T}_N \cdot (\boldsymbol{T}_{N-1} \cdot (\ldots(\boldsymbol{T}_1 \cdot \boldsymbol{X}))) - \boldsymbol{A}\|_2^2 + \Phi(\boldsymbol{A}) + \sum_{j=1}^{N} \{\mu \|\boldsymbol{T}_j\|_F^2 - \lambda \log\det(\boldsymbol{T}_j)\} \quad (6.3)$$

where $j = 1, .., N$ denotes the different layers of the N layer deep convolutional transform network. The solution to the problem in (6.3) can be obtained using an alternating proximal minimization algorithm [97].

With this brief introduction to CTL and DCTL, the subsequent section presents the details of the proposed methods for MISR.

6.3 Convolutional Transform Learning for Multi-Modal Image Super-Resolution

We address the same problem of MISR defined in our previous work (Chapter 4, Section 4.3.1). We follow the same naming convention for the different imaging modalities for uniformity and ease of reference. The proposed formulations employing CTL and DCTL frameworks are presented below.

6.3.1 Proposed MISR method using Convolutional Transform Learning (CTL-MISR)

Utilizing the concepts of CTL stated above, the proposed method learns dedicated convolutional transforms, $\boldsymbol{S}$ and $\boldsymbol{G}$ for the LR images of target modality $\boldsymbol{X}$ and

the HR images of guidance modality $\boldsymbol{Y}$, respectively. Since different modalities capture the same scene of interest, though they contain distinct features, they still share some common features, e.g., edges, texture, and shapes, that can be leveraged for image super-resolution. To facilitate this, the resulting modality-specific features $\boldsymbol{A}$ and $\boldsymbol{B}$ are combined with the help of non-convolutional transform $\boldsymbol{T}_f$ [44]. Here, $\boldsymbol{T}_f$ acts as a fully connected layer that learns the cross-modal relationship to generate the super-resolved (HR) images of the target modality $\boldsymbol{Z}$. The proposed method adopts a supervised learning paradigm to address MISR; the training and test phases are discussed below.

6.3.1.1 Training Phase

Using the knowledge of $\boldsymbol{Z}$ during training, joint learning is carried out that extracts the low frequency content from $\boldsymbol{X}$ and high frequency content from $\boldsymbol{Y}$ and combines them to efficiently synthesize $\boldsymbol{Z}$. The proposed joint optimization formulation is expressed as:

$$\begin{aligned}\min_{\boldsymbol{S},\boldsymbol{G},\boldsymbol{A},\boldsymbol{B},\boldsymbol{T}_f} & \|\boldsymbol{S}\cdot\boldsymbol{X}-\boldsymbol{A}\|_2^2+\|\boldsymbol{G}\cdot\boldsymbol{Y}-\boldsymbol{B}\|_2^2+\gamma\left\|\Psi(\boldsymbol{T}_f[\boldsymbol{A}^T|\boldsymbol{B}^T]^T)-\boldsymbol{Z}\right\|_2^2 \\ & +\Phi(\boldsymbol{A})+\Phi(\boldsymbol{B})+\mu_s\|\boldsymbol{S}\|_F^2-\lambda_s\log\det(\boldsymbol{S})+\mu_g\|\boldsymbol{G}\|_F^2 \\ & -\lambda_g\log\det(\boldsymbol{G})+\mu\|\boldsymbol{T}_f\|_F^2-\lambda\log\det(\boldsymbol{T}_f)\end{aligned} \tag{6.4}$$

where the first two terms learn the 2D transforms on $\boldsymbol{X}$ and $\boldsymbol{Y}$, to generate the features $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively. The third term learns the 1D fusing transform $\boldsymbol{T}_f$ that combines the features from the individual modalities to directly obtain $\boldsymbol{Z}$, with the help of hyperparameter γ and Sigmoid activation function Ψ. The fourth and fifth terms act as a regularizer on features $\boldsymbol{A}$ and $\boldsymbol{B}$, where Φ is the ReLU activation function. The remaining terms with hyperparameters $\mu_s, \lambda_s, \mu_g, \lambda_g, \mu$

and λ are associated with the additional constraints on the transforms $\boldsymbol{S}$, $\boldsymbol{G}$ and $\boldsymbol{T}_f$, respectively for learning effective representations.

It is important to note that since $\boldsymbol{Z}$ is known during training, the learning of $\boldsymbol{T}_f$ cannot result in a trivial or degenerate solution, thus eliminating the need for additional constraints on the fusing transform $\boldsymbol{T}_f$. The formulation in (6.4) is modified as:

$$\begin{aligned}\min_{\boldsymbol{S},\boldsymbol{G},\boldsymbol{A},\boldsymbol{B},\boldsymbol{T}_f} &\|\boldsymbol{S}\cdot\boldsymbol{X}-\boldsymbol{A}\|_2^2+\|\boldsymbol{G}\cdot\boldsymbol{Y}-\boldsymbol{B}\|_2^2+\gamma\left\|\Psi(\boldsymbol{T}_f[\boldsymbol{A}^T|\boldsymbol{B}^T]^T)-\boldsymbol{Z}\right\|_2^2 \\ &+\Phi(\boldsymbol{A})+\Phi(\boldsymbol{B})+\mu_s\|\boldsymbol{S}\|_F^2-\lambda_s\log\det(\boldsymbol{S})+\mu_g\|\boldsymbol{G}\|_F^2-\lambda_g\log\det(\boldsymbol{G})\end{aligned} \tag{6.5}$$

The key steps of CTL-MISR method are captured in the block diagram shown in Fig. 6.1. Notice that max-pooling is omitted in this formulation to prevent the loss of finer details, which are crucial for the super-resolution task. The problem in (6.5) can be solved using the Adaptive Moment Estimation (Adam) optimizer [98].

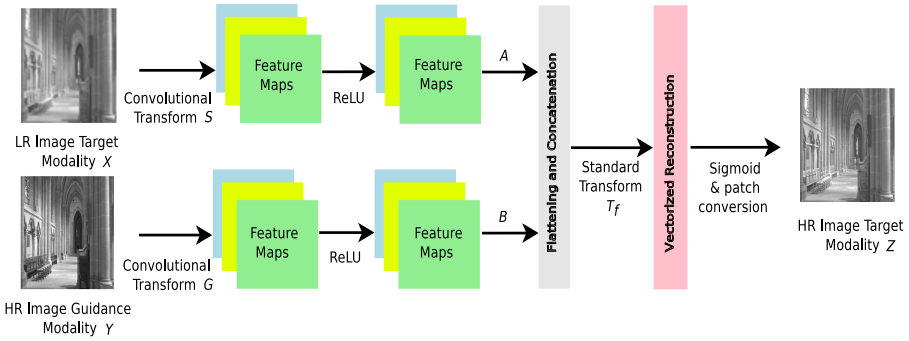


Figure 6.1: Block Diagram of the proposed CTL-MISR Method

6.3.1.2 Test Phase

During testing, given the LR target image $\boldsymbol{X}_{test}$ and HR guidance image $\boldsymbol{Y}_{test}$, first the features $\boldsymbol{A}_{test}$ and $\boldsymbol{B}_{test}$ are computed by transforming $\boldsymbol{X}_{test}$ and $\boldsymbol{Y}_{test}$ using convolutional transforms $\boldsymbol{S}$ and $\boldsymbol{G}$, respectively. Subsequently, the non-convolutional transform $\boldsymbol{T}_f$ is applied on the flattened and concatenated features

($\boldsymbol{A}_{test}$ and $\boldsymbol{B}_{test}$) to generate the HR target image $\boldsymbol{Z}_{test}$.

6.3.2 Proposed MISR method using Deep Convolutional Transform Learning (DCTL-MISR)

The proposed method employs the DCTL framework to exploit the correlation among different modalities in a supervised setting for MISR. Given the knowledge of HR images of target modality $\boldsymbol{Z}$, deep convolutional transforms $\boldsymbol{S}$ and $\boldsymbol{G}$ are learned for LR images of target modality $\boldsymbol{X}$ and HR images of guidance modality $\boldsymbol{Y}$, respectively. The associated coefficients $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively are augmented and a fusing non-convolutional transform $\boldsymbol{T}_f$ [44] is learned that acts as a fully connected layer to generate the HR images $\boldsymbol{Z}$. A joint learning is carried out in the training phase that learns the cross-modal relationship between the different imaging modalities. Similar to CTL-MISR method 6.3.1, the learned transforms are employed in the test phase to generate the HR of target modality $\boldsymbol{Z}$. More details on these two phases are presented in the subsequent sections.

