Structured sparse representations for supervised and unsupervised learning

Zeng, Yijie

2020

Zeng, Y. (2020). Structured sparse representations for supervised and unsupervised learning. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/143420>

<https://doi.org/10.32657/10356/143420>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 19:31:27 SGT

Structured Sparse Representations for Supervised and Unsupervised Learning

---

ZENG YIJIE

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2020

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

30 September 2019

.....

Date

曾艺杰

.....

ZENG YIJIE

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

30 September 2019

.....

Date

.....

Prof. Huang Guang-Bin

Authorship Attribution Statement

This thesis contains material from 3 paper(s) published in the following peer-reviewed journal(s) in which I am listed as an author.

Chapter 3 is published as Yijie Zeng, Yue Li, Jichao Chen, Xiaofan Jia, and Guang-Bin Huang, “ELM embedded discriminative dictionary learning for image classification,” *Neural Networks*, Mar. 2020. DOI: 10.1016/j.neunet.2019.11.015.

The contributions of the co-authors are as follows:

- Professor Huang Guang-Bin pointed out the research direction and supervised the work. He also reviewed the manuscript.
- I proposed the main algorithm and performed all the experiments at the School of Electrical and Electronic Engineering. I also analyzed the results.
- I prepared the manuscript draft. The draft was reviewed together with Mr. Li Yue, Mr. Chen Jichao and Ms. Jia Xiaofan.

Chapter 4 is published as Yijie Zeng, Jichao Chen, Yue Li, Yuanyuan Qing, and Guang-Bin Huang, “Clustering via Adaptive and Locality-constrained Graph Learning and Unsupervised ELM,” *Neurocomputing*, Mar. 2020. DOI: 10.1016/j.neucom.2020.03.045.

The contributions of the co-authors are as follows:

- Professor Huang Guang-Bin provided the initial research direction and edited the manuscript.
- I proposed the main algorithm and performed all the experiments at the School of Electrical and Electronic Engineering. I also analyzed the results.
- I prepared the manuscript draft. The draft was reviewed together with Mr. Chen Jichao, Mr. Li Yue and Ms. Qing Yuanyuan.

Chapter 5 is published as Yijie Zeng, Jichao Chen, and Guang-Bin Huang, “Slice-based Online Convolutional Dictionary Learning,” *IEEE Transactions on Cybernetics*, 2019. DOI: 10.1109/TCYB.2019.2931914.

The contributions of the co-authors are as follows:

- Professor Huang Guang-Bin provided the initial project direction and supervised the work. He also reviewed the manuscript.
- I proposed the main algorithm and performed all the experiments at the School of Electrical and Electronic Engineering. I also analyzed the results.
- I prepared the manuscript draft. The draft was reviewed together with Mr. Chen Jichao.

August 30, 2020

.....

Date

曾艺杰

.....

ZENG YIJIE

Acknowledgements

I would like to express my sincere gratitude to my supervisor Professor Huang Guang-Bin for supervising my research for the Ph.D. period. He has pointed out the research directions that I am now gratefully working on, and his encouragement and inspirations is valuable to me for completing this thesis. I would like to express my heartfelt appreciations to my parents for their understanding and support for my Ph.D.

I would like to thank Mr. Chen Jichao and Mr. Li Yue, with whom I have closely worked, for all the helpful discussions and insightful ideas. I would like to thank my former and present colleagues, Mr. Zhou Xiaosong, Dr. Zhou Xiangzeng, Mr. Zeng Guodong, Ms. Jia Xiaofan, Mr. Xu Rui, in STE-NTU Corp Lab, with whom I have spent happy and meaningful days working together for research ideas and projects. I would like to thank all the staffs in STE-NTU Corp Lab, especially the lab manager Mr. Alex Pek, for their help and support throughout these years. I would like to thank Dr. Wang Siqu, with whom I have collaborated closely and discussed enthusiastically for research directions. His passions to research have always motivated me to work hard at high standard.

Finally, I would like to thank Ms. Qing Yuanyuan for all the love and support and encouragement. Without her this thesis would not have been completed so smoothly.

Bet on sparsity.

— Trevor Hastie, Robert Tibshirani, Jerome Friedman, *The Elements of Statistical Learning, 2nd Edition.*

Summary

Many problems in machine learning (ML) and computer vision (CV) deal with large amounts of data with variations and noise for underlying tasks. For example, object detection requires filtering semantic objects from noisy and varying backgrounds, and the ImageNet challenge requires to build a classification model on top of over one million images. Finding good representations of data can not only improve the performance of ML systems, but also indicate proper choices of algorithms for different problems. On the other hand, representing the data in an effective and efficient way is challenging, partly due to the intrinsic complexity of the high dimensional data distributions. Natural images are known to lie on a low dimensional manifold, but it is extremely difficult to construct a generative model that properly describes its structure from the high dimensional pixel space. Video data is even more complex given the temporal variations among frames. Therefore, building a scalable data representation has become a critical part of the ML processing pipeline, where there has been a huge amount of work addressing on this topic in the past.

This thesis focuses on a specific data representation technique named sparse coding (SC) which has been found effective for natural signals such as images. The concept is to decompose signals into linear combinations of only a few elements from a large collection of predefined vectors. The goal of this thesis is to explore inherent sparse structures in machine learning problems, and design effective algorithms that utilize the derived sparse representations for solving supervised and unsupervised learning tasks.

The first part of the thesis investigates the topic of dictionary learning. The idea of dictionary learning is to construct (or learn) a set of “atomic vectors” that are over-complete on the data space, so that each signal can be efficiently represented and reconstructed via only a small subset of them. The dictionary determines the sparse pattern which in turn reveals the intrinsic structure of the signal space. In this thesis, a discriminative dictionary learning approach is proposed for image

classification which explicitly enforces the coding vectors to possess similar sparsity structure among the same class. This is achieved via a novel weighted class specific group sparsity regularization imposed on each class of the samples. The group sparsity property removes unwanted features from the same group simultaneously and keeps only important ones for the whole group, thus the regularization is able to improve uniformity of the sparse patterns of signals belonging to the same class and suppress “support overlaps” of different classes. Combined with feature representations via Extreme Learning Machine (ELM) on the input space and a maximum margin criterion (MMC) regularization on the output space, a unified supervised feature learning framework named ELM embedded Discriminative Dictionary Learning (ELM-DDL) is presented in this thesis which is effective in sparse feature representations. This thesis shows that ELM-DDL can reveal group structures robust to different conditions and separate samples favorably from different groups, achieving state-of-the-art performance on benchmark image datasets.

The second part of the thesis explores the structural information in unsupervised graph learning. Constructing an informative graph is beneficial to many unsupervised tasks, including clustering analysis and manifold learning. When the graph is sparse, the connectivities can be determined via nonzero entries of the graph representation, which indicate the local structures of data. In this thesis, sparsity of data graph is achieved and exploited via a simple constraint on locality of each samples, because locality naturally leads to sparsity. The locality-constrained coding method focuses on nearest neighbors and gives rise to a distance regularized feature representation combined with local decomposition. When a rank constraint is further imposed on the Laplacian of the data graph to enforce the graph’s connected components to match a predefined class number, the learned graph exhibits clear cluster structure and is easy to partition. It is demonstrated that the proposed graph learning method, termed Adaptive Locality-constrained Clustering (ALC), generates more structured graph compared with predefined ones and provides better clustering performance on benchmark datasets. This thesis also explores the use of unsupervised ELM (US-ELM) on top of the learned graph to learn a flexible and discriminative data embedding that is beneficial for clustering.

The third part of the thesis focuses on the topic of convolutional dictionary learning (CDL). CDL is a special variant of the ordinary dictionary learning technique, where the obtained dictionary is applied on sparse codes via convolution instead

of traditional matrix multiplication. The technique is particularly important to structural inputs such as images (natural or hyper-spectral), since the convolution operator is shift-invariant and does not break the spatial configurations. This part of the thesis is particularly dedicated to exploring online methods for efficient CDL algorithms where input signals are provided in a streaming fashion to update the dictionary, because the batch mode CDL is time and memory-consuming in nature. The methodology adopted in this part is a systematic extension of classical unsupervised online sparse coding algorithms to the convolutional sparse coding domain, with efficient optimizations achieved via a localized “slice-based” representation of sparse features. Specifically, a slice-based online convolutional dictionary learning (SOCDL) method is proposed for unsupervised image processing tasks such as image reconstruction and inpainting. The expected reconstruction cost with respect to the dictionary is approximated via a surrogate function to enable online learning, and under the slice-based representation a second-order stochastic approximation approach is proposed for optimization. The approximation efficiently generates a dictionary sequence which converges favorably with desired characteristics for image reconstruction. Convergence analysis is presented with theoretical justifications, and the computational complexity is among the lowest within the context of online CDL algorithms with least memory consumptions. Finally, this thesis proposes a novel masked version of SOCDL which can perform online learning on incomplete data for image inpainting.

Contents

Acknowledgements	v
Summary	vii
List of Figures	xiii
List of Tables	xv
Symbols and Acronyms	xvii
1 Introduction	1
1.1 Research Background	2
1.2 Objectives and Contributions	4
1.3 Thesis Organization	7
2 Literature Review	8
2.1 Notations	9
2.2 Sparse Dictionary Learning	10
2.2.1 Sparse Coding and Dictionary Learning	10
2.2.2 Discriminative Dictionary Learning	13
2.2.3 Online Learning for Dictionary Learning	16
2.2.4 Convolutional Sparse Coding and Convolutional Dictionary Learning	20
2.2.5 Connections to Deep Learning	22
2.3 Alternating Direction Method of Multipliers	26
2.3.1 ADMM Algorithm	26
2.3.2 Convergence	29
2.3.3 ADMM for Sparse Coding	30
2.4 Extreme Learning Machine	33
2.4.1 Regularized Extreme Learning Machine for Regression and Classification	36
2.4.2 Extreme Learning Machine based Autoencoder	36
2.4.3 Extreme Learning Machine for Unsupervised Learning	38

3	ELM Embedded Discriminative Dictionary Learning for Image Classification	40
3.1	Background and Motivations	41
3.1.1	Contributions	43
3.2	Proposed Method	43
3.2.1	ELM Mapping as Nonlinear Transformations	44
3.2.2	Discriminative Projection	45
3.2.3	Discriminative Sparse Regularization	45
3.2.4	Proposed Formulation	46
3.2.5	Classification Scheme	48
3.3	Optimizations of ELM-DDL	49
3.3.1	Update of β	49
3.3.2	Update of $\mathbf{A}$	51
3.3.3	Update of $\mathbf{D}$	53
3.3.4	Computational Complexity	53
3.4	Experiments	55
3.4.1	Performance and Comparisons	55
3.4.1.1	The Extended Yale B Dataset	57
3.4.1.2	The AR Face Dataset	59
3.4.1.3	The COIL-100 Dataset	59
3.4.1.4	The Fashion-MNIST Dataset	61
3.4.2	Visualizations of Representations	62
3.4.3	The Effect of ELM Embedding	64
3.4.3.1	Dictionary Learning without ELM	64
3.4.3.2	Dictionary Learning without MMC	65
3.4.3.3	The Effect of Number of Hidden Neurons	65
3.4.4	The Effect of λ_2 and λ_3	66
3.4.5	The Effect of SRC Predictor	66
3.4.6	Comparisons with Deep Learning based Approaches	67
3.5	Summary	69
4	Clustering via Adaptive and Locality-constrained Graph Learning and Unsupervised ELM	71
4.1	Background and Motivations	72
4.1.1	Contributions	74
4.2	Proposed Method	75
4.2.1	Graph Learning Formulation	75
4.2.2	Clustering with Learned Graph	79
4.3	Optimizations of ELM-ALC	79
4.4	Experiments	84
4.4.1	Datasets	85
4.4.2	Experimental Settings	85
4.4.3	Clustering Performance	91

4.4.4	Comparisons between ELM based Graph Learning Methods	94
4.4.5	Validations of Components	95
4.4.6	Sensitivity Analysis	98
4.5	Summary	102
5	Slice-based Online Convolutional Dictionary Learning	103
5.1	Background and Motivations	104
5.1.1	Contributions	106
5.2	Proposed Method	106
5.2.1	Slice-base Representations	107
5.2.2	Algorithm Outline	108
5.2.3	Dictionary Update	109
5.2.4	Comparison with Patch-based Online Dictionary Learning	114
5.2.5	Complexity Analysis and Comparison with OCDL algorithms	114
5.2.5.1	Spatial Surrogate Algorithms	115
5.2.5.2	Frequency Surrogate Algorithms	117
5.2.6	Convergence and Acceleration	118
5.2.7	Handling Boundaries and Learning from Incomplete Data	123
5.3	Experiments	123
5.3.1	The Effect of Slice-based Dictionary Update	125
5.3.2	Image Reconstruction on Small Datasets	127
5.3.3	Image Reconstruction on Large Dataset	127
5.3.4	Validation of Parameter Selection	132
5.3.5	Image Inpainting	136
5.4	Summary	138
6	Conclusions and Future Works	142
6.1	Conclusions	143
6.2	Future Works	145
	List of Publications	148
	Bibliography	150

List of Figures

3.1	Illustrations of the benchmark datasets: Extended Yale B, AR Face, COIL-100 and Fashion-MNIST.	54
3.2	Convergence of objective values of ELM-DDL on each benchmark dataset for 60 iterations.	56
3.3	Example faces from the Extended Yale B dataset. The first row corresponds to test images from the same person with different lighting conditions. The second row shows sparse representations of each image with a small $\lambda_4 = 0.005$. The dictionary is learned with $\lambda_2 = \lambda_3 = 0.01$. The last row is computed with a dictionary with $\lambda_2 = 0.01$ and $\lambda_3 = 0$	60
3.4	t-SNE visualizations of first ten persons of the Extended Yale B dataset. Data points of the same color belong to the same class. (a) corresponds to the original data points, (b) corresponds to ELM outputs $\beta\mathbf{H}$ from (3.9) with learned β , and (c) is the sparse coding vectors from (3.10) with learned $\mathbf{D}$	63
3.5	Accuracy of ELM-DDL on Extended Yale B with different number of hidden neurons (L).	66
3.6	Average accuracies of ELM-DDL for different choices of λ_2 and λ_3 for each dataset. The x and y axis are in log scale.	70
4.1	Illustrations of 5-NN graph on UMIST dataset: (a) created with self-tuning strategy [242]; (b) learned by proposed method.	77
4.2	Accuracy values with respect to different output dimensions on benchmark datasets.	93
4.3	Accuracies of variants of ALC on benchmark datasets.	96
4.4	Comparisons of different graph types for manifold regularizations.	97
4.5	Accuracies of ALC on each image dataset (UMIST/ORL/POLLEN/USP-ST/MNIST05/ETH80) with different values of η and γ . The number of nearest neighbors is chosen to give the best performance.	99
4.6	Accuracies of ALC on each UCI dataset (ECHOLI/CARS/DIABETES/SEGMENT/VEHICLE/CONTROL) with different values of η and γ . The number of nearest neighbors is chosen to give the best performance.	100
4.7	Accuracies of ALC with different values of nearest neighbors k . The values of η and γ are chosen to give the best performance.	101

5.1	Stripe dictionary Ω with $n = 6$ and $m = 3$	107
5.2	Comparison between SOCDL and OCDL for different dictionary size n and number of dictionary atoms m	126
5.3	Test set objective with respect to time step on <i>Fruit</i> and <i>City</i>	128
5.4	Visualizations of dictionaries learned on Fruit (Figure (a) to (e)) and City (Figure (f) to (j)).	129
5.5	Test set objective and PSNR with respect to time step on <i>CIFAR10</i> and <i>STL10</i>	130
5.6	Performance of SOCDL on CIFAR10 under different parameters: $p = 0, 1, 5, 10, 20$ for grayscale and color images (Figure 5.6a and 5.6b), $L = 5, 10, 20, 50, 100$ for grayscale and color images (Figure 5.6c and 5.6d).	131
5.7	Test set objective of different algorithms learned from masked data.	137
5.8	Dictionary learned from masked VOC07 dataset images by different online CSC algorithms.	137
5.9	Inpainting illustrations on <i>Parrots</i> using masked SOCDL. The dictionary is learn on each corrupted image with a size of $n = 8 \times 8$ and $m = 64$ atoms. λ is set to 0.05.	139
5.10	Inpainting illustrations on <i>Sails</i> using masked SOCDL. The dictionary is learn on each corrupted image with a size of $n = 8 \times 8$ and $m = 64$ atoms. λ is set to 0.05.	140

List of Tables

3.1	Specifications of benchmark datasets for image classification.	55
3.2	Recognition results (%) on four benchmark datasets: the Extended Yale B, AR Face, COIL-100 and Fashion-MNIST. All results are measured with the best hyperparameter settings. Final accuracy is averaged over 10 runs.	58
3.3	Test accuracies (%) of ELM-DDL under different experimental settings. “ <i>Baseline</i> ” corresponds to the best result in Table 3.2. “ <i>w.o. ELM</i> ” refers to the case where no ELM embedding is used. “ <i>w.o. MMC</i> ” means training the model with $\lambda_4 = 0$. “ <i>Lin. Pred.</i> ” is achieved by substitution of a linear predictor for SRC during testing. All results are averaged over 10 experiments.	64
3.4	Comparisons of the proposed ELM-DDL framework with deep learning based approaches: WTA-CAE and VGG-Small.	68
4.1	Dataset Specificities	84
4.2	Accuracy results (in %) of baseline methods (K-Means/SC) and graph learning methods (CAN/PCAN/CLR_L1/CLR_L2). For POLLEN and ECOLI, CAN cannot find the correct number of clusters, so their accuracy values are not reported. All results are reported as <i>mean $\pm$ standard deviation</i> over 10 runs.	86
4.3	Accuracy results (in %) of subspace clustering methods (SSC/SSC-OMP/LRR), ELM based graph learning method (ELM-CLR) and the proposed ALC/ELM-ALC. For each dataset, the highest mean accuracy among all clustering methods in Table 4.2 and this table is highlighted in bold. All results are reported as <i>mean $\pm$ standard deviation</i> over 10 runs.	87
4.4	NMI results (in %) of baseline methods (K-Means/SC) and graph learning methods (CAN/PCAN/CLR_L1/CLR_L2). For each dataset, the highest mean NMI among all clustering methods in Table 4.5 and this table is highlighted in bold. All results are reported as <i>mean $\pm$ standard deviation</i> over 10 runs.	88

4.5	NMI results (in %) of subspace clustering methods (SSC/SSC-OMP/LRR), ELM based graph learning method (ELM-CLR) and the proposed ALC/ELM-ALC. For each dataset, the highest mean NMI among all clustering methods in Table 4.4 and this table is highlighted in bold. All results are reported as <i>mean $\pm$ standard deviation</i> over 10 runs.	89
4.6	Configurations of ALC variants. The check mark ($\checkmark$) indicates that the algorithm possesses the corresponding capability.	94
5.1	An overview of parameters for SOCDL.	115
5.2	Comparison for time and space complexity of proposed algorithm with existing online CDL algorithms. N is the signal size, n is the size for each dictionary atom, m is the number of dictionary atoms, K is the number of samples for each batch, and J is the iteration number.	116
5.3	Datasets Used for Experiments	124
5.4	PSNR (in dB) of test images restored by dictionaries trained on VOC2007 via SGD-based OCDL and the proposed SOCDL.	133
5.5	PSNR (in dB) of test images restored by dictionaries trained on each corrupted image.	134

Symbols and Acronyms

Symbols

$\mathcal{N}_i$	the index set of the neighbors of agent i
$\mathbf{D}, \mathbf{D}_t$	dictionary in matrix form
$\mathbf{d}$	dictionary in vector form
$\odot$	element-wise multiplication between two vectors

Acronyms

ML	Machine Learning
CV	Computer Vision
MP	Matching Pursuit
OMP	Orthogonal Matching Pursuit
BPDN	Basis Pursuit Denoising
CBPDN	Convolutional Basis Pursuit Denoising
LASSO	Least Absolute Shrinkage and Selection Operator
MOD	Method of Optimal Directions
SC	Sparse Coding
ELM	Extreme Learning Machine
US-ELM	Unsupervised Extreme Learning Machine
CSC	Convolutional Sparse Coding
CDL	Convolutional Dictionary Learning
OCDL	Online Convolutional Dictionary Learning
ADMM	Alternating Direction Method of Multipliers
FISTA	Fast Iterative Shrinkage-Thresholding Algorithm
BCD	Block Coordinate Descent

SGD	Stochastic Gradient Descent
ELM-DDL	ELM embedded Discriminative Dictionary Learning
ALC	Adaptive Locality-constrained Clustering
ELM-ALC	Adaptive Locality-constrained Clustering with ELM
SOCDL	Slice-based Online Convolutional Dictionary Learning
SOCDL-Mask	Masked Slice-based Online Convolutional Dictionary Learning
i.i.d.	independent and identically distributed
<i>a.s.</i>	almost sure convergence of a random sequence
w.r.t.	with respect to
<i>s.t.</i>	subject to

Chapter 1

Introduction

This chapter gives a brief introduction to the topics that this thesis is most interested in. Section [1.1](#) investigate sparse coding with respect to its background and previous research. Section [1.2](#) describes the objective of this thesis, i.e. to explore sparse structures for solving supervised and unsupervised ML problems, based on which the main contributions are presented. Section [1.3](#) summarizes the organization of this thesis.

1.1 Research Background

Data representation plays an important role in signal processing and statistical learning. In either case, various amounts of prior knowledge are required to build a representation model, or moreover, to learn the underlying data distributions. The amount of prior knowledge one can adopt varies based on the problem definition. In the case when little is known, a natural choice is to seek for *sparsity* of linear codings on top of the pre-defined *dictionary* as a general prior.

The quest of sparsity for linear decomposition of input signals has appeared in many fields. It has been found in [4], [5] that sparse coding can be used to model the receptive fields of simple cells in the mammalian primary visual cortex. For statistical learning, finding a sparse solution to linear regression leads to the *LASSO* problem [6] which improves subset selection and model interpretability. In signal processing, a sparse representation of the input signal is central to the topic of *Compressive Sensing* [7], [8]. Specifically, the signal can be effectively recovered from the representation and the given measurement matrix as long as it is sufficiently sparse and the dictionary satisfies some “incoherence” constraints.

To enable effective sparse coding for various tasks, it is important to carefully select a suitable dictionary. This has attracted a series of research, such as the wavelets series [9] for various image processing tasks, and in a broader sense, the sparse representation classifier [10] for face recognition. On the other hand, the shift to *learning* a dictionary that adapts to given models and signals from *predefining* one has been given more attention, because the learned dictionary is more task-specific, and often yield a sparser representation. The method of dictionary learning has provided state-of-the-art results in low-level signal and image processing tasks, such as image denoising [11], restoration [12], [13], super resolution [14], as well as high level computer vision tasks [15]–[18].

Technically, the objective of dictionary learning is to minimize a reconstruction objective with respect to the sparse codes and the dictionary jointly, where the sparse codes are representations of each input signal, and the dictionary is shared among data. It is also possible to enforce additional regularizations apart from reconstructions to encode knowledge of underlying tasks in the objective. These regularizations are meant to be “discriminative” in the sense that they are designed for the specific task rather than general reconstructions and should improve

task-related performance. This is particularly important for computer vision applications [19], where the coding vectors are responsible for a definitive high-level task (e.g. classification) other than reconstruction, and additional information (e.g. labels of images) is usually available and can be utilized. Research on task-related regularizations can be found in [18], [20]–[25], to name a few.

A particularly interesting topic for sparse representation is to model the intrinsic structures of problems and data via sparsity-inducing regularizations. These regularizations aim at encouraging a specific sparse pattern which are shared by similar data signals. Typical formulations include group sparsity for ℓ_1 norm [26], [27], ℓ_0 norm [28], ℓ_∞ norm [29], etc.

The dictionary learning model established in [4], [5] is based on learning on natural image patches. Convolutional sparse coding (CSC) extends the sparse representation to full images by learning “slices” at all spatial locations of an image. The slices are decomposed by a shared dictionary and all slices are aggregated to reconstruct the input image. CSC can be also regarded as a structured sparse representation, since it considers the spatial correlations among overlapping patches within the same image, and the spatial structures are retained in the resulting feature maps, in a similar way to the hierarchical structure of convolutional neural networks [30]. Recently, research on CSC has focused on solving the sparse coding problem [31]–[33], learning the dictionary efficiently [34]–[36], as well as on different applications [37]–[40].

Structured sparse representations often refer to extending the sparsity-inducing regularizations on individual variables (e.g. ℓ_1) to block-sparsity-inducing norm on grouped variables (e.g. $\ell_{1,2}$) to encode structural correlations among different features [41]. To this end, the obtained model not only focuses on sparsity, but also incorporates prior information that may reveal more intricate structural information about the data. A simple and effective extension of the LASSO problem is the group LASSO [26] which models sparsities within each group in a linear model. The group structured $\ell_{2,1}$ norm is used to build an efficient supervised or unsupervised feature selection [42], [43] model. Moreover, hierarchical sparse coding and dictionary learning models can be formulated [44], [45].

1.2 Objectives and Contributions

The previous section briefly reviews the research background of sparse representations and dictionary learning in general. This section will focus on specific topics that attract research attentions. Addressing these topics leads to the objectives of this thesis, and this section describes major contributions to achieve these objectives.

Sparse coding for computer vision and pattern recognition has been giving impressive performance on image classification, partly because image patches are near low dimensional manifolds, and a semantically rich representations should recover a small sets of atoms (from a properly learned dictionary) as the support features, leading naturally to sparse representations [19]. This indicates that learning a discriminative dictionary is central for sparse coding based image classification.

Improving discriminations of dictionary learning can be addressed in different aspects. First and most importantly, task-related constraints can be directly imposed on the dictionary to encode semantic information. This includes learning class-specific dictionaries [46], [47], low-rank ones [48], or ones that possess both properties [22], [49], [50]. Second, discriminative regularizations can be applied on the sparse coding vectors which are directly responsible for classification. The sparse codes can be regularized via class discrimination [20], [51], label consistency [18], graph regularization [21], [23], support discrimination [25], [52], [53], etc. Last, the sparse codes can be learned on transformed feature vectors instead of raw image pixels. For example, Yang et al. [15] solves sparse coding on SIFT descriptors, and [16] is based on whitened image patches. Following this line of research, the first objective of the thesis is to extend previous work and to design effective and efficient class-aware regularizations to improve performance of discriminative dictionary learning for image classification.

Sparse coding can be regarded as a unsupervised feature learning method [16]. When no class discrimination is involved, the coding vectors are computed purely on image patches (or transformed features), and then a classifier (e.g. SVM [54], ELM [55], etc.) is learned on top of them. This property motivates the question of whether sparse coding is applicable *directly* to unsupervised learning tasks.

Learning an informative graph can effectively address many problems in unsupervised learning, e.g. clustering analysis and manifold learning. Traditionally graph learning for clustering involves constructing the data graph via k-nearest-neighbors or ϵ -ball, and spectral approaches are applied to partition data into clusters [56]. One of the most important properties of an efficient graph representation is that the connectivity should be sparse so that it characterizes locality relations and enables efficient storage [19]. *Sparse subspace clustering* exploits the *self-expressiveness* property of data and expresses each sample as a sparse representation of other samples [57], leading to the construction of an ℓ_1 graph. Without learning a dictionary, this self-expressive approach can yield a sparse and effective graph for subspace clustering. If the representations given by self-expressiveness are regarded as *feature descriptors*, then the aforementioned task-specific regularizations on sparse representations are applicable to the unsupervised graph learning in a similar way as those for supervised tasks. Therefore, the second objective of this thesis is to explore the possibility of extending the unsupervised feature learning methods to graph-based clustering with exploitation of data self-expressiveness, and incorporate priors that capture the structured information of data distributions.

Convolutional sparse coding (CSC) is an important topic for image analysis utilizing spatial structures. Similar to patch-based sparse coding, it is critical for CSC to learn a shift-invariant dictionary, leading to the research on convolutional dictionary learning (CDL) [36]. For unsupervised learning or image processing, CDL is notorious for large memory consumptions, since it goes through spatial locations of *all* images to update the dictionary. This inspires the research on *online convolutional dictionary learning (OCDL)*, and several works are available [58]–[61] as an extension of online learning methods for patch-based dictionary learning [62]. Most of them rely on a *surrogate objective* which approximates the true reconstruction cost and is easy to optimize w.r.t. the convolutional dictionary. However, some of them still have high complexity with large memory footprints, and not all of them can be generalized to learn on masked images. The last objective of the thesis, therefore, is to provide a novel OCDL method which has lower time complexity theoretically and empirically, and explore the possibility of online learning on masked or corrupted images.

In view of the above discussion, the contributions of this thesis are summarized as follows:

1. To address the problem of discriminative dictionary learning, this thesis proposes *ELM embedded Discriminative Dictionary Learning (ELM-DDL)* which is applied on the task of image classification. Specifically, a novel weighted class-specific group-sparsity-inducing norm is proposed to regularize the sparse coding vectors. The regularization promotes uniformity of the sparse patterns of samples belong to the same class and suppresses support overlaps of different classes. The input signals are transformed to a nonlinear feature space via Extreme Learning Machine (ELM), whose output embeddings are regularized via a maximum margin criterion (MMC) to learn a output projection matrix. The embeddings are simple to learn and discriminative with small within-class scatters and large between-class variance. The whole algorithm is optimized via ADMM, and empirical results show that the proposed approach achieve state-of-the-art results compared with other dictionary learning methods.
2. An adaptive graph learning method called *Adaptive and Locality-constrained Clustering with ELM (ELM-ALC)* is proposed in this thesis to address the problem of graph learning for data clustering. Specifically, this thesis proposes to adopt *Locality-constrained Linear Coding (LLC)* [63] as an effective feature representation to learn the data graph. The LLC descriptor is naturally sparse, leading to a formulation similar to subspace clustering combined with self-expressiveness of data samples, but it is further regularized via local distances. The graph constructed by the LLC descriptors is updated adaptively with rank constraints on the Laplacian of the graph to effectively exploit the cluster structures of data. The algorithm incorporates manifold regularizations over the learned descriptors to improve smoothness of representations from similar samples. The final learned graph is combined with the unsupervised ELM (US-ELM) for cluster computation. Extensive evaluations are conducted on benchmark datasets and show superior results of the proposed algorithms compared with graph learning counterparts.
3. This thesis proposes a *Slice-based Online Convolutional Dictionary Learning (SOCDL)* algorithm to address the problem of online CDL for the task of image reconstruction and image inpainting. The algorithm adopts the “slice-based” CSC representations and handle the sparse codes locally with structural information. A surrogate function is constructed to approximate the

expected objective, and given the slice-based representations, an online update scheme is proposed which is compact and efficient. Theoretical analysis is given to show the convergence behavior of the proposed algorithm, which shows that it has lower time complexity than existing counterpart methods on image reconstructions. An extension of SOCDL is given to enable online learning on masked images, and evaluations are conducted on the image inpainting task.

1.3 Thesis Organization

The thesis is organized as follows:

- Chapter 1 investigates the background of the sparse coding approaches, and presents the objective and contributions of the thesis.
- Chapter 2 presents an extensive review of literature of the following topics: sparse dictionary learning, alternating direction method of multipliers, and Extreme Learning Machine (ELM).
- Chapter 3 introduces the ELM embedded discriminative dictionary learning method referred to as ELM-DDL.
- Chapter 4 introduces the adaptive and locality-constrained graph learning method with a combination of US-ELM for clustering, referred to as ELM-ALC.
- Chapter 5 introduces a online method for convolutional dictionary learning based on local slice representations referred to as SOCDL.
- Chapter 6 concludes the thesis with discussions on future research directions.

Chapter 2

Literature Review

This chapter gives an extensive literature review on related topics to the thesis. The notations used throughout this thesis are defined in Section 2.1. After that Section 2.2 investigates the topic of sparse dictionary dictionary which is the main focus of this thesis. Section 2.3 presents the alternating direction method of multipliers (ADMM) as a common and effective approach for convex optimization, which is extensively used by subsequent works in the thesis. Section 2.4 discusses Extreme Learning Machine (ELM) for supervised and unsupervised learning.

2.1 Notations

The following notation conventions are used throughout this thesis unless otherwise specified. Vectors are denoted by lower case bold letters, e.g. $\mathbf{x}$, and matrices are denoted by upper case bold ones, e.g. $\mathbf{X}$. Given a vector $\mathbf{x} \in \mathbb{R}^n$, the j -th entry of $\mathbf{x}$ is denoted by $\mathbf{x}_j$ or $\mathbf{x}[j]$. Also given a matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$, the i -th row is denoted by $\mathbf{X}^i$, the j -th column is $\mathbf{X}_j$, and the element at the (i, j) entry is given by $\mathbf{X}_{ij}$ ¹. Given an index set $\Lambda \subseteq \{1, \dots, n\}$, the vector $\mathbf{x}_\Lambda \in \mathbb{R}^{|\Lambda|}$ denotes the entries of $\mathbf{x}$ whose indices belong to Λ . Similarly, $\mathbf{X}_\Lambda \in \mathbb{R}^{m \times |\Lambda|}$ denotes a sub-matrix of $\mathbf{X}$ whose column indices belong to Λ . The vector norm ℓ_p with $p \geq 1$ is defined in the usual way:

$$\|\mathbf{x}\|_p \triangleq \left(\sum_{i=1}^n |\mathbf{x}_i|^p \right)^{1/p}. \quad (2.1)$$

The ℓ_0 (pseudo-)norm is defined as

$$\|\mathbf{x}\|_0 \triangleq |\{j \mid \mathbf{x}_j \neq 0\}|. \quad (2.2)$$

For matrices, the Frobenius norm is given by

$$\|\mathbf{X}\|_F \triangleq \left(\sum_{i=1}^m \sum_{j=1}^n |\mathbf{X}_{ij}|^2 \right)^{1/2}. \quad (2.3)$$

Define the $\ell_{1,2}$ and $\ell_{2,1}$ norm of a matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ as follows²:

$$\|\mathbf{X}\|_{1,2} \triangleq \sum_{i=1}^m \|\mathbf{X}^i\|_2 = \sum_{i=1}^m \left(\sum_{j=1}^n |\mathbf{X}_{ij}|^2 \right)^{1/2} \quad (2.4)$$

$$\|\mathbf{X}\|_{2,1} \triangleq \sum_{j=1}^n \|\mathbf{X}_j\|_2 = \sum_{j=1}^n \left(\sum_{i=1}^m |\mathbf{X}_{ij}|^2 \right)^{1/2}. \quad (2.5)$$

The indicator function $\iota(\cdot)$ is defined on a set $\mathcal{C}$

$$\iota_{\mathcal{C}}(\mathbf{x}) = \begin{cases} 0 & \mathbf{x} \in \mathcal{C} \\ \infty & \mathbf{x} \notin \mathcal{C} \end{cases}. \quad (2.6)$$

¹The (i, j) entry of $\mathbf{X}$ is sometimes denoted as $\mathbf{x}_{ij}$ when no confusion is made.

²Note that the definitions of $\ell_{1,2}$ and $\ell_{2,1}$ here do not conform to $\ell_{p,q}$ for arbitrary $p, q \geq 1$ in general, and is only applicable within the context of this thesis.