6.3.2.1 Training Phase

In this phase, the cross-modal relationship between different imaging modalities is learned that essentially extracts the low-frequency information from $\boldsymbol{X}$ and high-frequency information from $\boldsymbol{Y}$ and effectively combines them to synthesize $\boldsymbol{Z}$. The proposed joint optimization formulation for MISR employing the N layer DCTL is expressed as follows:

$$
\begin{aligned}
&\min_{\boldsymbol{S}_{j's},\boldsymbol{G}_{j's},\boldsymbol{A},\boldsymbol{B},\boldsymbol{T}_f} \|\boldsymbol{S}_N\cdot(\boldsymbol{S}_{N-1}\cdot(\ldots(\boldsymbol{S}_1\cdot\boldsymbol{X})))-\boldsymbol{A}\|_2^2+\Phi(\boldsymbol{A})+\Phi(\boldsymbol{B})\\
&+\|\boldsymbol{G}_N\cdot(\boldsymbol{G}_{N-1}\cdot(\ldots(\boldsymbol{G}_1\cdot\boldsymbol{Y})))-\boldsymbol{B}\|_2^2+\gamma\left\|\Psi(\boldsymbol{T}_f[\boldsymbol{A}^T|\boldsymbol{B}^T]^T)-\boldsymbol{Z}\right\|_2^2\\
&+\mu\,\|\boldsymbol{T}_f\|_F^2-\lambda\log\det(\boldsymbol{T}_f)+\sum_{j=1}^{N}\{\mu_s\|\boldsymbol{S}_j\|_F^2-\lambda_s\log\det(\boldsymbol{S}_j)\}\\
&+\sum_{j=1}^{N}\{\mu_g\|\boldsymbol{G}_j\|_F^2-\lambda_g\log\det(\boldsymbol{G}_j)\}
\end{aligned}
\tag{6.6}
$$

Here, the first four terms are for learning the deep transforms $\boldsymbol{S}$ and $\boldsymbol{G}$ and their associated features $\boldsymbol{A}$ and $\boldsymbol{B}$ from $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively. The fifth term is for learning the fusing transform $\boldsymbol{T}_f$ on the features obtained from the individual modalities to generate $\boldsymbol{Z}$. Here, the penalty function Φ is a ReLU activation and Ψ denotes a Sigmoid function. The remaining terms are related to the additional constraints on the transforms that allows unique filters to be learnt. The hyperparameters $\mu_s, \lambda_s, \mu_g, \lambda_g, \mu$, λ and γ control the tradeoff between the data fidelity and regularization terms.

Since $\boldsymbol{Z}$ is known during training, the fusing transform $\boldsymbol{T}_f$ can never result in a trivial or degenerate solution; hence the additional constraints on $\boldsymbol{T}_f$ in (6.6) can be relaxed, resulting in the modified formulation:

$$
\begin{aligned}
&\min_{\boldsymbol{S}_{j's},\boldsymbol{G}_{j's},\boldsymbol{A},\boldsymbol{B},\boldsymbol{T}_f} \|\boldsymbol{S}_N\cdot(\boldsymbol{S}_{N-1}\cdot(\ldots(\boldsymbol{S}_1\cdot\boldsymbol{X})))-\boldsymbol{A}\|_2^2+\Phi(\boldsymbol{A})+\Phi(\boldsymbol{B})\\
&+\|\boldsymbol{G}_N\cdot(\boldsymbol{G}_{N-1}\cdot(\ldots(\boldsymbol{G}_1\cdot\boldsymbol{Y})))-\boldsymbol{B}\|_2^2+\gamma\left\|\Psi(\boldsymbol{T}_f[\boldsymbol{A}^T|\boldsymbol{B}^T]^T)-\boldsymbol{Z}\right\|_2^2\\
&+\sum_{j=1}^{N}\{\mu_s\,\|\boldsymbol{S}_j\|_F^2-\lambda_s\log\det(\boldsymbol{S}_j)\}+\sum_{j=1}^{N}\{\mu_g\,\|\boldsymbol{G}_j\|_F^2-\lambda_g\log\det(\boldsymbol{G}_j)\}
\end{aligned}
\tag{6.7}
$$

The above problem can be solved using Adam optimizer [98]. Fig. 6.2 presents the block diagram view of the proposed N layer DCTL-MISR method.

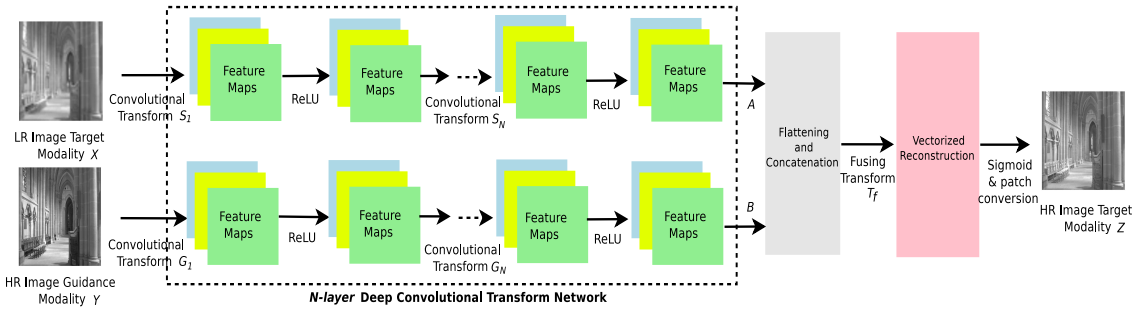


Figure 6.2: Block Diagram of the proposed DCTL-MISR Method

6.3.2.2 Test Phase

In this phase, given the LR target images $\boldsymbol{X}_{test}$ and the HR guidance images $\boldsymbol{Y}_{test}$, deep features $\boldsymbol{A}_{test}$ and $\boldsymbol{B}_{test}$ are generated by applying the learned deep convolutional transforms $\boldsymbol{S}$ and $\boldsymbol{G}$, respectively. These features are flattened and concatenated, and the fusing transform $\boldsymbol{T}_f$ is applied to generate the HR target images $\boldsymbol{Z}_{test}$.

6.4 Results Discussion

The proposed CTL-MISR and DCTL-MISR methods are evaluated for MISR using the same two datasets, RGB-MS [79] and RGB-NIR [80] discussed in Chapter 4, Section 4.4.1. Also, the benchmark methods as mentioned in Chapter 4, Section 4.4.2 are considered for performance evaluation. Additionally, comparisons with the shallow and deep variants of Joint Coupled Transform Learning (JCTL-MISR, JCDTL-MISR) (Chapter 4, Section 4.3.2 and 4.3.3) are provided to highlight the improvement gained with the convolutional transform variants. Kindly note that 3 layer JCDTL-MISR configuration is considered for comparison as it gave the best results for the datasets considered in this work. Few more methods like, Multi-modal Convolutional Dictionary Learning (MCDL [82]),

JCCDL-MISR (discussed in Chapter 5, Section 5.3) and deep learning-based Guided Pixel to Pixel Transformation (Pix to Pix) [99] are employed for performance comparison. In total, ten different MISR techniques based on filtering and learning-based paradigms (deep learning and sparse representation learning) are employed for benchmarking.

The data pre-processing remains the same as presented in Chapter 4, Section 4.4.3. In this work, grid search is used for tuning the hyperparameters of all the methods for optimal performance. The best results on both datasets were achieved with 5 filters of kernel size 3×3 for the proposed CTL-MISR and 4 filters of kernel (atom) size 8×8 for the MCDL. For DCTL-MISR, a 3 layer deep configuration with 5 filters of kernel size 3×3 gave optimal results for RGB-NIR dataset. On the other hand, a 5 layer DCTL-MISR configuration with 9 filters of kernel size 3×3 provided the best results for for RGB-MS dataset. The training for both the proposed methods was performed with a batch size of 32 and 50 epochs with hyperparameters values $\mu_s = \mu_g = 0.0001$, $\lambda_s = \lambda_g = 0.001$ and $\gamma = 1$ employing Adam optimizer with learning rate = 0.001. Kindly note that same kernel size and number of filters are considered for all the deep layers of DCTL-MISR.

Tables 6.1 and 6.2 summarize the reconstruction results for five random images obtained with different methods on both datasets, assessed in terms of PSNR (dB) and SSIM metrics. The results for RGB-MS are reported for 640 nm. The highest and second highest values of the PSNR and SSIM are highlighted in bold and underline, respectively. The proposed DCTL-MISR method displays

Table 6.1: MISR Results with RGB-NIR for 4×

RGB-NIR Dataset										
	Indoor 4		Indoor 5		Indoor 11		Indoor 16		Indoor21	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Proposed DCTL-MISR (N = 3)	**33.909**	0.933	**35.221**	0.949	31.648	0.911	**36.443**	**0.951**	**32.561**	0.921
Proposed CTL-MISR	32.649	0.922	33.513	0.941	31.534	0.908	34.219	0.941	31.696	0.910
JCCDL-MISR (Chapter 5)	31.644	0.942	32.058	0.958	**31.711**	**0.925**	31.801	0.944	31.403	**0.924**
MCDL [82]	28.932	0.902	28.307	0.908	28.874	0.842	30.601	0.914	28.787	0.865
JCDTL-MISR (Chapter 4)	30.629	0.941	30.580	0.951	30.384	0.903	33.070	0.950	29.668	0.915
JCTL-MISR (Chapter 4)	30.341	0.915	30.414	0.937	28.879	0.896	32.018	0.925	29.554	0.895
Coupled DL [68]	30.021	0.905	29.865	0.915	27.808	0.893	31.438	0.916	29.442	0.893
DJF [65]	26.958	0.898	27.804	0.899	27.015	0.817	30.149	0.890	27.619	0.853
Pix to Pix [99]	27.702	0.942	33.726	0.966	26.515	0.919	31.071	0.941	25.703	0.903
JR [72]	26.271	0.841	25.076	0.939	22.864	0.815	23.502	0.867	21.626	0.794
GF [70]	29.854	**0.946**	32.052	**0.971**	27.589	0.901	31.916	0.938	27.133	0.909
JBF [71]	26.354	0.919	31.283	0.968	26.480	0.906	30.431	0.929	25.746	0.902