2.2 Sparse Dictionary Learning

2.2.1 Sparse Coding and Dictionary Learning

An underdetermined linear system is represented as $\mathbf{s} = \mathbf{D}\mathbf{x}$, where $\mathbf{D} \in \mathbb{R}^{n \times m}$ and $n < m$. $\mathbf{s} \in \mathbb{R}^n$ is the observed signal and the objective is to look for a suitable $\mathbf{x} \in \mathbb{R}^m$ that can properly reconstruct the observation. If assuming that $\mathbf{D}$ is full rank, then the system has infinitely many solutions $\mathbf{x}$. For general problem with little prior information, one can enforce sparse constraints on $\mathbf{x}$ to yield a unique or a proper solution. This leads to the following regularized problem:

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{s} - \mathbf{D}\mathbf{x}\|_2^2 + \lambda \psi(\mathbf{x}). \quad (2.7)$$

Here $\psi(\cdot)$ is some sparsity-inducing function and λ is a trade-off parameter. The significance of seeking a sparse solution to underdetermined least square problems has been elaborated in Section 1.1. To encourage the solution to be sparse, one can choose the sparse regularization $\psi(\cdot)$ to be the ℓ_0 norm in (2.2), which counts the nonzero elements in $\mathbf{x}$:

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{s} - \mathbf{D}\mathbf{x}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{x}\|_0 \leq T \quad (2.8)$$

or

$$\min \|\mathbf{x}\|_0 \quad \text{s.t.} \quad \|\mathbf{s} - \mathbf{D}\mathbf{x}\|_2 \leq \epsilon \quad (2.9)$$

where T is a *hard bound* on number of nonzero elements that $\mathbf{x}$ maximally allows to have, and ϵ is a predefined error tolerance. It can be proved that solving the ℓ_0 regularized least square problem is actually NP-hard [64], and one can only solve the problem approximately. This inspires a series of greedy methods that approximate the minimizer in a stepwise fashion, including the Matching Pursuit (MP) [65], Orthogonal Matching Pursuit (OMP) [66], iterative hard-thresholding [67], etc.

Another option is to seek for the convex relaxation of (2.2) [68], leading to the following three formulations of ℓ_1 regularized problem:

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{s} - \mathbf{D}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad (2.10)$$

or

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{s} - \mathbf{D}\mathbf{x}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{x}\|_1 \leq \eta \quad (2.11)$$

or

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \quad \text{s.t.} \quad \|\mathbf{s} - \mathbf{D}\mathbf{x}\|_2 \leq \epsilon. \quad (2.12)$$

The problem (2.10) is known as Lasso [6] in statistical learning and basis pursuit [69] in signal processing. Compared with (2.8), problem (2.10) is more flexible and scalable since it is convex. A wide range of solvers for the ℓ_1 regularized problem is available, to name a few: the coordinate descent method [70], the iterative soft-thresholding method [71], [72], the LARS algorithm [73], ADMM [74], etc.

The coding schemes (2.8) or (2.10) are often referred to as **sparse coding** or **sparse representations**. This thesis uses both terms indiscriminately, and in most case refers to the ℓ_1 regularized problem (2.10) since it is easier to solve.

As mentioned in Section 1.1, learning a suitable dictionary $\mathbf{D}$ that adapts to the signals $\{\mathbf{s}\}$ is important for exploiting structural information of the data distribution, especially for image processing tasks [11]. Concretely, **dictionary learning** refers to the following sparse modeling problem:

$$\min_{\mathbf{D} \in \mathcal{C}, \{\mathbf{x}_i\}_{i=1}^N} \sum_{i=1}^N \frac{1}{2} \|\mathbf{s}_i - \mathbf{D}\mathbf{x}_i\|_2^2 + \lambda \psi(\mathbf{x}_i). \quad (2.13)$$

Here ψ is a sparsity-inducing regularization, and $\mathcal{C}$ is a constraint set to prevent the norm of $\mathbf{x}_i$ or $\mathbf{D}$ from being unbounded:

$$\mathcal{C} = \{\mathbf{D} = [\mathbf{d}_1, \dots, \mathbf{d}_n] \in \mathbb{R}^{n \times m} : \|\mathbf{d}_j\|_2 \leq 1, \forall j\}. \quad (2.14)$$

The constraint set $\mathcal{C}$ can also be designed to incorporate prior information about the underlying data distributions. In many cases, the learned dictionary $\mathbf{D} = [\mathbf{d}_1, \dots, \mathbf{d}_m] \in \mathbb{R}^{n \times m}$ is often chosen to be *over-complete*, i.e. the number of atoms m is more than the dimensionality of each atom n . The choice of ψ reflects prior knowledge for the underlying problem and the dataset $\{\mathbf{s}_i\}_{i=1}^N$. To promote sparsity (as the main focus of this thesis, otherwise Tikhonov or total variations regularizations are also available), the function $\psi(\mathbf{x}) = \sum_j \log(\epsilon + \mathbf{x}[j]^2)$ is used in [4],

and more popular choices include the ℓ_1 norm as in (2.10) or the Elastic-Net formulation [75]. In addition, feature interactions of $\mathbf{x}_i$ can be considered via block sparsity-inducing regularization, e.g. grouped [26] or tree-structured [45] regularizations.

Given $\mathbf{S} = [\mathbf{s}_1, \dots, \mathbf{s}_N]$ and $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]$, a similar formulation of dictionary learning can be given as:

$$\min_{\mathbf{D} \in \mathcal{C}, \mathbf{X}} \frac{1}{2} \|\mathbf{S} - \mathbf{D}\mathbf{X}\|_F^2 + \lambda \Psi(\mathbf{X}). \quad (2.15)$$

If the regularization function Ψ is defined as

$$\Psi(\mathbf{X}) \triangleq \sum_{i=1}^N \psi(\mathbf{x}_i), \quad (2.16)$$

then (2.15) is equivalent to (2.13). On the other hand, the formulation (2.15) opens up connections with the topic of *matrix factorization* [76] as identified by [62]. In addition, the regularization Ψ can be defined more flexibly than the additive model (2.16). For example, one can enforce different regularizations on each class of data to increase discriminations of the learned dictionary [25], [47]. This will be investigated in more details in Section 2.2.2.

To solve (2.13) or (2.15), typical approaches involve alternately updating the sparse coding vectors $\mathbf{X}$ and the dictionary $\mathbf{D}$. At each iteration when the dictionary is fixed, it amounts to solving the sparse coding problem (2.7) if Ψ is separable, where dedicated solvers have been introduced above. When Ψ is not separable, specific solvers can be also proposed to solve the structural sparse regularization problem. When the sparse coding vectors are fixed, (2.13) or (2.15) becomes a least square problem with regularization defined by $\mathcal{C}$, which can be minimized via the method of optimal directions (MOD) [77], [78], K-SVD [79], block coordinate descent [62] (see also [80, Algorithm 11]), etc.

At each step of alternating optimization, problem (2.13) or (2.15) requires solving all the sparse coding vectors $\{\mathbf{x}_i\}_{i=1}^N$ to update the dictionary $\mathbf{D}$. It is also possible to perform *online* update, i.e. the signals $\{\mathbf{s}_i\}_{i=1}^N$ are utilized in a streaming fashion to update $\mathbf{D}$. Typical methods include stochastic gradient descent [5] and stochastic approximations [62]. They are described in more details in Section 2.2.3.

2.2.2 Discriminative Dictionary Learning

Dictionary learning is an important method for image classification. Technically, the objective of dictionary learning is to minimize a reconstruction objective with respect to the sparse codes and the dictionary jointly, where the sparse codes are representations of each input image, and the dictionary is shared among the image dataset. It is also possible to enforce additional regularizations apart from reconstructions to encode knowledge of underlying tasks in the objective. These regularizations are meant to be “*discriminative*” in the sense that they are designed for the specific task rather than general reconstructions and should improve task-related performance. This is particularly important for computer vision applications, where the coding vectors are responsible for a definitive high-level task (e.g. classification) other than reconstruction, and additional information (e.g. labels of images) is usually available and can be utilized.

To annotate the problem (2.15) with label information, one can solve the following problem:

$$\min_{\mathbf{D} \in \mathcal{C}, \mathbf{X}} \frac{1}{2} \|\mathbf{S} - \mathbf{DX}\|_F^2 + \Psi_s(\mathbf{X}) + \Psi_d(\mathbf{X}) \quad (2.17)$$

where Ψ_s is the sparsity-inducing term and Ψ_d is the discrimination term. Some popular choices of Ψ_d are discussed here:

- **Discriminative K-SVD (D-KSVD)** [20]: The D-KSVD approach proposes to learn a linear predictor simultaneously when performing dictionary learning:

$$\min_{\mathbf{D} \in \mathcal{C}, \mathbf{X}} \|\mathbf{S} - \mathbf{DX}\|_F^2 + \alpha \|\mathbf{WX} - \mathbf{Y}\|_F^2 \quad \text{s.t.} \quad \|\mathbf{X}\|_0 \leq T. \quad (2.18)$$

Here $\mathbf{Y}$ is the collection of labels for all samples $\mathbf{S}$, and $\mathbf{W}$ is the parameter matrix to learn. Solving (2.18) enforces the sparse codes from the same class to be mapped to the same ground truth label, and in turn improve the learned dictionary.

- **Label Consistent K-SVD (LC-KSVD)** [18]: LC-KSVD builds on top of D-KSVD with an additional consistency term. Assume that each atom $\mathbf{d}_i$

belongs to one of C classes, then LC-KSVD solves

$$\min_{\mathbf{D} \in \mathcal{C}, \mathbf{X}} \|\mathbf{S} - \mathbf{D}\mathbf{X}\|_F^2 + \alpha \|\mathbf{W}\mathbf{X} - \mathbf{Y}\|_F^2 + \beta \|\mathbf{H}\mathbf{X} - \mathbf{Q}\|_F^2 \quad \text{s.t.} \quad \|\mathbf{X}\|_0 \leq T. \quad (2.19)$$

Here $\mathbf{H}$ is the parameter matrix to learn, and $\mathbf{Q}_{ij}$ is one if and only if $\mathbf{d}_i$ and $\mathbf{s}_j$ belongs to the same class. In this way, besides the linear predictor in D-KSVD, LC-KSVD promotes consistency of the sparse coding vectors via a regression term to the consistency target $\mathbf{Q}$.

- **Fisher Discrimination Dictionary Learning (FDDL)** [47]: The FDDL approach consists of two discrimination promoting terms. Consider a *structured* dictionary $\mathbf{D} = [\mathbf{D}_1, \dots, \mathbf{D}_C]$ similar to that of LC-KSVD where each atom belongs to one of C classes, and a partition of samples $\mathbf{S} = [\mathbf{S}_1, \dots, \mathbf{S}_C]$ where $\mathbf{S}_c$ is the samples from the c -th class, FDDL aims to solve:

$$\begin{aligned} \min_{\mathbf{D} \in \mathcal{C}, \{\mathbf{X}_c\}_{c=1}^C} \sum_{c=1}^C \left(\|\mathbf{S}_c - \mathbf{D}\mathbf{X}_c\|_F^2 + \|\mathbf{S}_c - \mathbf{D}_c\mathbf{X}_c^{(c)}\|_F^2 + \sum_{j \neq c} \|\mathbf{D}_j\mathbf{X}_c^{(j)}\|_F^2 \right) \\ + \lambda_1 \|\mathbf{X}\|_1 + \lambda_2 \left(\text{tr}[\mathbf{S}_w(\mathbf{X}) - \mathbf{S}_b(\mathbf{X})] + \eta \|\mathbf{X}\|_F^2 \right). \end{aligned} \quad (2.20)$$

For class c , the sparse coding vectors $\mathbf{X}_c$ can be further decomposed as $\mathbf{X}_c = [\mathbf{X}_c^{(1)}, \dots, \mathbf{X}_c^{(C)}]^T$ where each block $\mathbf{X}_c^{(j)}$ corresponds to the atoms $\mathbf{D}_j$. In this case for a well-trained dictionary $\mathbf{D}$, it should satisfy $\mathbf{S}_c \approx \mathbf{D}\mathbf{X}_c = \sum_j \mathbf{D}_j\mathbf{X}_c^{(j)}$. The minimization of $\|\mathbf{S}_c - \mathbf{D}_c\mathbf{X}_c^{(c)}\|_F^2 + \sum_{j \neq c} \|\mathbf{D}_j\mathbf{X}_c^{(j)}\|_F^2$ is referred to as *discriminative fidelity term*, which enforces that the samples $\mathbf{S}_c$ should be well reconstructed by atoms from the c -th class, i.e. $\mathbf{D}_c\mathbf{X}_c^{(c)}$, and poorly reconstructed by $\mathbf{D}_j\mathbf{X}_c^{(j)}$. The second component $\text{tr}[\mathbf{S}_w(\mathbf{X}) - \mathbf{S}_b(\mathbf{X})] + \eta \|\mathbf{X}\|_F^2$ is the *discriminative coefficient term*, where $\mathbf{S}_w(\mathbf{X})$ and $\mathbf{S}_b(\mathbf{X})$ is the *within-class scatter* and *between-class scatter* respectively:

$$\mathbf{S}_w(\mathbf{X}) = \sum_{c=1}^C \sum_{\mathbf{x}_k \in \mathbf{X}_c} (\mathbf{x}_k - \bar{\mathbf{x}}_c)(\mathbf{x}_k - \bar{\mathbf{x}}_c)^T \quad (2.21)$$

$$\mathbf{S}_b(\mathbf{X}) = \sum_{c=1}^C N_i(\bar{\mathbf{x}}_c - \bar{\mathbf{x}})(\bar{\mathbf{x}}_c - \bar{\mathbf{x}})^T. \quad (2.22)$$

$\bar{\mathbf{x}}_c$ and $\bar{\mathbf{x}}$ are the mean vectors of $\mathbf{X}_c$ and $\mathbf{X}$ respectively. Therefore, the discriminative coefficient term suppresses intra-class variance and promotes inter-class variance to increase class discrimination.

- **Support Discrimination Dictionary Learning (SDDL)** [25]: SDDL models interactions among features within the same class and between different classes:

$$\min_{\mathbf{D} \in \mathcal{C}, \mathbf{X}} \|\mathbf{S} - \mathbf{DX}\|_F^2 + w_1 \sum_{c=1}^C \|\mathbf{X}_c\|_{1,2} + w_2 \sum_{c=1}^C \sum_{\mathbf{x}_p \in \mathbf{X}_c, \mathbf{x}_q \notin \mathbf{X}_c} \|\mathbf{x}_p \odot \mathbf{x}_q\|_0. \quad (2.23)$$

Note that unlike LC-KSVD and FDDL, SDDL does not assume a structured dictionary with class specific atoms. Instead, it imposes structural constraints on the obtained sparse coding vectors. The first term $\|\mathbf{X}_c\|_{1,2}$ enforces that features from the same class share *similar supports*, i.e. vectors from the same class should have the same pattern of nonzero elements. The minimization of the second term $\|\mathbf{x}_p \odot \mathbf{x}_q\|_0$ indicates that features from different classes (i.e. $\mathbf{x}_p$ and $\mathbf{x}_q$) should have *minimal overlapping supports*. The two terms jointly learn a disentangled support which in turn produces a highly discriminative dictionary.

The formulation (2.17) imposes discriminative regularizations via Ψ_d on the sparse coding vectors $\mathbf{X}$, which in turn improve the dictionary during alternating update step. On the other hand, it is also possible to regularize the dictionary directly and consider the sub-dictionary setting $\mathbf{D} = [\mathbf{D}_1, \dots, \mathbf{D}_C]$ used in LC-KSVD and FDDL:

- **Dictionary Learning with Structured Incoherence (DLSI)** [46]: DLSI enforces that sub-dictionaries $\mathbf{D}_i$ from different classes should be different. This is achieved via an *incoherence* term:

$$\min_{\mathbf{D} \in \mathcal{C}, \mathbf{X}} \|\mathbf{S} - \mathbf{DX}\|_F^2 + \lambda \|\mathbf{X}\|_1 + \eta \sum_{i \neq j} \|\mathbf{D}_i^T \mathbf{D}_j\|_F^2. \quad (2.24)$$

The last term suppresses the pair-wise inner product between atoms from different classes, which means that sub-dictionaries from different classes should be as distinct as possible, thus the resulting sparse coding vectors tend to produce different activations for different classes.

- **Discriminative Dictionary Learning with Low-Rank Regularization ($\mathbf{D}^2\mathbf{L}^2\mathbf{R}^2$)** [22]: $\mathbf{D}^2\mathbf{L}^2\mathbf{R}^2$ annotates the objective of SDDL with low-rank

constraint on the sub-dictionaries. Denote the objective of FDDL in (2.20) as $\mathcal{L}_{\text{FDDL}}(\mathbf{D}, \mathbf{X}; \mathbf{S})$, then $\mathbf{D}^2\text{L}^2\text{R}^2$ minimizes:

$$\min_{\mathbf{D} \in \mathcal{C}, \mathbf{X}} \mathcal{L}_{\text{FDDL}}(\mathbf{D}, \mathbf{X}; \mathbf{S}) + \alpha \sum_{c=1}^C \|\mathbf{D}_c\|_*. \quad (2.25)$$

Since the dictionary atoms are initialized from the training samples which are often correlated, the ones from the same class should be low-rank. The low-rank constraint can also help exploit the unique structured information for each class c .

2.2.3 Online Learning for Dictionary Learning

In this part the online learning approaches for dictionary learning are investigated which are based on [17], [62]. The quest for online learning on the dictionary learning domain comes from two aspects:

1. In practice, the number of signals N in (2.13) is usually large especially for applications on natural images. For examples, when the signals $\mathbf{s}_i$ represent image patches, the dictionary $\mathbf{D}$ is learned on $N = 4 \times 10^5$ patches [16]. Online learning provides an efficient technique for solving or well approximating (2.13).
2. As point out in [81] and discussed in [17], instead of minimizing the *empirical cost* (2.13) on a fixed dataset $\{\mathbf{s}_i\}_{i=1}^N$, one is more interested in minimizing the *expected cost* on a (probably infinite) signal space $\mathcal{S}$ given the objective $\mathcal{L}(\mathbf{s}, \mathbf{D})$:

$$\min_{\mathbf{D} \in \mathcal{C}} F(\mathbf{D}) \triangleq \mathbb{E}_{\mathbf{s} \sim \mathcal{S}} [\mathcal{L}(\mathbf{s}, \mathbf{D})]. \quad (2.26)$$

To minimize (2.26) constrained on $\mathcal{C}$ (as defined in (2.14)), the first-order projected stochastic gradient descent (SGD) can be applied to update $\mathbf{D}$:

$$\mathbf{D}^{(\tau)} = \Pi_{\mathcal{C}} [\mathbf{D}^{(\tau-1)} - \eta^{(\tau)} \nabla_{\mathbf{D}} \mathcal{L}(\mathbf{s}^{(\tau)}, \mathbf{D}^{(\tau-1)})], \quad (2.27)$$

where $\mathbf{D}^{(\tau)}$ is the dictionary obtained at time step τ , $\eta^{(\tau)}$ is the learning rate, $\Pi_{\mathcal{C}}$ is the orthogonal projection onto the constraint set $\mathcal{C}$, and $\mathbf{s}^{(\tau)}$ is the i.i.d. sample

---

Algorithm 1: Online Dictionary Learning [62].

Data: Initialize $\mathbf{D}^{(0)}$ randomly, $\mathbf{A}^{(0)} = 0$, $\mathbf{B}^{(0)} = 0$;

1 for $\tau = 1, \dots, t$ **do**
2   Sample $\mathbf{s}^{(\tau)}$ from $\mathcal{S}$;

3   Solve sparse coding problem:

$$\mathbf{x}^{(\tau)} \triangleq \operatorname{argmin} \frac{1}{2} \|\mathbf{s}^{(\tau)} - \mathbf{D}^{(\tau-1)} \mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1$$

which can be efficient optimized via LARS [73];

4 $\mathbf{A}^{(\tau)} = \mathbf{A}^{(\tau-1)} + \mathbf{x}^{(\tau)} \mathbf{x}^{(\tau)\top}$;

5 $\mathbf{B}^{(\tau)} = \mathbf{B}^{(\tau-1)} + \mathbf{s}^{(\tau)} \mathbf{x}^{(\tau)\top}$;

6   Compute $\mathbf{D}^{(\tau)}$ using block coordinate descent with $\mathbf{D}^{(\tau-1)}$ as warm start:

$$\begin{aligned} \mathbf{D}^{(\tau)} &= \operatorname{argmin}_{\mathbf{D} \in \mathcal{C}} \mathcal{F}^{(\tau)}(\mathbf{D}) \\ &= \operatorname{argmin}_{\mathbf{D} \in \mathcal{C}} \frac{1}{\tau} \left[\frac{1}{2} \operatorname{tr}[\mathbf{D}^\top \mathbf{D} \mathbf{A}^{(\tau)}] - \operatorname{tr}[\mathbf{D}^\top \mathbf{B}^{(\tau)}] \right]. \end{aligned} \quad (2.28)$$

7 return $\mathbf{D}^{(t)}$

---

from $\mathcal{S}$. The SGD based optimization approach is effective in terms of convergence speed [61], [82], [83] when the learning rate η is carefully chosen.

Online Dictionary Learning

If the target is to reconstruct the signals in the signal space $\mathcal{S}$, the functional $\mathcal{L}$ can be simply chosen as

$$\mathcal{L}(\mathbf{s}, \mathbf{D}) = \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{s} - \mathbf{D} \mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1. \quad (2.29)$$

In this case, Mairal et al. [62] proposed online learning based on stochastic approximations, specifically utilizing the problem structures for effectively approximating the expected cost (2.26). Define the following function as the *surrogate* for F :

$$\mathcal{F}^{(t)}(\mathbf{D}) \triangleq \frac{1}{t} \sum_{\tau=1}^t \left[\frac{1}{2} \|\mathbf{s}^{(\tau)} - \mathbf{D} \mathbf{x}^{(\tau)}\|_2^2 + \lambda \|\mathbf{x}^{(\tau)}\|_1 \right], \quad (2.30)$$

where $\mathbf{x}^{(\tau)}$ is the sparse coding vectors computed at previous steps. The algorithm is outlined in Algorithm 1. The observations from [62] are two-fold:

1. The minimization of the surrogate function (2.30) on the constrained set $\mathcal{C}$ can be performed efficiently via block coordinate descent, which is parameter free. Moreover, the optimization does not require to store all $\mathbf{s}_i$ and $\mathbf{x}_i$, but only $\mathbf{A}^{(t)} = \sum_i \mathbf{x}_i \mathbf{x}_i^T$ and $\mathbf{B}^{(t)} = \sum_i \mathbf{s}_i \mathbf{x}_i^T$, which is memory efficient.
2. When assumptions regarding the regularities of $\mathcal{S}$ and $\mathbf{D}$ are satisfied (see [62, Section 4.1]), the $\{\mathbf{D}^{(\tau)}\}_{\tau=1}^t$ obtained by minimizing the surrogate function $\mathcal{F}^{(t)}$ converges to stationary points of (2.26), which is summarized in the following theorem:

Theorem 2.1 ([62]). *Suppose assumptions (A)-(C) in [62, Section 4.1] holds, and $\{\mathbf{D}^{(\tau)}\}_{\tau=1}^t$ denotes the dictionaries obtained from algorithm 1, then when t goes to infinity, $F(\mathbf{D}^{(t)})$ converges a.s. and the distance between $\mathbf{D}^{(t)}$ and the set of stationary points of F converges a.s. to 0.* $\square$

Task-Driven Dictionary Learning

For image processing applications [11], [12], the main focus is to reconstruct the image patches $\mathbf{s}$ with a suitable dictionary $\mathbf{D}$. For others problems, especially visual recognition [83], one may wish to apply the obtained sparse coding vectors for high level vision tasks, and they care less about the reconstruction performance. Furthermore, the label $\mathbf{y}$ from a label set $\mathcal{Y}$ corresponding to each signal $\mathbf{s} \in \mathcal{S}$, e.g. the category of each image, is usually available. In this case, the functional $\mathcal{L}$ in (2.26) is not limited to the form (2.29), but to define some *supervised loss* $\mathcal{L}_s$ (e.g. SVM [54], ELM [55]) parameterized by some model parameters $\mathbf{W}$.

Mairal et al. [17] proposed the task-driven dictionary learning to deal with the supervised case. Specifically, the expected cost (2.26) is extended to an expected risk minimization problem defined jointly on $\mathbf{W}$ and $\mathbf{D}$:

$$\min_{\mathbf{D} \in \mathcal{C}, \mathbf{W}} F_s(\mathbf{D}, \mathbf{W}) \triangleq \mathbb{E}_{\mathbf{y} \sim \mathcal{Y}, \mathbf{s} \sim \mathcal{S}} \left[\mathcal{L}_s(\mathbf{y}, \mathbf{W}, \mathbf{x}^*(\mathbf{s}, \mathbf{D})) \right] + \frac{\nu}{2} \|\mathbf{W}\|_F^2, \quad (2.31)$$

where $\mathbf{x}^*$ is defined as the minimization of an *elastic-net* formulation [75]:

$$\mathbf{x}^*(\mathbf{s}, \mathbf{D}) \triangleq \underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{s} - \mathbf{D}\mathbf{x}\|_2^2 + \lambda_1 \|\mathbf{x}\|_1 + \frac{\lambda_2}{2} \|\mathbf{x}\|_2^2. \quad (2.32)$$

It is natural to regard $\mathbf{x}^*$ as a function of $\mathbf{s}$ *parameterized* by $\mathbf{D}$. By minimizing (2.31), the dictionary $\mathbf{D}$ is adapted to the underlying task, thus it exhibits more discriminative information than the one optimized for reconstruction.

The dictionary $\mathbf{D}$ can be optimized via projected SGD in (2.27) as long as F_s is differentiable w.r.t. $\mathbf{D}$. This is indeed the case given mild assumptions, which is summarized in the following theorem:

Theorem 2.2 ([17]). *Assume that $\lambda_2 > 0$ and the following assumptions hold:*

1. *The probability density $\mathbf{p}(\mathbf{y}, \mathbf{s})$ is defined on a compact support $K_{\mathcal{Y}} \times K_{\mathcal{S}} \subseteq \mathcal{Y} \times \mathcal{S}$.*
2. *If $\mathcal{Y}$ is a subset of finite-dimensional real vector space, then $\mathbf{p}$ is continuous and $\mathcal{L}_s$ is twice continuously differentiable; If $\mathcal{Y}$ is a final set, then $\mathbf{p}(\mathbf{y}, \cdot)$ is continuous and $\mathcal{L}_s(\mathbf{y}, \cdot)$ is twice continuously differentiable for any $\mathbf{y} \in \mathcal{Y}$.*

Then F_s is differentiable w.r.t. $\mathbf{D}$ and $\mathbf{W}$ respectively. Specifically,

$$\nabla_{\mathbf{W}} F_s(\mathbf{D}, \mathbf{W}) = \mathbb{E}[\nabla_{\mathbf{W}} \mathcal{L}_s(\mathbf{y}, \mathbf{W}, \mathbf{x}^*)] \quad (2.33)$$

$$\nabla_{\mathbf{D}} F_s(\mathbf{D}, \mathbf{W}) = \mathbb{E}[-\mathbf{D}\boldsymbol{\beta}^* \mathbf{x}^{*\top} + (\mathbf{s} - \mathbf{D}\mathbf{x}^*)\boldsymbol{\beta}^{*\top}] \quad (2.34)$$

where $\boldsymbol{\beta}^*$ satisfies

$$\boldsymbol{\beta}_{\Lambda^c}^* = 0 \quad \text{and} \quad \boldsymbol{\beta}_{\Lambda}^* = (\mathbf{D}_{\Lambda}^{\top} \mathbf{D}_{\Lambda} + \lambda_2 \mathbf{I})^{-1} \nabla_{\mathbf{x}_{\Lambda}^*} \mathcal{L}_s(\mathbf{y}, \mathbf{W}, \mathbf{x}^*), \quad (2.35)$$

and

$$\Lambda = \{j \mid \mathbf{x}^*[j] \neq 0\} \quad (2.36)$$

is the nonzero entries of $\mathbf{x}^*(\mathbf{s}, \mathbf{D})$. □

2.2.4 Convolutional Sparse Coding and Convolutional Dictionary Learning

Section 2.2.1 has discussed the background corresponding to traditional sparse coding and dictionary learning approaches. The discussion is based on the underdetermined linear system $\mathbf{s} = \mathbf{D}\mathbf{x}$, where $\mathbf{D} = [\mathbf{d}_1, \dots, \mathbf{d}_m] \in \mathbb{R}^{n \times m}$ with $n < m$. For naturally signals, e.g. images, the system is more suitably modeled as $\mathbf{s} = \sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_i$ with observed $\mathbf{s}$ as the whole image instead of image patches, where $*$ denotes the convolution operator. The representation is to utilize the *translation-invariance* property [84] and to build a *deconvolutional network* [85], [86] for visual recognition.

Convolutional Sparse Representations

Given the dictionary atoms $\{\mathbf{d}_i\}_{i=1}^m$, the **convolutional sparse coding** (CSC) problem is the convolutional counterpart of the ordinary sparse coding problem in (2.7). Specifically, the ℓ_0 regularized problem is [87]

$$\min_{\{\mathbf{x}_i\}_{i=1}^m} \frac{1}{2} \|\mathbf{s} - \sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_i\|_2^2 \quad \text{s.t.} \quad \|\mathbf{X}\|_0 \leq T. \quad (2.37)$$

The ℓ_1 regularized problem (also referred to as convolutional basis pursuit denoising (CBPDN) [32]) is

$$\min_{\{\mathbf{x}_i\}_{i=1}^m} \frac{1}{2} \|\mathbf{s} - \sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_i\|_2^2 + \lambda \sum_{i=1}^m \|\mathbf{x}_i\|_1. \quad (2.38)$$

Recently, efficient optimization approaches have been proposed for solving the computational expensive problem (2.38) [31]–[33], [88], [89] which are based on ADMM [90]. CSC has shown superior performance on various image processing tasks, including super resolution [38], image decomposition [91], image fusion [40], image inpainting [33], [92], etc. It is also popular for vision tasks [85], [93], [94].

Convolutional Dictionary Learning

The problem of **Convolutional Dictionary Learning** (CDL) is to solve dictionary learning problem regarding the convolutional dictionary atoms $\{\mathbf{d}_i\}_{i=1}^m$ on the dataset $\{\mathbf{s}_k\}_{k=1}^K$:

$$\min_{\{\mathbf{x}_{i,k}\}, \{\mathbf{d}_i\}} \sum_{k=1}^K \left[\frac{1}{2} \left\| \mathbf{s}_k - \sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_{i,k} \right\|_2^2 + \lambda \sum_{i=1}^m \|\mathbf{x}_{i,k}\|_1 \right] \quad \text{s.t.} \quad \mathbf{d}_i \in \mathcal{C}. \quad (2.39)$$

The optimization framework follows the alternating update methodology for dictionary learning problem. On the other hand, unlike the convolutional sparse coding problem (2.38) which can be solved very efficient on frequency domain via FFT, solving the dictionary update problem requires special care. Efficient dictionary update methods include [34], [35], [92], [95] which are all based on ADMM.

A detailed summary and comparison of different convolutional dictionary learning solvers is given in [36]. Efficient solvers for various CSC/CDL problems based on ADMM and FISTA [72] include the SPORCO library [96].

Masked Convolutional Dictionary Learning

The need for considering a mask for convolutional dictionary learning stems either from the hope of dealing missing or incomplete data [33], [35], or from the concern of boundary artifacts caused by the circular boundary conditions of convolution operation [89]. In either case, one can simulate a spatial (known) mask represented as a vector $\mathbf{W}_k \in \mathbb{R}^N$ for each signal $\mathbf{s}_k$:

$$\min_{\{\mathbf{x}_{i,k}\}, \{\mathbf{d}_i\}} \sum_{k=1}^K \left[\frac{1}{2} \left\| \mathbf{W}_k \odot (\mathbf{s}_k - \sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_{i,k}) \right\|_2^2 + \lambda \sum_{i=1}^m \|\mathbf{x}_{i,k}\|_1 \right] \quad \text{s.t.} \quad \mathbf{d}_i \in \mathcal{C}. \quad (2.40)$$

Optimizations for the corresponding sparse coding problem include mask decoupling [33], [97] and additive mask simulation [89]. The dictionary update approaches associated with (2.40) are discussed in [36].

Online Convolutional Dictionary Learning

Similar to the discussions in Section 2.2.3, it is possible to generalize convolutional dictionary learning to the online scenario. This is necessary because CDL is particularly time and memory consuming compared to traditional dictionary learning (2.13). Generalizing the online approach proposed in [62] amounts to defining the CDL functional:

$$\mathcal{L}_{\text{CDL}}(\mathbf{x}, \mathbf{D}; \mathbf{s}) \triangleq \frac{1}{2} \left\| \sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_i - \mathbf{s} \right\|_2^2 + \lambda \sum_{i=1}^m \|\mathbf{x}_i\|_1, \quad (2.41)$$

and update the dictionary $\mathbf{D}$ either via first order projected SGD approach (2.27) or via minimizing the surrogate function by second order stochastic approximations:

$$\mathcal{F}^{(t)}(\mathbf{D}) \triangleq \frac{1}{t} \sum_{\tau=1}^t \left[\frac{1}{2} \left\| \sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_i^{(\tau)} - \mathbf{s}^{(\tau)} \right\|_2^2 + \lambda \sum_{i=1}^m \|\mathbf{x}_i^{(\tau)}\|_1 \right]. \quad (2.42)$$

The first order methods for online CDL are discussed in [61], and second order methods are presented in [58]–[61]. Moreover, Chapter 5 of this thesis will present an efficient online approach for CDL with the least time and memory complexity among the aforementioned approaches.

Additionally, Zeng et al. [98] generalize the approach in [17] for end-to-end optimizations of the CDL counterpart of the expected risk minimization (2.31) for supervised learning.

2.2.5 Connections to Deep Learning

Sparse dictionary learning has been well studied for more than twenty years. In recent years, the topic of deep learning [99] has become a heated one and has attracted unparalleled attention from both academia and industry. Although this thesis focuses on theories and applications of sparse coding and dictionary learning, it is worthwhile to briefly review deep methods for supervised and unsupervised learning, and in particular those that relates closely to the classical sparse coding methods.

The rapid developments of deep learning methods are attributed to two aspects: the developments of parallel computing units, i.e. GPUs, that enable the training of neural networks with extremely large number of parameters, and the availability of “*big data*”, so that massive data driven approaches become possible. Benefitting from these two aspects, the famous AlexNet [100], which is essentially a convolutional neural network (CNN) [30], achieved the best accuracy performance on image classification task on ImageNet [101], which is by the time the largest image dataset with over a million annotated images, marking the beginning of “deep learning era”. Since then, deep learning approaches have become dominant in *all* statistical machine learning tasks, including supervised learning [102]–[104], semi-supervised learning [105]–[107], unsupervised learning [108]–[112], reinforcement learning [113], [114], and more. It has also become the *de facto* approach for various domains, such as classification [103], object detection [115], semantic segmentation [116], instance segmentation [117], object tracking [118], pose estimation [119], video recognition [120], [121], image retrieval [122], person re-identification [123], to name a few.

Connections of Sparse Representations to Deep Learning

In a broader sense, the method of sparse representations falls into the category of “representation learning” [124], where the task is to construct or learn for the input a set of representations, or features, which should possess the property of *discriminative-ness*, namely, that the distinctions of different entities, e.g. the classes they belong to, should be reflected by the dissimilarities of their corresponding features. It has been found that the sparse representations of image patches are well suited for this task since ideally the statistically different patches should be represented sparsely by different dictionary atoms, so the sparse coding vectors are naturally discriminative. Sparse coding approaches has been successfully applied to visual recognition problems and achieved state-of-the-art results at that time [15], [125], [126].