Table 6.2: MISR Results with RGB-MS for 4×

RGB-MS Dataset										
	Imge6		Imge7		Imgf5		Imgf7		Imgh3	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Proposed DCTL-MISR (N = 5)	**34.271**	**0.891**	**42.674**	0.984	**43.051**	**0.982**	**36.191**	**0.948**	**45.453**	**0.985**
Proposed CTL-MISR	33.597	0.883	39.969	0.973	40.979	0.972	34.508	0.933	43.186	0.981
JCCDL-MISR (Chapter 5)	31.779	0.856	36.732	0.928	38.984	0.954	34.830	0.907	40.361	0.961
MCDL [82]	25.477	0.605	28.955	0.709	33.426	0.880	27.894	0.722	35.564	0.896
JCDTL-MISR (Chapter 4)	31.441	0.841	35.496	0.899	37.522	0.947	33.339	0.888	39.403	0.948
JCTL-MISR (Chapter 4)	31.049	0.835	33.222	0.889	36.277	0.939	31.964	0.864	37.140	0.941
Coupled DL [68]	28.793	0.814	32.669	0.877	34.239	0.906	31.401	0.878	36.107	0.920
DJF [65]	20.968	0.828	26.732	0.938	32.588	0.824	23.851	0.902	30.788	0.924
Pix to Pix [99]	29.726	0.875	42.218	**0.988**	37.861	0.978	31.721	0.942	37.421	0.974
JR [72]	26.519	0.814	32.781	0.889	33.933	0.890	29.295	0.804	33.999	0.922
GF [70]	25.332	0.774	29.709	0.869	31.706	0.901	28.045	0.880	33.518	0.807
JBF [71]	25.535	0.746	29.655	0.799	32.411	0.886	28.874	0.839	34.461	0.902

superior performance for RGB-MS dataset with an average improvement > 3 dB in PSNR and $> 1\%$ in SSIM against the best-performing (non CTL-based) benchmark method. However, although the PSNR is consistently good for RGB-NIR dataset with an average PSNR > 2 dB (over the best-performing benchmark), the SSIM value is high for most of the images. It is important to note that in comparison to the JCCDL-MISR method (detailed in Chapter 5, Section 5.3), the proposed CTL-MISR method alone yields an average improvement ≈ 1 dB in PSNR and $\approx 1\%$ in SSIM on RGB-NIR dataset; and ≈ 2 dB in PSNR and $\approx 2\%$ in SSIM on RGB-MS dataset. One can notice that filtering based

techniques (unsupervised) namely, GF and Pix to Pix display improved SSIM for this case as they focus on improving the edge information. The proposed shallow (single-layer) variant CTL-MISR gives the second best performance for most of the images.

Among the learning-based MISR techniques, single-stage methods like Coupled DL, JCTL-MISR, JCDTL-MISR, JCCDL-MISR, CTL-MISR and DCTL-MISR that employ a joint-learning paradigm report improved performance compared to the two-stage MCDL approach. In most cases, the results are even superior to the filtering-based (unsupervised) methods. This can be attributed to the fact that joint learning facilitates effective mapping to be learned between the different modalities that yield improved results. It is important to note that the two-stage MCDL requires learning 6 convolutional dictionaries and 3 associated sparse coefficients. In contrast, the proposed CTL-MISR approach achieves superior performance with reduced complexity by learning only 3 convolutional transforms and 2 associated sparse coefficients. It can be seen that CTL-based formulations (CTL-MISR and DCTL-MISR) outperform the non-convolutional TL-based methods (JCTL-MISR and JCDTL-MISR). This can be attributed to the potential of convolutional filters in extracting effective representations from different modalities for enhanced reconstruction. For illustration, Fig. 6.3 presents the reconstruction results for one test image obtained from different methods, along with the error maps with respect to ground truth. One can observe the improved enhancement achieved by the proposed methods compared to other benchmark techniques.

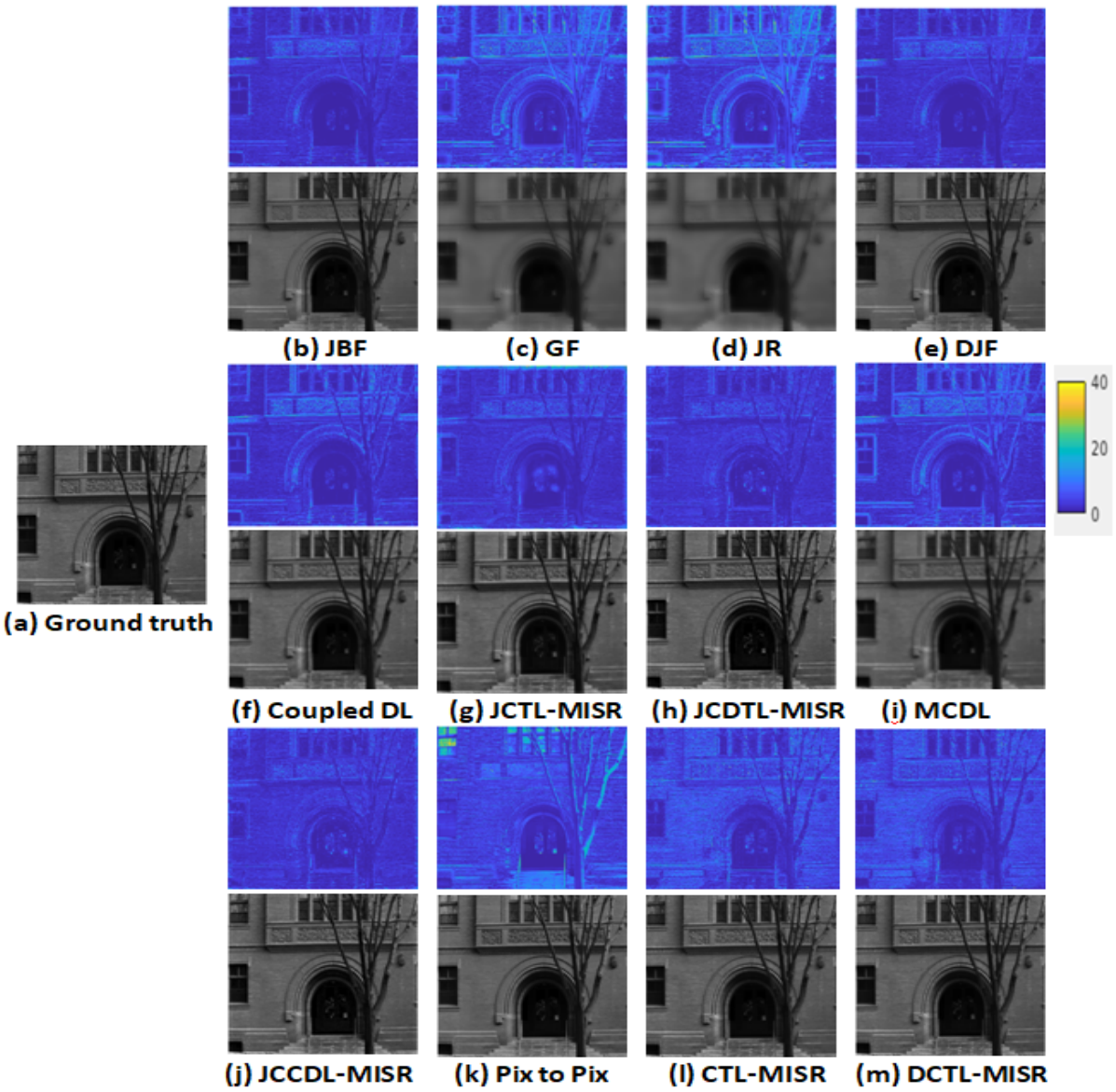


Figure 6.3: Error Maps and Reconstructed images for 'Imge6' image of RGB-MS dataset

The proposed CTL-MISR and DCTL-MISR methods are implemented using PyTorch on an AMD Ryzen 5 4500U CPU@2.3GHz with 16 GB RAM. CTL-MISR and DCTL-MISR took $\approx$ 10 and 15 minutes, respectively, for training across 50 iterations over 30 image pairs, compared to 1-2 hours taken by the MCDL method. Also, the proposed methods took $\approx$ 30 milliseconds for testing a single image pair as opposed to $\approx$ 2.5 seconds taken by the MCDL method. This demonstrates the computational advantage gained over the MCDL approach. Fig.

6.4 shows that CTL-MISR converges quickly, within few iterations. A similar convergence is observed for the deep variant.

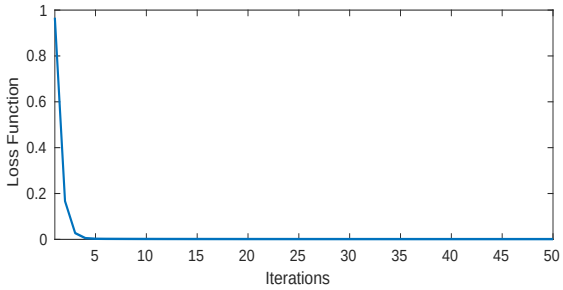


Figure 6.4: Convergence Plot of the CTL-MISR Method

6.4.1 Optimal Parameter Settings

We conducted various experiments with different configurations of CTL-MISR and DCTL-MISR to study the effect of the number of filters and kernel size on the MISR performance. Additionally, the effect of noise on the reconstruction capability of CTL-MISR method was studied. The analysis of the optimal parameter settings for DCTL-MISR also included the effect of the number of layers on the reconstruction results.

6.4.1.1 Experiments with CTL-MISR

A. Effect of Number of Filters (M): In this analysis, we explore the performance of various configurations of CTL-MISR, considering 3, 5, and 9 filters with a constant kernel size of 3×3. The results presented in Table 6.3 A for two test images from the RGB-NIR dataset reveal that CTL-MISR with $M = 5$ (values highlighted in bold) gives optimal performance in both metrics. The suboptimal performance of other configurations can be attributed to the inability of fewer filters to learn robust representations and challenges in generalization with more filters, especially with limited training data.