On the other hand, classical deep learning approaches for representation learning involves learning an *autoencoder* [127]–[131], where a deep network is learned to reconstruct the input itself. The autoencoder consists of an *encoder* and a *decoder* (often symmetric in design), where the former is to map input into a *latent space*, and the latter is to reconstruct input from the latent space. After training the

autoencoder, the latent activations are used as representations of input samples and the decoder is discarded. Autoencoders and dictionary learning approaches have the following connections:

1. The process of sparse coding can be seen as *encoding* the input image patches into some sparse features. The difference is that sparse representations are usually overcomplete while the structure of autoencoder is often *squeezing* in the sense that the hidden dimension of the autoencoder is smaller than that of the input to learn nontrivial representations. Regarding sparse representations as feature encoders have profound influence on the design of visual recognition systems, particularly in [125] which shows that sparse coding as encoders provides better performance than sparse autoencoders under same settings.
2. Sparse coding can also be seen as *decoding* the coding vectors into the original image patches by adopting the reconstruction loss as autoencoders do. In this sense, the overcomplete sparse coding vectors should also be approximated by transformations of the input features, which acts as a parameterization function similar to the encoders of the autoencoder. The learning process of such model is elaborated in [93], [128] which is similar to that of an autoencoder with decoupled encoder and decoder reconstruction. This process is sometimes referred to as *predictive sparse decomposition* [83].
3. In a simplistic sense, linear autoencoders perform principal component analysis (PCA) of the input data [132], which in turn can be interpreted in a probabilistic way as *factor analysis* [133]:

$$\mathbf{x} = \mathbf{D}\boldsymbol{\alpha} + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2 \mathbf{I}).$$

Assuming x are normalized (i.e. $\bar{\mathbf{x}} = 0$), PCA models the latent code $\boldsymbol{\alpha}$ using a Gaussian prior, i.e. $\boldsymbol{\alpha} \sim \mathcal{N}(0, \sigma_\alpha^2 \mathbf{I})$. On the other hand, the sparse coding model can be regraded as modeling the latent code using a sparsity inducing Laplace prior, i.e. $\boldsymbol{\alpha} \sim \exp(-\lambda \|\boldsymbol{\alpha}\|)$, and recovering $\boldsymbol{\alpha}$ using maximum *a posteriori* estimation (MAP): $\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}}{\operatorname{argmax}} p(\boldsymbol{\alpha} | \mathbf{x})$ [5], [124].

Recent developments of deep learning have moved forward from layer-wise pretraining of deep networks using autoencoders [127], [130] to effective end-to-end training

of deep neural networks [100], [103], and representation learning favors the methodology of *self-supervised learning* [110]–[112] over the reconstruction based methods used by autoencoders. The following describes advantages and disadvantages of recent deep learning approaches compared with sparse coding:

1. Deep learning approaches, as have discussed previously, are data driven in the sense that massive data from the target domain are required to learn a usable model for the specific task [100], [134]. On the other hand, since representations on image patches are naturally sparse, SC can yield good performance on low data regime. For example, SC can perform well on visual recognition even if the dictionary are purely random image patches [125].
2. Neural network based methods are at present still considered as “*black box*” approaches, that is, there is no direct interpretations on *why* the network gives a specific prediction. In addition, it is known that deep networks can easily overfit (even random) labels [135], and they are vulnerable to adversarial examples [136], [137]. Sparsity based methods, on the other hand, have solid foundations and guarantees [68], and the solution has explicit statistical interpretations [6], [26], [75].
3. In practice, deep learning approaches are more scalable than sparse coding methods, considering that SC based methods can hardly handle large dataset such as ImageNet (note also that there exists successful attempts, e.g. [60]). This is partly due to the fact that the latter is a linear model while the former has a hierarchical structure with an extremely high dimensional latent space, and partly because deep learning models can benefit numerously from GPU’s parallel computation capacity. While there are attempts of generalizing sparse coding by “going deep” while maintaining its interpretability, e.g. [138]–[141], or “unrolling” the optimization procedure (usually ISTA [71], [72]) for more efficient learning [142], [143], the performance gap between the two methodologies are still large.

2.3 Alternating Direction Method of Multipliers

This part of the thesis is dedicated to an important convex optimization technique named *alternating direction method of multipliers* (ADMM) [90], that is extensively used in subsequent chapters: Chapter 3 utilizes ADMM for solving structured sparse coding problem (see Section 3.3), and Chapter 4 solves a nonnegative ℓ_2 regularized problem via ADMM (see Section 4.3). ADMM is able to formulate a given (additive) convex problem into several subproblems which are solved iteratively. The subproblems are often chosen to be simple and *decoupled* so that they can be solved independently and efficiently. The decomposition methodology of ADMM makes it particularly suitable for *distributed* convex optimization, where numerous simple subproblems can be solved simultaneously via decentralized workers. Additionally, the increasingly large amount of data require that the underlying algorithm should be scalable enough to process them efficiently in a distributed or decentralized manner, where ADMM is suited for such situation.

The contents of this part are mainly based on [90], where the main algorithm and convergence of ADMM is investigated in Section 2.3.1 and Section 2.3.2 respectively. Finally, the applications on solving sparse coding and convolutional sparse coding are demonstrated.

2.3.1 ADMM Algorithm

This part discusses the computational practice of general ADMM algorithm. The ADMM algorithm solves the following problem:

$$\min_{\mathbf{x}, \mathbf{z}} f(\mathbf{x}) + g(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{Ax} + \mathbf{Bz} = \mathbf{c}. \quad (2.43)$$

The dimensionality of each variable is given by $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{z} \in \mathbb{R}^m$, $\mathbf{A} \in \mathbb{R}^{p \times n}$, $\mathbf{B} \in \mathbb{R}^{p \times m}$ and $\mathbf{c} \in \mathbb{R}^p$. A basis assumption regarding (2.43) is that $f(\mathbf{x})$ and $g(\mathbf{z})$ should be convex. The *augmented Lagrangian* [144] is

$$\mathcal{L}_\rho(\mathbf{x}, \mathbf{z}, \mathbf{y}) \triangleq f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^T(\mathbf{Ax} + \mathbf{Bz} - \mathbf{c}) + \frac{\rho}{2} \|\mathbf{Ax} + \mathbf{Bz} - \mathbf{c}\|_2^2. \quad (2.44)$$

In the sense of methods of multipliers, $\mathbf{x}$ and $\mathbf{z}$ are referred to as *primal variables* and $\mathbf{y}$ is the *dual variable*. $\rho > 0$ is a *penalty parameter*. Instead of solving jointly for $\mathbf{x}$ and $\mathbf{z}$, ADMM updates $\mathbf{x}$ and $\mathbf{z}$ alternately with the following iterations:

$$\mathbf{x}^{(k+1)} = \underset{\mathbf{x}}{\operatorname{argmin}} \mathcal{L}_\rho(\mathbf{x}, \mathbf{z}^{(k)}, \mathbf{y}^{(k)}) \quad (2.45)$$

$$\mathbf{z}^{(k+1)} = \underset{\mathbf{z}}{\operatorname{argmin}} \mathcal{L}_\rho(\mathbf{x}^{(k+1)}, \mathbf{z}, \mathbf{y}^{(k)}) \quad (2.46)$$

$$\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \rho(\mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z}^{(k+1)} - \mathbf{c}). \quad (2.47)$$

This is similar to using a Gauss-Seidel pass over the primal variables $(\mathbf{x}, \mathbf{z})$ to update them separately, hence justifying the name *alternating direction*. The advantage of variable splitting is that when the objective is additive in the form of $f(\mathbf{x}) + g(\mathbf{z})$,

Scaled Form

A more convenient formulation involves combining the linear and quadratic term of (2.43) by scaling the dual variable $\mathbf{y}$. Define the residual $\mathbf{r} \triangleq \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c}$ and note

$$\mathbf{y}^T \mathbf{r} + (\rho/2) \|\mathbf{r}\|_2^2 = (\rho/2) \|\mathbf{r} + \mathbf{u}\|_2^2 - (\rho/2) \|\mathbf{u}\|_2^2, \quad (2.48)$$

where $\mathbf{u} \triangleq (1/\rho)\mathbf{y}$ is the *scaled dual variable*. Replacing the dual variable $\mathbf{y}$ in (2.45) to (2.47) by $\mathbf{u}$, the update iteration is

$$\mathbf{x}^{(k+1)} = \underset{\mathbf{x}}{\operatorname{argmin}} \left[f(\mathbf{x}) + (\rho/2) \|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}^{(k)} - \mathbf{c} + \mathbf{u}^{(k)}\|_2^2 \right] \quad (2.49)$$

$$\mathbf{z}^{(k+1)} = \underset{\mathbf{z}}{\operatorname{argmin}} \left[g(\mathbf{z}) + (\rho/2) \|\mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z} - \mathbf{c} + \mathbf{u}^{(k)}\|_2^2 \right] \quad (2.50)$$

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z}^{(k+1)} - \mathbf{c}. \quad (2.51)$$

The formulation (2.49) to (2.51) is referred to as *scaled form* of ADMM. In practice, this thesis will use the scaled form for solving optimization problems (see Algorithm 2 and Algorithm 3).

Stopping Criterion

One major concern in the algorithmic considerations of ADMM regards when the iterations (2.45)-(2.47) (or (2.49)-(2.51)) should terminate. Define the *primal residual* $\mathbf{r}^{(k+1)} \triangleq \mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z}^{(k+1)} - \mathbf{c}$ and the *dual residual* $\mathbf{s}^{(k+1)} \triangleq \rho \mathbf{A}^T \mathbf{B}(\mathbf{z}^{(k+1)} - \mathbf{z}^{(k)})$ at the $(k+1)$ -th iteration. It can be seen that $\mathbf{r}^{(k)}$ corresponds to the primal feasibility condition of the ADMM problem (2.43), and $\mathbf{s}^{(k)}$ actually corresponds to the dual feasibility conditions (see [90, Section 3.3] for details). [90] suggests a stopping criterion based on the norm of $\mathbf{r}^{(k)}$ and $\mathbf{s}^{(k)}$, i.e.

$$\|\mathbf{r}^{(k)}\|_2 \leq \epsilon_{\text{pri}} \quad \text{and} \quad \|\mathbf{s}^{(k)}\|_2 \leq \epsilon_{\text{dua}}, \quad (2.52)$$

where the tolerances $\epsilon_{\text{pri}} > 0$ and $\epsilon_{\text{dua}} > 0$ are chosen as

$$\begin{aligned} \epsilon_{\text{pri}} &= \sqrt{p}\epsilon_{\text{abs}} + \epsilon_{\text{rel}} \max\{\|\mathbf{A}\mathbf{x}^{(k)}\|_2, \|\mathbf{B}\mathbf{z}^{(k)}\|_2, \|\mathbf{c}\|_2\}, \\ \epsilon_{\text{dua}} &= \sqrt{n}\epsilon_{\text{abs}} + \epsilon_{\text{rel}} \|\mathbf{A}^T \mathbf{y}^{(k)}\|_2, \end{aligned} \quad (2.53)$$

and the absolute and relative $\epsilon_{\text{abs}}/\epsilon_{\text{rel}}$ are determined depending on the applications.

Updating Penalty Parameter

In practice the penalty parameter ρ can be varied to improve the convergence. In [90] the following update scheme is recommended based on the relative value of primal and dual residuals:

$$\rho^{(k+1)} = \begin{cases} \rho^{(k)} \cdot \tau_{\text{inc}} & \|\mathbf{r}^{(k)}\|_2 > \mu \|\mathbf{s}^{(k)}\|_2 \\ \rho^{(k)} / \tau_{\text{dec}} & \|\mathbf{s}^{(k)}\|_2 > \mu \|\mathbf{r}^{(k)}\|_2 \\ \rho^{(k)} & \text{otherwise} \end{cases} \quad (2.54)$$

Typical choices of hyperparameters are $\mu = 10$ and $\tau_{\text{inc}} = \tau_{\text{dec}} = 2$. In [145] a more complicated penalty update scheme is discussed.

It should be noted that when updating the penalty parameter in the scaled form of ADMM, the scaled dual parameter $\mathbf{u} = (1/\rho)\mathbf{y}$ should be correspondingly rescaled after updating ρ with (2.54), i.e. $\mathbf{u}^{(k)}$ should be divided by τ_{inc} if $\rho^{(k)}$ is multiplied by τ_{inc} , and vice versa.

ADMM with Nonconvex Objectives or Nonlinear Constraints

The above discussions focus on convex formulation of (2.43) with linear constraints $\mathbf{Ax} + \mathbf{Bz} = \mathbf{c}$. This applies to most, but not all scenarios whenever augmented Lagrange multipliers are required. Specifically, the ADMM algorithm is applied to solve (3.14), (3.23) and (4.11) in subsequent chapters. Easy examinations could find that problem (3.23) and (4.11) have convex objectives with linear constraints, while the objective of (3.14) is potentially not convex, and its constraint is nonlinear.

There have been discussions on applying ADMM with nonlinear constraints [146]. Furthermore, [147] discusses several specific forms of nonlinear constraints, and [148], [149] focus specifically on the useful generative priors [150] as nonlinear constraints. On the other hand, research on nonconvex objectives of ADMM has been active [151]–[155]. Last but not least, recently a general framework for nonconvex optimization problem (2.55) with nonlinear constraints is proposed in [156]:

$$\min_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{x}) \quad \text{s.t.} \quad \mathcal{A}(\mathbf{x}) = 0, \quad (2.55)$$

where g is convex, f is nonconvex and $\mathcal{A}$ is a nonlinear operator. This framework is helpful for studying the convergence behavior of (3.14) although in practice it is solved with the same treatment as the linear and convex case (2.43).

2.3.2 Convergence

The convergence analysis of ADMM iterations (2.49)–(2.51) given in [90, Appendix A] and [157] is briefly discussed here. Denote the optimal value of problem (2.43) as

$$p^* = \inf \{ f(\mathbf{x}) + g(\mathbf{z}) \mid \mathbf{Ax} + \mathbf{Bz} = \mathbf{c} \}, \quad (2.56)$$

Suppose the following assumptions are satisfied:

1. The functions $f(\mathbf{x})$ and $g(\mathbf{z})$ (with extended ranges $\mathbb{R} \cup \{+\infty\}$) are closed, proper and convex.
2. The (unaugmented) Lagrangian $\mathcal{L}_0$ has a saddle point $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*)$ such that for all $(\mathbf{x}, \mathbf{z}, \mathbf{y})$

$$\mathcal{L}_0(\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}) \leq \mathcal{L}_0(\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*) \leq \mathcal{L}_0(\mathbf{x}, \mathbf{z}, \mathbf{y}^*).$$

Then the following convergence results are given in [90] when $k \rightarrow \infty$:

- The primal residual $\mathbf{r}^{(k)} \rightarrow 0$.
- The objective $f(\mathbf{x}^{(k)}) + g(\mathbf{z}^{(k)}) \rightarrow p^*$.
- The dual variable $\mathbf{y}^{(k)} \rightarrow \mathbf{y}^*$ which is a dual optimal point.

2.3.3 ADMM for Sparse Coding

To finish the discussion, this part demonstrates on the application of ADMM for sparse coding [74] and convolutional sparse coding problems [158].

Sparse Representations

The standard ℓ_1 regularized basis pursuit denoising (BPDN) [69] with given dictionary $\mathbf{D}$ is formulated as

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{D}\mathbf{x} - \mathbf{s}\|_2^2 + \lambda \|\mathbf{x}\|_1. \quad (2.57)$$

The ADMM form is written as

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{D}\mathbf{x} - \mathbf{s}\|_2^2 + \lambda \|\mathbf{z}\|_1 \quad \text{s.t.} \quad \mathbf{x} - \mathbf{z} = 0. \quad (2.58)$$

The (scaled) ADMM iterates are

$$\begin{aligned} \mathbf{x}^{(k+1)} &= \underset{\mathbf{x}}{\operatorname{argmin}} (1/2) \|\mathbf{D}\mathbf{x} - \mathbf{s}\|_2^2 + (\rho/2) \|\mathbf{x} - \mathbf{z}^{(k)} + \mathbf{u}^{(k)}\|_2^2 \\ \mathbf{z}^{(k+1)} &= \underset{\mathbf{z}}{\operatorname{argmin}} \lambda \|\mathbf{z}\|_1 + (\rho/2) \|\mathbf{x}^{(k+1)} - \mathbf{z} + \mathbf{u}^{(k)}\|_2^2 \\ \mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \mathbf{x}^{(k+1)} - \mathbf{z}^{(k+1)}. \end{aligned} \quad (2.59)$$

The $\mathbf{x}$ -step is quadratic and has a closed form solution $\mathbf{x}^{(k+1)} = (\mathbf{D}^T \mathbf{D} + \rho \mathbf{I})^{-1} (\mathbf{D}^T \mathbf{s} + \rho(\mathbf{z}^{(k)} - \mathbf{u}^{(k)}))$. The $\mathbf{z}$ -step is given by

$$\mathbf{z}^{(k+1)} = \mathcal{S}_{\lambda/\rho}(\mathbf{x}^{(k+1)} + \mathbf{u}^{(k)}) \quad (2.60)$$

where $\mathcal{S}_\nu$ is the (element-wise) *soft thresholding operator* given by

$$\mathcal{S}_\nu(x) = \begin{cases} x - \nu & x > \nu \\ 0 & -\nu \leq x \leq \nu \\ x + \nu & x < -\nu \end{cases}. \quad (2.61)$$

Note that in general soft thresholding $\mathcal{S}_\nu$ is the *proximal operator* [159] of the ℓ_1 norm. When regularization is shifted from ℓ_1 to other constraints, the iterates (2.59) still hold by substituting the soft thresholding operator in (2.60) with the proximal operator associated with such constraint. For instance, the group sparse coding problem (3.22) in Chapter 3 is solved with ADMM and the proximal operator of the ℓ_2 norm (3.30).

Convolutional Sparse Representations

ADMM is the standard approach for solving the convolutional sparse coding problem (2.38). The procedure is similar to (2.58) and (2.59). The ADMM form is written as

$$\begin{aligned}
\min_{\{\mathbf{x}_i\}, \{\mathbf{z}_i\}} \quad & \frac{1}{2} \left\| \sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_i - \mathbf{s} \right\|_2^2 + \lambda \sum_{i=1}^m \|\mathbf{z}_i\|_1, \\
\text{s.t.} \quad & \mathbf{x}_i = \mathbf{z}_i, \forall 1 \leq i \leq m.
\end{aligned} \tag{2.62}$$

The iterations are

$$\begin{aligned}
\{\mathbf{x}_i\}^{(k+1)} &= \underset{\{\mathbf{x}_i\}}{\operatorname{argmin}} (1/2) \left\| \sum_i \mathbf{d}_i * \mathbf{x}_i - \mathbf{s} \right\|_2^2 + (\rho/2) \sum_i \|\mathbf{x}_i - \mathbf{z}_i^{(k)} + \mathbf{u}_i^{(k)}\|_2^2 \\
\mathbf{z}_i^{(k+1)} &= \underset{\mathbf{z}_i}{\operatorname{argmin}} \lambda \|\mathbf{z}_i\|_1 + (\rho/2) \|\mathbf{x}_i^{(k+1)} - \mathbf{z}_i + \mathbf{u}_i^{(k)}\|_2^2 \\
\mathbf{u}_i^{(k+1)} &= \mathbf{x}_i^{(k+1)} - \mathbf{z}_i^{(k+1)} + \mathbf{u}_i^{(k)}.
\end{aligned} \tag{2.63}$$

The $\mathbf{z}_i$ -step can be computed via (2.60). The general form for $\mathbf{x}_i$ in (2.63) is

$$\min_{\{\mathbf{x}_i\}} \frac{1}{2} \left\| \sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_i - \mathbf{s} \right\|_2^2 + \frac{\rho}{2} \sum_{i=1}^m \|\mathbf{x}_i - \mathbf{y}_i\|_2^2. \tag{2.64}$$

Bristow et al. [31] proposed to solve the specific $\mathbf{x}$ -step (2.64) efficiently using FFT. Specifically, denote $\hat{\mathbf{d}}_i$, $\hat{\mathbf{x}}_i$, $\hat{\mathbf{s}}$ and $\hat{\mathbf{y}}_i$ as the DFT of $\mathbf{d}_i$, $\mathbf{x}_i$, $\mathbf{s}$ and $\mathbf{y}_i$ respectively. Then the convolution theorem guarantees that (2.64) is equivalent to

$$\min_{\{\hat{\mathbf{x}}_i\}} \frac{1}{2} \left\| \sum_{i=1}^m \hat{\mathbf{D}}_i \hat{\mathbf{x}}_i - \hat{\mathbf{s}} \right\|_2^2 + \frac{\rho}{2} \sum_{i=1}^m \|\hat{\mathbf{x}}_i - \hat{\mathbf{y}}_i\|_2^2, \tag{2.65}$$

where $\hat{\mathbf{D}}_i = \operatorname{diag}(\hat{\mathbf{d}}_i)$. The transform can be done efficiently via FFT in $\mathcal{O}(mN \log(N))$. Furthermore, denote $\hat{\mathbf{D}} = [\hat{\mathbf{D}}_1, \dots, \hat{\mathbf{D}}_m]$, $\hat{\mathbf{x}} = [\hat{\mathbf{x}}_1, \dots, \hat{\mathbf{x}}_m]$ and $\hat{\mathbf{z}} = [\hat{\mathbf{z}}_1, \dots, \hat{\mathbf{z}}_m]$, it can be easily seen that the solution to (2.65) satisfies:

$$(\hat{\mathbf{D}}^H \hat{\mathbf{D}} + \rho \mathbf{I})^{-1} \hat{\mathbf{x}} = \hat{\mathbf{D}}^H \hat{\mathbf{s}} + \rho \hat{\mathbf{z}}. \tag{2.66}$$

Noting that $\hat{\mathbf{D}}$ is the concatenation of m diagonal matrices of size $N \times N$, Wohlberg et al. [32], [160] proposed to rearrange the independent system in (2.66). Each system can be solved efficiently via the Sherman-Morrison formula in $\mathcal{O}(m)$, in total

$\mathcal{O}(mN)$ for computing $\hat{\mathbf{x}}$. Therefore, the computation of $\mathbf{x}$ in (2.64) is dominated by FFT, which is $\mathcal{O}(mN \log(N))$.

2.4 Extreme Learning Machine

The traditional single layer feed-forward networks (SLFNs) provide powerful performance on classification and regression tasks. This is due to the expressive power of the nonlinear transformations. The universal approximation theorem [161] guarantees that SLFNs can well approximate continuous functions with mild conditions. In practice, the parameters of SLFNs are learned via SGD using back-propagation (BP) [162]. Since the parameter space is non-convex, the learning procedure often traps in stationary points of the optimization problem and oscillates. Recently, Huang et al. proposed Extreme Learning Machine (ELM) [55], [163]–[167] as a simple and effective learning approach based on SLFNs. ELM is essentially an SLFN with randomly generated and fixed neurons between the input and hidden layer. In this case, the optimization problem is convex with respect to the learnable output weights and the optimization can be carried out very efficiently by solving a least square problem. Despite its simplicity, ELM also has the universal approximation property [164] and provides good classification and regression capabilities [166]. ELM has been found successful in many domains, including supervised classification and regression tasks [166]–[168], representation learning [169]–[172], semi-supervised learning [173], dimension reduction [174], clustering [173], [175], [176], one-class learning [177], [178], unsupervised feature selection [179] etc. It has also demonstrated competitive performance in various applications, such as visual recognition [167], [170], [171], super resolution [180], video anomaly detection [178], outlier detection [181], machine fault diagnosis [182], 3D shape classification [183], [184], etc. Some good reviews regarding the topic of Extreme Learning Machine are presented in [185]–[188].

Given the data $(\mathbf{x}, \mathbf{y})$ defined on the joint input and label space $\mathcal{X} \times \mathcal{Y}$, ELM assigns fixed random neurons to each feature and encodes the j -th hidden neuron ($1 \leq j \leq L$) as [55]:

$$\mathbf{h}_j = G(\mathbf{a}_j, b_j, \mathbf{x}), \quad (2.67)$$

where $\mathbf{a}_j$ and b_j are hidden weights generated from some predefined data distribution. The nonlinear activation function G can be chosen as [166]:

- **Sigmoid function:**

$$G(\mathbf{a}, b, \mathbf{x}) \triangleq \frac{1}{1 + \exp(-(\mathbf{a} \cdot \mathbf{x} + b))}.$$

- **Hard-limit function:**

$$G(\mathbf{a}, b, \mathbf{x}) \triangleq \mathbb{1}_{\{\mathbf{a} \cdot \mathbf{x} + b \geq 0\}}.$$

- **Gaussian function:**

$$G(\mathbf{a}, b, \mathbf{x}) \triangleq \exp(-b\|\mathbf{x} - \mathbf{a}\|_2^2).$$

- **Multiquadric function:**

$$G(\mathbf{a}, b, \mathbf{x}) \triangleq (\|\mathbf{x} - \mathbf{a}\|_2^2 + b^2)^{1/2}.$$

After encoding the hidden space, effective learning can be achieved via solving the following ℓ_2 loss objective for the output weight $\boldsymbol{\beta} \in \mathbb{R}^{L \times C}$:

$$\min_{\boldsymbol{\beta}} \|\mathbf{H}\boldsymbol{\beta} - \mathbf{Y}\|_F^2, \quad (2.68)$$

where $\mathbf{H} \in \mathbb{R}^{N \times L}$ is the output of hidden layer with each row corresponding to the input sample $\mathbf{x}_i$:

$$\mathbf{H} = \begin{bmatrix} G(\mathbf{a}_1, b_1, \mathbf{x}_1) & G(\mathbf{a}_2, b_2, \mathbf{x}_1) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_1) \\ G(\mathbf{a}_1, b_1, \mathbf{x}_2) & G(\mathbf{a}_2, b_2, \mathbf{x}_2) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ G(\mathbf{a}_1, b_1, \mathbf{x}_N) & G(\mathbf{a}_2, b_2, \mathbf{x}_N) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_N) \end{bmatrix} \quad (2.69)$$

and $\mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_N]^T \in \mathbb{R}^{N \times C}$ is the collections of regression values (for regression problems) or one-hot encodings (for classification problems) $\mathbf{y}_i$ associated with each $\mathbf{x}_i$.

It can be seen that since the hidden neuron weights $\mathbf{a}_j$ and b_j are fixed, problem (2.68) is convex and admits a global minimum $\boldsymbol{\beta}^*$. Moreover, since the ℓ_2 loss is used, an analytic solution exists and avoids applying BP³. The optimal weight $\boldsymbol{\beta}^*$ can be obtained simply by:

$$\boldsymbol{\beta}^* = \mathbf{H}^\dagger \mathbf{Y}. \quad (2.70)$$

Finally, the output function corresponding to an input $\mathbf{x}$ is

$$f_{\text{ELM}}(\mathbf{x}) = \mathbf{h}(\mathbf{x})\boldsymbol{\beta}^* = \mathbf{h}(\mathbf{x})\mathbf{H}^\dagger \mathbf{Y} \quad (2.71)$$

where $\mathbf{h}(\mathbf{x}) = [G(\mathbf{a}_1, b_1, \mathbf{x}), G(\mathbf{a}_2, b_2, \mathbf{x}), \dots, G(\mathbf{a}_L, b_L, \mathbf{x})]$.

The universal approximation property [161], [189], [190] of ELM are proved in [164], which roughly states that any piecewise continuous function can be approximated by randomly generated function bases. After the advent of Extreme Learning Machine, *Random Features* (RF) [191] was proposed with the purpose of efficiently approximating kernel machines. RF is almost the same as ELM, except that it approximates a kernel function instead of a regression or classification target. Later, the universal approximation properties of random features similar to those of ELM [164] were rediscovered in [192], [193]. Their work has attracted some attentions and some recent developments of RF discuss its representation capability. For example, in [154], [194] the authors show separately that for certain types of functions, the RF model requires an exponential number of neurons for effective approximation. Based on similar insights that *curse of dimensionality* may harm RF or ELM's capabilities, in [195] they propose deep semi-random features as an alternative, which is shown to have better expressive power than RF, and better ganeralization error bound than common deep neural networks. In a different sense, in [196] the authors study the characteristics of ELM via *Random Matrix Theory*. They show that the training and testing errors of ELM converge to deterministic values when sample size and number of hidden neuron become large.

---

³Note that computing the least square solution still requires an iterative method such as SVD.

Subsequent contents in this section will be focusing on applying ELM in various domains, i.e. supervised learning (Section 2.4.1), representation learning (Section 2.4.2) and unsupervised learning (Section 2.4.3).

2.4.1 Regularized Extreme Learning Machine for Regression and Classification

The naive ELM formulation (2.68) provides universal approximation capabilities [164] and good regression/classification performance [55]. On the other hand, the solution β^* can be made stabler by penalizing the output weight. From ridge regression point of view, regularizing the norm of β decreases the variance of the solution (with slight increase of bias). Additionally, as noted in [197] and discussed in [166], SLFNs with small training error achieves better generalization performance for smaller norm of weights. Therefore Huang et al. proposed to generalize (2.68) to the regularized case:

$$\min_{\beta} \|\mathbf{H}\beta - \mathbf{Y}\|_F^2 + \frac{1}{C} \|\beta\|_F^2. \quad (2.72)$$

The formulation (2.72) can be applied for regression and multiclass classification, depending on the value of $\mathbf{Y}$. It can be also noted that (2.72) is also convex and the normal equation is always positive definitive with smallest eigenvalue $\lambda_{\min} \geq 1/C$. Like (2.68), the formulation (2.72) also admits an analytic solution:

$$\beta^* = (\mathbf{H}^T \mathbf{H} + \frac{\mathbf{I}}{C})^{-1} \mathbf{H}^T \mathbf{Y}. \quad (2.73)$$

The output function is

$$f_{\text{ELM}}(\mathbf{x}) = \mathbf{h}(\mathbf{x})\beta^* = \mathbf{h}(\mathbf{x})(\mathbf{H}^T \mathbf{H} + \frac{\mathbf{I}}{C})^{-1} \mathbf{H}^T \mathbf{Y}. \quad (2.74)$$

2.4.2 Extreme Learning Machine based Autoencoder

The Extreme Learning Machine is originally proposed for supervised learning tasks, e.g. regression and classification. Generalizing ELM for feature representation is

not trivial, and specific approaches are required for extracting useful information for the randomly assigned ELM feature space $\mathbf{H}$.

Kasun et al. [169], [174] proposed an ELM based autoencoder [131] named ELM-AE to exploit the representation capability of ELM. An autoencoder can be regarded also as a single layer feed-forward neural network with the input-to-hidden transformation as the “encoder” and the hidden-to-output transformation as the “decoder”. The target of the autoencoder is to reconstruct the original input distribution, in the hope that the hidden layer can capture non-trivial and disentangled representations of the original data.

ELM-AE shares such idea whereas the encoder during training is randomly assigned and fixed. Specifically, consider that the input data are encoded into a low dimensional space with the randomly assigned matrices $\mathbf{A}$ and $\mathbf{b}$:

$$\mathbf{h}(\mathbf{x}) = G(\mathbf{x}\mathbf{A} + \mathbf{b}) \quad \text{with} \quad \mathbf{A}^T \mathbf{A} = \mathbf{I}, \mathbf{b}^T \mathbf{b} = 1. \quad (2.75)$$

Note that the orthogonality of $\mathbf{A}$ and $\mathbf{b}$ preserves the Euclidean distances among the samples. The encoded features $\mathbf{H} = [\mathbf{h}(\mathbf{x}_1)^T, \dots, \mathbf{h}(\mathbf{x}_N)^T]^T$ are decoded with linear decoder $\boldsymbol{\beta}_{\text{AE}}$:

$$\min_{\boldsymbol{\beta}_{\text{AE}}} \|\mathbf{H}\boldsymbol{\beta}_{\text{AE}} - \mathbf{X}\|_F^2. \quad (2.76)$$

Note that compared with (2.68) where the target is the corresponding label $\mathbf{Y}$, in (2.76) the target is the input data themselves. The formulation is a standard autoencoder except that the hidden weights are generated and fixed. In this case, the learning of decoder can be done efficiently via

$$\boldsymbol{\beta}_{\text{AE}}^* = \mathbf{H}^\dagger \mathbf{X}. \quad (2.77)$$

The linear autoencoder learns the variance information of data. By contrast, it is proved in [174] that linear ELM-AE tries to learn the between-class scatter matrix of data. Therefore it retains meaningful information and exhibits discriminative capability.

The output transformation achieved by ELM-AE is to project the input data $\mathbf{X}$ with the obtained decoder weights β_{AE}^* :

$$\mathbf{X}_{\text{out}} = \mathbf{X}\beta_{\text{AE}}^{*\text{T}}. \quad (2.78)$$

2.4.3 Extreme Learning Machine for Unsupervised Learning

Other than supervised methods where ELM has been found effective [55], [166], [167], ELM has been widely applied for unsupervised learning. Section 2.4.2 has presented the ELM based autoencoder which can be applied for dimension reduction and unsupervised feature learning. In the following some ELM based unsupervised learning approaches are investigated which show competitive performance.

Unsupervised ELM for Clustering

Semi-supervised learning (SSL) and unsupervised learning (UL) approaches often build on top of a manifold regularization framework [198], i.e. the representations $\mathbf{h}_i$ and $\mathbf{h}_j$ of two samples should be close to each other if $\mathbf{x}_i$ and $\mathbf{x}_j$ are themselves close. In this scenario, two concerns arise naturally for clustering approaches defined on the manifold:

1. How to select or learn a proper similarity matrix $\mathbf{W}$ to regularize the distance;
2. How to learn a suitable feature representation that is discriminative for the applications.

Spectral clustering [199] is a classical approach tackling on the second concern, which uses eigenvectors of the Laplacian of the similarity matrix as feature representations. Huang et al. proposed unsupervised ELM (US-ELM) [173] to perform spectral clustering on ELM feature space, which is more discriminative. Specifically, US-ELM solves the following objective:

$$\begin{aligned} \min_{\beta} \quad & \|\beta\|_F^2 + \alpha \text{tr}[\beta \mathbf{H} \mathbf{L} \mathbf{H}^T \beta^T] \\ \text{s.t.} \quad & \beta \mathbf{H} \mathbf{H}^T \beta^T = \mathbf{I}. \end{aligned} \quad (2.79)$$

Here $\mathbf{H}$ is the ELM feature transformation given by (2.69), and $\mathbf{L}$ is Laplacian matrix of some predefined similarity matrix. In essence US-ELM replaces the original features with ELM embeddings $\mathbf{H}\boldsymbol{\beta}$ which are shown to be more discriminative. To perform clustering, K-Means is used to assign centroid labels on the obtained $\mathbf{H}\boldsymbol{\beta}$. US-ELM has empirically been shown to provide better clustering performance than spectral clustering and other counterparts.

ELM with Constrained Laplacian Rank for Graph Learning

US-ELM proposes a framework using ELM embeddings to learn feature representations. Building or learning a proper similarity matrix or *graph* that addresses the intrinsic characteristics of the manifold is equally important and addresses the first concern above. Typical clustering methods including spectral clustering and US-ELM utilize a predefined graph. By contrast, Liu et al. [176] proposed ELM-CLR, which is an effective *graph learning* method via adaptive graph regularizations and dual data representations. Given data samples $\{\mathbf{x}_i\}_{i=1}^N$, instead of predefining the data graph for clustering, ELM-CLR proposes to solve the following objective:

$$\begin{aligned} \min_{\mathbf{W}} \quad & \sum_{i,j=1}^N (\bar{\mathbf{d}}_{ij} \bar{\mathbf{p}}_{ij} \mathbf{w}_{ij} + \gamma \mathbf{w}_{ij}^2) \\ \text{s.t.} \quad & \mathbf{w}_{ij} \geq 0, \mathbf{w}_i^T \mathbf{1} = 1, \text{rank}(\mathbf{L}) = N - C. \end{aligned} \quad (2.80)$$

Here $\mathbf{L}$ is the Laplacian matrix for the data similarity matrix $\mathbf{W} = [\mathbf{w}_{ij}]_{i,j=1}^N$. $\bar{\mathbf{d}}_{ij}$ and $\bar{\mathbf{p}}_{ij}$ are *normalized pairwise distances* defined on the original feature space and *ELM feature space* between the i -th and the j -th sample:

$$\begin{aligned} \mathbf{d}_{ij} &= \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 & \mathbf{p}_{ij} &= \|\mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta} - \mathbf{h}(\mathbf{x}_j)\boldsymbol{\beta}\|_2^2 \\ \bar{\mathbf{d}}_{ij} &= \sqrt{\frac{\mathbf{d}_{ij}}{\|\mathbf{d}_i\|_2} \cdot \frac{\mathbf{d}_{ji}}{\|\mathbf{d}_j\|_2}} & \bar{\mathbf{p}}_{ij} &= \sqrt{\frac{\mathbf{p}_{ij}}{\|\mathbf{p}_i\|_2} \cdot \frac{\mathbf{p}_{ji}}{\|\mathbf{p}_j\|_2}} \end{aligned} \quad (2.81)$$

Note that instead of regularizing the obtained features $\mathbf{H}\boldsymbol{\beta}$ with predefined weights $\mathbf{W}$ in (2.79), ELM-CLR regularizes the similarity weight $\mathbf{W}$ via distances with dual representations, i.e. $\bar{\mathbf{d}}_{ij}$ and $\bar{\mathbf{p}}_{ij}$. Therefore, the obtained graph should capture more structural information for the underlying data manifold.