B. Effect of Kernel Size: Here, different kernel sizes of 3×3, 5×5, and 9×9 are considered, with M = 5. It can be seen from Table 6.3 B that 3×3 and 5×5 display comparable performance for both the test images, with the former marginally better in terms of PSNR (values highlighted in bold). Since larger kernels tend to lose some local details, one can observe a decline in performance for 9×9 configuration.

Table 6.3: Effect of Number of Filters (M) and Kernel Size on CTL-MISR

A. Effect of M with Kernel Size = 3×3

	Indoor 4		Indoor 5	
(M)	PSNR	SSIM	PSNR	SSIM
3	31.506	0.922	31.901	0.937
5	**32.649**	**0.922**	**33.513**	**0.941**
9	31.777	0.913	33.051	0.935

B. Effect of Kernel Size with M = 5

Kernel Size	Indoor 4		Indoor 5	
	PSNR	SSIM	PSNR	SSIM
3×3	**32.649**	**0.922**	**33.513**	**0.941**
5×5	32.325	0.922	33.319	0.941
9×9	32.034	0.913	32.157	0.929

C. Noise Sensitivity Analysis: To evaluate the impact of noise on the reconstruction performance of the proposed method, Gaussian noise with zero mean and varying variances (0, 0.001, and 0.0001) was added to the LR test images, following the approach in [68]. The model, trained on noise-free LR images, was directly tested on the noisy inputs. Fig. 6.5 illustrates the reconstruction results obtained using CTL-MISR for a representative noisy LR test image. The corresponding PSNR and SSIM values for both the LR and reconstructed images are summarized in Table 6.4. The results indicate that the method successfully enhances image resolution even under noisy conditions. For a broader comparison, Fig. 6.6 presents the average PSNR and SSIM values for the RGB-NIR dataset across all sparse representation learning-based methods. It is observed that while all methods perform well at lower noise levels, their effectiveness diminishes as noise increases. Among them, the proposed CTL-MISR consistently achieves

the highest PSNR and the second-highest SSIM values across all noise levels.

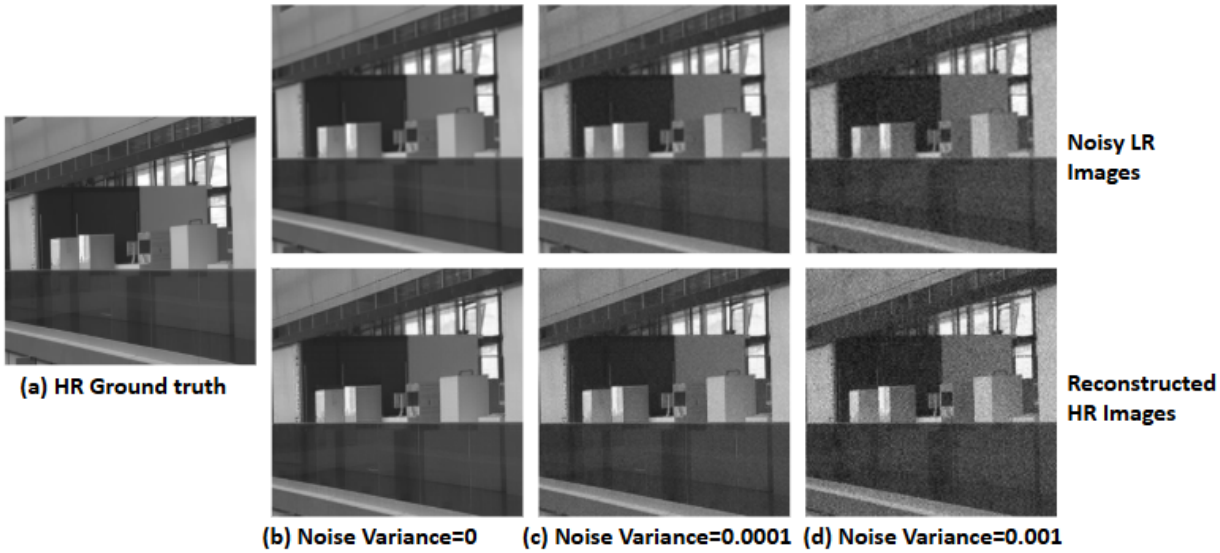


Figure 6.5: Results for 'Indoor 4' image of RGB-NIR dataset with the noisy LR image at the top and reconstructed HR image at the bottom.

Table 6.4: Reconstruction Performance with Noisy LR Images for 'Indoor 4'

	Noise Variance=0		Noise Variance=0.0001		Noise Variance=0.001	
Image	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
LR	29.87	0.89	27.84	0.86	26.91	0.64
CTL-MISR	32.65	0.92	32.47	0.88	27.76	0.74

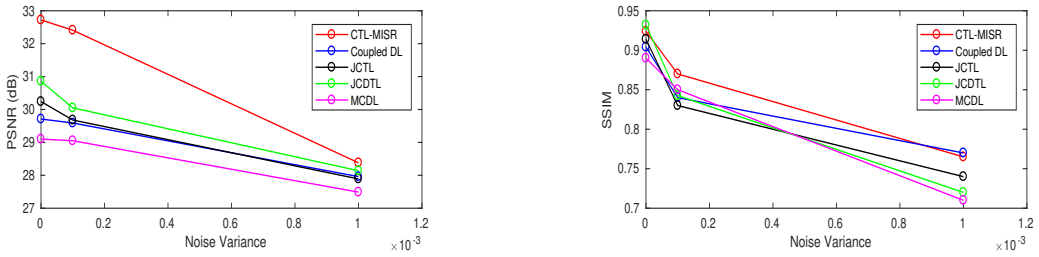


Figure 6.6: Performance Comparison with Noisy Test Images

6.4.1.2 Experiments with DCTL-MISR

The average values of PSNR and SSIM obtained with different DCTL-MISR configurations computed over all the test images presented in Tables 6.1 and 6.2 are reported in Tables 6.5, 6.6 and 6.7. Note that a 3 layer and 5 DCTL-MISR configuration is considered for RGB-NIR and RGB-MS, respectively, in Tables 6.6 and 6.7 since they gave the best results.

A. Number of Layers, N: Experiments are conducted considering different number of layers in the DCTL-MISR configuration. Here, a 3×3 kernel is used with 5 filters for RGB-NIR and 9 filters for RGB-MS, which gave optimal performance as reported in Tables 6.1 and 6.2. Table 6.5 shows that while a 3 layer DCTL-MISR configuration works best for RGB-NIR, a 5 layer DCTL-MISR configuration is best suited for RGB-MS dataset (values highlighted in bold). Notice the improved reconstruction ability of deep configuration (DCTL-MISR) over the shallow CTL counterpart (CTL-MISR). Deep configuration learns rich information, so the performance keeps improving as we go deep until saturation is observed. Going deep beyond that point does not yield any significant improvement; infact, a slight degradation in performance is observed for both datasets, which can be due to overfitting on limited data. Notice an improvement > 1 dB in PSNR and $> 1\%$ in SSIM gained with the deep architecture over the shallow (single-layer) counterpart.

Table 6.5: Effect of Number of Layers (N) on DCTL-MISR

# Layers (N)	**RGB-NIR**		**RGB-MS**	
	PSNR	SSIM	PSNR	SSIM
1	32.552	0.922	37.228	0.956
2	32.051	0.913	38.124	0.961
3	**33.653**	**0.933**	38.098	0.956
4	33.651	0.929	39.045	0.963
5	33.612	0.928	**39.344**	**0.966**
6	33.641	0.931	38.765	0.963

B. Number of Filters, M: Here, different DCTL-MISR configurations considering 3, 5, 9 and 12 filters with a constant kernel size of 3×3 are analyzed for performance comparison. Table 6.6 shows that 5 and 9 filters result in optimal performance for RGB-NIR and RGB-MS, respectively (values highlighted in

bold). The performance is sub-optimal for other configurations since fewer filters struggle to learn effective representation, and more filters face problems with generalization, especially with limited data.

Table 6.6: Effect of Number of Filters (M) on DCTL-MISR

# Filters (M)	**RGB-NIR**		**RGB-MS**	
	PSNR	SSIM	PSNR	SSIM
3	33.294	0.932	37.067	0.939
5	**33.653**	**0.933**	38.493	0.956
9	33.228	0.925	**39.344**	**0.966**
12	32.871	0.918	36.845	0.954

C. Kernel Size: In this analysis, different kernel sizes of 3×3, 5×5, and 9×9 are studied with M = 5, 9 for RGB-NIR and RGB-MS datasets, respectively. In Table 6.7, one can observe that 3×3 kernel performs best for both datasets (values highlighted in bold). Since we consider image patches of size 16×16, larger kernels tend to lose some local details; hence, a decline in performance is observed for other configurations.

Table 6.7: Effect of Kernel Size on DCTL-MISR

Kernel Size	**RGB-NIR** (M = 5)		**RGB-MS** (M = 9)	
	PSNR	SSIM	PSNR	SSIM
3×3	**33.653**	**0.933**	**39.344**	**0.966**
5×5	33.479	0.931	37.876	0.959
9×9	33.208	0.922	37.348	0.951

6.5 Summary

Two novel CTL-based methods namely, CTL-MISR and DCTL-MISR are presented to generate the HR image of target modality from its LR counterpart, leveraging guidance from the HR image of another modality. Joint optimization frameworks are proposed to simultaneously learn the transforms and representations that capture both similarities and differences among various imaging modalities and effectively fuses them for super-resolution tasks. Compared to

conventional learning-based methods, these frameworks require fewer learning parameters making them suitable for real-life applications with limited data. Experimental results demonstrate the superior performance of the proposed methods compared to benchmarks, highlighting its resilience to noise and its capability to learn effectively, even with limited data.