Chapter 3

ELM Embedded Discriminative Dictionary Learning for Image Classification

Chapter 3 investigates the topic of dictionary learning where discrimination is achieved via exploiting structures of the sparse coding vectors within and between predefined classes, and effective features are obtained through ELM embeddings. Section 3.1 describes existing discriminative dictionary learning approaches and their limitations. Section 3.2 proposes a novel dictionary learning formulation for image classification, and Section 3.3 discusses efficient optimization schemes for the proposed method. Section 3.4 justifies the design and performance of the proposed algorithm. The contents of this chapter are based on the following paper:

- Yijie Zeng, Yue Li, Jichao Chen, Xiaofan Jia, and Guang-Bin Huang, “ELM embedded discriminative dictionary learning for image classification,” *Neural Networks*, Mar. 2020. DOI: 10.1016/j.neunet.2019.11.015.

3.1 Background and Motivations

Feature representation [124] is a central topic for the task of image classification. Natural images usually come with large amounts of variability and noise because of varied illuminations and viewpoints. Finding an efficient representation that is robust to such variations is important for classification tasks. Typically, such representations include hand crafted features such as SIFT [200] or HOG [201], or features learned from a deep neural network and large data [100]. They have been found successful for image classification in the past twenty years. Although deep learning based approaches have been widely used in recent years, they still suffer from over-fitting when the underlying dataset is small.

Sparse representation can be regarded as a type of feature learning for natural signals, which is especially effective when little or no prior knowledge is known about the underlying dataset. The sparse representation method aims to find a set of filters that can approximate each sample linearly with only a portion of its elements. It has been found to be particularly effective for image and vision processing [80] for two reasons: 1) Sparse coding can be used to model the receptive fields of simple cells in the mammalian primary visual cortex [4], [5]; 2) Image patches often reside near a low dimensional manifold, and a semantically rich representations will recover such subspace as a small sets of atoms (from a properly learned dictionary), leading naturally to sparse representations [19]. In this chapter, image classification is considered as the underlying task to validate the proposed algorithm.

Among all the necessary components for a good sparse representation, dictionary learning is most important because the atoms represent examples of the target dataset. This part of the thesis utilizes *discriminative dictionary learning* (DDL), where label information is incorporated for learning a dictionary suitable for image classification. The importance and typical methods for DDL have been explored in Section 2.2.2. Most of them enforce class-specific constraints on sparse codes from the same class to achieve discrimination for the image classification task. A common approach is to transform the coding vectors into their class labels with a linear regressor, and incorporate the linear prediction loss into the reconstruction

loss [20], [51]. It is also favorable to require the coding vectors to be similar via different regularizations such as manifold regularization [21], [23], Fisher criterion [47], [202], distance of pairwise support coding vectors [24], etc.

Since the sparse pattern of the coding vectors describes atom assignment to each sample, requiring similar samples to share overlapping sparse patterns also enforces similarity, which in turn achieves discrimination among different classes [25]. This idea can be implemented via a simple group sparse $\ell_{1,2}$ norm on the coding vectors, which enforces the coding matrix of the same class to be row-sparse. Furthermore, in SDDL [25] (also see Section 2.2.2), an ℓ_0 norm is applied on the overlap of coding vectors from different classes to promote dissimilarities among different classes. Numerically the ℓ_0 norm in SDDL is unstable and hard to optimize.

It is also noted that few has discussed the problem of input feature representations for dictionary learning approaches. Adopting a dedicated feature learning method specifically designed for each dataset requires large domain knowledge. This is in contrast to common dictionary learning approaches which only assumes sparsity of features. Therefore, sparse coding and dictionary learning methods often only apply a fixed linear transformation, e.g. random-face [10], PCA, before learning the codes. Applying a learned feature transformation such as a projection matrix has been proved to improve the performance [203], but this method often falls into the categories of dimension reduction [204]–[206] or manifold regularization [50], [207].

Inspired by this idea, this thesis tries to adopt a simple and *nonlinear* feature transformation in conjunction with dictionary learning approaches. Specifically, Extreme Learning Machine (ELM) [55] with orthogonal output weights is applied as feature mappings. ELM as an efficient regression/classification algorithm is popular. Also, applying ELM as feature embedding has been found useful for clustering tasks [173], [175], as discussed in Section 2.4.3. It will be shown that such idea is also beneficial for dictionary learning. The advantages of ELM embeddings over linear projections are two-fold: 1) The random and high dimensional nonlinear mappings provides expressive representations suitable for learning the dictionary, and, 2) learning the linear output weight is as efficient as learning a linear projection, which avoids optimizing on a non-convex manifold.

3.1.1 Contributions

This chapter makes the following contributions:

- It proposes a novel weighted class specific $\ell_{1,2}$ norm for discriminative dictionary learning which promotes similarities of coding vectors within the same class and dissimilarities among different classes. The proposed $\ell_{1,2}$ norm is unified and robust compared to the $\ell_{1,2} + \ell_0$ optimization in SDDL.
- It shows that ELM features can be utilized as a simple and effective approach to enhance dictionary learning. Compared to linear projection, the nonlinear ELM embedding achieves *feature diversity* on the high dimensional hidden space and *computational efficiency* on the low dimensional output space.
- It shows that the sparse representation classifier (SRC) can be utilized on the *shared* dictionary for image classification as long as the structured regularizations produce discriminative sparse codes, and can achieve superior performance.

3.2 Proposed Method

The following objective is proposed for dictionary learning:

$$\begin{aligned} \min_{\beta, \mathbf{A}, \mathbf{D}} \quad & \frac{1}{2} \|\beta \mathbf{H} - \mathbf{D} \mathbf{A}\|_F^2 + \mathcal{F}_1(\beta) + \mathcal{F}_2(\mathbf{A}) \\ \text{s.t.} \quad & \beta \beta^T = \mathbf{I}, \|\mathbf{d}_i\|_2 = 1, \forall i. \end{aligned} \quad (3.1)$$

Here $\mathcal{F}_1$ and $\mathcal{F}_2$ are regularizations for $\beta \in \mathbb{R}^{n \times L}$ and $\mathbf{A} \in \mathbb{R}^{m \times N}$ respectively. $\mathbf{H} = [\mathbf{h}_1, \dots, \mathbf{h}_N] \in \mathbb{R}^{L \times N}$ is the nonlinear transformations of input samples $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N] \in \mathbb{R}^{d \times N}$ as in (2.67):

$$\mathbf{h}_i = f(\mathbf{W} \mathbf{x}_i + \mathbf{b}) \quad (3.2)$$

where $\mathbf{W} \in \mathbb{R}^{L \times d}$ and $\mathbf{b} \in \mathbb{R}^L$ are random matrices generated from continuous probability distribution [55]. $f(\cdot)$ is an element-wise nonlinear activation function. $\mathbf{D} = [\mathbf{d}_1, \dots, \mathbf{d}_m] \in \mathbb{R}^{n \times m}$ is the dictionary with m atoms, and $\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_N] \in \mathbb{R}^{m \times N}$ are coding vectors.

3.2.1 ELM Mapping as Nonlinear Transformations

The transformation (3.2) projects input samples $\mathbf{x}_i$ into high dimensional feature space $\mathbf{h}_i$ via a random nonlinear mapping. After that, a projection matrix $\beta \in \mathbb{R}^{n \times L}$ with orthonormal rows is learned to transform the high dimensional features $\mathbf{h}_i$ to the output space $\beta \mathbf{h}_i \in \mathbb{R}^n$. Notice the output space is determined by the regularization $\mathcal{F}_1$. On the obtained output space, dictionary learning is conducted with $\mathcal{F}_2$ as sparse constraints and atoms $\mathbf{d}_i$ with unit norm. The ELM embedding $\beta \mathbf{H}$ for dictionary learning has two advantages:

1. The feature transformation (3.2) is simple and expressive. The high dimensional features $\mathbf{H}$ are obtained efficiently which are critical for high performance [203] without tuning parameters. Since the transformation is nonlinear, the universal approximation capability [164] guarantees that the underlying data distribution can be accurately approximated.
2. Although the hidden features $\mathbf{h}_i \in \mathbb{R}^L$ are high dimensional, dictionary learning is performed solely on a low dimensional subspace $\mathcal{O} \subseteq \mathbb{R}^n$, which is achieved via the projection matrix β . Optimizing β amounts to *searching for the best subspace* that is most suited for learning a discriminative dictionary.

Compared with a linear mapping such as PCA or LDA, ELM transformation captures more discriminative features in a high dimensional latent space. On the other hand, other nonlinear transformations are also available, such as kernel PCA [208] or neural networks [209]. Kernel based methods such as kernel PCA look at the reproducing kernel Hilbert space only and is not adaptive to the dictionary learning process. Neural network based methods have more than two layers of parameters to tune, which are unstable (the sigmoid activation often saturates without proper initializations), complicated (it has a large number of hyper-parameters to search, e.g. learning rate, weight decay, etc.), and are easily trapped in stationary points during optimization. Compared with other nonlinear transformations, the ELM mapping $\beta \mathbf{H}$ is *linear* to the hidden space which enables efficient learning and is adaptive to the dictionary learning objective in (3.1).

3.2.2 Discriminative Projection

A discriminative and representative subspace $\beta\mathbf{H}$ is important for dictionary learning, which is determined by the regularization $\mathcal{F}_1$. In the simplest form where $\mathcal{F}_1(\beta) = 0$, (3.1) projects to the output space without class information. Incorporating the class information naturally leads to an LDA-like regularization. However, as noted in [210], LDA suffers from the *small sample size problem*, so the *maximum margin criterion* (MMC) [210] is applied instead which has similar class separation effect:

$$\mathcal{F}_1(\beta) = \frac{\lambda_1}{2} \text{tr}[\beta(\mathbf{S}_w - \mathbf{S}_b)\beta^T]. \quad (3.3)$$

Here λ_1 is a hyperparameter, and $\mathbf{S}_w$ and $\mathbf{S}_b$ are the within-class scatter matrix and between-class scatter matrix on the *hidden space* $\mathcal{H}$ defined as:

$$\mathbf{S}_w = \sum_{c=1}^C \sum_{i \in \mathcal{H}_c} (\mathbf{h}_i - \bar{\mathbf{h}}_c)(\mathbf{h}_i - \bar{\mathbf{h}}_c)^T \quad (3.4)$$

$$\mathbf{S}_b = \sum_{c=1}^C N_c (\bar{\mathbf{h}}_c - \bar{\mathbf{h}})(\bar{\mathbf{h}}_c - \bar{\mathbf{h}})^T \quad (3.5)$$

where $\bar{\mathbf{h}}_c$ and $\bar{\mathbf{h}}$ are mean vectors of $\mathcal{H}_c$ and $\mathcal{H}$ respectively.

The effect of (3.3) is to minimize the within-class scatter and to maximize the between-class scatter on the *output space* $\mathcal{O}$. Note that the formulation of (3.3) is different from that of [47]. In [47] the Fisher discrimination is computed on $\mathbf{A}$ to maximize the class separability of the coding space, while (3.3) aims to separate the classes of features alongside dictionary learning.

3.2.3 Discriminative Sparse Regularization

The idea of class specific $\ell_{1,2}$ norm is motivated by multi-task learning where similar tasks share similar sparse patterns [25]. In the context of dictionary learning, the row-sparse structure enforces selecting the same dictionary atoms for the same class. To further promote dissimilarities between different classes, Liu et al. [25]

proposed to suppress the overlapping supports of coding vectors from different classes via ℓ_0 minimization. However, their scheme (2.23) is hard to optimize, unstable and sensitive to the optimization approach.

As an alternative, a simple and robust sparse regularization is investigated for discriminative dictionary learning. Specifically, $\mathcal{F}_2$ can be modeled as

$$\mathcal{F}_2(\mathbf{A}) = \sum_{c=1}^C \left(\lambda_2 + \lambda_3 \|\mathbf{a}_{-c}^{(i)}\|_2 \right) \|\mathbf{a}_c^{(i)}\|_2, \quad (3.6)$$

where $\mathbf{a}_c^{(i)}$ and $\mathbf{a}_{-c}^{(i)}$ are the i -th row of $\mathbf{A}_c$ and $\mathbf{A}_{-c}$, which represents coding vectors that belong to class c and those that *do not* belong to class c , respectively. $\lambda_2 > 0$ and $\lambda_3 > 0$ are two trade-off parameters.

Note that when $\mathbf{a}_{-c}^{(i)}$ is large, i.e. the i -th dictionary atom is well adopted for classes other than c , then (3.6) enforces that $\mathbf{a}_c^{(i)}$ is more likely to be suppressed, which ultimately maximizes class separability. On the other hand, if $\lambda_2 = 0$, then (3.6) achieves zero when the class support is *completely* separated, regardless of concrete values of $\mathbf{A}$.

Note that when $\lambda_3 = 0$, (3.6) is equivalent to (2.23) with $w_2 = 0$, thus it can be regarded as a weighted version of (2.23). Define $\mathbf{W}_c \in \mathbb{R}^{m \times m}$ as a diagonal matrix whose i -th diagonal entry is $[\mathbf{W}_c]_{ii} = \lambda_2 + \lambda_3 \|\mathbf{a}_{-c}^{(i)}\|_2$, then

$$\mathcal{F}_2(\mathbf{A}) = \sum_{c=1}^C \|\mathbf{W}_c \mathbf{A}_c\|_{1,2}. \quad (3.7)$$

3.2.4 Proposed Formulation

The final formulation of the proposed ELM-DDL framework (3.1) is given as follows:

$$\begin{aligned} \min_{\boldsymbol{\beta}, \mathbf{A}, \mathbf{D}} & \frac{1}{2} \|\boldsymbol{\beta} \mathbf{H} - \mathbf{D} \mathbf{A}\|_F^2 + \\ & \frac{\lambda_1}{2} \text{tr}[\boldsymbol{\beta}(\mathbf{S}_w - \mathbf{S}_b)\boldsymbol{\beta}^T] + \sum_{c=1}^C \|\mathbf{W}_c \mathbf{A}_c\|_{1,2} \\ \text{s.t.} & \quad \boldsymbol{\beta} \boldsymbol{\beta}^T = \mathbf{I}, \|\mathbf{d}_i\|_2 = 1, \forall i. \end{aligned} \quad (3.8)$$

We make a few remarks regarding the formulation (3.8):

- For the proposed formulation (3.8), both $\mathcal{F}_1$ and $\mathcal{F}_2$ are designed as discriminative regularizations. However, they serve very different purposes: the discriminative projection term $\mathcal{F}_1$ learns a suitable projection for feature representations, and the discriminative sparse regularization term $\mathcal{F}_2$ regularizes the sparse coding vectors to learn a discriminative dictionary.
- The proposed method does not consider class specific regularizations on the dictionary $\mathbf{D}$. Additional constraints on $\mathbf{D}$ such as structural incoherence [46] can increase performance as identified by [50] and discussed in Section 2.2.2. However, this also increases the complexity of the proposed method, making it more difficult to optimize. Moreover, unlike [46], dictionary learning methods based on support discrimination (such as [25] and ours) do not assign dictionary atoms to each class. Therefore, the discussion within this thesis makes no structured regularizations on the dictionary.
- Additionally, it requires access to *all* data in the dataset for optimizing (3.8). This is not required for updating dictionary $\mathbf{D}$ or projection matrix β , but is required by the scatter matrices $\mathbf{S}_w$ and $\mathbf{S}_b$ which model variances among the whole dataset, and is required by learning the sparse code $\mathbf{A}$ which enforces that samples within the same class should have the same sparsity pattern. In this sense, the proposed discriminative dictionary learning method in Algorithm 2 is not very suitable for large image classification datasets where loading all samples are prohibited. Nevertheless, it is an interesting future direction for developing an online learning procedure for approximating the objective (3.8), with treatments similar to [62] or those in Chapter 5. If such generalization is possible then it would be worthwhile to test the ELM-DDL framework on large scale image datasets such as ImageNet.

Before describing the detailed optimization approach for solving (3) in Section 3.3, classification approaches for the obtained representation $\mathbf{A}$ for the supervised learning task, i.e. image classification, is first discussed.

3.2.5 Classification Scheme

After learning β and $\mathbf{D}$ on the training set, each test sample $\mathbf{x}_{\text{test}}$ is first embedded using ELM:

$$\mathbf{h}_{\text{test}} = \beta f(\mathbf{W}\mathbf{x}_{\text{test}} + \mathbf{b}) \quad (3.9)$$

The output feature $\mathbf{h}_{\text{test}}$ is used to compute the coding vector with Lasso:

$$\mathbf{a}_{\text{test}} = \underset{\mathbf{a}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{h}_{\text{test}} - \mathbf{D}\mathbf{a}\|_2^2 + \lambda_4 \|\mathbf{a}\|_1. \quad (3.10)$$

To classify $\mathbf{a}_{\text{test}}$, previous methods learn a linear classifier $\mathbf{C}$ on the training set and classify the test coding vectors with $\mathbf{C}$. Since for those methods the dictionary is shared among all classes, it is hard to apply the sparse representation classifier (SRC) [10] which explicitly requires sub-dictionaries for each class. This chapter investigates simple heuristics which allows the use of SRC. For class c the dictionary atoms are simply selected for which the coding vectors have *non-zero supports* to form the sub-dictionary. Concretely, define the index set as follows:

$$\mathcal{I}_c = \{i \mid \|\mathbf{a}_c^{(i)}\|_2 \neq 0\}. \quad (3.11)$$

Then the sub-dictionary $\mathbf{D}_c$ can be defined as

$$\mathbf{D}_c = \{\mathbf{d}_i \mid i \in \mathcal{I}_c\} \quad (3.12)$$

Since each sample in $\mathbf{A}_c$ has unified sparse support, the atoms corresponding to the support indices can be regarded as representatives of class c . SRC can be utilized for each sample:

$$l_{\text{pred}} = \underset{c}{\operatorname{argmin}} \|\mathbf{h}_{\text{test}} - \mathbf{D}_c \mathbf{a}_{\text{test},c}\|_2, \quad (3.13)$$

where $\mathbf{a}_{\text{test},c} = \{[\mathbf{a}_{\text{test}}]_i \mid i \in \mathcal{I}_c\}$ selects indices of $\mathbf{a}_{\text{test}}$ from $\mathcal{I}_c$, and l_{pred} is the predicted label for $\mathbf{x}_{\text{test}}$.

Note that (3.12) does not guarantee that the sub-dictionaries are non-overlapping, but empirically such scheme is stable and provide slight improvement over the linear predictor $\mathbf{C}$. Empirical validations will be given in Section 3.4.5.

3.3 Optimizations of ELM-DDL

The optimization of (3.8) is divided into three subproblems to update the projection β , coding vectors $\mathbf{A}$ and the dictionary $\mathbf{D}$. The subproblems are optimized alternately with the other variables fixed in an iterative fashion. Details of the algorithm is presented in in Algorithm 2.

3.3.1 Update of β

Assuming that $\mathbf{A}$ and $\mathbf{D}$ are fixed, the objective of β becomes

$$\begin{aligned} \mathcal{L}_\beta &= \frac{1}{2} \|\beta \mathbf{H} - \mathbf{D} \mathbf{A}\|_F^2 + \frac{\lambda_1}{2} \text{tr}[\beta(\mathbf{S}_w - \mathbf{S}_b)\beta^T] \\ \text{s.t. } &\beta \beta^T = \mathbf{I}. \end{aligned} \quad (3.14)$$

The above problem can be solved by ADMM. The ADMM problem is

$$\begin{aligned} \min_{\gamma, \beta} & \frac{1}{2} \|\gamma \mathbf{H} - \mathbf{D} \mathbf{A}\|_F^2 + \frac{\lambda_1}{2} \text{tr}[\gamma(\mathbf{S}_w - \mathbf{S}_b)\gamma^T] \\ \text{s.t. } & \gamma = \beta, \beta \beta^T = \mathbf{I} \end{aligned} \quad (3.15)$$

where $\gamma \in \mathbb{R}^{n \times L}$. The (scaled) augmented Lagrangian is

$$\begin{aligned} \min_{\gamma, \beta, \mathbf{u}} & \frac{1}{2} \|\gamma \mathbf{H} - \mathbf{D} \mathbf{A}\|_F^2 + \frac{\lambda_1}{2} \text{tr}[\gamma(\mathbf{S}_w - \mathbf{S}_b)\gamma^T] + \frac{\rho}{2} \|\gamma - \beta + \mathbf{u}\|_F^2 \\ \text{s.t. } & \beta \beta^T = \mathbf{I} \end{aligned} \quad (3.16)$$

where $\mathbf{u} \in \mathbb{R}^{n \times L}$ is the dual variable, and $\rho > 0$ is the penalty weight. At step $k+1$ each variable can be updated via

---

Algorithm 2: ELM Embedded Discriminative Dictionary Learning (ELM-DDL)

Data: Input samples $\mathbf{X} \in \mathbb{R}^{d \times N}$, random matrices $\mathbf{W} \in \mathbb{R}^{L \times d}$ and $\mathbf{b} \in \mathbb{R}^L$, output dimension n , number of atoms m , hyperparameters $\lambda_1, \lambda_2, \lambda_3$.

```

1 for  $i = 1, \dots, N$  do
2    $\mathbf{h}_i = f(\mathbf{W}\mathbf{x}_i + \mathbf{b})$ 
3 Compute scatter matrices  $\mathbf{S}_w$  (3.4) and  $\mathbf{S}_b$  (3.5);
4 Initialize  $\mathbf{D}$  with K-Means,  $\beta$  with PCA and  $\mathbf{A} = 0$ ;
5 for  $t = 1, \dots, T$  do
6   while Not converged do
7     Compute  $\gamma^{(k+1)}$ :
8       
$$\gamma^{(k+1)} = \left( \mathbf{D}\mathbf{A}\mathbf{H}^T + \rho(\beta^{(k)} - \mathbf{u}^{(k)}) \right) \left( \mathbf{H}\mathbf{H}^T + \lambda_1(\mathbf{S}_w - \mathbf{S}_b) + \rho\mathbf{I} \right)^{-1}$$

9     Project  $\beta^{(k+1)}$  using SVD:
10      
$$\beta^{(k+1)} = \mathbf{U}\mathbf{V}^T \quad \text{s.t.} \quad [\mathbf{U}, \Sigma, \mathbf{V}] = \text{SVD}(\gamma^{(k+1)} + \mathbf{u}^{(k)})$$

11     Update  $\mathbf{u}^{(k+1)}$  with (3.19) and  $\rho$ ;
12   for  $c = 1, \dots, C$  do
13     while Not converged do
14       Compute  $\mathbf{G}_c^{(k+1)}$ :
15        
$$\mathbf{G}_c^{(k+1)} = \left( \mathbf{D}^T\mathbf{D} + \rho\mathbf{I} \right)^{-1} \left( \mathbf{D}^T\beta\mathbf{H}_c + \rho(\mathbf{A}_c^{(k)} - \mathbf{U}_c^{(k)}) \right)$$

16       Compute the  $i$ -th row of  $\mathbf{A}_c^{(k+1)}$  with
17        
$$\nu^{(i)} = 1 - (\lambda_2 + \lambda_3 \|\mathbf{a}_{-c}^{(i)}\|_2) / (\rho \|\mathbf{g}_c^{(i)} + \mathbf{u}_c^{(i)}\|_2):$$

18        
$$\mathbf{a}_c^{(i)} = \begin{cases} \nu^{(i)}(\mathbf{g}_c^{(i)} + \mathbf{u}_c^{(i)}) & \nu^{(i)} > 0 \\ 0 & \text{otherwise} \end{cases}$$

19       Update  $\mathbf{U}_c^{(k+1)}$  with (3.27) and  $\rho$ ;
20     Update  $\mathbf{D}$  (3.31) with constrained MOD;
21 return  $\beta, \mathbf{D}$ 
```

$$\boldsymbol{\gamma}^{(k+1)} = \underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \frac{1}{2} \|\boldsymbol{\gamma} \mathbf{H} - \mathbf{D} \mathbf{A}\|_F^2 + \frac{\lambda_1}{2} \operatorname{tr}[\boldsymbol{\gamma}(\mathbf{S}_w - \mathbf{S}_b) \boldsymbol{\gamma}^T] + \frac{\rho}{2} \|\boldsymbol{\gamma} - \boldsymbol{\beta}^{(k)} + \mathbf{u}^{(k)}\|_F^2 \quad (3.17)$$

$$\boldsymbol{\beta}^{(k+1)} = \underset{\boldsymbol{\beta} \boldsymbol{\beta}^T = \mathbf{I}}{\operatorname{argmin}} \|\boldsymbol{\gamma}^{(k+1)} - \boldsymbol{\beta} + \mathbf{u}^{(k)}\|_F^2 \quad (3.18)$$

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \boldsymbol{\gamma}^{(k+1)} - \boldsymbol{\beta}^{(k+1)} \quad (3.19)$$

Subproblem (3.17) has a closed form solution:

$$\boldsymbol{\gamma}^{(k+1)} = \left(\mathbf{D} \mathbf{A} \mathbf{H}^T + \rho(\boldsymbol{\beta}^{(k)} - \mathbf{u}^{(k)}) \right) \left(\mathbf{H} \mathbf{H}^T + \lambda_1(\mathbf{S}_w - \mathbf{S}_b) + \rho \mathbf{I} \right)^{-1} \quad (3.20)$$

Solving subproblem (3.18) is based on the following lemma:

Lemma 3.1. *The solution of*

$$\min_{\mathbf{X}} \|\mathbf{X} - \mathbf{Y}\|_F^2 \quad s.t. \quad \mathbf{X} \mathbf{X}^T = \mathbf{I}$$

is given by

$$\mathbf{X} = \mathbf{U} \mathbf{V}^T$$

where $\mathbf{Y} = \mathbf{U} \boldsymbol{\Sigma} \mathbf{V}^T$ is the singular value decomposition (SVD) of $\mathbf{Y}$. $\square$

Therefore the solution to (3.18) is given by

$$\boldsymbol{\beta}^{(k+1)} = \mathbf{U} \mathbf{V}^T \quad (3.21)$$

where $\mathbf{U} \boldsymbol{\Sigma} \mathbf{V}^T = (\boldsymbol{\gamma}^{(k+1)} + \mathbf{u}^{(k)})$ is the SVD of $\boldsymbol{\gamma}^{(k+1)} + \mathbf{u}^{(k)}$.

3.3.2 Update of $\mathbf{A}$

The objective of $\mathbf{A}$ is further divided into C classes, which are optimized for each class alternately with the other ones fixed. For class c the objective of $\mathbf{A}_c$ is:

$$\mathcal{L}_{\mathbf{A}_c} = \frac{1}{2} \|\boldsymbol{\beta} \mathbf{H}_c - \mathbf{D} \mathbf{A}_c\|_F^2 + \|\mathbf{W}_c \mathbf{A}_c\|_{1,2}. \quad (3.22)$$

Applying ADMM formulates the problem as follows:

$$\min_{\mathbf{G}_c, \mathbf{A}_c} \frac{1}{2} \|\beta \mathbf{H}_c - \mathbf{D} \mathbf{G}_c\|_F^2 + \|\mathbf{W}_c \mathbf{A}_c\|_{1,2} \quad \text{s.t.} \quad \mathbf{G}_c = \mathbf{A}_c \quad (3.23)$$

where $\mathbf{G}_c \in \mathbb{R}^{m \times N_c}$. The scaled augmented Lagrangian is

$$\min_{\mathbf{G}_c, \mathbf{A}_c, \mathbf{U}_c} \frac{1}{2} \|\beta \mathbf{H}_c - \mathbf{D} \mathbf{G}_c\|_F^2 + \|\mathbf{W}_c \mathbf{A}_c\|_{1,2} + \frac{\rho}{2} \|\mathbf{G}_c - \mathbf{A}_c + \mathbf{U}_c\|_F^2, \quad (3.24)$$

where $\mathbf{U}_c \in \mathbb{R}^{m \times N_c}$ is the dual variable and $\rho > 0$. At step $k + 1$:

$$\mathbf{G}_c^{(k+1)} = \underset{\mathbf{G}_c}{\operatorname{argmin}} \frac{\rho}{2} \|\mathbf{G}_c - \mathbf{A}_c^{(k)} + \mathbf{U}_c^{(k)}\|_F^2 + \frac{1}{2} \|\beta \mathbf{H}_c - \mathbf{D} \mathbf{G}_c\|_F^2 \quad (3.25)$$

$$\mathbf{A}_c^{(k+1)} = \underset{\mathbf{A}_c}{\operatorname{argmin}} \frac{\rho}{2} \|\mathbf{G}_c^{(k+1)} - \mathbf{A}_c + \mathbf{U}_c^{(k)}\|_F^2 + \|\mathbf{W}_c \mathbf{A}_c\|_{1,2} \quad (3.26)$$

$$\mathbf{U}_c^{(k+1)} = \mathbf{U}_c^{(k)} + \mathbf{G}_c^{(k+1)} - \mathbf{A}_c^{(k+1)} \quad (3.27)$$

Similarly (3.25) has a closed form solution:

$$\mathbf{G}_c^{(k+1)} = \left(\mathbf{D}^T \mathbf{D} + \rho \mathbf{I} \right)^{-1} \left(\mathbf{D}^T \beta \mathbf{H}_c + \rho (\mathbf{A}_c^{(k)} - \mathbf{U}_c^{(k)}) \right). \quad (3.28)$$

For (3.26), since $\mathbf{W}_c$ is diagonal and fixed (since $\mathbf{A}_{-c}$ is fixed), the problem can be solved in a row-wise fashion:

$$\mathbf{a}_c^{(i)} = \mathcal{P}_{[\mathbf{W}_c]_{ii}/\rho} \left(\mathbf{q}_c^{(i)} \right) = \max \left(\|\mathbf{q}_c^{(i)}\|_2 - \frac{[\mathbf{W}_c]_{ii}}{\rho}, 0 \right) \cdot \frac{\mathbf{q}_c^{(i)}}{\|\mathbf{q}_c^{(i)}\|_2}, \quad (3.29)$$

where $\mathbf{g}_c^{(i)}$, $\mathbf{a}_c^{(i)}$ and $\mathbf{u}_c^{(i)}$ are the i -th row of $\mathbf{G}_c^{(k+1)}$, $\mathbf{A}_c^{(k+1)}$ and $\mathbf{U}_c^{(k)}$ respectively, and $\mathbf{q}_c^{(i)} = \mathbf{g}_c^{(i)} + \mathbf{u}_c^{(i)}$. $\mathcal{P}_\lambda(\mathbf{x})$ is the *proximal operator* [159] of the ℓ_2 norm $\|\cdot\|_2$:

$$\mathcal{P}_\lambda(\mathbf{x}) \triangleq \max(1 - \lambda/\|\mathbf{x}\|_2, 0) \mathbf{x}. \quad (3.30)$$

3.3.3 Update of $\mathbf{D}$

Finally, the dictionary $\mathbf{D}$ is minimized via

$$\mathcal{L}_{\mathbf{D}} = \frac{1}{2} \|\beta \mathbf{H} - \mathbf{D} \mathbf{A}\|_F^2 \quad \text{s.t.} \quad \|\mathbf{d}_i\| = 1, \forall i. \quad (3.31)$$

Requiring columns of the dictionary to be unit vectors avoids the magnitude of the coding vectors to be arbitrary small. (3.31) is a quadratic programming problem which can be solved efficiently using Method of Optimal Direction (MOD) [77] or K-SVD [79]. In this chapter MOD with constraint $\|\mathbf{d}_i\|_2 = 1$ is used.

3.3.4 Computational Complexity

The computation of ELM-DDL is dominated by line 7, 8, 12 and 15 in Algorithm 2. Given $\beta \in \mathbb{R}^{n \times L}$, $\mathbf{H} \in \mathbb{R}^{L \times N}$, $\mathbf{A} \in \mathbb{R}^{m \times N}$, $\mathbf{D} \in \mathbb{R}^{n \times m}$, the computational complexity is discussed for each part based on a single loop of both the ADMM iterations and dictionary learning. In line 7, computing $\mathbf{D} \mathbf{A} \mathbf{H}^T$ requires $\mathcal{O}(Nmn + LNn)$, computing $\mathbf{H} \mathbf{H}^T$ requires $\mathcal{O}(L^2N)$ (which can be computed once since $\mathbf{H}$ is fixed), and the inverse requires $\mathcal{O}(L^3)$. In line 8, SVD requires $\mathcal{O}(Ln^2)$ both for decomposition and multiplication. In line 12, the inverse can be computed once for all C classes, leading to $\mathcal{O}(m^2n + m^3)$, and the multiplication $\mathbf{D}^T \beta \mathbf{H}_c$ requires $\mathcal{O}(Lmn + LN_c m)$, in total $\mathcal{O}(Lmn + LNm)$ for all classes (computing $\mathbf{D}^T \beta$ can be shared). In line 15, solving $\mathbf{D}$ amounts to solving the least square problem (3.31), which results in $\mathcal{O}(Nm^2 + Nmn + m^3)$.

To summarize, computing $\mathbf{H} \mathbf{H}^T$ requires $\mathcal{O}(L^2N)$, and others require $\mathcal{O}(L^3 + (m + n)(LN + Ln + Nm + m^2))$ for each iteration. In practice, the number of hidden neuron is usually set to be large, thus the major part of computations will fall onto line 7 in Algorithm 2, which is $\mathcal{O}(L^3)$ for each iteration.

(A) Extend Yale B

(B) AR Face

(c) COIL-100

(D) Fashion-MNIST

FIGURE 3.1: Illustrations of the benchmark datasets: Extended Yale B, AR Face, COIL-100 and Fashion-MNIST.

TABLE 3.1: Specifications of benchmark datasets for image classification.

Datasets	No. Samples	Image Size
Extended Yale B	2414	192×168
AR Face	2600	165×120
COIL-100	7200	32×32
Fashion-MNIST	70000	28×28

3.4 Experiments

In this section, the proposed method is evaluated on benchmark datasets which are illustrated in Figure 3.1. The designs of ELM embedding, the weighted $\ell_{1,2}$ norm and the heuristic SRC classifier are also empirically validated. The algorithm has four hyperparameters: λ_1 , λ_2 , λ_3 and λ_4 . For each dataset, cross validation is applied to select the best parameters. After that the algorithm is run 10 times with different data partitions and initializations and report the average accuracy.

The proposed method is compared with SRC¹ [10], K-SVD [79], D-KSVD [20], LC-KSVD [18], DLSI [46], FDDL [47], SVGDL [24] and SDDL [25]. SRC is non-parametric and applies the training samples directly as the dictionary. D-KSVD and LC-KSVD are based on the K-SVD algorithms with different discriminative regularizations. DLSI adds incoherence constraints on sub-dictionaries of each class. FDDL employs Fisher discrimination and SVGDL further generalizes this idea with a support vector formulation. SDDL applies the $\ell_{1,2}$ norm as regularizations to constrain the supports of the coding vectors. For the Extended Yale B dataset and AR Face dataset, results in the literatures are reported since the experimental setting is the same. For the COIL-100 and Fashion-MNIST dataset, each algorithm is fine-tuned for the best parameters and the average results are reported over 10 runs.

3.4.1 Performance and Comparisons

This section makes evaluations on the following datasets: the Extended Yale B dataset [211], [212] and the AR face dataset [213], [214] for face recognition, and COIL-100 [215] and Fashion-MNIST [216] for object classification. Specifications of

¹SRC uses all training samples as the dictionary.

FIGURE 3.2: Convergence of objective values of ELM-DDL on each benchmark dataset for 60 iterations.

each dataset are given in Table 5.3, and illustrations of them are given in Figure 3.1. For each dataset, the objective value of ELM-DDL in (3.8) is computed with respect to each iteration. The results are given in Figure 3.2, which shows that ELM-DDL has a smooth convergence behavior across all benchmarks. Most of them converge to a small value within 10 to 20 iterations.