Chapter 7

Conclusion

This dissertation proposed novel DA approaches that utilize dictionaries, transforms, and their convolutional variants to address classification and inverse problems. A brief summary of the key contributions and potential directions for future research is provided below.

7.1 Summary of Contribution

In this section, we will briefly summarize the chapter-wise contribution in the area of DA addressing the two research problems discussed in this dissertation. Additionally, the broader applicability of the proposed methods is examined, and their limitations are discussed, both of which offer valuable directions for future research.

7.1.1 Unsupervised Domain Adaptation Via Subspace Interpolating Deep Dictionary Learning for Classification

Chapter 2 introduced a deep DL-based subspace interpolation (DDL-UDA) method that links the source and target domain data for unsupervised adaptation. Deep dictionaries could learn rich and reliable data representations that assisted

in learning the source-to-target mapping more effectively, thereby addressing the domain shift problem. The requisite formulation and the solution steps were detailed. Experiments conducted on the bearing dataset for the challenging adaptation between *different but related machines* scenarios demonstrated the potential of the proposed DDL-UDA method for UDA. DDL-UDA significantly outperformed all benchmark methods, suggesting its applicability to real-life industrial applications.

7.1.2 Unsupervised Domain Adaptation Via Subspace Interpolating Transform Learning for Classification

Chapter 3 explored the use of TL for subspace modeling and introduced novel transform-based subspace interpolation methods for UDA using TL and deep TL frameworks. The approach involves modeling the source and target domain data as low-dimensional subspaces using shallow (single-layer) and deep transforms, respectively. These methods learn intermediate domains to bridge the gap between the two domains, facilitating the generation of domain-invariant features for cross-domain classification tasks. Detailed formulations employing TL and DTL and their corresponding closed-form updates were provided. Experiments on bearing datasets demonstrated the effectiveness of the proposed methods in addressing the challenging adaptation between *different but related machines*. An accuracy improvement of $\approx 5\%$ and $>10\%$ was achieved against the dictionary-based counterparts and the best-performing DCNN, respectively. The results highlighted the potential of the proposed methods in learning reliable data representations, particularly in the limited data scenario, rendering them suitable

for real-world industrial applications.

7.1.3 Supervised Domain Adaptation Via Joint Coupled Transform Learning for Multi-Modal Image Super-Resolution

In Chapter 4, two novel joint optimization formulations for MISR utilizing the TL framework: JCTL-MISR and its deep variant JCDTL-MISR were introduced. Through joint optimization, the transforms for individual modalities were learned while ensuring that the coefficients of the modalities are related to each other, thus capturing cross-modal dependencies. All the solution steps and necessary closed-form updates for the proposed formulations were presented. Results obtained from two publicly available datasets, RGB-NIR and RGB-MS, showcased the enhanced reconstruction performance of the proposed techniques compared to benchmark methods.

7.1.4 Supervised Domain Adaptation Via Joint Coupled Convolutional Dictionary Learning for Multi-Modal Image Super-Resolution

In Chapter 5, we introduced a joint optimization technique employing coupled CDL (JCCDL-MISR) to enhance LR images from the target modality by integrating information from HR images of the guidance modality. The coupling helped to extract the low-frequency information from the LR images of the target modality and high-frequency information from the HR images of the guidance modality. These two sources of information were effectively fused to synthesize the HR images of the target modality. The solution steps and closed-form updates of the proposed optimization problem were presented. This methodology

harnessed the superior representation of CDL, combined with systematically formulated joint optimization steps, to achieve enhanced performance and the capability to operate effectively with limited data. The results show that the proposed method, trained with only 10 image pairs, surpasses benchmark methods trained with more image pairs.

7.1.5 Supervised Domain Adaptation Via Convolutional Transform Learning for Multi-Modal Image Super-Resolution

In Chapter 6, we introduced two novel methods based on CTL, namely CTL-MISR and DCTL-MISR, aimed at generating HR images of a target modality from their LR counterparts by leveraging guidance from HR images of another modality. We proposed joint optimization frameworks to simultaneously learn transforms and representations, capturing both similarities and differences among various imaging modalities and effectively fusing them for super-resolution tasks. Unlike conventional learning-based methods, the proposed frameworks required fewer learning parameters, making them suitable for real-world applications with limited data. Experimental results demonstrated the superior performance of the proposed methods compared to benchmarks. They exhibited resilience to noise and demonstrated effective learning, even with limited data.

In summary, this thesis introduces both unsupervised and supervised domain adaptation approaches. To provide further context and highlight the broader relevance of the work, potential applications to other domains and limitations are discussed below.

1. Unsupervised domain adaptation via subspace interpolation using dictionaries and transforms for machine inspection: While our work focuses on machine fault diagnosis, the underlying approach is domain-agnostic and can be extended to various other fields where distributional shifts are common. While prior art on subspace based adaptation methods [1, 25] covers vision problem like, face recognition, object recognition, a few other applications include:

(i) Medical imaging – addressing domain shifts due to equipment differences, imaging protocols, or demographic variations.

(ii) Speech recognition – adapting to differences in accents and recording conditions.

(iii) Autonomous driving – managing variability in sensors and driving environments across regions.

(iv) Remote sensing and agriculture – handling data from different satellites or seasons.

Limitation: Subspace interpolation methods often assume that the domain shift can be modeled by smooth transformations in subspace. In some cases, target distributions may include outliers that may violate these assumptions, leading to poor adaptation performance.

2. Supervised domain adaptation using coupled dictionaries and transforms for MISR: The core methodologies underlying the proposed models for image enhancement and fusion are fundamentally extensible to broader domains, like:

(i) Medical Imaging: Multimodal approaches like PET-MRI or CT-MRI can benefit from joint representations to enhance diagnostic resolution.

(ii) Remote Sensing: Fusion of modalities like SAR and optical imagery for fine-grained analysis in change detection or land cover classification.

(iii) Surveillance and Security: Infrared and RGB data fusion can be improved using these frameworks to enhance visibility under poor lighting conditions.

Limitation: These methods need registered multi-modal image pairs and are not flexible for unregistered image domains. In many real-world scenarios, perfect alignment or pairing may not be available.

Future work could address these challenges to improve generalization ability of the proposed methods across domains.

7.2 Future Work

In this Section, some other tasks that can be taken up in the future are spelled out. Rather than mentioning them at a high level, an effort has been made to point out the directions toward solving the extensions or new threads for the research work of this thesis.

1. Following the work in [25] that employs domain-adaptive dictionaries, domain-adaptive transforms can be explored for UDA, which learns the common transform and domain-specific transforms via subspace interpolation to model the domain shift between the source and target domain. Fig. 7.1 shows the block diagram of domain-adaptive TL for UDA. Here, $\boldsymbol{T}^c$ is the common transform and $\boldsymbol{T}^0, .., \boldsymbol{T}^m, .., \boldsymbol{T}^t$ are the domain-specific transforms with coefficients $\boldsymbol{Z}^0, .., \boldsymbol{Z}^m, .., \boldsymbol{Z}^t$ and $\boldsymbol{\Gamma}^0, .., \boldsymbol{\Gamma}^m, .., \boldsymbol{\Gamma}^t$ for the respective transforms across the different subspaces. The common transform will be shared by all the domains and hence will extract domain-shared features. The domain-specific transforms,

incoherent to the common transform, will extract domain-specific features capturing the domain changes. This method will aid in learning more compact and reconstructive transforms, thereby addressing the domain shift more effectively.

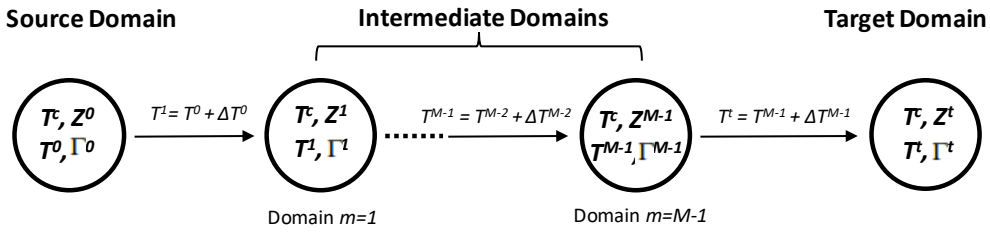


Figure 7.1: UDA via Interpolation using Domain-adaptive TL

2. A deep extension of the proposed JCCDL-MISR method can be explored to study the performance with the deep network. Here, considering a 3 layer deep CDL architecture, the data $\{\boldsymbol{x}_i\}_{i=1}^{N}$ with N measurements each of n dimension, is expressed as $\boldsymbol{x}_i = \sum_{m=1}^{M} \boldsymbol{d}_m{}^1 * \boldsymbol{d}_m{}^2 * \boldsymbol{d}_m{}^3 * \boldsymbol{a}_{m,i}$, where $\{\boldsymbol{d}_m\}_{m=1}^{M}$ are the dictionary filters and $\{\boldsymbol{a}_{m,i}\}_{m=1}^{M}$ are the set of coefficients. With $\boldsymbol{D}_m{}^1 \boldsymbol{D}_m{}^2 \boldsymbol{D}_m{}^3 \boldsymbol{a}_{m,i} = \boldsymbol{d}_m{}^1 * \boldsymbol{d}_m{}^2 * \boldsymbol{d}_m{}^3 * \boldsymbol{a}_{m,i}$, using the matrix notation similar to (5.2) in Chapter 5, Section 5.2, the deep dictionary formulation is given as:

$$\min_{\boldsymbol{D}^1, \boldsymbol{D}^2, \boldsymbol{D}^3, \boldsymbol{A}} \frac{1}{2} \left\| \boldsymbol{X} - \boldsymbol{D}^1 \boldsymbol{D}^2 \boldsymbol{D}^3 \boldsymbol{A} \right\|_F^2 + \lambda \left\| \boldsymbol{A} \right\|_1 \tag{7.1}$$

In the deep JCCDL-MISR formulation (block diagram shown in Fig. 7.2), the modality-specific convolutional dictionaries $\boldsymbol{S}$ and $\boldsymbol{G}$ can be made deep using (7.1) employing appropriate number of layers for learning the fine structure details of each modality. The corresponding deep features $\boldsymbol{A}$ and $\boldsymbol{B}$ can be fused using single-layer coupled convolutional dictionaries $\boldsymbol{W}$ and $\boldsymbol{V}$ to synthesize the HR of target modality, by employing a joint optimization formulation (similar to JCCDL-MISR).

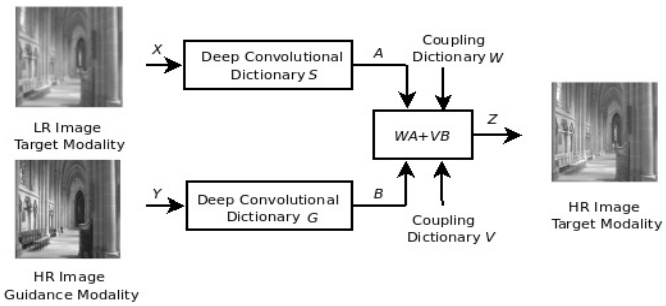


Figure 7.2: Block Diagram of the deep JCCDL-MISR Method

3. Semi-supervised domain adaptation using TL formulation can be explored to incorporate the limited labels of the target data for improved performance over the unsupervised methods. Since the data from the source and target domain may not be correlated in the original space (due to domain shift), the data from both domains can be projected to a common low-dimensional feature space while maintaining the manifold structure of the data, similar to the work in [24]. With the help of available labels, class-discriminant transform can be learned on the projected feature space to help with classification. The block diagram of TL-based semi-supervised DA is presented in Fig. 7.3.

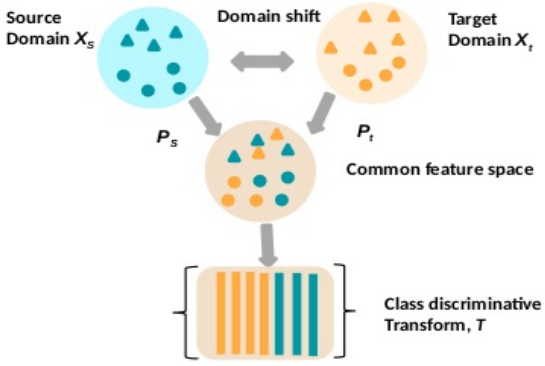


Figure 7.3: Block Diagram of the TL-based Semi-Supervised DA

A joint optimization formulation can be developed that will learn the projections of the source and target domain data along with a common transform, expressed as:

$$\{\boldsymbol{T}^{\star}, \boldsymbol{P}_s^{\star}, \boldsymbol{P}_t^{\star}, \boldsymbol{Z}_s^{\star}, \boldsymbol{Z}_t^{\star}\} = \underset{\boldsymbol{T}, \boldsymbol{P}_s, \boldsymbol{P}_t, \boldsymbol{Z}_s, \boldsymbol{Z}_t}{\min} \|\boldsymbol{T}(\boldsymbol{P}_s \boldsymbol{X}_s) - \boldsymbol{Z}_s\|_F^2 + \|\boldsymbol{T}(\boldsymbol{P}_t \boldsymbol{X}_t) - \boldsymbol{Z}_t\|_F^2$$
$$+ \gamma(\|\boldsymbol{X}_s - (\boldsymbol{P}_s{}^T \boldsymbol{P}_s)\boldsymbol{X}_s\|_F^2 + \|\boldsymbol{X}_t - (\boldsymbol{P}_t{}^T \boldsymbol{P}_t)\boldsymbol{X}_t\|_F^2) + \lambda(\|\boldsymbol{T}\|_F^2 - \log\det \boldsymbol{T})$$
$$+ \mu_s \|\boldsymbol{Z}_s\|_0 + \mu_t \|\boldsymbol{Z}_t\|_0 \tag{7.2}$$

where, $\boldsymbol{X}_s$, $\boldsymbol{X}_t$ are the source and target domain data, $\boldsymbol{P}_s$, $\boldsymbol{P}_t$ are the associated projection matrices, $\boldsymbol{T}$ is the common transform and $\boldsymbol{Z}_s$, $\boldsymbol{Z}_t$ are the coefficients associated with the transformed source and target data. The first two terms in (7.2) ensure the transform $\boldsymbol{T}$ is learned such that it reduces the reconstruction error in the projected space for both domains. The third term preserves energy in the original signal and ensures less information loss while transforming the data to a common subspace. The other term prevents trivial solutions, controls the condition number of $\boldsymbol{T}$, and enforces sparsity on $\boldsymbol{Z}_s$ and $\boldsymbol{Z}_t$. To avoid degenerate solutions, we ensure that rows of the projection matrices, $\boldsymbol{P}_s$ and $\boldsymbol{P}_t$ are orthogonal and normalized to unit-norm. The method can be extended to handle multiple domains.

Later, the common transform $\boldsymbol{T}$ in (7.2) can be made class-discriminant, by expressing $\boldsymbol{T} = [\boldsymbol{T}_1 \;\; \boldsymbol{T}_2 \;\; .. \;\; \boldsymbol{T}_C]$, where C is the number of classes. Additional constraints can be added to the learning formulation in (7.2) to encourage data samples from the same class to be analyzed by the transform of the same class and penalize the analysis from transforms of other classes. During the test phase, the data sample can be assigned to class i, for which the reconstruction error of the features in the projected space, computed using the class transform $\boldsymbol{T}_i$ and its associated sparse coefficients $\boldsymbol{z}_i$, is minimum.

4. Ethical considerations and bias evaluation of domain adaptation and super-resolution techniques warrant thorough investigation, particularly for mission-critical applications such as healthcare and defense. While this thesis focuses on advancing multimodal image super-resolution using joint coupled transforms and dictionary learning, there are potential risks of bias and feature hallucination in sensitive domains. Although the proposed methods are designed to preserve modality-specific features and inter-modal correlations, which may help mitigate bias—they do not incorporate explicit control mechanisms to prevent the generation of artificial yet realistic content. It is important to emphasize that all methods were validated solely on publicly available benchmark datasets and are not intended for direct application in mission-critical scenarios without extensive downstream evaluation. As such, the broader societal and ethical implications lie beyond the scope of this work but are identified as promising directions for future research.

References

[1] J. Ni, Q. Qiu, and R. Chellappa, "Subspace interpolation via dictionary learning for unsupervised domain adaptation," in *IEEE Computer Vision and Pattern Recognition Conference (CVPR)*, 2013, pp. 692–699.

[2] P. Gupta and M. K. Pradhan, "Fault detection analysis in rolling element bearing: A review," *Materials Today: Proceedings*, vol. 4, no. 2, pp. 2085–2094, 2017.

[3] T. Zonta, C. A. da Costa, R. da R. Righi, M. J. de Lima, E. S. da Trindade, and G. P. Li, "Predictive maintenance in the industry 4.0: A systematic literature review," *Computers & Industrial Engineering*, vol. 150, p. 106889, 2020.

[4] F. Alves, H. Badikyan, H. António Moreira, J. Azevedo, P. M. Moreira, L. Romero, and P. Leitão, "Deployment of a smart and predictive maintenance system in an industrial case study," in *IEEE International Symposium on Industrial Electronics (ISIE)*, 2020, pp. 493–498.

[5] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning

and its applications to machine health monitoring," *Mechanical Systems and Signal Processing*, vol. 115, pp. 213–237, 2019.

[6] C. Chen, Z. H. Zhu, J. Shi, N. Lu, and B. Jiang, "Dynamic predictive maintenance scheduling using deep learning ensemble for system health prognostics," *IEEE Sensors Journal*, vol. 21, no. 23, pp. 26 878–26 891, 2021.

[7] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[8] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Transactions on Intelligent Systems and Technology*, vol. 11, no. 5, pp. 1–46, 2020.

[9] G. Csurka, *A Comprehensive Survey on Domain Adaptation for Visual Applications*. Cham: Springer International Publishing, 2017, pp. 1–35.

[10] H. Guan and M. Liu, "Domain adaptation for medical image analysis: a survey," *IEEE Transactions on Biomedical Engineering*, vol. 69, no. 3, pp. 1173–1185, 2021.

[11] X. Cao, B. Chen, and N. Zeng, "A deep domain adaption model with multi-task networks for planetary gearbox fault diagnosis," *Neurocomputing*, vol. 409, pp. 173–190, 2020.

[12] X. Li, W. Zhang, Q. Ding, and J.-Q. Sun, "Multi-layer domain adaptation method for rolling bearing fault diagnosis," *Signal Processing*, vol. 157, pp. 180–197, 2019.