The accuracies of the proposed ELM-DDL as well as comparison methods on benchmarks in Table 3.1 are summarized in Table 3.2.

3.4.1.1 The Extended Yale B Dataset

The Extended Yale B dataset [211] has 2414 frontal face images collected from 38 people, each containing about 64 images. This dataset is challenging because the images are collected under different lighting, facial expressions and occlusions. The illustrations of the dataset can be found in Figure 3.1a. Following common experimental settings [18], the images are normalized and cropped into 192×168 pixels [212]. Then they are encoded with random-face features [10] with an output dimension $n = 504$. Half of the images for each person are randomly selected for training and the rest for testing. The number of dictionary atoms is set to be $m = 570$, corresponding to 15 atoms per person.

For the proposed method, the hidden dimension of ELM is $L = 2000$. The nonlinear random mapping then is projected to an output space with dimension $n = 504$, which is aligned with the random-face features. The regularization parameters are chosen as $\lambda_1 = 0.01$, $\lambda_2 = 0.01$, $\lambda_3 = 0.01$ and $\lambda_4 = 0.001$ for the Extended Yale B dataset.

The experimental results are summarized in column 2 of Table 3.2. It can be seen that the proposed method achieves the best result among all compared methods. Also note that the margin between the proposed method and SDDL is not large (about 0.56%). However, SDDL suppresses the overlap support of different classes via ℓ_0 minimization, which is approximated with the iterative reweighted ℓ_2 minimization. This approach is unstable and is sensitive to a regularization factor ϵ . The proposed method avoids ℓ_0 minimization and is stable among multiple runs (with standard deviation of 0.23% over 10 runs). On the other hand, the $\ell_{1,2}$ based methods (SDDL and ELM-DDL) achieve large accuracy gains over other methods.

TABLE 3.2: Recognition results (%) on four benchmark datasets: the Extended Yale B, AR Face, COIL-100 and Fashion-MNIST. All results are measured with the best hyperparameter settings. Final accuracy is averaged over 10 runs.

Methods	Extended Yale B	AR Face	COIL-100	Fashion-MNIST
SRC [10]	97.20	97.50	85.11	83.43
K-SVD [79]	93.10	86.30	71.50	80.08
D-KSVD [20]	94.10	88.80	83.35	80.34
LC-KSVD [18]	95.00	93.70	83.77	80.39
FDDL [47]	94.92	94.10	83.23	82.68
DLSI [46]	94.64	89.80	86.50	83.94
SVGD [24]	95.70	96.00	75.88	80.65
SDDL [25]	98.50	98.00	86.13	80.13
ELM-DDL	99.06	98.95	86.57	84.32

The proposed method has over 1.8% accuracy gain over SRC, which has an unfair advantage of larger dictionary size. This demonstrates the effectiveness of the proposed discriminative sparse regularization.

In addition, an illustration is given in Figure 3.3 showing example faces (from the test split) as well as the corresponding coding vectors obtained with ELM-DDL. Although each image corresponds to different lighting conditions, the proposed algorithm gives high response from the coding vectors on the same group of atoms, even though the dictionary is shared among all categories.

3.4.1.2 The AR Face Dataset

The AR Face dataset [213] contains over 4000 color images from 126 people. It can be observed from Figure 3.1b that they include large variations such as lightings, expressions and wearables (sun glasses and scarf). Following the standard experimental setting, the experiment uses a subset which contains 2600 images from 50 male and 50 female. There are 26 facial images of size 165×120 for each person, which are projected to $n = 540$ via random-face features [10]. For each person, 20 images are randomly selected for training and the other 6 for testing. The number of dictionary atoms is set to be $m = 500$.

The hidden dimension is $L = 2000$ for the proposed ELM-DDL. The output dimension is set to be $n = 540$. The regularization parameters are chosen as $\lambda_1 = 0.01$, $\lambda_2 = 0.001$, $\lambda_3 = 0.06$, $\lambda_4 = 0.01$. The experimental results is shown at column 3 of Table 3.2. The proposed ELM-DDL outperforms all other methods. Also, the SDDL and ELM-DDL algorithms achieve large margins against other dictionary learning methods, showing the effectiveness of support discrimination.

3.4.1.3 The COIL-100 Dataset

This part evaluates the performance of the proposed approach on object recognition with the COIL-100 dataset [215]. The COIL-100 dataset contains 7200 images belonging to 100 objects. Each object has 72 images which are captured from different views with clean background. An example of the dataset can be found in Figure 3.1c. Following standard practice, 10 out of 72 images from each object are randomly selected for training and the rest for testing. Each image is resized to

FIGURE 3.3: Example faces from the Extended Yale B dataset. The first row corresponds to test images from the same person with different lighting conditions. The second row shows sparse representations of each image with a small $\lambda_4 = 0.005$. The dictionary is learned with $\lambda_2 = \lambda_3 = 0.01$. The last row is computed with a dictionary with $\lambda_2 = 0.01$ and $\lambda_3 = 0$.

32×32 pixels and converted to grayscale. For comparison methods, PCA is used to reduce the dimension to 512. All 100 object categories are included for training and evaluation, and the number of dictionary atoms is set to be 1000.

For the proposed ELM-DDL, the hidden dimension is $L = 2000$. The hidden features are projected to an output space with equal dimension with the PCA output, i.e. $n = 512$. The regularization parameters are chosen as $\lambda_1 = 0.1$, $\lambda_2 = 0.002$, $\lambda_3 = 0.03$, $\lambda_4 = 0.4$, which are determined via cross validation.

The performance is shown on column 4 of Table 3.2. The proposed method has outperformed the others on COIL-100. The naive K-SVD has the worst performance while the discriminative variants, i.e. D-KSVD and LC-KSVD, achieve significantly high results. This shows the importance of discrimination for dictionary learning. SRC gives superior performance over the K-SVD based methods, despite that SRC has the same number of samples as the dictionary atoms. One possible reason is that the COIL-100 dataset has a clean background with a regular structure, so that a query image can be well constructed via only a few others. The DLSI algorithm also gives a competitive performance which is close to ELM-DDL, where they only applies structural incoherence over the dictionary as regularizations. This means that the proposed algorithm can still be improved with additional constraints on the dictionary atoms.

3.4.1.4 The Fashion-MNIST Dataset

Another popular dataset for object recognition is the Fashion-MNIST dataset [216], which aims to be a drop-in replacement for the classical MNIST dataset [30]. The Fashion-MNIST dataset contains 60000 training images and 10000 testing images. Each of them is a 28×28 grayscale image belonging to one of ten categories of fashion dresses. An illustration can be found in Figure 3.1d.

The performance is evaluated under a restricted experimental setting. Specifically, 600 images per class are randomly selected for training and the whole test set is used for evaluation. For comparison methods, the images are projected to a 512-dim subspace with PCA. The number of dictionary atoms is set to be 300, which is small compared to a total size of 6000 training samples.

For the proposed method, the hidden dimension is $L = 2000$ and the projected output dimension to be $n = 512$. The regularization parameters are $\lambda_1 = 0.1$, $\lambda_2 = 0.05$, $\lambda_3 = 0.5$, $\lambda_4 = 0.01$.

The experimental results can be found in the last column of Table 3.2. The proposed ELM-DDL gives the highest accuracy. Specifically ELM-DDL performs better than the SRC method, which has an unfair advantage of applying 6000 training samples while others having only 300 atoms. Note also that SDDL does not yield a superior result against the other methods as it does on previous datasets. One possible reason is that the support size (i.e. the number of dictionary atoms) is not large enough for SDDL to describe the test partition.

3.4.2 Visualizations of Representations

To demonstrate the effectiveness of ELM-DDL, the learned representations from solving (3.10) are visualized, The visualizations are compared with those of ELM output results $\beta\mathbf{H}$ and of original data points. Specifically, test samples $\mathbf{X}$ of the first ten classes from the Extended Yale B dataset are used and are encoded via ELM in (3.9) to yield the representations $\beta\mathbf{H}$ with learned β . The ELM representations are further used to compute the sparse representations $\mathbf{A}$ via (3.10) with learned dictionary $\mathbf{D}$. Finally, the original samples, ELM outputs and sparse coding vectors are embedded into two dimensions using t-SNE [217].

The visualization results are given in Figure 3.4. The original data points are cluttered with mixed structures. When encoded with random mappings and projected with learned β however, the representations become more separable. The shows that the learned projection with MMC constraints is able to generate discriminative outputs, even though ELM is not directly used for the classification task. Some data points from the same classes under such representations are still far from each other (note the orange data points in Figure 3.4b), and samples from different classes may still mix with each other (note the light blue and light green ones).

The outputs of sparse coding remedy these cases and exhibit clear clusters with highly separable representations, as shown in Figure 3.4c. This means that 1) the learned dictionary $\mathbf{D}$ can produce highly discriminative sparse representations

FIGURE 3.4: t-SNE visualizations of first ten persons of the Extended Yale B dataset. Data points of the same color belong to the same class. (a) corresponds to the original data points, (b) corresponds to ELM outputs $\beta\mathbf{H}$ from (3.9) with learned β , and (c) is the sparse coding vectors from (3.10) with learned $\mathbf{D}$.

TABLE 3.3: Test accuracies (%) of ELM-DDL under different experimental settings. “*Baseline*” corresponds to the best result in Table 3.2. “*w.o. ELM*” refers to the case where no ELM embedding is used. “*w.o. MMC*” means training the model with $\lambda_4 = 0$. “*Lin. Pred.*” is achieved by substitution of a linear predictor for SRC during testing. All results are averaged over 10 experiments.

Settings	Extended Yale B	AR Face	COIL-100	Fashion-MNIST
<i>Baseline</i>	99.06	98.95	86.57	84.32
<i>w.o. ELM</i>	97.12	97.02	85.66	81.44
<i>w.o. MMC</i>	98.77	98.28	85.41	84.13
<i>Lin. Pred.</i>	98.35	98.70	86.48	81.40

even if no high-level label information is used to constrain its atoms, and 2) the class specific weighted $\ell_{1,2}$ norm (3.6) is effective for discriminative learning. The visualization results thus empirically justify the design of the proposed ELM-DDL approach for image classification.

3.4.3 The Effect of ELM Embedding

Since ELM is an important component in the proposed framework, this part investigates the importance of the learned projection β and the ELM embedding $\beta\mathbf{H}$.

3.4.3.1 Dictionary Learning without ELM

The proposed method *without* ELM embedding corresponds to a simple support discriminative model:

$$\min_{\mathbf{A}, \mathbf{D}} \frac{1}{2} \|\mathbf{X} - \mathbf{DA}\|_F^2 + \sum_{c=1}^C \|\mathbf{W}_c \mathbf{A}_c\|_{1,2}. \quad (3.32)$$

The performance of this setting is given in Table 3.3 (see the row “*w.o. ELM*”). It can be observed that without the ELM embedding, the accuracies on each dataset is decreased by 2% on average compared to the baseline results. This proves that the simple random encoding and projection is important and gives a more discriminative feature representation.

3.4.3.2 Dictionary Learning without MMC

As further explorations, the performance is evaluated under the scenario that ELM is applied *without* the MMC criterion. This can be simply achieved by setting $\lambda_1 = 0$. This setting investigates the importance of nonlinear embedding compared to the sensitivity of class separability. The subproblem with respect to β now becomes

$$\min_{\beta} \|\beta \mathbf{H} - \mathbf{D}\mathbf{A}\|_F^2 \quad \text{s.t.} \quad \beta \beta^T = \mathbf{I}. \quad (3.33)$$

This means that at each step β is minimized to reconstruct the dictionary and the coding vectors from the previous step. The performance of this setting on each benchmark dataset is given in Table 3.3 (see the row “*w.o. MMC*”). It can be seen that the performance without MMC is slightly lower than the baseline. However, the decrease is not as significant as removing ELM. Also the performance is competitive among all dictionary learning methods. This means that ELM plays a more important role in dictionary learning. The discriminative regularization, on the other hand, provides moderate improvement over the single ELM performance.

3.4.3.3 The Effect of Number of Hidden Neurons

The performance of ELM-DDL with respect to different number of hidden neurons (L) on the Extended Yale B dataset is demonstrated in Figure 3.5. It can be seen that when L takes a small value, the performance drops significantly to around 92%, which is far behind the case of not applying ELM. This means that for a small number of neurons, randomness dominates the features and the discrimination is overwhelmed. On the other hand, a relatively large value of L has guaranteed the discriminative performance of ELM, and it remains stable with increase of number of neurons. For all experiments a sufficiently large number of neurons (i.e. $L = 2000$) is used to guarantee the performance.

FIGURE 3.5: Accuracy of ELM-DDL on Extended Yale B with different number of hidden neurons (L).

3.4.4 The Effect of λ_2 and λ_3

In this part the importance of a proper choice of λ_2 and λ_3 for the proposed ELM-DDL is validated, which controls the sparsity of the coding vectors. A simple illustration on the Extended Yale B dataset is given in Figure 3.3. A proper combination of λ_2 and λ_3 can constrain the response on a small group of atoms. On the other hand, when setting $\lambda_3 = 0$ as in the third row of Figure 3.3, the codes are less concentrated and exhibit more fluctuations.

More detailed results are given in Figure 3.6 which shows the accuracy values with respect to different choices of λ_2 and λ_3 on each benchmark dataset. The algorithm is more sensitive to λ_2 which controls sparsity within the same class. On the other hand, when a suitable value of λ_3 is selected, the proposed method should be able to yield a reasonable performance. Also, for Extended Yale B, AR Face and COIL-100, the parameters are more stable to parameter choices while for Fashion-MNIST, the parameters should be selected with greater care.

3.4.5 The Effect of SRC Predictor

Finally the effect of the sparse representation classifier (SRC) is empirically investigated for classification [10]. The usual approach for coding vector classification

is to learn a simple linear predictor $\mathbf{C}$, i.e. with the obtained coding vectors $\mathbf{A}$ in Section 3.3.2, the classifier weight $\mathbf{C} \in \mathbb{R}^{C \times m}$ is learned via

$$\min_{\mathbf{C}} \|\mathbf{CA} - \mathbf{Y}\|_F^2 + \eta \|\mathbf{C}\|_F^2 \quad (3.34)$$

where $\mathbf{Y} \in \mathbb{R}^{C \times N}$ is the one-hot encoding of the labels of each training sample. The solution is

$$\mathbf{C}^T = \left(\mathbf{AA}^T + \eta \mathbf{I} \right)^{-1} \mathbf{AY}^T. \quad (3.35)$$

Although the linear predictor is favored by existing methods such as [18], [25], SRC is preferable since it is nonparametric (it does not requires tuning η) and gives superior performance in many cases [10]. Moreover, the coding vectors from training set are learned with the class specific weighted $\ell_{1,2}$ norm while the test codes $\mathbf{a}_{\text{test}}$ in (3.10) are computed using LASSO with ℓ_1 norm. Therefore, the linear predictor $\mathbf{C}$ may not be suited for the two dissimilar domains, while SRC focuses on reconstruction error and is more suitable.

The performance is compared between applying linear predictor (3.34) and the SRC classifier (3.13) to the learned coding vectors on each benchmark dataset. The results are shown in Table 3.3 (in the row “*Lin. Pred.*”). Overall, the use of SRC improves performance. For some dataset such as AR Face and COIL-100, SRC does not give significant boost (only about 0.1%). On the other hand, SRC is critical to the Fashion-MNIST dataset, which improves the performance for 2.9%.

3.4.6 Comparisons with Deep Learning based Approaches

In Section 2.2.5 deep learning (DL) based approaches are reviewed, and their connections and differences with sparse representation based methods are discussed. As said, deep learning repeatedly achieves state-of-the-art results on standard CV tasks, advancing sparsity-based methods by a large margin (check for performance gap between [126] and [218] for example). Here a brief comparison of the proposed ELM-DDL with deep learning based approaches are presented for completeness. The comparison is not meant for striving for state-of-the-art results on Table 3.1

TABLE 3.4: Comparisons of the proposed ELM-DDL framework with deep learning based approaches: WTA-CAE and VGG-Small.

Algorithm	Extended Yale B	AR Face	COIL-100	Fashion-MNIST
<i>ELM-DDL</i>	99.06	98.95	86.57	84.32
<i>WTA-CAE</i> [219]	88.23	98.67	85.74	85.18
<i>VGG-Small</i> [102]	98.71	99.58	95.96	88.32

by exploiting all possible configurations and tricks in modern deep learning network training, but rather for checking the performance capabilities given a *sane* DL settings, and for understanding how far the one-layer sparse dictionary learning model can achieve compared to the modern multilayer perceptron. Specifically, the following two DL based configurations are adopted:

- Winner-take-all convolutional autoencoders (WTA-CAE) [219]. This is a CAE with 3 convolutional layers for encoding and 1 layer for decoding. Sparsity is injected onto the hidden representations via a “winner-take-all” strategy. In this configuration, the WTA-CAE is trained with mean square loss to learn features on each dataset, and an SVM [54] is trained on top of learned features for classification. Note that the structure of WTA-CAE is particularly similar to the proposed ELM-DDL framework, where the ELM step can be seen as a two-layer encoder (one random and one adaptive), and the dictionary learning step can be seen as decoding with sparsity constraints.
- A tailored 3-layer VGG network (VGG-Small) [102]. This is the classical end-to-end setting suitable for learning datasets with a small size. The smaller size of the network compared with a standard 16-layer one prevents it from overfitting, and additionally batch normalization [220] is inserted after convolutional layers for better convergence, which is a common practice. More complicated designs, e.g. residual connections [103], SE blocks [221], etc., are not adopted since they are not very beneficial to datasets in Table 3.1. The network is trained end-to-end with cross-entropy loss on each dataset for classification.

Results on each dataset are summarized in Table 3.4. The proposed ELM-DDL is comparable with WTA-CAE with better performance on Extended Yale B, AR Face and COIL-100. This demonstrates that ELM-DDL captures structural information

of the training data as discussed in Section 3.2.3, considering the fact that ELM-DDL has fewer tunable parameters than WTA-CAE, and the feature space of ELM-DDL for classification (i.e. dictionary size) is much smaller than that of WTA-CAE. On the other hand, the state-of-the-art VGG-Small network achieves best accuracy results among 3 out of 4 benchmark datasets, which demonstrates the power of deep neural networks. Note that for object recognition datasets, i.e. COIL-100 and Fashion-MNIST, where class differences are notable, the performance gap between VGG-Small and the proposed approach is prominent, while for face recognition datasets i.e. Extended Yale B and AR Face, where differences between classes are subtle, the results between ELM-DDL and VGG-Small are comparable. Especially on low data realm (note that for the Extended Yale B dataset only half samples are used for training as described in Section 3.4.1.1), ELM-DDL actually achieves the best results among three methods. This suggests that sparsity-based methods are still competitive at capturing minute difference in dataset samples even if only a few can be accessed.

3.5 Summary

This chapter proposes ELM embedded discriminative dictionary learning (ELM-DDL) as a novel combination of ELM embedding with a new support discriminative dictionary learning algorithm. It addresses the topic of utilizing a structured sparse regularization for producing a discriminative dictionary and enhancing the performance of dictionary learning for image classification. It also demonstrates the representation capabilities of ELM features. To achieve these objectives, Section 3.2 proposes a dictionary learning scheme that applies a weighted class specific $\ell_{1,2}$ norm that guarantees minimal overlap among coding vectors from different classes and similarity within the same class. The dictionary is learned a nonlinear embedding which efficiently encodes random features on high dimensional space and projects them onto output space, with an MMC criterion to enforce class separation. A modified SRC classifier is applied for final classification. Section 3.3 describes detailed optimization steps for ELM-DDL. Section 3.4 demonstrates the effectiveness of ELM-DDL by extensive experiments on face recognition and object classification benchmark datasets, where ELM-DDL achieves better results than state-of-the-art dictionary learning algorithms.

FIGURE 3.6: Average accuracies of ELM-DDL for different choices of λ_2 and λ_3 for each dataset. The x and y axis are in log scale.

Chapter 4

Clustering via Adaptive and Locality-constrained Graph Learning and Unsupervised ELM

Chapter 4 introduces a simple and effective graph learning method based on adaptive graph regularizations, where the structures of the data similarity matrix are exploited via the sparse and distance-regularized locality-constrained linear coding. Section 4.1 reviews existing graph learning approaches and their limitations. To remedy existing problems, Section 4.2 proposes a novel graph learning method and utilizes unsupervised ELM for clustering, and Section 4.3 describes detailed optimization procedures. Section 4.4 performs extensive experiments over a wide range of benchmark datasets and compares the proposed method to a large number of existing graph learning approaches. The contents of this chapter are based on the following paper:

- Yijie Zeng, Jichao Chen, Yue Li, Yuanyuan Qing, and Guang-Bin Huang, “Clustering via Adaptive and Locality-constrained Graph Learning and Unsupervised ELM,” *Neurocomputing*, Mar. 2020. DOI: 10.1016/j.neucom.2020.03.045.

4.1 Background and Motivations

Clustering refers to the task of grouping different entities based on commonly shared properties. It is a straightforward method for data mining and statistical data analysis since clustering provides an exploratory view of data. The most important component for clustering is the pair-wise similarities of the underlying data. Depending on the structures of the underlying data as well as the nature of the tasks at hand, different similarity measures can be defined that lead to different clustering results. Various clustering methods have been proposed, such as K-means clustering [222], spectral clustering [223], [224], hierarchical clustering [225], density-based clustering [226], etc., to address different aspects of the clustering problem. They have also been widely applied in various fields including image segmentation [56], bioinformatics [227], outlier detection [228], natural language processing [229], etc.

Data can be represented as a *graph* where each sample is a node and similarities are measured by the connectivities between pairs of nodes. Therefore, learning or constructing an effective graph that captures the intrinsic relationship among the data is an important tool for solving the clustering problem. Technically, one can use a *similarity matrix* to describe a graph. The nonzero entries within the matrix indicate connected components and their values measure levels of connectivities. After defining the similarity matrix, a dedicated clustering algorithm can be applied to yield the class predictions. For example, spectral clustering [223] computes the eigenvectors of the Laplacian of the similarity matrix to form a low-dimensional data representations, on which the class indicators can be calculated.

One way to determine the similarity matrix is the distance between samples where smaller distances indicate higher resemblance. For graph learning specifically, each entry in the similarity matrix can be set as the pair-wise distance of connect samples. Furthermore, it is beneficial to build a sparse graph where only significant connectivities are allowed in order to yield a more efficient data representation. This inspires the construction of the k -NN graph where each node is connected only to its k nearest neighbors, or the ϵ -graph where each node is connected to neighbors within distance ϵ [224].

Instead of considering pair-wise distance based similarities, one can also adopt a *synthesis* or *reconstruction* view and represent each sample as a linear combination

of a few similar ones. The effectiveness of this approach lies in the fact that data can be efficiently reconstructed by those from the same subspace. Locally linear embedding [230] represents each sample as a linear combination of its nearest neighbors. The representations are used to learn a low-dimensional embedding for dimension reduction. Instead of choosing nearest neighbors, sparse subspace clustering (SSC) [57] enforces ℓ_1 constraint on the reconstruction coefficients to yield sparse vectors. The representations are normalized to construct a data graph for clustering.

SSC constructs an efficient ℓ_1 graph that is both sparse and capable of partitioning data from disjoint subspaces. However, as discussed in [231], the ℓ_1 graph also enforces sparsity *within* the connectivities of the same cluster and breaks the correct clustering structures. An alternative approach, such as LLE, is to resort to representations on a local scale and to exploit information at nearest neighbors. However, LLE does not consider a direct similarity metric (i.e. distances) among different samples, and is more effective for nonlinear dimension reduction rather than graph learning. Wang et al. [63] proposed Locality-constrained Linear Coding (LLC) as an effective method for unsupervised feature learning and dictionary learning. LLC reconstructs each sample by similar ones, and it additionally regularizes the obtained descriptor with pair-wise distances, which is a favorable characteristic for learning a data similarity matrix. This chapter investigates the generalization of LLC to graph learning, which is achieved by setting the dictionary as collections of all samples. The LLC formulation utilizes local representation capability through k -NN graph (as LLE does) and incorporating similarity information via local distances.

The ℓ_1 graph or the k -NN graph optimizes each column of the graph independently. Likewise, features of LLC are aggregated from each samples to construct the graph directly. On the other hand, when these high level feature descriptors are regarded as *embeddings* of a data graph, they are often assumed to be *smooth* on some low-dimensional data manifold and should be close to each other if the original samples are similar. The smoothness assumption can be modeled through manifold regularizations [198] on the learned representations, which has been explored also in [21] for sparse coding, [232] for graph representations, in [233] for subspace clustering, and is widely used for unsupervised tasks as discussed in Section 2.4.3. This chapter investigates the manifold regularizations via a *predefined*

graph on the learned LLC features to incorporate prior beliefs on the underlying data distribution and to model the intrinsic similarities among samples.

Traditional graph construction methods depend on observations of data distributions to build a fixed similarity matrix. This graph is independent of the downstream task, e.g. clustering, which is performed as a separated approach. Recently Nie et al. [234] propose to *adaptively* update the similarity matrix with a rank constraint on its Laplacian to incorporate the cluster information simultaneously when learning the graph. When the rank constraint is satisfied, data points are automatically partitioned into c clusters without resorting to discretization methods for class assignments. Adaptive graph regularization is a powerful approach and can be combined with other methods to learn better graphs. In [235] the data graph is learned to match an initial similarity matrix with adaptive regularizations. In [236] the adaptive graph is also constrained by the low-rank representation (LRR) model [237]. In [238] the adaptive regularization is combined with spectral embedded representations [239] for more discriminative data representations. In this chapter, adaptive graph regularizations are utilized to learn a data graph with clearer structures.

It has been shown in Chapter 3 that ELM provides discriminative features for dictionary learning and supervised image classification. The random mapping can provide more discriminative embeddings than the original features. Chapter 2.4.3 also discusses the applications of ELM for unsupervised learning and in particular clustering [173], [175], [176], [240], [241]. Specifically, in unsupervised ELM (US-ELM) [173], feature embedding is learned on ELM feature space rather than on original samples for spectral clustering. To utilize such characteristics, the use of US-ELM as the clustering method after learning the data graph is investigated in this chapter.

4.1.1 Contributions

This chapter makes the following contributions:

- It generalizes locality-constrained linear coding (LLC) as feature representation approach for graph learning. The representation enforces a locally sparse

connection on the data graph that exhibits high discrimination power and is easy to optimize.

- It shows that it is possible to impose a rank constraint on the Laplacian of the *localized* data graph so that the connected components match the predefined class number. The graph is *learned* adaptively to exhibit clearer clustering structures.
- It proposes to regularize the graph components as high-level feature representations via manifold regularizations, which enforce smoothness among descriptors of adjacent samples.
- It demonstrates that US-ELM instead of spectral clustering as the clustering approach can empirically improve discriminations and the clustering performance.

4.2 Proposed Method

4.2.1 Graph Learning Formulation

Given N samples $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]$ belonging to c classes, the data graph $\mathbf{S}$ is learned using LLC for graph representation and adaptive neighbor regularization for encoding clustering information as follows:

$$\begin{aligned} \min_{\mathbf{S}} \quad & \frac{1}{2} \|\mathbf{X} - \mathbf{X}\mathbf{S}\|_F^2 + \frac{\eta}{2} \sum_{i=1}^N \|\mathbf{d}_{xi} \odot \mathbf{s}_i\|_2^2 \\ \text{s.t.} \quad & \mathbf{s}_{ij} \geq 0, \mathbf{s}_{ii} = 0, \text{rank}(\mathbf{L}_{\mathbf{S}}) = N - c. \end{aligned} \tag{4.1}$$

Here $\mathbf{s}_i \in \mathbb{R}^N$ is the i -th column of $\mathbf{S}$. $\mathbf{d}_{xi} \in \mathbb{R}^N$ is the distance of $\mathbf{x}_i$ to all other samples $\mathbf{x}_j$, $1 \leq j \leq N$. The formulation (4.1) leverages *self-expressiveness* of data samples, i.e. reconstruction of each $\mathbf{x}_i$ via other samples, and distance based regularizations, where the weights between distant samples are suppressed. Additionally, the rank constraint on the Laplacian of $\mathbf{S}$ ensure that the optimal graph has exactly c components so that clustering can be performed effectively.

Solving (4.1) with the rank constraint requires iteratively updating the data graph, which is computationally intensive for dense graph $\mathbf{S}$. Moreover, as [19], [57], [199] have pointed out, a sparse graph can convey structured information for classification and clustering. Therefore, instead of modeling relationship between each pair of samples, each $\mathbf{s}_i$ is nonzero only on k nearest neighbors of $\mathbf{x}_i$. This leads to a k -NN graph with adaptively computed weights as follows. For notational convenience, define $\mathbf{S} = [\hat{\mathbf{s}}_1, \dots, \hat{\mathbf{s}}_N]$ and define $\mathbf{s}_i \in \mathbb{R}^k$ as the k nonzero elements of $\hat{\mathbf{s}}_i$, i.e. $\hat{\mathbf{s}}_{i,i_k} = \mathbf{s}_{ik}$ if $i_k \in \mathcal{N}_i$ and 0 otherwise. The optimization problem is

$$\begin{aligned} \min_{\mathbf{S} \in \mathcal{C}} \quad & \frac{1}{2} \sum_{i=1}^N \|\mathbf{x}_i - \mathbf{X}_i \mathbf{s}_i\|_2^2 + \frac{\eta}{2} \sum_{i=1}^N \|\mathbf{d}_{xi} \odot \mathbf{s}_i\|_2^2 \\ \text{s.t.} \quad & \mathbf{s}_{ij} \geq 0, \text{rank}(\mathbf{L}_\mathbf{S}) = N - c \end{aligned} \quad (4.2)$$

where $\mathbf{X}_i = [\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_k}] \in \mathbb{R}^{n \times k}$ consists of samples from k nearest neighbors of $\mathbf{x}_i$, and

$$\mathbf{d}_{xi} = [\text{dist}(\mathbf{x}_i, \mathbf{x}_{i_1}), \dots, \text{dist}(\mathbf{x}_i, \mathbf{x}_{i_k})]^T \in \mathbb{R}^k$$

denotes the distance of $\mathbf{x}_i$ to its k nearest neighbors. $\mathcal{C}$ is a constraint set on the graph so that the i -th column of $\mathbf{S}$ is nonzero only at $\mathcal{N}_i$. Compared with (4.1), problem (4.2) only needs to optimize $k \ll N$ elements for each column of $\mathbf{S}$ at each iteration which is far more efficient. The formulation also enforces the learned graph to focus more at local representations, while maintaining the global rank constraint on graph Laplacian.

Two modifications are made on (4.2) to improve the graph learning performance. First, the Frobenius norm is replaced with ℓ_2 norm (*not squared*) for data reconstruction which provides more robust representations. Second, manifold regularizations [198] are adopted on columns of $\mathbf{S}$ to improve the smoothness of the representation. Specifically, the weighted sum of pair-wise distances of $\{\mathbf{s}_i\}_{i=1}^N$ are minimized where the weights are defined by a given data graph. This is equivalent to minimizing $\text{tr}[\mathbf{S} \mathbf{L} \mathbf{S}^T]$ where $\mathbf{L}$ is the Laplacian of some predefined graph $\mathbf{S}'$ [199]. The final graph learning problem is

$$\begin{aligned} \min_{\mathbf{S} \in \mathcal{C}} \quad & \sum_{i=1}^N \|\mathbf{x}_i - \mathbf{X}_i \mathbf{s}_i\|_2 + \frac{\eta}{2} \sum_{i=1}^N \|\mathbf{d}_{xi} \odot \mathbf{s}_i\|_2^2 + \frac{\gamma}{2} \text{tr}[\mathbf{S} \mathbf{L} \mathbf{S}^T] \\ \text{s.t.} \quad & \mathbf{s}_{ij} \geq 0, \text{rank}(\mathbf{L}_\mathbf{S}) = N - c. \end{aligned} \quad (4.3)$$

(A) Self-tuning graph

(B) Proposed

FIGURE 4.1: Illustrations of 5-NN graph on UMIST dataset: (a) created with self-tuning strategy [242]; (b) learned by proposed method.

Here η and γ are two trade-off parameters. The proposed graph learning approach (4.3) is termed *Adaptive Locality-constrained Clustering* (ALC). An illustration of differences between a predefined k -NN graph and the graph learned via (4.3) is given in Figure 4.1. The graph learned by ALC is more structured with fewer incorrect connections. In addition, the proposed method has the following advantages compared with other graph learning algorithms:

1. SSC [57] and LRR [237] focus on utilizing *self-expressiveness* of subspaces and exploiting reconstructions of each samples via a few others. While theoretically sound, these methods fail to consider a local connectivity relationship, and they cannot effectively incorporate a class number c when it is known, whereas (4.3) explicitly partitions the graph into c components via the rank constraint, and it enforces a locally-constrained linear coding which exploits local information.
2. The idea of adaptive building the data graph is extensively investigated in CAN [234] and CLR [235], which ALC also achieves via constraining the rank of the Laplacian matrix $\mathbf{L}_S$. In addition, the proposed method incorporates self-expressiveness via reconstructions of each sample, which provides cluster information from a new aspect during graph learning.

Additionally, some remarks can be made regarding the formulation (4.3):

1. Similar to LLE [230], the formulation (4.3) seeks to reconstruct each $\mathbf{x}_i$ by its nearest neighbors. Instead of applying the obtained weights for learning a low-dimensional embedding, the weights themselves are useful as data representations and as components of a data graph, given that the cluster information is encoded via rank constraint.
2. The distance constraint is given as $\sum_i \|\mathbf{d}_{xi} \odot \mathbf{s}_i\|_2^2 = \sum_{ij} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \mathbf{s}_{ij}^2$ when using Euclidean distances, which is similar to the objective of CAN [234], except that the power of $\mathbf{s}_{ij}$ is 1 in CAN and 2 in the proposed case.
3. From feature learning point of view, each $\mathbf{s}_i$ can be regarded as a representation of $\mathbf{x}_i$. The representations are assumed to be smooth on some low-dimensional manifold and are regularized by a predefined graph $\mathbf{S}'$. In practice $\mathbf{S}'$ is defined also on the k -NN map, i.e. $\mathbf{s}_i$ is connected only to

$\mathbf{s}_j, j \in \mathcal{N}_i$. Since $\mathbf{s}_i$ carries information of its nearest neighbors, the manifold regularization in fact models the relationship between $\mathcal{N}_i$ and $\mathcal{N}_j$ beyond nearest neighbors.

4. It is also worth noting that in practice (4.3) is generally non-convex and the rank constraint can only be approximately achieved, especially under the positivity constraint. Nevertheless, solving (4.3) gives a locally connect data graph $\mathbf{S}$ that is well partitioned.

4.2.2 Clustering with Learned Graph

After learning the graph $\mathbf{S}$, spectral clustering (SC) [224] can be performed on $\mathbf{S}$ to group $\{\mathbf{x}_i\}_{i=1}^N$ into c different centroids, which is a common practice for subspace clustering [57], [237]. Specifically for $\mathbf{F} \in \mathbb{R}^{N \times c}$, SC minimizes

$$\min_{\mathbf{F} \in \mathbb{R}^{N \times c}, \mathbf{F}^T \mathbf{F} = \mathbf{I}_c} \text{tr}[\mathbf{F}^T \mathbf{L}_S \mathbf{F}] \quad (4.4)$$

and then K-means clustering is performed on rows of $\mathbf{F}$.

Alternatively, Huang et al. [173] proposed *unsupervised ELM* (US-ELM) for embedding and clustering, which is reviewed in Section 2.4.3. This chapter reports both clustering results given by spectral clustering and US-ELM on the learned graph for comparison, which are referred to as ALC and ELM-ALC respectively. Experiments in Section 4.4 will demonstrate that ALC and ELM-ALC are expertise in different datasets, but overall they both achieve superior results compared with other graph-based clustering methods.

4.3 Optimizations of ELM-ALC

This section presents the optimization approach for solving (4.3). The iterative scheme in [234] is adopted to update the graph by adaptively increasing the weight of the rank constraint. Specifically, enforcing the rank constraint $\text{rank}(\mathbf{L}_S) = N - c$ is equivalent to requiring

---

Algorithm 3: Adaptive Locality-constrained Clustering (ALC)

Data: Input samples $\mathbf{X} \in \mathbb{R}^{n \times N}$, predefined Laplacian matrix $\mathbf{L}$, number of nearest neighbors k , number of classes c , hyperparameters η, γ .

```

1 for  $i = 1, \dots, N$  do
2   └─ Compute  $\mathbf{x}_i$ 's nearest neighbors  $\mathcal{N}_i$  and distance to each neighbor  $\mathbf{d}_{xi}$ ;
3 Initialize  $\mathbf{S}$  as a binary  $k$ -NN map;
4 for  $t = 1, \dots, T$  do
5   └─ Update  $\mathbf{F}$  by computing the  $c$  eigenvectors corresponding to  $c$  smallest
      eigenvalues of  $\mathbf{L}\mathbf{S}$ ;
6   for  $i = 1, \dots, N$  do
7     └─ Fix  $\mathbf{s}_j$  with  $j \neq i$  and compute  $\mathbf{d}_{fi}$  and  $\mathbf{d}_{\ell i}$ ;
8     └─ Update  $\mathbf{s}_i$  by solving (4.11) with ADMM;
9 return Data graph  $\mathbf{S}$  with  $c$  connected components.
```

$$\sigma_1(\mathbf{L}_\mathbf{S}) = \dots = \sigma_c(\mathbf{L}_\mathbf{S}) = 0, \sigma_{c+1}(\mathbf{L}_\mathbf{S}) > 0 \quad (4.5)$$

where σ_i is the i -th smallest eigenvalue of $\mathbf{L}_\mathbf{S}$. Therefore, one can enforce $\sum_{i=1}^c \sigma_i(\mathbf{L}_\mathbf{S})$ to be as small as possible as a soft alternative to (4.5). Note that [243]

$$\sum_{i=1}^c \sigma_i(\mathbf{L}_\mathbf{S}) = \min_{\mathbf{F} \in \mathbb{R}^{N \times c}, \mathbf{F}^T \mathbf{F} = \mathbf{I}} \text{tr}[\mathbf{F}^T \mathbf{L}_\mathbf{S} \mathbf{F}]. \quad (4.6)$$

Therefore, (4.3) can be approximated by introducing a new variable $\mathbf{F} \in \mathbb{R}^{N \times c}$:

$$\begin{aligned} \min_{\mathbf{S} \in \mathcal{C}, \mathbf{F}} & \sum_{i=1}^N \|\mathbf{x}_i - \mathbf{X}_i \mathbf{s}_i\|_2 + \frac{\eta}{2} \sum_{i=1}^N \|\mathbf{d}_{xi} \odot \mathbf{s}_i\|_2^2 \\ & + \frac{\gamma}{2} \text{tr}[\mathbf{S} \mathbf{L} \mathbf{S}^T] + \lambda \text{tr}[\mathbf{F}^T \mathbf{L}_\mathbf{S} \mathbf{F}] \\ \text{s.t.} \quad & \mathbf{s}_{ij} \geq 0, \mathbf{F}^T \mathbf{F} = \mathbf{I}. \end{aligned} \quad (4.7)$$

Here λ is a large number to penalize the constraint of $\mathbf{F}$ so that the obtained Laplacian matrix $\mathbf{L}_\mathbf{S}$ has multiplicity of c for the smallest eigenvalue 0. In practice, exact multiplicity of c cannot be guaranteed since $\sigma_{c+1}(\mathbf{L}_\mathbf{S})$ is not explicitly guaranteed to be positive. The optimizations in [234] suggest adjusting dynamically the value of λ based on the values of $\sum_{i=1}^c \sigma_i(\mathbf{L}_\mathbf{S})$ and $\sum_{i=1}^{c+1} \sigma_i(\mathbf{L}_\mathbf{S})$ at each iteration, so that the condition (4.5) is approximately achieved.

To solve $\mathbf{S}$ when $\mathbf{F}$ is fixed, (4.7) can be transformed to

$$\begin{aligned} \min_{\mathbf{S} \in \mathcal{C}} & \sum_{i=1}^N \|\mathbf{x}_i - \mathbf{X}_i \mathbf{s}_i\|_2 + \frac{\eta}{2} \sum_{i=1}^N \|\mathbf{d}_{xi} \odot \mathbf{s}_i\|_2^2 \\ & + \frac{\gamma}{2} \text{tr}[\mathbf{S} \mathbf{L} \mathbf{S}^T] + \frac{\lambda}{2} \sum_i \sum_{j \in \mathcal{N}_i} \|\mathbf{f}_i - \mathbf{f}_j\|_2^2 \mathbf{s}_{ij} \\ \text{s.t.} \quad & \mathbf{s}_{ij} \geq 0. \end{aligned} \quad (4.8)$$

Here $\mathbf{f}_i \in \mathbb{R}^c$ is the i -th row of $\mathbf{F}$, $1 \leq i \leq N$. Define $\mathbf{d}_{fi} \in \mathbb{R}^k$ as all distances of $\mathbf{f}_i$ to $\mathbf{f}_j$ where $j \in \mathcal{N}_i$. At each iteration each column $\mathbf{s}_i$ is updated in a Gauss-Seidel fashion with other columns of $\mathbf{S}$ fixed. Note that

$$\begin{aligned}
\text{tr}[\mathbf{S}\mathbf{L}\mathbf{S}^T] &= \sum_{i,j=1}^N \ell_{ij} \hat{\mathbf{s}}_i^T \hat{\mathbf{s}}_j = \sum_i \ell_{ii} \hat{\mathbf{s}}_i^T \hat{\mathbf{s}}_i + \sum_{i \neq j} \ell_{ij} \hat{\mathbf{s}}_i^T \hat{\mathbf{s}}_j \\
&= \sum_i \ell_{ii} \mathbf{s}_i^T \mathbf{s}_i + \sum_i \mathbf{s}_i^T \mathbf{d}_{\ell i}
\end{aligned} \tag{4.9}$$

where $\mathbf{d}_{\ell i} = \mathcal{R}_{\mathcal{N}_i} \sum_{j \neq i} \ell_{ij} \hat{\mathbf{s}}_j$, and $\mathcal{R}_{\mathcal{N}_i}(\mathbf{x}) : \mathbb{R}^N \rightarrow \mathbb{R}^k$ is the selection function for entries of $\mathbf{x}$ in $\mathcal{N}_i$ (since $\hat{\mathbf{s}}_{ij} = 0$ for $j \notin \mathcal{N}_i$). When $\mathbf{s}_j$ ($j \neq i$) is fixed, (4.8) can be solved independently for each i , $1 \leq i \leq N$:

$$\begin{aligned}
\min_{\mathbf{s}_i} \|\mathbf{x}_i - \mathbf{X}_i \mathbf{s}_i\|_2 + \frac{\eta}{2} \|\mathbf{d}_{xi} \odot \mathbf{s}_i\|_2^2 + \frac{\gamma}{2} \ell_{ii} \mathbf{s}_i^T \mathbf{s}_i + \frac{\gamma}{2} \mathbf{d}_{\ell i}^T \mathbf{s}_i + \frac{\lambda}{2} \mathbf{d}_{fi}^T \mathbf{s}_i \\
\text{s.t. } \mathbf{s}_{ij} \geq 0.
\end{aligned} \tag{4.10}$$

Problem (4.10) can be solved iteratively via ADMM [90]. For notational convenience the subscript i is omitted. Define $\mathbf{d} \triangleq (\gamma \mathbf{d}_{\ell i} + \lambda \mathbf{d}_{fi})/2$ and $\mathbf{q} \triangleq \eta \mathbf{d}_{xi} + \gamma \ell_{ii} \mathbf{1}$, then (4.10) can be transformed to

$$\min_{\mathbf{s}} \|\mathbf{x} - \mathbf{X}\mathbf{s}\|_2 + (1/2) \mathbf{s}^T \text{diag}(\mathbf{q}) \mathbf{s} + \mathbf{d}^T \mathbf{s} + \iota(\mathbf{s}) \tag{4.11}$$

where $\iota(\mathbf{s}) = 0$ if $\mathbf{s} \geq 0$ and ∞ otherwise. To solve (4.11), define auxiliary variables $\mathbf{e} \triangleq \mathbf{X}\mathbf{g} - \mathbf{x}$, $\mathbf{g} = \mathbf{s}$, and (4.11) is equivalent to

$$\begin{aligned}
\min_{\mathbf{s}, \mathbf{e}, \mathbf{g}} \|\mathbf{e}\|_2 + (1/2) \mathbf{g}^T \text{diag}(\mathbf{q}) \mathbf{g} + \mathbf{d}^T \mathbf{g} + \iota(\mathbf{s}) \\
\text{s.t. } \mathbf{e} = \mathbf{X}\mathbf{g} - \mathbf{x}, \mathbf{g} = \mathbf{s}.
\end{aligned} \tag{4.12}$$

The ADMM formulation is:

$$\begin{aligned}
\min_{\mathbf{s}, \mathbf{e}, \mathbf{g}, \mathbf{u}, \mathbf{v}} \|\mathbf{e}\| + \frac{1}{2} \mathbf{g}^T \text{diag}(\mathbf{q}) \mathbf{g} + \mathbf{d}^T \mathbf{g} + \iota(\mathbf{s}) \\
+ \frac{\rho}{2} \|\mathbf{X}\mathbf{g} - \mathbf{e} - \mathbf{x} + \mathbf{u}\|_2^2 + \frac{\rho}{2} \|\mathbf{g} - \mathbf{s} + \mathbf{v}\|_2^2.
\end{aligned} \tag{4.13}$$

Here $\mathbf{u}$ and $\mathbf{v}$ are dual variables and $\rho > 0$ is a penalty weight. Solving (4.13) amounts to optimizing w.r.t. $\mathbf{s}$, $\mathbf{e}$ and $\mathbf{g}$, and updating the dual variable iteratively. At step $k+1$ each variable can be updated as

$$\mathbf{g}^{(k+1)} = \underset{\mathbf{g}}{\operatorname{argmin}} \frac{1}{2} \mathbf{g}^T \operatorname{diag}(\mathbf{q}) \mathbf{g} + \mathbf{d}^T \mathbf{g} \quad (4.14)$$

$$+ \frac{\rho}{2} \|\mathbf{X}\mathbf{g} - \mathbf{e}^{(k)} - \mathbf{x} + \mathbf{u}^{(k)}\|_2^2 + \frac{\rho}{2} \|\mathbf{g} - \mathbf{s}^{(k)} + \mathbf{v}^{(k)}\|_2^2 \quad (4.15)$$

$$\mathbf{e}^{(k+1)} = \underset{\mathbf{e}}{\operatorname{argmin}} \|\mathbf{e}\|_2 + \frac{\rho}{2} \|\mathbf{X}\mathbf{g}^{(k+1)} - \mathbf{e} - \mathbf{x} + \mathbf{u}^{(k)}\|_2^2 \quad (4.16)$$

$$\mathbf{s}^{(k+1)} = \underset{\mathbf{s}}{\operatorname{argmin}} \iota(\mathbf{s}) + \frac{\rho}{2} \|\mathbf{g}^{(k+1)} - \mathbf{s} + \mathbf{v}^{(k)}\|_2^2 \quad (4.17)$$

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \mathbf{X}\mathbf{g}^{(k+1)} - \mathbf{e}^{(k+1)} - \mathbf{x}^{(k+1)} \quad (4.18)$$

$$\mathbf{v}^{(k+1)} = \mathbf{v}^{(k)} + \mathbf{g}^{(k+1)} - \mathbf{s}^{(k+1)} \quad (4.19)$$

The $\mathbf{g}$ sub-problem (4.15) has a closed-form solution:

$$\mathbf{g}^{(k+1)} = \left(\mathbf{X}^T \mathbf{X} + \mathbf{I} + \frac{\operatorname{diag}(\mathbf{q})}{\rho} \right)^{-1} \left(\mathbf{X}^T (\mathbf{e}^{(k)} + \mathbf{x} - \mathbf{u}^{(k)}) + \mathbf{s}^{(k)} - \mathbf{v}^{(k)} - \frac{\mathbf{d}}{\rho} \right). \quad (4.20)$$

The solution to $\mathbf{e}$ sub-problem (4.16) is

$$\mathbf{e}^{(k+1)} = \mathcal{P}_{1/\rho}(\mathbf{X}\mathbf{g}^{(k+1)} - \mathbf{x} + \mathbf{u}^{(k)}) \quad (4.21)$$

where

$$\mathcal{P}_\lambda(\mathbf{x}) = \max(1 - \lambda/\|\mathbf{x}\|_2, 0)\mathbf{x}$$

is the *proximal operator* [159] of the ℓ_2 norm $\|\cdot\|_2$.

For $\mathbf{s}$ sub-problem (4.17), the solution is

$$\mathbf{s}^{(k+1)} = \max(\mathbf{g}^{(k+1)} + \mathbf{v}^{(k)}, 0). \quad (4.22)$$

The final algorithm for solving (4.3) is summarized in algorithm 3. Some discussions regarding the computational complexity of ALC are given as follows. The computation of ALC is dominated by line 5 and 8 in algorithm 3. For each iteration, line 5 computes the eigendecompositions of the Laplacian matrix $\mathbf{L}_\mathbf{S}$ which typically takes $\mathcal{O}(N^3)$ for a full system. Line 8 solves an ADMM problem at each iteration, and the computations fall on solving (4.15), i.e. the $\mathbf{g}$ sub-problem.

TABLE 4.1: Dataset Specificities

Type	Datasets	No. Samples	No. Features	No. Classes
Image	UMIST	575	644	20
	ORL	400	1024	40
	POLLEN	630	625	7
	USPST	2007	256	10
	MNIST05	3495	784	10
	ETH80	3280	1024	8
UCI	ECOLI	336	343	8
	CARS	392	8	3
	DIABETES	768	8	2
	SEGMENT	2310	19	7
	VEHICLE	846	18	4
	CONTROL	600	60	6

The closed-form solution (4.20) incurs a complexity of $\mathcal{O}(k^2N + k^3)$. Considering that the number of ADMM iterations is fixed, the complexity of algorithm 3 is $\mathcal{O}(N^3 + k^2N + k^3)$, or $\mathcal{O}(N^3)$ with the number of nearest neighbors $k \ll N$. In practice, since $\mathbf{S}$ has only k nonzeros at each column, utilizing the sparse pattern of $\mathbf{S}$ can further reduce the complexity of eigendecompositions.

4.4 Experiments

This section evaluates the proposed algorithm on benchmark datasets, described in Section 4.4.1, and compares with state-of-the-art graph-based clustering algorithms. Descriptions of the experimental settings and comparison methods are given in Section 4.4.2. Main experimental results are discussed in Section 4.4.3 with comparisons between graph-based clustering methods. An empirical evaluation of ELM based clustering methods is given in Section 4.4.4. Validations of ALC's different components, including the distance constraint, manifold regularizations, as well as different predefined graphs for manifold regularizations are given in Section 4.4.5. Finally A sensitivity analysis of trade-off parameters is presented in Section 4.4.6.

4.4.1 Datasets

The proposed method is evaluated on six image datasets, i.e. UMIST [244], ORL [245], POLLEN [246], USPST [247], MNIST05 [30], ETH80 [248], and six UCI datasets [249], including ECOLI, CARS, DIABETES, SEGMENT, VEHICLE and CONTROL. Specifically, the UMIST database and the ORL database contain face images from 20 and 40 individuals respectively. The POLLEN database contains images of 7 classes of pollen grain. The USPST database is the test set of the handwriting recognition dataset USPS. The MNIST05 database is a subset (5% sampling) of the MNIST digit dataset. Details regarding the dataset size, number of features for each sample and number of classes for each dataset are summarized in Table 4.1.

4.4.2 Experimental Settings

For evaluating the proposed method and comparing with others, the clustering accuracy (ACC) and the Normalized Mutual Information (NMI) are adopted as the evaluation metrics to measure the performance of different clustering methods. ACC and NMI are common clustering measure which can be found to be widely used, for example [176], [234], [250]–[253]. ACC computes the percentage of correctly grouped data samples when compared with ground truth labels. This amounts to finding the “best match” between the predictions and ground truths. Specifically, the accuracy is defined as

$$ACC = \frac{\sum_{i=1}^N \delta(\mathbf{y}_i, \text{map}(\hat{\mathbf{y}}_i))}{N}, \quad (4.23)$$

where $\mathbf{y}_i$ is the ground truth label for $\mathbf{x}_i$, and $\hat{\mathbf{y}}_i$ is the predicted label given by a clustering algorithm. The function $\text{map}(\cdot)$ maps the indices of the predicted labels so that they matches the ground truth partition maximally.

On the other hand, NMI aims to measure the amount of common information shared by two random variables. It is widely used for measuring the quality of clustering algorithms by comparing prediction results to ground truth assignments. Specifically NMI is compute as

TABLE 4.2: Accuracy results (in %) of baseline methods (K-Means/SC) and graph learning methods (CAN/PCAN/CLR_L1/CLR_L2). For POLLEN and ECOLI, CAN cannot find the correct number of clusters, so their accuracy values are not reported. All results are reported as *mean* $\pm$ *standard deviation* over 10 runs.

Dataset	K-Means	SC	CAN	PCAN	CLR_L1	CLR_L2
UMIST	42.66 $\pm$ 1.47	68.64 $\pm$ 3.02	83.96 $\pm$ 0.00	68.17 $\pm$ 0.00	74.09 $\pm$ 0.00	74.09 $\pm$ 0.00
ORL	54.60 $\pm$ 2.32	67.08 $\pm$ 0.78	52.50 $\pm$ 0.00	59.50 $\pm$ 0.00	61.25 $\pm$ 0.00	63.25 $\pm$ 0.00
POLLEN	47.86 $\pm$ 3.38	50.32 $\pm$ 0.00	—	43.49 $\pm$ 0.00	38.73 $\pm$ 0.00	41.11 $\pm$ 0.00
USPST	67.23 $\pm$ 0.09	67.41 $\pm$ 0.00	75.54 $\pm$ 0.00	64.52 $\pm$ 0.00	73.89 $\pm$ 0.00	76.73 $\pm$ 0.00
MNIST05	52.66 $\pm$ 0.28	66.98 $\pm$ 0.00	62.00 $\pm$ 0.00	55.22 $\pm$ 0.00	50.41 $\pm$ 0.00	57.34 $\pm$ 0.00
ETH80	50.05 $\pm$ 0.16	55.02 $\pm$ 0.75	17.32 $\pm$ 0.00	49.39 $\pm$ 0.00	31.95 $\pm$ 0.00	41.07 $\pm$ 0.00
ECOLI	56.25 $\pm$ 2.79	58.93 $\pm$ 0.00	—	77.68 $\pm$ 0.00	69.35 $\pm$ 0.00	67.26 $\pm$ 0.00
CARS	44.13 $\pm$ 0.00	46.43 $\pm$ 0.00	46.94 $\pm$ 0.00	61.48 $\pm$ 0.00	46.43 $\pm$ 0.00	46.43 $\pm$ 0.00
DIABETES	66.80 $\pm$ 0.00	64.58 $\pm$ 0.00	65.23 $\pm$ 0.00	66.02 $\pm$ 0.00	64.84 $\pm$ 0.00	64.84 $\pm$ 0.00
SEGMENT	66.57 $\pm$ 0.04	67.90 $\pm$ 0.02	41.08 $\pm$ 0.00	57.57 $\pm$ 0.00	52.03 $\pm$ 0.00	52.94 $\pm$ 0.00
VEHICLE	36.69 $\pm$ 0.06	42.32 $\pm$ 0.00	40.07 $\pm$ 0.00	41.25 $\pm$ 0.00	42.55 $\pm$ 0.00	42.67 $\pm$ 0.00
CONTROL	56.83 $\pm$ 0.00	67.33 $\pm$ 0.00	57.50 $\pm$ 0.00	68.00 $\pm$ 0.00	72.17 $\pm$ 0.00	72.17 $\pm$ 0.00

TABLE 4.3: Accuracy results (in %) of subspace clustering methods (SSC/SSC-OMP/LRR), ELM based graph learning method (ELM-CLR) and the proposed ALC/ELM-ALC. For each dataset, the highest mean accuracy among all clustering methods in Table 4.2 and this table is highlighted in bold. All results are reported as *mean ± standard deviation* over 10 runs.

Dataset	SSC	SSC-OMP	LRR	ELM-CLR	ALC	ELM-ALC
UMIST	59.81 ± 1.81	56.82 ± 1.20	45.36 ± 0.91	75.65 ± 0.00	85.39 ± 0.00	85.39 ± 0.00
ORL	75.65 ± 2.20	73.03 ± 1.98	73.63 ± 2.29	68.25 ± 0.00	69.07 ± 0.72	73.70 ± 1.77
POLLEN	54.33 ± 0.13	49.43 ± 0.15	44.37 ± 0.08	54.41 ± 0.44	50.95 ± 0.13	60.57 ± 1.52
USPST	65.82 ± 0.00	75.49 ± 0.00	73.00 ± 0.02	84.67 ± 0.12	87.94 ± 0.00	92.54 ± 0.02
MNIST05	67.27 ± 0.10	64.41 ± 0.01	56.46 ± 0.08	67.19 ± 3.64	72.22 ± 0.00	68.63 ± 2.00
ETH80	47.08 ± 0.05	56.82 ± 0.00	53.97 ± 0.02	55.66 ± 0.52	58.23 ± 0.00	54.71 ± 1.01
ECOLI	59.49 ± 0.41	59.85 ± 0.09	58.13 ± 0.28	79.49 ± 0.09	79.76 ± 0.00	73.01 ± 5.43
CARS	62.40 ± 0.22	49.23 ± 0.00	57.14 ± 0.00	46.94 ± 0.00	64.29 ± 0.00	65.48 ± 0.12
DIABETES	68.88 ± 0.00	67.32 ± 0.00	67.97 ± 0.00	67.41 ± 0.21	66.93 ± 0.00	69.00 ± 0.04
SEGMENT	16.04 ± 0.05	14.93 ± 0.04	72.16 ± 0.00	69.78 ± 0.38	72.86 ± 0.00	75.57 ± 7.54
VEHICLE	42.91 ± 0.00	40.09 ± 0.00	41.96 ± 0.00	41.97 ± 0.07	51.06 ± 0.00	47.45 ± 0.26
CONTROL	66.17 ± 0.00	67.33 ± 0.18	60.50 ± 0.00	71.42 ± 0.24	83.83 ± 0.00	90.98 ± 5.14

TABLE 4.4: NMI results (in %) of baseline methods (K-Means/SC) and graph learning methods (CAN/PCAN/CLR_L1/CLR_L2). For each dataset, the highest mean NMI among all clustering methods in Table 4.5 and this table is highlighted in bold. All results are reported as *mean $\pm$ standard deviation* over 10 runs.

Dataset	K-Means	SC	CAN	PCAN	CLR_L1	CLR_L2
UMIST	64.71 $\pm$ 0.96	86.56 $\pm$ 1.39	90.54 $\pm$ 0.00	81.09 $\pm$ 0.00	87.49 $\pm$ 0.00	87.49 $\pm$ 0.00
ORL	73.72 $\pm$ 1.40	81.97 $\pm$ 0.39	70.34 $\pm$ 0.00	75.91 $\pm$ 0.00	78.51 $\pm$ 0.00	80.38 $\pm$ 0.00
POLLEN	41.48 $\pm$ 0.95	45.19 $\pm$ 0.00	0.00 $\pm$ 0.00	40.51 $\pm$ 0.00	33.48 $\pm$ 0.00	39.49 $\pm$ 0.00
USPST	64.59 $\pm$ 0.08	79.35 $\pm$ 0.00	78.24 $\pm$ 0.00	71.81 $\pm$ 0.00	75.08 $\pm$ 0.00	81.03 $\pm$ 0.00
MNIST05	48.14 $\pm$ 0.25	68.34 $\pm$ 0.00	65.95 $\pm$ 0.00	56.87 $\pm$ 0.00	60.62 $\pm$ 0.00	64.86 $\pm$ 0.00
ETH80	47.65 $\pm$ 0.19	52.19 $\pm$ 0.45	17.32 $\pm$ 0.00	43.50 $\pm$ 0.00	26.48 $\pm$ 0.00	36.27 $\pm$ 0.00
ECOLI	51.23 $\pm$ 1.59	54.32 $\pm$ 0.00	0.00 $\pm$ 0.00	66.51 $\pm$ 0.00	62.27 $\pm$ 0.00	58.58 $\pm$ 0.00
CARS	24.39 $\pm$ 0.00	25.46 $\pm$ 0.00	24.81 $\pm$ 0.00	29.86 $\pm$ 0.00	25.50 $\pm$ 0.00	25.50 $\pm$ 0.00
DIABETES	5.12 $\pm$ 0.00	4.84 $\pm$ 0.00	0.22 $\pm$ 0.00	4.86 $\pm$ 0.00	0.22 $\pm$ 0.00	0.22 $\pm$ 0.00
SEGMENT	61.13 $\pm$ 0.03	64.96 $\pm$ 0.00	41.57 $\pm$ 0.00	56.50 $\pm$ 0.00	55.09 $\pm$ 0.00	56.14 $\pm$ 0.00
VEHICLE	10.02 $\pm$ 0.02	17.54 $\pm$ 0.00	15.07 $\pm$ 0.00	17.90 $\pm$ 0.00	18.83 $\pm$ 0.00	16.67 $\pm$ 0.00
CONTROL	69.98 $\pm$ 0.00	75.73 $\pm$ 0.00	74.21 $\pm$ 0.00	77.52 $\pm$ 0.00	76.33 $\pm$ 0.00	79.08 $\pm$ 0.00

TABLE 4.5: NMI results (in %) of subspace clustering methods (SSC/SSC-OMP/LRR), ELM based graph learning method (ELM-CLR) and the proposed ALC/ELM-ALC. For each dataset, the highest mean NMI among all clustering methods in Table 4.4 and this table is highlighted in bold. All results are reported as *mean $\pm$ standard deviation* over 10 runs.

Dataset	SSC	SSC-OMP	LRR	ELM-CLR	ALC	ELM-ALC
UMIST	75.97 $\pm$ 0.90	73.65 $\pm$ 1.32	65.60 $\pm$ 0.32	87.33 $\pm$ 0.45	92.11 $\pm$ 0.00	92.31 $\pm$ 0.00
ORL	87.60 $\pm$ 0.70	86.03 $\pm$ 0.92	85.32 $\pm$ 1.13	81.56 $\pm$ 0.00	83.14 $\pm$ 0.44	85.80 $\pm$ 0.82
POLLEN	49.01 $\pm$ 0.26	41.49 $\pm$ 0.16	45.52 $\pm$ 0.02	52.10 $\pm$ 0.53	50.56 $\pm$ 0.06	49.61 $\pm$ 1.16
USPST	75.86 $\pm$ 0.00	75.87 $\pm$ 0.02	70.08 $\pm$ 0.02	81.41 $\pm$ 0.10	84.08 $\pm$ 0.00	85.43 $\pm$ 0.06
MNIST05	67.51 $\pm$ 0.11	64.94 $\pm$ 0.02	51.82 $\pm$ 0.03	67.88 $\pm$ 1.13	71.62 $\pm$ 0.00	67.46 $\pm$ 2.37
ETH80	51.82 $\pm$ 0.08	58.51 $\pm$ 0.00	45.86 $\pm$ 0.03	51.06 $\pm$ 0.66	57.76 $\pm$ 0.00	53.87 $\pm$ 0.69
ECOLI	50.38 $\pm$ 0.10	52.43 $\pm$ 0.00	48.26 $\pm$ 0.25	69.46 $\pm$ 0.00	69.31 $\pm$ 0.00	62.81 $\pm$ 2.77
CARS	21.49 $\pm$ 0.00	6.67 $\pm$ 0.00	16.70 $\pm$ 0.00	27.43 $\pm$ 0.06	27.91 $\pm$ 0.00	28.22 $\pm$ 2.25
DIABETES	10.18 $\pm$ 0.00	10.67 $\pm$ 0.00	9.74 $\pm$ 0.00	5.18 $\pm$ 0.11	8.25 $\pm$ 0.00	13.23 $\pm$ 0.04
SEGMENT	1.50 $\pm$ 0.07	0.79 $\pm$ 0.04	64.04 $\pm$ 0.06	67.71 $\pm$ 0.11	68.84 $\pm$ 0.00	73.25 $\pm$ 6.36
VEHICLE	16.61 $\pm$ 0.00	15.38 $\pm$ 0.00	12.47 $\pm$ 0.00	18.26 $\pm$ 2.40	27.48 $\pm$ 0.00	20.95 $\pm$ 0.32
CONTROL	73.29 $\pm$ 0.00	61.48 $\pm$ 0.10	60.99 $\pm$ 0.00	79.75 $\pm$ 0.21	82.37 $\pm$ 0.00	85.99 $\pm$ 0.76

$$NMI = \frac{MI(\mathbf{Y}, \hat{\mathbf{Y}})}{\max(\mathbb{H}(\mathbf{Y}), \mathbb{H}(\hat{\mathbf{Y}}))} \quad (4.24)$$

where $\mathbf{Y}$ and $\hat{\mathbf{Y}}$ are ground truth labels and predicted labels for the whole dataset respectively. $MI(\cdot, \cdot)$ denotes *mutual information* between two distributions, and $\mathbb{H}(\cdot)$ denotes *entropy*.

The proposed graph learning algorithm is compared with the following ones: K-Means, SC [223], CAN/PCAN [234], CLR_L1/CLR_L2 [235], SSC [57], SSC-OMP [254], LRR [237] and ELM-CLR [176]. K-Means is classic clustering method served as a baseline in this Section. The spectral clustering (SC) predicts labels on a predefined k -NN graph. The Clustering with Adaptive Neighbors (CAN) and Projected CAN (PCAN) update the data graph adaptively with rank constraint. The Constrained Laplacian Rank (CLR) minimizes the distance between the rank constrained data graph with a predefined one. The Sparse Subspace Clustering (SSC) expresses each data point as linear combinations of others with ℓ_1 constraint. The Sparse Subspace Clustering with Orthogonal Matching Pursuit (SSC-OMP) solves an ℓ_0 regularized subspace clustering problem. The Low Rank Representation (LRR) enforces low rank constraints on the graph from subspace clustering. The ELM-CLR utilizes dual data representations for learning a graph. The performance of ALC and ELM-ALC are measured and the accuracy results are compared with the above algorithms. Following [176], the following experimental settings are adopted:

1. For each algorithm, the best accuracy is given by searching the parameter grid. The accuracy value is averaged over 10 runs.
2. For image datasets in Table 4.1, each image is normalized to $[-1, 1]$. For UCI datasets in Table 4.1, each feature is rescaled to $[-1, 1]$.
3. For all methods using K-means for final stage clustering, it is repeated for 100 times the one with smallest objective value is chosen as common practice.
4. For all adaptive graph learning methods, i.e. CAN/PCAN, CLR_L1/CLR_L2, ELM-CLR, ALC/ELM-ALC, the number of iterations is 30. Failure of algorithm is reported if the obtained graph has incorrect number of clusters.

5. For SC, the self-tuning Gaussian method [242] is adopted for constructing the predefined graph.
6. For ELM based methods, i.e. ELM-CLR and ELM-ALC, the number of hidden neurons is set to be 1000 for UCI datasets and 2000 for image datasets, and sigmoid function is used as the nonlinear activation.
7. For ALC and ELM-ALC, the initial graph for CLR [235] is set as the predefined graph in (4.3) for manifold regularizations.

The values of parameters for each algorithm are given as follows:

1. For all graph-based methods, the number of nearest neighbors is selected from $\{3, \dots, 10, 12, 15, 20, 25, 30\}$.
2. For PCAN, ELM-CLR and ELM-ALC, the output dimension is selected from $\{2, 4, 8, 16, 32\} \cup \{c\}$.
3. For SSC and LRR, the regularization parameter are selected from $\{2^{-10}, 2^{-8}, \dots, 2^8, 2^{10}\}$.
4. For ELM-CLR and ELM-ALC, the regularization parameter (the parameter α in (2.79)) is selected from $\{10^{-4}, 10^{-3}, \dots, 10^4\}$.
5. For ALC and ELM-ALC, η is selected from $\{2^{-4}, 2^{-3}, \dots, 2^4\}$ and γ is selected from $\{2^{-10}, 2^{-8}, \dots, 2^8, 2^{10}\}$.

4.4.3 Clustering Performance

The clustering is performed on the datasets given in Table 4.1. Accuracy results (ACC) with baseline methods as well as graph learning approaches are summarized in Table 4.2 and 4.3, where the accuracies are reported as *mean $\pm$ standard deviation*. In general, ALC and ELM-ALC achieve the best results on most datasets (except ORL). Specifically, ALC prevails on 5 datasets and ELM-ALC gives the best performance on 7 datasets. The accuracy boost of ALC/ELM-ALC against comparison methods ranges from 1.4% (on UMIST) to maximally 19% (on CONTROL), with typically 3% accuracy gain. This shows that the graph obtained via (4.3) is effective in capturing cluster structures. On ORL and DIABETES,

ELM-ALC provides comparable results to subspace clustering methods, i.e. SSC and LRR, which demonstrates that 1) self-expressiveness as the main property of subspace clustering methods is important for clustering, and 2) the reconstruction term in ALC also exhibits self-expressiveness property.

ALC provides a large boost to baseline methods, i.e. K-Means and SC. SC performs eigendecompositions on a fixed Gaussian k -NN graph whose width is determined automatically via self-tuning approaches [242]. ALC also perform eigendecompositions on a k -NN graph, but its weights are learned adaptively. This proves that the learned weights are beneficial for the clustering task. It can be seen that ALC learns a better graph than CAN or CLR. In fact CAN or CLR does not provide a significant advantage over the fixed graph spectral clustering method. CAN focuses on reducing the weighted sum of pair-wise distances, and it does not consider reconstructions of each sample by others, which are important for effective graph representations. CLR can be seen as a refinement of the predefined graph by gradual incorporations of rank constraints, and it does not add additional regularizations as priors for clustering. The proposed method learns a graph that not only considers both distance constraints and knowledge of a predefined graph, but also incorporates representative information of the nearest neighbors.

Compared with ALC that uses spectral clustering for predicting class labels, ELM-ALC with the same graph can give better results on more than a half of benchmark datasets. ELM-ALC uses US-ELM as post-processing for computing clustering results, and overall the obtained random features provides more discriminative representations than original ones. On the other hand, ELM-ALC does not always obtain the best results, especially on large image datasets such as MNIST05 and ETH80. One possible reason is that the objective for graph learning and US-ELM are different, and the graph from solving (4.3) may not be optimal for ELM feature based clustering.

In addition, the NMI results with baseline methods as well as graph learning approaches are summarized in Table 4.4 and 4.5. The results are also reported as *mean $\pm$ standard deviation*. In this measure ALC and ELM-ALC give best results among 8 out of 12 datasets, and the gap between the remaining 4 and the best ones is small. The performance is consistent with the one given by ACC (note that SSC also achieves the best result on ORL). It can be also seen that ELM-CLR provides a competitive performance under the NMI metric. Specifically, it prevails in the

FIGURE 4.2: Accuracy values with respect to different output dimensions on benchmark datasets.

TABLE 4.6: Configurations of ALC variants. The check mark (✓) indicates that the algorithm possesses the corresponding capability.

	ℓ_2 Formulation	Manifold Regularization	US-ELM for Clustering
ALC- Fr			
ALC- η	✓		
ALC	✓	✓	
ELM-ALC	✓	✓	✓

POLLEN and E-COIL dataset, and is close to the proposed methods (ALC and ELM-ALC) in several benchmark datasets. Therefore, it would be worthwhile to make a more detailed comparison between ELM-CLR and ELM-ALC, as presented in the following section.

4.4.4 Comparisons between ELM based Graph Learning Methods

A more detailed comparison is made between ELM based methods, i.e. ELM-CLR and ELM-ALC. Specifically, the clustering accuracies are plotted for each method with varying ELM output dimension from $\{2, 4, 8, 16, 32\}$. Other parameters are selected to provide the best clustering results. The results are averaged over 10 runs with different initializations.