[13] X. Li, W. Zhang, and Q. Ding, "A robust intelligent fault diagnosis method for rolling element bearings based on deep distance metric learning," *Neurocomputing*, vol. 310, pp. 77–95, 2018.

[14] X. Wang, H. He, and L. Li, "A hierarchical deep domain adaptation approach for fault diagnosis of power plant thermal system," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 9, pp. 5139–5148, 2019.

[15] H. S. Farahani, A. Fatehi, and M. A. Shoorehdeli, "On the application of domain adversarial neural network to fault detection and isolation in power plants," in *IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2020, pp. 1132–1138.

[16] X. Yu, Z. Zhao, X. Zhang, C. Sun, B. Gong, R. Yan, and X. Chen, "Conditional adversarial domain adaptation with discrimination embedding for locomotive fault diagnosis," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–12, 2021.

[17] M. Ragab, Z. Chen, M. Wu, H. Li, C. K. Kwoh, R. Yan, and X. Li, "Adversarial multiple-target domain adaptation for fault classification," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–11, 2020.

[18] Z. Zhao, Q. Zhang, X. Yu, C. Sun, S. Wang, R. Yan, and X. Chen, "Applications of unsupervised deep transfer learning to intelligent fault diagnosis: A survey and comparative study," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–28, 2021.

[19] J. Zhu, N. Chen, and C. Shen, "A new deep transfer learning method for

bearing fault diagnosis under different working conditions," *IEEE Sensors Journal*, vol. 20, no. 15, pp. 8394–8402, 2020.

[20] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," *Advances in Neural Information Processing Systems*, vol. 31, pp. 1647 – 1657, 2018.

[21] R. Gopalan, R. Li, and R. Chellappa, "Domain adaptation for object recognition: An unsupervised approach," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 999–1006.

[22] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *IEEE Computer Vision and Pattern Recognition Conference (CVPR)*, 2012, pp. 2066–2073.

[23] Q. Qiu, V. M. Patel, P. Turaga, and R. Chellappa, "Domain adaptive dictionary learning," in *Computer Vision – ECCV*, vol. 7575. Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012, pp. 631–645.

[24] S. Shekhar, V. M. Patel, H. V. Nguyen, and R. Chellappa, "Generalized domain-adaptive dictionaries," in *IEEE Computer Vision and Pattern Recognition Conference (CVPR)*, 2013, pp. 361–368.

[25] H. Xu, J. Zheng, A. Alavi, and R. Chellappa, "Cross-domain visual recognition via domain adaptive dictionary learning," *CoRR*, vol. abs/1804.04687, 2018.

[26] B. Yang, A. Ma, and P. Yuen, "Domain-shared group-sparse dictionary

learning for unsupervised domain adaptation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, pp. 7453–7460, 2018.

[27] H. Tang, H. Liu, W. Xiao, and N. Sebe, "When dictionary learning meets deep learning: Deep dictionary learning and coding network for image recognition with limited data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 5, pp. 2129–2141, 2021.

[28] R. Yan, F. Shen, C. Sun, and X. Chen, "Knowledge transfer for rotary machine fault diagnosis," *IEEE Sensors Journal*, vol. 20, no. 15, pp. 8374–8393, 2020.

[29] W. A. Smith and R. B. Randall, "Rolling element bearing diagnostics using the case western reserve university data: A benchmark study," *Mechanical Systems and Signal Processing*, vol. 64, pp. 100–131, 2015.

[30] C. Lessmeier, J. K. Kimotho, D. Zimmer, and W. Sextro, "Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification," in *PHM Society European Conference*, vol. 3, no. 1, 2016.

[31] A. Kumar and R. Kumar, "Vibration and acoustic data for defect cases of the cylindrical roller bearing (nbc: Nu205e)," IEEE DataPort, 2022.

[32] X. Chen, R. Yang, Y. Xue, M. Huang, R. Ferrero, and Z. Wang, "Deep transfer learning for bearing fault diagnosis: A systematic review since 2016," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–21, 2023.

[33] K. Feng, J. Ji, Y. Zhang, Q. Ni, Z. Liu, and M. Beer, "Digital twin-driven intelligent assessment of gear surface degradation," *Mechanical Systems and Signal Processing*, vol. 186, p. 109896, 2023.

[34] S. Tariyal, A. Majumdar, R. Singh, and M. Vatsa, "Deep dictionary learning," *IEEE Access*, vol. 4, pp. 10 096–10 109, 2016.

[35] A. Majumdar and R. Ward, "Robust greedy deep dictionary learning for ecg arrhythmia classification," in *International Joint Conference on Neural Networks (IJCNN)*, 2017, pp. 4400–4407.

[36] K. Engan, S. O. Aase, and J. H. Husoy, "Method of optimal directions for frame design," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 5, 1999, pp. 2443–2446.

[37] M. Aharon, M.Elad, and A. M. Bruckstein, "K-SVD and its non-negative variant for dictionary design," in *Wavelets XI*, vol. 5914, International Society for Optics and Photonics. SPIE, 2005, p. 591411.

[38] T. Zhang, "Sparse recovery with orthogonal matching pursuit under rip," *IEEE Transactions on Information Theory*, vol. 57, no. 9, pp. 6215–6221, 2011.

[39] E. Gur, S. Sabach, and S. Shtern, "Convergent nested alternating minimization algorithms for nonconvex optimization problems," *Mathematics of Operations Research*, vol. 48, no. 1, p. 53–77, 2023.

[40] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal

processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, vol. 49. Springer New York, 2011, pp. 185–212.

[41] R. Zhao, D. Wang, R. Yan, K. Mao, F. Shen, and J. Wang, "Machine health monitoring using local feature-based gated recurrent unit networks," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1539–1548, 2018.

[42] S. Ravishankar, B. Wen, and Y. Bresler, "Online sparsifying transform learning part i: Algorithms," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 625–636, 2015.

[43] S. Ravishankar and Y. Bresler, "Learning overcomplete sparsifying transforms for signal processing," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 3088–3092.

[44] S. Ravishankar and Y. Bresler, "Learning sparsifying transforms," *IEEE Transactions on Signal Processing*, vol. 61, no. 5, pp. 1072–1086, 2013.

[45] S. Ravishankar and Y. Bresler, "Online sparsifying transform learning part ii: Convergence analysis," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 637–646, 2015.

[46] S. Ravishankar and Y. Bresler, "Learning doubly sparse transforms for images," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 4598–4612, 2013.

[47] J. Maggu and A. Majumdar, "Kernel transform learning," *Pattern Recognition Letters*, vol. 98, pp. 117 – 122, 2017.

[48] Q. Fang, "A note on the condition number of a matrix," *Journal of Computational and Applied Mathematics*, vol. 157, no. 1, pp. 231–234, 2003.

[49] Maggu, J. and Majumdar, A., "Greedy deep transform learning," in *IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 1822–1826.

[50] Maggu, J. and Majumdar, A., "Unsupervised deep transform learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 6782–6786.

[51] G. Taylor, R. Burmeister, Z. Xu, B. Singh, A. Patel, and T. Goldstein, "Training neural networks without gradients: A scalable admm approach," in *International Conference on International Conference on Machine Learning (ICML)*, 2016, p. 2722–2731.

[52] A. Sharma and B. P. Shrivastava, "Different techniques of image sr using deep learning: A review," *IEEE Sensors Journal*, vol. 23, no. 3, pp. 1724–1733, 2023.

[53] G. A. Shaw and H. K. Burke, "Spectral imaging for remote sensing," *Lincoln Laboratory Journal*, vol. 14, no. 1, pp. 3–28, 2003.

[54] A. F. Goetz, "Three decades of hyperspectral remote sensing of the earth: A personal view," *Remote Sensing of Environment*, vol. 113, pp. S5–S16, 2009.

[55] T. Zhang, W. Wei, B. Zhao, R. Wang, M. Li, L. Yang, J. Wang, and Q. Sun, "A reliable methodology for determining seed viability by using

hyperspectral data from two sides of wheat seeds," *Sensors*, vol. 18, no. 3, p. 813, 2018.

[56] B. Zhang, D. Wu, L. Zhang, Q. Jiao, and Q. Li, "Application of hyperspectral remote sensing for environment monitoring in mining areas," *Environmental Earth Sciences*, vol. 65, no. 3, pp. 649–658, 2012.

[57] B. Park, K. C. Lawrence, W. R. Windham, D. P. Smith, and P. W. Feldner, "Hyperspectral imaging for food processing automation," in *Imaging Spectrometry VIII*, vol. 4816. The International Society for Optical Engineering, 2002, pp. 308–316.

[58] G. Lu and B. Fei, "Medical hyperspectral imaging: a review," in *Journal of Biomedical Optics*, vol. 19, no. 1. International Society for Optics and Photonics, 2014, p. 010901.

[59] D. B. Malkoff and W. R. Oliver, "Hyperspectral imaging applied to forensic medicine," in *Spectral Imaging: Instrumentation, Applications, and Analysis*, vol. 3920. International Society for Optics and Photonics, 2000, pp. 108–116.

[60] L. Gómez-Chova, D. Tuia, G. Moser, and G. Camps-Valls, "Multimodal classification of remote sensing images: A review and future directions," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1560–1584, 2015.

[61] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1521–1527, 2001.

[62] J. Sun, Z. Xu, and H. Y. Shum, "Image super-resolution using gradient profile prior," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[63] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *International Conference on Curves and Surfaces*. Springer, 2010, pp. 711–730.