The results are shown in Figure 4.2. In general, ELM-ALC gives better performance for the optimal dimension for most benchmark datasets (except EHT80 and ECOLI), as also verified in Table 4.3. Taking the whole dimensionality range into consideration, ELM-CLR is comparable on several datasets, but overall ELM-ALC takes the lead on more benchmarks. Moreover, ELM-CLR does not have good results on UCI datasets with few number of features (less than 60), i.e. CARS, DIABETES, SEGMENTS, VEHICLE and CONTROL, and is comparably good on others with feature numbers larger than 250. This means that ELM-CLR is effective when the original feature space is relatively complex. Otherwise the randomness may degrade the performance. The proposed method does not suffer from such limitation, and provide better accuracy results on CARS and DIABETES, both of which have 8 features only.

4.4.5 Validations of Components

In this part, the effectiveness of each component of ALC and ELM-ALC on the clustering task is validated via an ablation study. Apart from ALC/ELM-ALC, problem (4.2) is compared which uses the Frobenius norm for reconstruction and does not apply manifold regularization. This variant is termed ALC-*Fr*. The ablation study also considers the setting of ALC in (4.3) except that $\gamma = 0$, i.e. no manifold regularization, which is termed ALC- η since only the value of η is selected. Detailed configurations of aforementioned algorithms are summarized in Table 4.6. The result of ELM based method ELM-ALC is added as a reference. For each method, the study selects the best parameters for clustering and reports the best accuracy value averaged on 10 runs.

The results on benchmark datasets are shown in Figure 4.3. Overall, ALC-*Fr* is inferior on all image datasets and most UCI datasets (except ECOLI). The gap between ALC-*Fr* and ALC- η indicates that reconstruction using the ℓ_2 norm (not squared) is effective for learning graph weights. On the other hand, the performance between ALC- η and ALC is comparable, with observable accuracy gain of ALC against ALC- η on 6 out of 12 datasets. This means that graph regularization can provide improvement on learning the optimal graph, but the improvement is not as significant as the performance boost produced by the shift to ℓ_2 norm (not squared). In fact ALC- η compares favorably to other graph learning methods in Table 4.2 and 4.3, and it is more efficient than ALC since it does not have to select the value of γ .

In the experimental setting in Section 4.4.2, the initial graph of CLR is used as the predefined smoothing graph in (4.3) for manifold regularizations in order to fairly compare with CLR [235]. Here the impact of manifold regularizations is examined with different predefined graphs. Specifically, the following graphs are used to compute the Laplacian matrix $\mathbf{L}$ in (4.3):

- BINARY: the binary k -NN graph where only nearest neighbor entries are one.
- CLR: the initial graph used in [235].
- SELFTUNE: the self-tuning Gaussian method for building a k -NN graph [242].

FIGURE 4.3: Accuracies of variants of ALC on benchmark datasets.

FIGURE 4.4: Comparisons of different graph types for manifold regularizations.

The results of applying different graph types for manifold regularizations on benchmark datasets are given in Figure 4.4. All of them provide similar results on most datasets. The CLR graph has the highest accuracy on USPST, but it performs worst on ECOLI. The SELFTUNE graph is best on CONTROL, but on VEHICLE the accuracy value is the lowest. The BINARY graph, despite its simplest form, provides results comparable to the more complex ones on all datasets. Therefore, this Chapter simply uses the initial graph described in [235] as the “default” setting, and other options (BINARY and SELFTUNE) give similar results, indicating that the use of manifold regularizations rather than a specific choice of graph will improve clustering performance.

4.4.6 Sensitivity Analysis

This part analyzes the impact of choices of parameters. The ALC formulation (4.3) has three trade-off parameters: η , γ and number of nearest neighbors k . First, the accuracy values against different values of η and γ , with k selected to give the best performance, are plotted in Figure 4.5 and 4.6. It can be seen that on several datasets (e.g. DIABETES), the performance is stable across different values. Although the accuracy is set to zero if the obtained graph has incorrect number of components, there appears almost no failure mode in Figure 4.5 and 4.6, indicating that most parameter combinations have recovered the correct number of clusters. It can be seen that on most datasets, the best result is obtained at moderate value of η and large value of γ , meaning that both regularization terms are non-negligible for effective graph learning in ALC. This behavior is also in accordance with the ablation study in Figure 4.3.

Finally, to demonstrate the effect of different numbers of nearest neighbors, the accuracies against different values of k with η and γ chosen to give the best performance are shown in Figure 4.7. For most image datasets, the best accuracy values are obtained at a small value of k , leading a relatively sparse graph representation. For UCI datasets such as DIABETES and CONTROL, the best results are achieved with larger values of k . This may be because UCI data have less number of features, and a larger amount of adjacent samples are required to yield an effective representation of each data point.

FIGURE 4.5: Accuracies of ALC on each image dataset (UMIST/ORL/POLLEN/USPST/MNIST05/ETH80) with different values of η and γ . The number of nearest neighbors is chosen to give the best performance.

FIGURE 4.6: Accuracies of ALC on each UCI dataset (ECHOLI/CARS/DIABETES/SEGMENT/VEHICLE/CONTROL) with different values of η and γ . The number of nearest neighbors is chosen to give the best performance.

FIGURE 4.7: Accuracies of ALC with different values of nearest neighbors k . The values of η and γ are chosen to give the best performance.

4.5 Summary

This chapter proposes an adaptive and locality-constrained graph learning approach for clustering. Specifically, ALC uses spectral clustering to perform clustering, and ELM-ALC utilizes ELM-ALC. To achieve this objective, Section 4.2 generalizes locality-constrained linear coding (LLC) for graph component construction, which leverages expressiveness of nearest neighbors regularized via local distances. This results in a k -NN graph where the nonzero elements are learned via adaptively applying a rank constraint on the Laplacian matrix. The weighted pair-wise distances of LLC feature descriptors are minimized with manifold regularizations to yield smooth representations among adjacent samples. The use of US-ELM for unsupervised embedding and clustering is also investigated. Section 4.3 describes optimization approaches for ALC and ELM-ALC. Section 4.4 demonstrates the effectiveness of the proposed ALC/ELM-ALC for clustering on a wide range of benchmark datasets.

Chapter 5

Slice-based Online Convolutional Dictionary Learning

Chapter 5 investigates the use of a slice-based representation to efficiently solve the problem of online convolutional dictionary learning. Section 5.1 describes current and previous approaches for convolutional dictionary learning and online convolutional dictionary learning, where their complexities and limitations are discussed. Section 5.2 proposes to utilize the localized slice-based representation to construct a surrogate objective which provides stochastic approximations of the convolutional dictionary. The proposed online learning algorithm enables efficient convolutional dictionary learning with low time and memory complexity. Additionally, a masked version of online algorithm is also proposed to learn from incomplete data. Section 5.3 empirically validates the convergence performance of the proposed method on image reconstruction task, and the capabilities of learning from masked data on image inpainting task. The contents of this chapter are based on the following paper:

- Yijie Zeng, Jichao Chen, and Guang-Bin Huang, “Slice-based Online Convolutional Dictionary Learning,” *IEEE Transactions on Cybernetics*, 2019. DOI: 10.1109/TCYB.2019.2931914.

5.1 Background and Motivations

Sparse coding and dictionary learning have become important techniques in many fields, including signal processing and computer vision [19], [80]. The task of *dictionary learning* aims to learn a set of overcomplete filters with which a given signal can be represented as their sparse combinations. The representation is useful in many fields, including image denoising [11], [82], [255], image smoothing [256], image restoration [12], [257], super resolution [252] etc. Also it has been successfully applied to various computer vision tasks, such as image classification [15], [125], face recognition [10], anomaly detection [258], etc. Despite its success in the above fields, sparse coding is a “low-level” and linear model that typically operates on small patches when dealing with large images. It first extracts overlapping patches from a set of image to create a large and redundant dataset. Each patch is represented as a linear combination of few learned dictionary atoms. As a result, the patches are processed independently regardless of the fact that they may come from the same image and are related. Moreover, the spatial configuration inside an image is lost.

To overcome these two shortcomings, *deconvolutional network* [85] or *convolutional sparse coding* (CSC) [31] is proposed. Instead of operating on small image patches, CSC applies the filters on a sparse feature map to reconstruct the whole image. The feature map has the same size as the image and is *convolved* with the filters to preserve spatial consistency. Dependencies of overlapping patches are naturally handled since *all* patches are processed together. CSC has shown superior performance on various image processing tasks, including super resolution [38], image decomposition [91], image fusion [40], image inpainting [33], [92], etc.. In computer vision, it is also popular for the task of image classification [85], [93], [94].

Since all pixels in an image are required to learn the convolutional representation, solving CSC and the corresponding convolutional dictionary learning (CDL) problem is very computationally intensive, thus recent research has been focusing on an efficient solver for CSC and CDL problem. This has also been investigated in Section 2.2.4, where most efficient solvers for CSC and CDL are operated on frequency domain using ADMM [36], [160]. The procedure of applying ADMM on frequency domain for CSC has also been demonstrated in Section 2.3.3. On the

other hand, since the convolution operation is essentially equivalent to matrix multiplication between learned filters and image patches (with boundary condition), the CSC problem can be cast alternatively as a spatially consensus optimization problem, where the local sparse “stripe vector” is solved on each image patch, and consensus is obtained over the global feature map [87]. Likewise, a locally processed convolutional dictionary learning method can be proposed with ADMM and local dictionary update [35].

In practice, a common strategy when applying ADMM for CDL problems is to interleave CSC steps with dictionary learning steps for each ADMM loop [95]. Under this setting, for each ADMM iteration, the consensus method [34], [92] has time complexity $\mathcal{O}(KNm \log(N))$, where K is the number of images, N is the image size and m is the number of learned dictionary atoms. The locally processed CDL method [35] has time complexity $\mathcal{O}(KNmn + KNm\alpha^2)$, where n is the patch size, α is the internal LARS [73] update steps for sparse coding. It can be seen that for either case the time complexity scales linearly with respect to number of samples K . This corresponds to the fact that the dictionary is shared among all sparse codes whose value has to be refreshed whenever the dictionary is updated. Such constraint is highly undesirable in practice, since it limits the size of the dataset to learn the dictionary. Current convolutional dictionary learning algorithms could handle only up to tens of images on a normal PC, while recent datasets, e.g. ImageNet [100], contain more than one million images.

Such discrepancy motivates very recent research to seek an online method for convolutional dictionary learning (see also Section 2.2.4). Under online settings, the images are fed into the algorithm in a streaming fashion and the dictionary is updated incrementally. There is no need to store all the images in memory, thus online algorithms are able to handle orders of magnitude more images than their batch counterparts. Recent online convolutional dictionary learning algorithms are mainly divided into two branches: stochastic gradient descent (SGD) based [61] and stochastic approximation (SA) based [58]–[61]. The SGD based method uses gradients computed only from current data point to update the dictionary. The SA based method constructs a surrogate function to approximate the true objective, where all history information is stored in a fixed-sized matrix. The memory consumption is therefore not scaled with time. In practice the SGD based method

has lower time complexity for each step, but SA based method converges faster especially for larger dataset since it aggregates more information.

The method proposed in this chapter falls into the second category of online convolutional dictionary learning (OCDL). Existing surrogate-based methods either have large time complexity for updating history matrix [59], or suffer from careful parameter selection introduced by ADMM [60]. The proposed method is motivated by the slice-based representation of convolutional sparse model [35], [87], [259], [260]. The objective function is transformed to an equivalent form which can be decomposed into independently small sparse coding subproblems with global constraints. Each subproblem can be solved using traditional basis pursuit denoising [69]. Contrary to existing surrogate-based OCDL methods, the proposed method is faster in terms of time complexity, and few parameters are required to be tuned. Moreover, the locally proposed sparse representation suggests a close link between the proposed OCDL method and the popular online learning method for patch-based sparse coding problems [62].

5.1.1 Contributions

The contribution of this chapter can be summarized as follows:

- It proposes a novel slice-based online convolutional dictionary learning algorithm with a locally defined representation, which can be alternatively viewed as a generalization of patch-based online dictionary learning method.
- It proves theoretically that the proposed online algorithm reduces time complexity compared to existing methods, while yielding the same results.
- It shows that the proposed method can be combined with a mask simulation technique to learn from masked or incomplete data online.

5.2 Proposed Method

This section presents the components of the proposed online algorithm for convolutional dictionary learning. The algorithm builds on top of the slice-based representations of convolutional sparse coding [35], [87], discussed in Section 5.2.1, with

FIGURE 5.1: Stripe dictionary Ω with $n = 6$ and $m = 3$.

minimization on a surrogate function approximating the expected objective [62]. Such representations have been investigated on batch CDL problems but not on an online scenario yet. The algorithm outline is presented in Section 5.2.2 and 5.2.3, followed by comparisons with ODL approach [62] (Section 5.2.4). Comparisons with OCDL approaches in terms of time complexity is given in Section 5.2.5. Convergence analysis and accelerations of the algorithm are discussed in Section 5.2.6. Finally, boundary handling with the proposed method is discussed in Section 5.2.7.

5.2.1 Slice-base Representations

Convolutional dictionary learning (CDL) aims to learn m dictionary atoms $\{\mathbf{d}_i \in \mathbb{R}^n\}_{i=1}^m$ such that a given signal $\mathbf{s} \in \mathbb{R}^N$ can be approximated by $\mathbf{s} = \sum_i \mathbf{d}_i * \mathbf{x}_i$, where $\{\mathbf{x}_i \in \mathbb{R}^N\}_{i=1}^m$ are the corresponding sparse coding vectors. Considering the CDL functional $\mathcal{L}_{\text{CDL}}$ defined in (2.41), a *local* formulation can be described as the following transformations on the convolutional form (2.41). Let $\mathbf{z}_j = [\mathbf{x}_1[j], \dots, \mathbf{x}_m[j]]^T \in \mathbb{R}^m$ and $\mathbf{D}_l = [\mathbf{d}_1, \dots, \mathbf{d}_m] \in \mathbb{R}^{n \times m}$, then

$$\sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_i = \sum_{j=1}^N \mathbf{R}_j^T \mathbf{D}_l \mathbf{z}_j. \quad (5.1)$$

Here $\mathbf{R}_j \in \mathbb{R}^{n \times N}$ is a *patch extractor* that extracts the j -th patch of length n from a signal of length N , and $\mathbf{R}_j^T \in \mathbb{R}^{N \times n}$ *aggregates* a patch of length n on the j -th position¹. Then the *local* functional $\mathcal{L}_l$ for signal $\mathbf{s}$ and dictionary $\mathbf{D}_l$ can be defined as:

¹The concrete form of $\mathbf{R}_j$ matrix depends on how convolution is performed. This chapter assumes $\mathbf{R}_j = [\mathbf{O}_{n \times (j-n)}, \mathbf{I}_{n \times n}, \mathbf{O}_{n \times (N-j)}]$.

$$\mathcal{L}_l(\mathbf{z}, \mathbf{D}_l; \mathbf{s}) = \frac{1}{2} \left\| \sum_{j=1}^N \mathbf{R}_j^T \mathbf{D}_l \mathbf{z}_j - \mathbf{s} \right\|_2^2 + \lambda \sum_{j=1}^N \|\mathbf{z}_j\|_1. \quad (5.2)$$

The *local dictionary* $\mathbf{D}_l = [\mathbf{d}_1, \dots, \mathbf{d}_m] \in \mathbb{R}^{n \times m}$ is the column concatenation matrix of dictionary $\{\mathbf{d}_i\}_{i=1}^m$, and $\{\mathbf{z}_j\}_{j=1}^N$, $\mathbf{z}_j \in \mathbb{R}^m$ is the *local sparse code*. λ is a regularization parameter controlling sparsity of $\mathbf{z}_j$. The “slices” refer to vectors $\{\mathbf{D}_l \mathbf{z}_j\}_{j=1}^N$ which are consequently aggregated to reconstruct the signal $\mathbf{s}$ [35]. The equivalence of the objective $\mathcal{L}_l$ and the canonical CDL form $\mathcal{L}_{\text{CDL}}$ (2.41) is guaranteed by (5.1). This representation bridges convolutional sparse coding with traditional patch-based sparse coding methods. Sparse coding method aims to decompose *patches* by an over-complete dictionary, and convolutional sparse coding aims to represent a global signal $\mathbf{s}$ by aggregating local *slices* $\mathbf{y}_j \triangleq \mathbf{D}_l \mathbf{z}_j$.

5.2.2 Algorithm Outline

The proposed slice-based online convolutional dictionary learning algorithm (SOCDL) is outlined in algorithm 4. Similar to the discussions in Section 2.2.3, one is usually more interested in the *expected cost* on a signal space $\mathcal{S}$:

$$F(\mathbf{D}_l) = \mathbb{E}_{\mathbf{s} \sim \mathcal{S}} \left[\min_{\mathbf{z}} \mathcal{L}_l(\mathbf{z}, \mathbf{D}_l; \mathbf{s}) \right] \quad (5.3)$$

where $\mathcal{L}_l$ can be substituted with $\mathcal{L}_{\text{CDL}}$ since they are equivalent. Assume the dataset $\mathcal{S}$ is composed of i.i.d. samples $\mathbf{s}$, and each time one sample $\mathbf{s}^{(\tau)}$ is drawn from it. At time t a surrogate function (5.4) can be constructed to approximate the expected objective (5.3):

$$\mathcal{F}^{(t)}(\mathbf{D}_l) = \frac{1}{t} \sum_{\tau=1}^t \mathcal{L}_l(\mathbf{z}^{(\tau)}, \mathbf{D}_l; \mathbf{s}^{(\tau)}). \quad (5.4)$$

When a new signal $\mathbf{s}^{(t)}$ arrives at t -th time step, the sparse code $\{\mathbf{z}_j^{(t)}\}_{j=1}^N$ and dictionary $\mathbf{D}_l$ are updated as

$$\mathbf{z}^{(t)} = \underset{\mathbf{z}}{\operatorname{argmin}} \mathcal{L}_l(\mathbf{z}, \mathbf{D}_l^{(t-1)}; \mathbf{s}^{(t)}) \quad (5.5)$$

$$\mathbf{D}_l^{(t)} = \underset{\mathbf{D}_l \in \mathbb{R}^{n \times m}}{\operatorname{argmin}} \mathcal{F}_l^{(t)}(\mathbf{D}_l) + \iota_{\mathcal{C}}(\mathbf{D}_l). \quad (5.6)$$

Here $\mathcal{C} = \{\mathbf{D}_l \in \mathbb{R}^{n \times m} \mid \|\mathbf{d}_i\|_2 \leq 1, \forall i\}$ is the constraint set.

To solve (5.5), various methods have been discussed in the previous section, and this chapter simply applies CBPDN [158] for this problem, which has time complexity of $\mathcal{O}(mN \log(N))$ for each ADMM iteration for each sample.

5.2.3 Dictionary Update

At time t the subproblem (5.6) can be represented as

$$\begin{aligned} \mathbf{D}_l^{(t)} &= \underset{\mathbf{D}_l \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{t} \sum_{\tau=1}^t \frac{1}{2} \left\| \sum_j \mathbf{R}_j^T \mathbf{D}_l \mathbf{z}_j^{(\tau)} - \mathbf{s}^{(\tau)} \right\|_2^2 \\ &= \underset{\mathbf{D}_l \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{t} \sum_{\tau=1}^t \frac{1}{2} \sum_{i,j} \mathbf{z}_j^{(\tau)T} \mathbf{D}_l^T \mathbf{R}_j \mathbf{R}_i^T \mathbf{D}_l \mathbf{z}_i^{(\tau)} - \sum_j \mathbf{z}_j^{(\tau)T} \mathbf{D}_l^T \mathbf{R}_j \mathbf{s}^{(\tau)}. \end{aligned} \quad (5.7)$$

Note that

$$\begin{aligned} \mathbf{R}_j \sum_{k=1}^N \mathbf{R}_k^T \mathbf{D}_l \mathbf{z}_k &= \sum_{u=-(n-1)}^{n-1} \mathbf{K}_{-u} \mathbf{D}_l \mathbf{z}_{j+u} \\ &\triangleq \boldsymbol{\Omega} \boldsymbol{\gamma}_j. \end{aligned} \quad (5.8)$$

The first equality utilizes the fact that $\mathbf{K}_{j-k} \triangleq \mathbf{R}_j \mathbf{R}_k^T \in \mathbb{R}^{n \times n}$ is shift invariant and is non-zero if and only if $|j - k| < n$. The *stripe vector* $\boldsymbol{\gamma}_j \in \mathbb{R}^{(2n-1)m}$ and *stripe dictionary* $\boldsymbol{\Omega} \in \mathbb{R}^{n \times (2n-1)m}$ on the second equality can be represented as:

$$\boldsymbol{\Omega} = [\mathbf{K}_{n-1} \mathbf{D}_l, \dots, \mathbf{K}_0 \mathbf{D}_l, \dots, \mathbf{K}_{-n+1} \mathbf{D}_l] \quad (5.9)$$

$$\boldsymbol{\gamma}_j = [\mathbf{z}_{j-n+1}^T, \dots, \mathbf{z}_j^T, \dots, \mathbf{z}_{j+n-1}^T]^T \quad (5.10)$$

Algorithm 4: Slice-based Online Convolutional Dictionary Learning (SOCDL)

```

1 Initialize  $\mathbf{D}_l^{(0)}$  randomly,  $\mathbf{A}^{(0)} = \mathbf{0}$ ,  $\mathbf{B}^{(0)} = \mathbf{0}$ ;
2 for  $t = 1, \dots, T$  do
3   Sample  $\mathbf{s}^{(t)}$  from  $\mathcal{S}$ ;
4   Convolutional sparse coding of  $\mathbf{z}^{(t)}$  (5.5) using CBPDN:
5   
$$\mathbf{z}^{(t)} = \operatorname{argmin}_{\mathbf{z}} \mathcal{L}_l(\mathbf{z}, \mathbf{D}_l^{(t-1)}; \mathbf{s}^{(t)})$$

6   Construct stripe vector  $\{\gamma_j\}_{j=1}^N$  (5.10) and stripe dictionary  $\mathbf{\Omega}$  (5.9);
7   Update  $\mathbf{A}^{(t)}$  and  $\mathbf{B}^{(t)}$  using (5.13);
8   Update  $\mathbf{D}_l^{(t)}$  (5.6) using algorithm 5, with gradient (5.14) and  $\mathbf{D}_l^{(t-1)}$  as
    warm start:
9   
$$\mathbf{D}_l^{(t)} = \operatorname{argmin}_{\mathbf{D}_l} \mathcal{F}_l^{(t)}(\mathbf{D}_l) + \iota_{\mathcal{C}}(\mathbf{D}_l)$$

10 return  $\mathbf{D}_l^{(T)}$ 

```

---

which can also be found in [87]. The subscript of (5.10) is understood under modulo N . An illustration of $\mathbf{D}_l$ and $\mathbf{\Omega}$ can be found in Figure 5.1. A more canonical way of finding the stripe dictionary $\mathbf{\Omega}$ and its link with shift-invariance of convolutional sparse coding can be found in [87]. Combining (5.8) and (5.7)

$$\begin{aligned}\mathbf{D}_l^{(t)} &= \underset{\mathbf{D}_l \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{t} \sum_{\tau} \sum_j \frac{1}{2} \mathbf{z}_j^{(\tau)\mathrm{T}} \mathbf{D}_l^{\mathrm{T}} \mathbf{\Omega} \gamma_j - \mathbf{z}_j^{(\tau)\mathrm{T}} \mathbf{D}_l^{(\tau)} \mathbf{s}_j^{(\tau)} \\ &= \underset{\mathbf{D}_l \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{2} \operatorname{tr}[\mathbf{D}_l^{\mathrm{T}} \mathbf{\Omega} \mathbf{A}^{(t)}] - \operatorname{tr}[\mathbf{D}_l^{\mathrm{T}} \mathbf{B}^{(t)}].\end{aligned}\quad (5.11)$$

Here $\mathbf{s}_j^{(\tau)} = \mathbf{R}_j \mathbf{s}^{(\tau)}$ is the *patches* extracted from signal $\mathbf{s}^{(\tau)}$ at location j . $\mathbf{A}^{(t)} \in \mathbb{R}^{(2n-1)m \times m}$ and $\mathbf{B}^{(t)} \in \mathbb{R}^{n \times m}$ are defined as follows:

$$\begin{aligned}\mathbf{A}^{(t)} &= (1/t) \sum_{\tau=1}^t \sum_{j=1}^N \gamma_j^{(\tau)} \mathbf{z}_j^{(\tau)\mathrm{T}} \\ \mathbf{B}^{(t)} &= (1/t) \sum_{\tau=1}^t \sum_{j=1}^N \mathbf{s}_j^{(\tau)} \mathbf{z}_j^{(\tau)\mathrm{T}}\end{aligned}\quad (5.12)$$

The objective (5.11) is very similar to that of [62] for standard dictionary learning problems and [58]–[61] for convolutional dictionary learning. More detailed comparisons will be made in Section 5.2.4 and Section 5.2.5. Some remarks are made here:

- Unlike batch CDL algorithms where *all* sparse codes $\{\mathbf{z}^{(\tau)}\}_{\tau=1}^t$ have to be updated, at each time step, only one sparse code $\mathbf{z}^{(t)}$ from $\mathbf{s}^{(t)}$ are required for the online CDL algorithm.
- The dictionary $\mathbf{D}_l$ is updated using all information from previously learned sparse codes. Nevertheless, it does not require storing all sparse vectors $\{\mathbf{z}^{(\tau)}\}_{\tau=1}^t$, but only $\mathbf{A}^{(t)}$ and $\mathbf{B}^{(t)}$ which have fixed size. Moreover, they can be updated incrementally at each step:

$$\begin{aligned}\mathbf{A}^{(t)} &= \frac{t-1}{t} \mathbf{A}^{(t-1)} + \frac{1}{t} \sum_{j=1}^N \gamma_j^{(t)} \mathbf{z}_j^{(t)\mathrm{T}} \\ \mathbf{B}^{(t)} &= \frac{t-1}{t} \mathbf{B}^{(t-1)} + \frac{1}{t} \sum_{j=1}^N \mathbf{s}_j^{(t)} \mathbf{z}_j^{(t)\mathrm{T}}.\end{aligned}\quad (5.13)$$

It remains to solve optimization problem (5.11) at time t . This chapter applies the *Fast Iterative Shrinkage Thresholding Algorithm (FISTA)* [72]. The FISTA algorithm for solving problem (5.11) is given in algorithm 5. The gradient of the objective (i.e. $\mathcal{F}^{(t)}$) with respect to $\mathbf{D}_l$ is given by

$$\nabla_{\mathbf{D}_l} \mathcal{F}^{(t)}(\mathbf{D}_l) = \mathbf{\Omega} \mathbf{A}^{(t)} - \mathbf{B}^{(t)}. \quad (5.14)$$

The projection of each column $\mathbf{d}_j$ of $\mathbf{D}_l$ to the ℓ_2 ball $\mathcal{C}$ is given by:

$$\text{Proj}_{\mathcal{C}}(\mathbf{d}_j) = \frac{1}{\max(\|\mathbf{d}_j\|_2, 1)} \mathbf{d}_j. \quad (5.15)$$

The stopping criterion in algorithm 5 is defined as the fixed point residual (FPR) [61], [261] being lower than a pre-specified threshold ϵ , which is defined as

$$\text{FPR}^{(t)} \triangleq \|\mathbf{G}_{j+1} - \mathbf{Y}_j\|. \quad (5.16)$$

It should be noted that other iterative methods are also available, including Newton's method [78], block coordinate descent (BCD) [62], ADMM [60], etc. ADMM [90] algorithm requires very careful parameter selection [145] for fast convergence (as discussed in Section 2.3), and Newton's method requires inverting an $nm \times nm$ matrix at each time step. For BCD in [62], each column $\mathbf{d}_j$ of $\mathbf{D}_l$ is updated as

$$\mathbf{d}_j \leftarrow \text{Proj}_{\mathcal{C}}\left(\mathbf{d}_j - \alpha \mathbf{H}_j^{-1}(\mathbf{\Omega} \mathbf{a}_j - \mathbf{b}_j)\right), \quad (5.17)$$

where $\mathbf{a}_j$ and $\mathbf{b}_j$ is the j -th column of $\mathbf{A}^{(t)}$ and $\mathbf{B}^{(t)}$ respectively, and $\mathbf{H}_j \in \mathbb{R}^{n \times n}$ is the Hessian matrix corresponding to $\mathbf{d}_j$. α is the step size. Compared with (5.14), the above update rule requires computing the inverse of m matrices of size $n \times n$, whereas FISTA only requires first order gradients.

Algorithm 5: FISTA for solving dictionary update (5.11)

Input : Matrix $\mathbf{A}^{(t)}$ and $\mathbf{B}^{(t)}$, dictionary $\mathbf{D}_l^{(t-1)}$.

Initialize: $\mathbf{G}_0 = \mathbf{D}_l^{(t-1)}$, $\mathbf{Y}_0 = \mathbf{G}_0$, $t_0 = 1$.

1 **repeat**

2   Construct the stripe dictionary $\mathbf{\Omega}_j$ from $\mathbf{Y}_j$ using (5.9);

3   Compute gradient:

$$\nabla \mathcal{F}^{(t)}(\mathbf{Y}_j) = \mathbf{\Omega}_j \mathbf{A}^{(t)} - \mathbf{B}^{(t)}.$$

      Compute the next iterate with (5.15):

$$\mathbf{G}_{j+1} = \text{Proj}_{\mathcal{C}} \left(\mathbf{Y}_j - (1/L) \nabla \mathcal{F}_l^{(t)}(\mathbf{Y}_j) \right).$$

      Update the auxiliary variable $\mathbf{Y}_{j+1}$ with $t_{j+1} = (1 + \sqrt{1 + 4t_j^2})/2$:

$$\mathbf{Y}_{j+1} = \mathbf{G}_{j+1} + \frac{t_{j+1} - t_j}{t_j} (\mathbf{G}_{j+1} - \mathbf{G}_j).$$

4 **until** *stopping criterion is met*;

Output : $\mathbf{G}_J$, where J is the last iteration.

5.2.4 Comparison with Patch-based Online Dictionary Learning

In this part the proposed algorithm is compared with online dictionary learning (ODL) [62], which operates on image patches rather than the whole image. Consider image patches $\{\mathbf{s}_j^{(\tau)} | j \in I_\tau\}$ sampled from the image $\mathbf{s}^{(\tau)}$ at time step τ , and the corresponding sparse codes $\{\mathbf{z}_j^{(\tau)} | j \in I_\tau\}$ computed for each patch via LARS [73], where I_τ is the index set at τ -th step. The dictionary update problem at time t is given by

$$\mathbf{D}_l^{(t)} = \underset{\mathbf{D}_l \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{2} \operatorname{tr}[\mathbf{D}_l^T \mathbf{D}_l \mathbf{A}_{\text{odl}}^{(t)}] - \operatorname{tr}[\mathbf{D}_l^T \mathbf{B}_{\text{odl}}^{(t)}], \quad (5.18)$$

and the gradient of surrogate $\mathcal{F}_{\text{odl}}^{(t)}$ w.r.t. the dictionary $\mathbf{D}_l$ is given by

$$\begin{aligned} \nabla_{\mathbf{D}_l} \mathcal{F}_{\text{odl}}^{(t)} &= (1/t) \sum_{\tau=1}^t \sum_{j \in I_\tau} \mathbf{D}_l \mathbf{z}_j^{(\tau)} \mathbf{z}_j^{(\tau)T} - \mathbf{s}_j^{(\tau)} \mathbf{z}_j^{(\tau)T} \\ &\triangleq \mathbf{D}_l \mathbf{A}_{\text{odl}}^{(t)} - \mathbf{B}_{\text{odl}}^{(t)}. \end{aligned} \quad (5.19)$$

It can be seen that (5.14) and (5.19) have very similar form. Specifically, when I_τ iterates over all patches in $\mathbf{s}^{(\tau)}$ and assuming the sparse codes are the same, the two matrices $\mathbf{B}^{(t)}$ and $\mathbf{B}_{\text{odl}}^{(t)}$ are equal and the central $m \times m$ block of $\mathbf{A}^{(t)}$ is equal to $\mathbf{A}_{\text{odl}}^{(t)}$. The difference between two algorithms is that for ODL the local dictionary $\mathbf{D}_l$ is used while for SOCDL, the stripe dictionary $\mathbf{\Omega}$ is used which encodes contributions of the sparse code at current location as well as at neighboring ones. Therefore, one can regard SOCDL as a direct extension to the patch-based ODL method with additional shift invariance property.

5.2.5 Complexity Analysis and Comparison with OCDL algorithms

In this part a comparison is made between the proposed algorithm and other surrogate-based online convolutional dictionary learning algorithms [58], [60], [61].

TABLE 5.1: An overview of parameters for SOCDL.

Parameter	Explanation
n	The size of dictionary atoms.
m	Number of atoms in the dictionary.
N	The size of signals.
K	Number of signals.
λ	Regularization parameter for sparsity term.
p	Forgetting factor adjusting weights for each samples in (5.38).
L	Parameters controlling step size of FISTA in Algorithm 5.
T	Number of iterations for SOCDL.
ϵ_{CBPDN}	Stopping tolerance for CBPDN algorithm.
ϵ_{FISTA}	Stopping tolerance for FISTA of Algorithm 5.

All of them construct a surrogate function² to approximate the expected cost³ (5.3):

$$\mathcal{F}^{(t)}(\mathbf{d}) = (1/t) \sum_{\tau=1}^t \mathcal{L}_{\text{CDL}}(\mathbf{x}^{(\tau)}, \mathbf{d}; \mathbf{s}^{(\tau)}). \quad (5.20)$$

The time complexity is dominated by three parts: convolutional sparse coding (5.5), update of $(\mathbf{A}^{(t)}, \mathbf{B}^{(t)})$ (5.13), and dictionary update (5.6). Assume CSC is solved by the same methods with same time complexity $\mathcal{T}_{\text{CSC}}$. For Algorithm 4, updating $(\mathbf{A}^{(t)}, \mathbf{B}^{(t)})$ takes $\mathcal{O}(nm^2K)$ for N components and batch size of K , in total $\mathcal{O}(nm^2KN)$ operations. For FISTA the matrix multiplication in (5.13) takes $\mathcal{O}(n^2m^2)$ for each iteration.

5.2.5.1 Spatial Surrogate Algorithms

When the convolution is represented in *operator form*, i.e. $\sum_i \mathbf{d}_i * \mathbf{x}_i = \mathbf{X}\mathbf{d}$ where $\mathbf{X} \in \mathbb{R}^{N \times nm}$ is a matrix representing $\{\mathbf{x}_i\}_{i=1}^m$, the gradient of (5.20) w.r.t. $\mathbf{d}$ can be given as [58], [61]

$$\begin{aligned} \nabla_{\mathbf{d}} \mathcal{F}^{(t)} &= \mathbf{A}^{(t)} \mathbf{d} - \mathbf{B}^{(t)}, \text{ where} \\ \mathbf{A}^{(t)} &= (1/t) \sum_{\tau} \mathbf{X}^{(\tau)\text{T}} \mathbf{X}^{(\tau)}, \quad \mathbf{B}^{(t)} = (1/t) \sum_{\tau} \mathbf{X}^{(\tau)\text{T}} \mathbf{s}^{(\tau)} \end{aligned} \quad (5.21)$$

²For ease of discussion, in this section the dictionary atoms are represented by the *vector form* $\mathbf{d} \triangleq [\mathbf{d}_1^{\text{T}}, \dots, \mathbf{d}_m^{\text{T}}]^{\text{T}} \in \mathbb{R}^{nm}$ rather than the *matrix form* $\mathbf{D} \triangleq [\mathbf{d}_1, \dots, \mathbf{d}_m] \in \mathbb{R}^{n \times m}$.

³The expected cost (5.3) is defined w.r.t. the local CDL functional $\mathcal{L}_l$. Since $\mathcal{L}_l$ and $\mathcal{L}_{\text{CDL}}$ are equivalent, this section simply uses the same cost objective F to simplify the discussion.