[64] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 184–199.

[65] Y. Li, J. B. Huang, N. Ahuja, and M. H. Yang, "Joint image filtering with deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1909–1923, 2019.

[66] R. Dian, S. Li, A. Guo, and L. Fang, "Deep hyperspectral image sharpening," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 11, pp. 5345–5355, 2018.

[67] X. Wang, Q. Hu, Y. Cheng, and J. Ma, "Hyperspectral image super-resolution meets deep learning: A survey and perspective," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 8, pp. 1668–1691, 2023.

[68] P. Song, X. Deng, J. F. C. Mota, N. Deligiannis, P. L. Dragotti, and M. D. Rodrigues, "Multimodal image super-resolution via joint sparse representations induced by coupled dictionaries," *IEEE Transactions on Computational Imaging*, vol. 6, pp. 57–72, 2020.

[69] X. Deng and P. L. Dragotti, "Coupled ista network for multi-modal image super-resolution," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 1862–1866.

[70] K. He and X. Sun, J.and Tang, "Guided image filtering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, pp. 1397–1409, 06 2013.

[71] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele, "Joint bilateral upsampling," *ACM Transactions on Graphics (ToG)*, vol. 26, no. 3, pp. 96–102, 2007.

[72] X. Shen, Q. Yan, L. Xu, L. Ma, and J. Jia, "Multispectral joint image restoration via optimizing a scale map," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 12, pp. 2518–2530, 2015.

[73] Deng, X. and Dragotti, P. L., "Deep coupled ista network for multi-modal image super-resolution," *IEEE Transactions on Image Processing*, vol. 29, pp. 1683–1698, 2020.

[74] W. Wan, B. Zhang, M. Vella, J. F. Mota, and W. Chen, "Robust rgb-guided super-resolution of hyperspectral images via TV^3 minimization," *IEEE Signal Processing Letters*, vol. 29, pp. 957–961, 2022.

[75] M. Vella, B. Zhang, W. Chen, and J. F. C. Mota, "Enhanced hyperspectral image super-resolution via rgb fusion and tv-tv minimization," in *IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 3837–3841.

[76] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, p. 1–122, jan 2011.

[77] I. Daubechies, M. Defrise, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on Pure and Applied Mathematics*, vol. 57, no. 11, pp. 1413–1457, 2004.

[78] N. Gauraha, "Introduction to the lasso," *Resonance*, vol. 23, no. 4, pp. 439–464, 2018.

[79] A. Chakrabarti and T. Zickler, "Statistics of real-world hyperspectral images," in *IEEE Computer Vision and Pattern Recognition Conference (CVPR)*, 2011, pp. 193–200.

[80] M. Brown and S. Süsstrunk, "Multi-spectral sift for scene category recognition," in *IEEE Computer Vision and Pattern Recognition Conference (CVPR)*, 2011, pp. 177–184.

[81] C. Garcia-Cardona and B. Wohlberg, "Convolutional dictionary learning: A comparative review and new algorithms," *IEEE Transactions on Computational Imaging*, vol. 4, no. 3, pp. 366–381, 2018.

[82] F. Gao, X. Deng, M. Xu, J. Xu, and P. L. Dragotti, "Multi-modal convolutional dictionary learning," *IEEE Transactions on Image Processing*, vol. 31, pp. 1325–1339, 2022.

[83] F. G. Veshki and S. A. Vorobyov, "Efficient admm-based algorithms for convolutional sparse coding," *IEEE Signal Processing Letters*, vol. 29, pp. 389–393, 2022.

[84] D. R. Hunter, "Alternating minimization algorithms," *Wiley StatsRef: Statistics Reference Online*, pp. 1–10, 2014.

[85] B. Wohlberg, "Efficient algorithms for convolutional sparse representations," *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 301–315, 2016.

[86] W. W. Hager, "Updating the inverse of a matrix," *Society for Industrial and Applied Mathematics (SIAM) Review*, vol. 31, no. 2, pp. 221–239, 1989.

[87] Z. Wang, Z. Yan, and J. Yang, "Sgnet: Structure guided network via gradient-frequency awareness for depth map super-resolution," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 6, 2024, pp. 5823–5831.

[88] B. Wohlberg, "Sparse optimization research code (sporco)," https://purl.org/brendt/software/sporco, 2016.

[89] C. Guo, C. Li, J. Guo, R. Cong, H. Fu, and P. Han, "Hierarchical features driven residual learning for depth map super-resolution," *IEEE Transactions on Image Processing*, vol. 28, no. 5, pp. 2545–2557, 2019.

[90] Y. Cui, Q. Liao, W. Yang, and J. H. Xue, "Rgb guided depth map super-resolution with coupled u-net," in *2021 IEEE International Conference on Multimedia and Expo (ICME)*, 2021, pp. 1–6.

[91] B. Wang, Y. Zou, L. Zhang, Y. Li, Q. Chen, and C. Zuo, "Multimodal super-resolution reconstruction of infrared and visible images via deep learning," *Optics and Lasers in Engineering*, vol. 156, p. 107078, 2022.

[92] P. Gupta, J. Maggu, A. Majumdar, E. Chouzenoux, and G. Chierchia, "Confuse: Convolutional transform learning fusion framework for multi-channel data analysis," in *European Signal Processing Conference (EUSIPCO)*, 2021, pp. 1986–1990.

[93] P. Gupta, A. Majumdar, E. Chouzenoux, and G. Chierchia, "Superdeconfuse: A supervised deep convolutional transform based fusion framework for financial trading systems," *Expert Systems with Applications*, vol. 169, p. 114206, 2021.

[94] P. Gupta, J. Maggu, A. Majumdar, E. Chouzenoux, and G. Chierchia, "Deconfuse: a deep convolutional transform-based unsupervised fusion framework," *EURASIP Journal on Advances in Signal Processing*, vol. 26, 2020.

[95] J. Maggu, E. Chouzenoux, G. Chierchia, and A. Majumdar, "Convolutional transform learning," in *Neural Information Processing*. Springer International Publishing, 2018, pp. 162–174.

[96] J. Maggu, A. Majumdar, E. Chouzenoux, and G. Chierchia, "Deep convolutional transform learning," in *Neural Information Processing*. Cham: Springer International Publishing, 2020, pp. 300–307.

[97] H. Attouch, J. Bolte, and B. F. Svaiter, "Convergence of descent methods for

semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods," *Mathematical Programming*, 2011.

[98] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations*, 2015.

[99] R. D. Lutio, S. D'aronco, J. D. Wegner, and K. Schindler, "Guided super-resolution as pixel-to-pixel transformation," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 8828–8836.

Appendices

Appendix A

Appendix

A.1 Derivation for the closed-form updates for Chapter 2

Alternating Minimization (AM) approach [39] is employed to compute the updates for $\boldsymbol{\Delta D}_{m's}$ for the multiple dictionary layers. Here, we describe the detailed steps involved in computing the closed-form solution of one of the parameters: $\boldsymbol{\Delta D}_m^1$. The closed-form updates for other parameters can be obtained using a similar approach.

As discussed in Chapter 2, Section 2.1, the sub-problem for the update for $\boldsymbol{\Delta D}_m^1$ (referred to as (2.9)) is given as:

$$\boldsymbol{\Delta D}_m^1 \leftarrow \min_{\boldsymbol{\Delta D}_m^1} \|\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m\|_F^2 + \lambda \|\boldsymbol{\Delta D}_m^1\|_F^2 \tag{A.1}$$

Expanding the above problem (A.1) in terms of trace and equating the derivative with respect to $\boldsymbol{\Delta D}_m^1$ to 0, results in the following closed-form update for $\boldsymbol{\Delta D}_m^1$. The steps are detailed below.

Expanding (A.1) in terms of trace results in the following:

$$(\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m)^T (\boldsymbol{J}_m - \boldsymbol{\Delta D}_m^1 \boldsymbol{\Delta D}_m^2 \boldsymbol{\Delta D}_m^3 \boldsymbol{Z}_m) + \lambda((\boldsymbol{\Delta D}_m^1)^T \boldsymbol{\Delta D}_m^1) \tag{A.2}$$

$$= (\boldsymbol{J}_m)^T(\boldsymbol{J}_m) - (\boldsymbol{J}_m)^T(\boldsymbol{\Delta D}_m^1\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m) - (\boldsymbol{\Delta D}_m^1\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m)^T(\boldsymbol{J}_m)$$
$$+(\boldsymbol{\Delta D}_m^1\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m)^T(\boldsymbol{\Delta D}_m^1\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m) + \lambda((\boldsymbol{\Delta D}_m^1)^T\boldsymbol{\Delta D}_m^1) \tag{A.3}$$

Taking a derivative of (A.3) with respect to $\boldsymbol{\Delta D}_m^1$ results in:

$$-2\boldsymbol{J}_m(\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m)^T + 2(\boldsymbol{\Delta D}_m^1\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m)(\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m)^T$$
$$+2\lambda\boldsymbol{\Delta D}_m^1 \tag{A.4}$$

Equating the derivative in (A.4) to 0 and re-arranging the terms results in the following closed-form update for $\boldsymbol{\Delta D}_m^1$ (referred to as (2.10) in Chapter 2):

$$\boldsymbol{\Delta D}_m^1 = \boldsymbol{J}_m(\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m)^T[(\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m)(\boldsymbol{\Delta D}_m^2\boldsymbol{\Delta D}_m^3\boldsymbol{Z}_m)^T + \lambda\boldsymbol{I}]^{-1} \tag{A.5}$$