TABLE 5.2: Comparison for time and space complexity of proposed algorithm with existing online CDL algorithms. N is the signal size, n is the size for each dictionary atom, m is the number of dictionary atoms, K is the number of samples for each batch, and J is the iteration number.

Method	CSC	Matrix Update	Dictionary Update	Memory
OCDL-Frequency [59], [61]	$\mathcal{T}_{\text{csc}}$	$\mathcal{O}(m^2KN + mKN \log(N))$	$J \times \mathcal{O}(m^2N + mN \log(N))$	$\mathcal{O}(m^2N)$
OCSC [60]	$\mathcal{T}_{\text{csc}}$	$\mathcal{O}(m^2KN + mKN \log(N))$	$J \times \mathcal{O}(m^2N + mN \log(N))$	$\mathcal{O}(m^2N)$
OCDL-Spatial [58], [61]	$\mathcal{T}_{\text{csc}}$	$\mathcal{O}(n^2m^2KN)$	$J \times \mathcal{O}(n^2m^2)$	$\mathcal{O}(n^2m^2)$
SOC DL	$\mathcal{T}_{\text{csc}}$	$\mathcal{O}(nm^2KN)$	$J \times \mathcal{O}(n^2m^2)$	$\mathcal{O}(nm^2)$

and they update the dictionary with FISTA as well. Note that since the locally defined objective $\mathcal{L}_l$ in (5.2) is equivalent to the convolutional form $\mathcal{L}_{\text{CDL}}$ introduced in (2.41), the two surrogate function $\mathcal{F}$ in (5.20) and $\mathcal{F}_l$ in (5.4) are equal. This indicates that with the same data at each iteration, spatial surrogate algorithms will yield *exactly* the same dictionary as the proposed algorithm. However, notice that in (5.21) matrix update takes $\mathcal{O}(n^2 m^2 KN)$ and FISTA takes $\mathcal{O}(n^2 m^2)$ for each step. Therefore the proposed algorithm gives faster convergence than spatial surrogate methods with same performance.

5.2.5.2 Frequency Surrogate Algorithms

When convolution is computed in Fourier domain: $\text{FFT}\{\sum \mathbf{d}_i * \mathbf{x}_i\} = \sum \hat{\mathbf{X}}_i \hat{\mathbf{d}}_i = \hat{\mathbf{X}} \hat{\mathbf{d}}$, where $\hat{\mathbf{X}} \in \mathbb{R}^{N \times mN}$ is the concatenation of m diagonal matrices $\hat{\mathbf{X}}_i \in \mathbb{R}^{N \times N}$, the gradient of functional (2.41) can also be computed with *conjugate cogradient* on the Fourier domain [61]:

$$\nabla_{\mathbf{d}} \mathcal{L}_{\text{CDL}} = \text{IFFT}\left\{\nabla_{\hat{\mathbf{d}}}(1/2)\|\hat{\mathbf{X}}\hat{\mathbf{d}} - \hat{\mathbf{s}}\|_2^2\right\}.$$

In this case, it is possible to compute the conjugate cogradient of an equivalent surrogate function $\hat{\mathcal{F}}^{(t)}$ on Fourier domain:

$$\begin{aligned} \nabla_{\hat{\mathbf{d}}} \hat{\mathcal{F}}^{(t)} &= \hat{\mathbf{A}}^{(t)} \hat{\mathbf{d}} - \hat{\mathbf{B}}^{(t)}, \text{ where} \\ \hat{\mathbf{A}}^{(t)} &= (1/t) \sum_{\tau} \hat{\mathbf{X}}^{(\tau)\text{H}} \hat{\mathbf{X}}^{(\tau)}, \quad \hat{\mathbf{B}}^{(t)} = (1/t) \sum_{\tau} \hat{\mathbf{X}}^{(\tau)\text{H}} \hat{\mathbf{s}}^{(\tau)} \end{aligned}$$

and $\hat{\mathbf{d}}$ is computed via FISTA. The matrix update takes $\mathcal{O}(m^2 KN + mKN \log(N))$, and the FISTA step takes $\mathcal{O}(m^2 N + mN \log(N))$ for each step, utilizing sparse structure of $\hat{\mathbf{X}}^{(\tau)}$.

Alternatively, [60] discover that each row of $\hat{\mathbf{D}} = [\hat{\mathbf{d}}_1, \dots, \hat{\mathbf{d}}_m] \in \mathbb{R}^{N \times m}$ could be independently optimized with joint constraint. This results in maintaining N history matrix pairs $\hat{\mathbf{A}}_p^{(t)} \in \mathbb{R}^{m \times m}$ and $\hat{\mathbf{B}}_p^{(t)} \in \mathbb{R}^m$, and their proposed ADMM dictionary update algorithm has the same complexity as above.

A complete summary of time and space complexity for different algorithms are listed in Table 5.2. In general, the matrix update step for frequency based method

is faster than the proposed method, and dictionary update is slower. Moreover, the proposed method is more memory efficient in term of memory requirements for dictionary update. Also take note of comparisons in [61, Figure 11] which indicates that spatial surrogate algorithms are empirically faster.

5.2.6 Convergence and Acceleration

This part focuses on the main convergence result, which follows the same treatment in [61], [62]. The hope is that the sequence $\{\mathbf{D}_l^{(t)}\}$ generated by algorithm 4 should converge to minimizers of the expected cost (5.3). The following assumptions are made which are the same as those in [61]:

1. The signals $\mathbf{s}$ are drawn from a distribution p_S with a compact support.
2. A sufficient condition is satisfied to guarantee that the solution to the CSC problem (5.5) is unique.
3. The surrogate function $\mathcal{F}^{(t)}$ is strongly convex.

Some elaborations are needed regarding the sufficient condition before its formulation as Condition 5.1. The interpretation of the sufficient condition for Uniqueness of (5.5) is in essence equivalent to the treatment in [62, assumption (C)] for standard sparse coding and [61, appendix D.1.] for convolutional sparse coding. The optimality condition states that $\mathbf{z}$ is the minimizer of (5.2) if and only if

$$-\mathbf{D}_l^T \mathbf{R}_i (s - \sum \mathbf{R}_j^T \mathbf{D}_l \mathbf{z}_j) + \lambda \mathbf{u}_i = 0, \forall i, \quad (5.22)$$

where $\mathbf{u}_i \in \partial \|\mathbf{z}_i\|_1$. For arbitrary i this can further be represented as

$$\mathbf{D}_l^T \mathbf{\Omega} \boldsymbol{\gamma}_i = \mathbf{D}_l^T \mathbf{s}_i - \lambda \mathbf{u}_i \quad (5.23)$$

using symbols in Section 5.2.3. Since $\boldsymbol{\gamma}_i$ and $\boldsymbol{\gamma}_j$ may overlap, the above equation cannot be solved independently. Nevertheless, the linear equations can be gathered into a compact one:

$$\mathbf{M} \mathbf{z} = \mathbf{r} - \lambda \mathbf{u}, \quad (5.24)$$

where $\mathbf{r} = [\mathbf{s}_1^T \mathbf{D}_1, \dots, \mathbf{s}_N^T \mathbf{D}_1]^T$ and $\mathbf{u} \in \partial \|\mathbf{z}\|_1$. It is easy to check that $\mathbf{M}$ is positive semi-definite (actually $\mathbf{M} = \mathbf{D}^T \mathbf{D}$ where $\mathbf{D}$ is the *operator form* for $\{\mathbf{d}_i\}$ in convolution). Define Λ as the index set for non-zero elements of $\mathbf{z}$, then the minimizer $\mathbf{z}$ is unique if $\mathbf{M}_\Lambda$ is invertible, which is the matrix $\mathbf{M}$ with row and column indices not in Λ removed. The unique solution is

$$\mathbf{z}_\Lambda = \mathbf{M}_\Lambda^{-1}(\mathbf{r}_\Lambda - \lambda \text{sign}(\mathbf{z}_\Lambda)). \quad (5.25)$$

The subscript $(\cdot)_\Lambda$ indicates a vector with entries not in Λ removed, and $\mathbf{z}_i = 0$ for i not belonging to the index set Λ . Therefore, the sufficient condition can be formally stated as the following:

Condition 5.1. *For all $\mathbf{s} \in \mathcal{S}$ and all feasible dictionaries $\mathbf{D}_l \in \mathcal{C}$, the smallest eigenvalue of $\mathbf{M}_\Lambda$ is greater than or equal to a positive number κ .*

Given these assumptions a convergence result can be proved which guarantees that minimization of the approximated surrogate function (5.4) indeed yields the desired solution. Prior to that, some lemmas can be established:

Lemma 5.1. *Assume that the three assumptions in Section 5.2.6 hold, then for $\mathbf{D}_l \in \mathcal{C}$ and $\mathbf{s} \in \mathcal{S}$, the function*

$$\ell(\mathbf{D}_l, \mathbf{s}) = \min_{\mathbf{z}} \mathcal{L}_l(\mathbf{z}, \mathbf{D}_l; \mathbf{s}) \quad (5.26)$$

is continuously differentiable w.r.t. $\mathbf{D}_l$.

Proof. This is the direct consequence of assumption 3 and [62, Theorem 5]⁴. $\square$

Lemma 5.2. *Assume that the three assumptions in Section 5.2.6 hold, then for $\mathbf{D}_l \in \mathcal{C}$ and $\mathbf{s} \in \mathcal{S}$, the function*

$$\mathbf{z}(\mathbf{D}_l, \mathbf{s}) = \underset{\mathbf{z}}{\text{argmin}} \mathcal{L}_l(\mathbf{z}, \mathbf{D}_l; \mathbf{s}) \quad (5.27)$$

is uniformly Lipschitz on $\mathcal{C} \times \mathcal{S}$.

Proof. The proof is inspired by [62, Proposition 2] and [17, Proposition 3]. Since $\mathcal{C} \times \mathcal{S}$ is compact, one only needs to prove $\mathbf{z}$ is locally Lipschitz. Noting that $\mathbf{z}$ is

⁴Notice that Theorem 5 in [62] is adopted from [262] and originates in [263].

continuous in $\mathbf{D}_l$ and $\mathbf{s}$, there exists an open neighborhood $\mathbf{V}$ around $(\mathbf{D}_l, \mathbf{s})$ such that for all $(\mathbf{D}'_l, \mathbf{s}')$ in $\mathbf{V}$, $\mathbf{z}' = \operatorname{argmin} \mathcal{L}_l(\mathbf{z}, \mathbf{D}'_l; \mathbf{s}')$ has the same index set Λ as $\mathbf{z}$, and the sign pattern is the same. Using (5.22):

$$-\mathbf{D}'_l{}^T \mathbf{R}_i(\mathbf{s}' - \sum \mathbf{R}_j^T \mathbf{D}'_l \mathbf{z}'_j) + \lambda \mathbf{u}'_i = 0, \forall i. \quad (5.28)$$

Noting that $\mathbf{u}_{i, \Lambda_i} = \mathbf{u}'_{i, \Lambda_i}$, where Λ_i is the index set for $\mathbf{z}_i$ (and $\mathbf{z}'_i$), the rows of (5.22) and (5.28) are limited to Λ_i and subtract each other. Omitting tedious derivations, the final result is

$$\mathbf{D}_{l, \Lambda_i}^T \boldsymbol{\Omega}_{\Lambda_i}(\gamma_{i, \Lambda_i} - \gamma'_{i, \Lambda_i}) = \mathbf{v}_{i, \Lambda_i}, \forall i. \quad (5.29)$$

Here $\mathbf{v}_i \in \mathbb{R}^m$ is a vector constrained by

$$\|\mathbf{v}_i\| \leq c_1 \|\mathbf{D}_l - \mathbf{D}'_l\|_F + c_2 \|\mathbf{s} - \mathbf{s}'\|, \quad (5.30)$$

where c_1 and c_2 are constants irrelevant to $\mathbf{D}_l, \mathbf{D}'_l, \mathbf{s}, \mathbf{s}'$. Also the left hand side of (5.29) is exactly rows of (5.24), so aggregating all indices i gives

$$\mathbf{M}_\Lambda(\mathbf{z}_\Lambda - \mathbf{z}'_\Lambda) = \mathbf{v}_\Lambda, \quad (5.31)$$

where $\mathbf{v} = [\mathbf{v}_1^T, \dots, \mathbf{v}_N^T]^T$. Using the sufficient condition 5.1, one has

$$\begin{aligned} \|\mathbf{z} - \mathbf{z}'\| &\leq (1/\kappa) \|\mathbf{v}_\Lambda\| \\ &\leq (c'_1/\kappa) \|\mathbf{D}_l - \mathbf{D}'_l\|_F + (c'_2/\kappa) \|\mathbf{s} - \mathbf{s}'\|. \end{aligned} \quad (5.32)$$

This shows that $\mathbf{z}$ is locally Lipschitz, thus the proof is complete. $\square$

Lemma 5.3. *Assume that the three assumptions in Section 5.2.6 hold, then*

$$\|\mathbf{D}_l^{(t+1)} - \mathbf{D}_l^{(t)}\| = \mathcal{O}(1/t) \quad a.s. \quad (5.33)$$

Proof. This is direct since (5.11) and (5.18) from [62] have a similar form. Given $\mathcal{F}^{(t)}(\mathbf{D}_l) = \frac{1}{2} \operatorname{tr}[\mathbf{D}_l^T \boldsymbol{\Omega} \mathbf{A}^{(t)}] - \operatorname{tr}[\mathbf{D}_l^T \mathbf{B}^{(t)}]$ and defining $\mathcal{G}^{(t)} = \mathcal{F}^{(t)} - \mathcal{F}^{(t+1)}$, one has

$$\mathcal{G}^{(t)}(\mathbf{D}_l) = \frac{1}{2} \operatorname{tr}[\mathbf{D}_l^T \boldsymbol{\Omega}(\mathbf{A}^{(t)} - \mathbf{A}^{(t+1)})] - \operatorname{tr}[\mathbf{D}_l^T(\mathbf{B}^{(t)} - \mathbf{B}^{(t+1)})]. \quad (5.34)$$

Using the fact that $\mathbf{D}_l \in \mathcal{C}$, it can be seen that $\mathbf{\Omega}$ is bounded by a constant c_0 regardless of t . Besides, $\mathbf{A}^{(t)}$ and $\mathbf{B}^{(t)}$ are also uniformly bounded using assumption 1. Therefore, the function $\mathcal{G}^{(t)}$ is Lipschitz with constant $c_t = (c_0/2)\|\mathbf{A}^{(t)} - \mathbf{A}^{(t+1)}\| + \|\mathbf{B}^{(t)} - \mathbf{B}^{(t+1)}\|$.

To prove (5.3), observe that

$$\begin{aligned} \mathcal{F}^{(t)}(\mathbf{D}_l^{(t+1)}) - \mathcal{F}^{(t)}(\mathbf{D}_l^{(t)}) &= \left[\mathcal{F}^{(t)}(\mathbf{D}_l^{(t+1)}) - \mathcal{F}^{(t+1)}(\mathbf{D}_l^{(t+1)}) \right] + \\ &\quad \left[\mathcal{F}^{(t+1)}(\mathbf{D}_l^{(t+1)}) - \mathcal{F}^{(t+1)}(\mathbf{D}_l^{(t)}) \right] + \left[\mathcal{F}^{(t+1)}(\mathbf{D}_l^{(t)}) - \mathcal{F}^{(t)}(\mathbf{D}_l^{(t)}) \right] \\ &\leq \left[\mathcal{F}^{(t)}(\mathbf{D}_l^{(t+1)}) - \mathcal{F}^{(t+1)}(\mathbf{D}_l^{(t+1)}) \right] + \left[\mathcal{F}^{(t+1)}(\mathbf{D}_l^{(t)}) - \mathcal{F}^{(t)}(\mathbf{D}_l^{(t)}) \right] \\ &= \mathcal{G}^{(t)}(\mathbf{D}_l^{(t+1)}) - \mathcal{G}^{(t)}(\mathbf{D}_l^{(t)}) \\ &\leq c_t \|\mathbf{D}_l^{(t+1)} - \mathbf{D}_l^{(t)}\| \end{aligned} \tag{5.35}$$

The first inequality holds since $\mathbf{D}_l^{(t+1)}$ is the minimizer of $\mathcal{F}^{(t+1)}$. The second inequality is due to the fact that $\mathcal{G}^{(t)}$ is Lipschitz with c_t .

On the other hand, using assumption 3,

$$\mathcal{F}^{(t)}(\mathbf{D}_l^{(t+1)}) - \mathcal{F}^{(t)}(\mathbf{D}_l^{(t)}) \geq c_1 \|\mathbf{D}_l^{(t+1)} - \mathbf{D}_l^{(t)}\|^2 \tag{5.36}$$

holds for some constant c_1 . Combining (5.35) and (5.36),

$$\|\mathbf{D}_l^{(t+1)} - \mathbf{D}_l^{(t)}\| \leq c_t/c_1. \tag{5.37}$$

Finally, observing the update rule (5.13), it holds that $\|\mathbf{A}^{(t)} - \mathbf{A}^{(t+1)}\| = \mathcal{O}(1/t)$ and $\|\mathbf{B}^{(t)} - \mathbf{B}^{(t+1)}\| = \mathcal{O}(1/t)$ a.s., which indicates that $\|\mathbf{D}_l^{(t+1)} - \mathbf{D}_l^{(t)}\| = \mathcal{O}(1/t)$ almost surely. $\square$

Having the above lemmas at hand, the following convergence theorem regarding the sequence $\{\mathbf{D}_l^{(t)}\}$ can be derived:

Theorem 5.1. *Suppose assumptions 1 to 3 are satisfied. Let $\mathcal{F}^{(t)}(\mathbf{D}_l)$ denote the surrogate function (5.4), $F(\mathbf{D}_l)$ denote the expected cost (5.3), and $\{\mathbf{D}_l^{(t)}\}$ the dictionaries learned from algorithm 4, then when t goes to infinity,*

1. $\mathcal{F}^{(t)}(\mathbf{D}_l^{(t)})$ converges almost surely;
2. $F(\mathbf{D}_l^{(t)}) - \mathcal{F}^{(t)}(\mathbf{D}_l^{(t)})$ converges almost surely to 0;

3. $F(\mathbf{D}_l^{(t)})$ converges almost surely;
4. The distance between $\mathbf{D}_l^{(t)}$ and the set of stationary points of F converges to 0 almost surely.

Proof. When Lemma 5.1, 5.2 and 5.3 hold, the same treatment in [62, Lemma 1] can be directly applied, and the four convergence results are proved. $\square$

Accelerations of algorithm 4 are possible, and several computational issues are observed in [61]. Some of them are discussed in the following:

- It is found in [61] that the single step complexity is large for large input image s , and they propose to split the images into small ones to operate on. However, the splitting increases the number of images, and total amount of work is not reduced.
- Besides, they observe that OCDL algorithms often requires large FISTA iterations, and propose a diminishing tolerance scheme to decrease the threshold for FPR (5.16) for each time step. Although such scheme is effective in reducing FISTA iterations, in practice the FISTA update is fast compared to CSC step and matrix update step (Table 5.2), so optimizing on FISTA only results in marginal improvements.

This work only applies one acceleration approach, i.e. reweighing past data. In (5.4), the surrogate function $\mathcal{F}^{(t)}$ assigns equal weights to sparse codes $\{\mathbf{z}^{(\tau)}\}_{\tau=1}^t$, which is in turn obtained via “inaccurate” version of $\mathbf{D}_l$. It is proposed in [61], [62] that assigning smaller weights to older samples, and larger ones on more recent ones results in faster convergence, especially for large datasets. This amounts to use an exponential decay factor with power p for past information:

$$\tilde{\mathcal{F}}^{(t)}(\mathbf{D}_l) = \frac{1}{P^{(t)}} \sum_{\tau=1}^t (\tau/t)^p \mathcal{L}_l(\mathbf{z}^{(\tau)}, \mathbf{D}_l; \mathbf{s}^{(\tau)}) \quad (5.38)$$

where $P^{(t)} = \sum_{\tau=1}^t (\tau/t)^p$. $p > 0$ is the “forget factor”, and $p = 0$ corresponds to the original $\mathcal{F}$. It is shown in [61] that Theorem 5.1 also holds for the modified version of the surrogate function (5.38) and the corresponding learned dictionary $\mathbf{D}_l$. The effect of the forget factor will be empirically verified in Section 5.3.

5.2.7 Handling Boundaries and Learning from Incomplete Data

Boundary conditions and incomplete data can both be handled via incorporating a binary mask into the functional $\mathcal{L}_{\text{CDL}}$ [33], [89]:

$$\mathcal{L}_{\text{mask}}(\mathbf{x}, \mathbf{D}_l; \mathbf{s}, \mathbf{W}) = \frac{1}{2} \left\| \mathbf{W} \odot \left(\sum_{i=1}^m \mathbf{d}_i * \mathbf{x}_i - \mathbf{s} \right) \right\|_2^2 + \lambda \sum_{i=1}^m \|\mathbf{x}_i\|_1, \quad (5.39)$$

where $\mathbf{W} \in \mathbb{R}^N$ is a binary mask. The masked CSC is solved via *Mask Decoupling* [33], [89], and for dictionary learning, the mask $\mathbf{W}$ can be incorporated naturally in gradients of the surrogate function $\mathcal{F}^{(t)}$ [61]. However, such approach is impossible for the proposed local slice-based method since the mask is not shift-invariant. An alternative masked CDL representation is called *Additive Mask Simulation* (AMS) [89]:

$$\mathcal{L}_{\text{AMS}}(\mathbf{x}, \mathbf{D}_l; \mathbf{s}, \mathbf{W}) = \frac{1}{2} \left\| \sum_{i=0}^m \mathbf{d}_i * \mathbf{x}_i - \mathbf{s} \right\|_2^2 + \lambda \sum_{i=1}^m \|\mathbf{x}_i\|_1 + \iota_{\mathcal{W}}(\mathbf{x}_0). \quad (5.40)$$

Here $\mathbf{d}_0$ is an impulse response and $\mathcal{W} = \{\mathbf{x} \in \mathbb{R}^N \mid \mathbf{W}\mathbf{x} = \mathbf{0}\}$ constrains masked positions of $\mathbf{W}$ to zero. Equation (5.40) also simulates the same mask $\mathbf{W}$ when minimizing over $\{\mathbf{x}_i\}_{i=0}^m$ since $\mathbf{x}_0$ is unbounded at unmasked locations.

To generalize the SOCDL algorithm to the masked data, at each time step CBPDN with AMS (5.40) is applied to solve the CSC problem, then $\tilde{\mathbf{s}} = \mathbf{s} - \mathbf{d}_0 * \mathbf{x}_0$ is computed as the target signal to compute the gradients w.r.t. $\{\mathbf{d}_i\}_{i=1}^m$ as in (5.14). The proposed masked OCDL algorithm is named SOCDL-Mask, and a comparison of its performance with others will be given in the next section.

5.3 Experiments

This section compares the proposed algorithm with others on the task of *image reconstruction* and *image inpainting*. The datasets are listed in Table 5.3. The performance is validated on small datasets (*Fruit* and *City* [85]) as well as large ones (*CIFAR10* [264] and *STL10* [16]) in Section 5.3.2, followed by comparisons

TABLE 5.3: Datasets Used for Experiments

Name	No. train	No. test	Size
Fruit	10	4	100×100
City	10	4	100×100
CIFAR10	50000	10000	32×32
STL10	500	8000	96×96
VOC2007	5011	4952	—

between the proposed slice-based OCDL and equivalent online counterparts, as well as validations of acceleration approaches. This section also investigates the task of image inpainting by incorporating the proposed algorithm with mask simulation techniques and compare the performance on large image dataset (*PASCAL VOC2007* [265]).

Note that the experiments do not include large-scale datasets with large image sizes because larger images will greatly increase the time on CSC and consequently decrease the effect of online dictionary update comparisons among different algorithms. Besides, such datasets often require a large number of dictionary atoms m to capture the image statistics, which typically results in quadratic growth on time complexity of dictionary updates (see Table 5.2). In general, all previous online CDL approaches, including [58]–[61] and the proposed SOCDL, are not suitable for large-scale image datasets. A possible extension to existing OCDL methods are to update the dictionary via *slices*, and to develop an asynchronous algorithm to make update steps accessible with acceptable memory footprints.

The performance is measured by the value of the objective function (2.41) on test images minimized via CSC and trained dictionary:

$$\ell_{\text{recon}}(\mathbf{D}_l) = \sum_{\mathbf{s} \in \mathcal{S}_{\text{test}}} \min_{\mathbf{x}} \mathcal{L}_{\text{CDL}}(\mathbf{x}, \mathbf{D}_l; \mathbf{s}), \quad (5.41)$$

where the dictionary $\mathbf{D}_l$ is obtained at each time step on the training images. The parameters for the SOCDL algorithm are listed in Table 5.1. For tunable parameters such as p and L , they are selected via cross validation on a smaller subset from each dataset. Also, $\epsilon_{\text{CPBDN}} = \epsilon_{\text{FISTA}} = 10^{-3}$ is adopted for simplicity. The implementation of the proposed SOCDL algorithm is based on the Python

SPORCO library [96]. All experiments are performed on a PC with a 3.70 GHz CPU and 16 GB of memory.

5.3.1 The Effect of Slice-based Dictionary Update

Previously in Section 5.2.5 it has been demonstrated that the proposed SOCDL algorithm is equivalent to the surrogate function based OCDL method (OCDL-Surrogate) [61] which applies FISTA for dictionary update. The difference is that SOCDL accumulates a non-square matrix $\mathbf{A}^{(t)}$ which consists of stripe vectors, while for OCDL-Surrogate the $\mathbf{A}^{(t)}$ is square and approximates the Hessian of the surrogate function. The theoretical performance gap has been shown in Table 5.2, and in this part such a gap is validated empirically.

To minimize the influence of convolutional sparse coding, the available CUDA version of CBPDN [158] is utilized which operates only on grayscale images. The training images from CIFAR10 are converted into grayscale, normalized and Tikhonov regularization is applied. The SOCDL and OCDL-Surrogate are run with the same parameter settings, i.e. $\lambda = 0.2$, $p = 0$ and $L = 5$. The experiment varies the number of the dictionary atoms m and the atom size n to investigate how the different dictionary update steps affect the final performance. At each iteration a minibatch is randomly drawn with a size of 16 from CIFAR10 training set, and both algorithms are run for $T = 400$ iterations. The reconstruction error is computed on 1000 test samples for each time step.

The result is shown in Figure 5.2. It can be seen that with the same sparse coding step, the proposed algorithm is faster than OCDL-Surrogate for different settings. When the size and number of dictionary atoms are small, i.e. when $m = 100$ and $n = 8 \times 8$, the performance gap is small. However, when increasing m from 100 to 400, or increasing n from 8×8 to 13×13 , the gap becomes obvious and the proposed algorithm can be at most 3 times faster (note the log scale on time axis). This is due to the fact that OCDL-Surrogate requires $\mathcal{O}(n^2m^2)$ for each sample to update the matrices $\mathbf{A}^{(t)}$ and $\mathbf{B}^{(t)}$, while SOCDL only requires $\mathcal{O}(nm^2)$. This means that the proposed algorithm is particularly efficient when the dictionary size n is large.

(A) $m = 100, n = 8 \times 8$. (B) $m = 400, n = 8 \times 8$. (C) $m = 100, n = 13 \times 13$ .FIGURE 5.2: Comparison between SOCDL and OCDL for different dictionary size n and number of dictionary atoms m .

5.3.2 Image Reconstruction on Small Datasets

In this part, experiments are performed on *Fruit* and *City*, which are standard for CSC applications, to compare SOCDL with other CDL and OCDL algorithms. Each of the dataset has 10 training images and 4 test images. For fair comparisons, common experimental settings for *Fruit* and *City* are used as in [33], [60]. Specifically, each image is converted to grayscale, local contrast normalized and edge-tapered. A patch size n of 11×11 is used and the dictionary size m is 100. The algorithms are run with $\lambda = 1$.

The experiment compares the proposed SOCDL with the following online convolutional dictionary learning (OCDL) algorithms: SGD-based OCDL on frequency domain (OCDL-SGD) [61], surrogate-based OCDL on spatial domain (OCDL-Surrogate) [58], [61] and on frequency domain (OCSC) [60]. As OCDL has been found much faster than batch CDL algorithms [60], only one batch CDL algorithm (Batch-CDL) is included with consensus dictionary update [34], [92], which is among the fastest ones for batch learning [36].

The test set objective with respect to elapsed time is given in Figure 5.3a and 5.3b, and visualizations of learned dictionaries are in Figure 5.4. It can be seen that on both datasets SOCDL converges faster than all other surrogate-based algorithms. Specifically, the proposed algorithm is faster than OCDL-Surrogate even if the two algorithms yield exactly the same result at each iteration. Batch-CDL is the slowest among all algorithms, since at each step it should operate on all 10 training samples. Also it is observed that the proposed algorithm is comparable with first order method OCDL-SGD. One reason is that the sample size is small enough such that SGD is sufficient for the dictionary to converge quickly. In terms of elapsed time, OCDL-SGD requires the least training time because it does not require maintaining a history matrix or iterative thresholding. OCSC is slightly faster than SOCDL because it employs ADMM for dictionary update and stops for only few iterations.

5.3.3 Image Reconstruction on Large Dataset

This part performs experiments on large datasets, i.e. *CIFAR10* and *STL10*. The target is to validate the performance of the proposed algorithm under cases where

(A) Fruit

(B) City

FIGURE 5.3: Test set objective with respect to time step on *Fruit* and *City*.

FIGURE 5.4: Visualizations of dictionaries learned on Fruit (Figure (a) to (e)) and City (Figure (f) to (j)).

FIGURE 5.5: Test set objective and PSNR with respect to time step on *CIFAR10* and *STL10*.

FIGURE 5.6: Performance of SOCDL on CIFAR10 under different parameters: $p = 0, 1, 5, 10, 20$ for grayscale and color images (Figure 5.6a and 5.6b), $L = 5, 10, 20, 50, 100$ for grayscale and color images (Figure 5.6c and 5.6d).

(1) the size of the dictionary is large, (2) the patch size is large, and (3) the images are color. For each dataset, The test images are highpass filtered by subtracting lowpass component computed by Tikhonov regularization, as is commonly done in [36], [158]. For CIFAR10, the patch size is set to 11×11 with 144 dictionary atoms, and $\lambda = 0.2$. For STL10, the patch size is 8×8 with 400 dictionary atoms, and $\lambda = 0.1$. The experiment compares SOCDL with first order OCDL-SGD and second order OCDL-Surrogate. OCSC is not included because their code operates only on grayscale images. Batch-CDL is also not included because of memory limits.

The test objectives at each time step are given in Figure 5.5a and Figure 5.5b, which show that SOCDL outperforms OCDL-SGD and OCDL-Surrogate for both cases. Specifically, the proposed method outperforms first order SGD based method by a large margin on STL10, even though OCDL-SGD is much faster at each step. This is because the second order surrogate-based algorithms utilize history information which leads to faster and stabler convergence. In Figure 5.5c and Figure 5.5d the PSNR⁵ values between the original images and the reconstructed ones at each time step are plotted. It can be seen that the proposed algorithm is faster than others, and converges to higher PSNR values than SGD based methods.

5.3.4 Validation of Parameter Selection

This part evaluates the impact of several parameters for the convergence of the proposed algorithm. Specifically, two parameters are important for fast and robust convergence of the proposed algorithm: the forgetting factor p discussed in Section 5.2.6 and the step size $1/L$ for the FISTA dictionary update in algorithm 5. For fair comparisons, the base parameter values in Table 5.1 are set as $m = 100$, $n = 8 \times 8$, $\lambda = 0.2$, $T = 400$, $p = 0$, and $L = 5$ for grayscale images and $L = 25$ for color ones.

The experiment first empirically validates the effect of forgetting factor as an acceleration approach to SOCDL. Specifically, it applies the same routine in algorithm 4 and algorithm 5 for dictionary learning, but the history matrix $\mathbf{A}^{(t)}$ and $\mathbf{B}^{(t)}$ are updated with a decay factor:

---

⁵PSNR between original image I and recovered image I_{rec} is given by $\text{PSNR} = 10 \log_{10}(N/\|I - I_{\text{rec}}\|_2^2)$ since the image is normalized between 0 and 1.

TABLE 5.4: PSNR (in dB) of test images restored by dictionaries trained on VOC2007 via SGD-based OCDL and the proposed SOCDL.

	Barbara	House	Peppers	Mandrill	Monarch	Airplane	Parrots	Boy	Tulips	Kangaroo
OCDL-SGD-Mask	26.68	30.70	30.34	25.11	26.72	29.58	30.19	27.89	26.68	28.38
SOCDL-Mask	26.99	30.81	30.46	25.23	26.84	29.70	30.33	27.98	26.80	28.53

TABLE 5.5: PSNR (in dB) of test images restored by dictionaries trained on each corrupted image.

	Barbara	House	Peppers	Mandrill	Monarch	Airplane	Parrots	Boy	Tulips	Kangaroo
Batch-CDL-Mask	28.54	31.81	31.41	26.28	27.70	30.46	31.09	29.05	28.06	30.47
OCDL-SGD-Mask	28.56	31.70	31.38	26.29	27.70	30.38	31.05	29.14	28.06	30.55
SOCDL-Mask	28.51	31.71	31.30	26.27	27.73	30.50	30.92	29.21	28.05	30.53

$$\begin{aligned}
\mathbf{A}^{(t)} &= \frac{1}{\alpha^{(t)} + 1} \left[\alpha^{(t)} \mathbf{A}^{(t-1)} + \sum_{j=1}^N \gamma_j^{(t)} \mathbf{z}_j^{(t)\top} \right] \\
\mathbf{B}^{(t)} &= \frac{1}{\alpha^{(t)} + 1} \left[\alpha^{(t)} \mathbf{B}^{(t-1)} + \sum_{j=1}^N \mathbf{s}_j^{(t)} \mathbf{z}_j^{(t)\top} \right],
\end{aligned} \tag{5.42}$$

where $\alpha^{(t)} = (1 - 1/t)^p$ scales the weight of previous steps. A large forgetting factor p corresponds to more attention to the recent samples and a smaller one means the samples are contributed more equally to the learned dictionary. A series of values for p , i.e. $p = 0, 1, 5, 10, 20$, are chosen to see their influence on the algorithm. The algorithm is tested on both grayscale and color training images of CIFAR10, and is validated on a smaller subset on the test set containing 1000 images with the same channel number.

The performance of different p values on CIFAR10 is shown in Figure 5.6a and Figure 5.6b. It can be seen clearly that with the increase of p the algorithm will converge faster to a lower objective for both cases. This verifies the fact the recent updates correspond to more accurate dictionaries and should be used with heavier weights. On the other hand, when the value of p is large, i.e. $p > 5$, then the gain from increasing the weight of recent samples is negligible. In particular when $p > 10$, the update becomes unstable. This corresponds to the fact that when p is very large, only the most recent samples are used, and previous ones are largely forgotten. Therefore, choosing a moderate value of p is the key to fast convergence of SOCDL.

In a similar setting the effect of the step size $1/L$ is validated for FISTA on dictionary update. Although it is possible to incorporate back tracking as in [72] for each step of FISTA, in practice one only chooses a sufficiently large value of L and keep it fixed throughout the iterations. This is because L approximates the largest eigenvalue of the Hessian of the surrogate function $\mathcal{F}$, which is uniformly bounded on the feasible set. Specifically, L is fixed from a series of values: $L = 5, 10, 20, 50, 100$ and the algorithm is applied with the same training and test settings described above.

The results are shown in Figure 5.6c and Figure 5.6d. For grayscale images, with the increase of L , the proposed algorithm will converge more slowly. This is because

typically smaller step size requires more iterations to reach to the same precision. On the other hand, the behavior for color images are different. It can be seen in Figure 5.6d that for $L = 5$ and $L = 10$ the algorithm actually converges more slowly than other case, and for $L = 20, 50$ and 100 there is no significant difference among them. This indicates that for dictionaries with color channels, the FISTA algorithms requires larger value of L to guarantee its convergence. Nevertheless after several iterations for all choices of L the algorithm manages to converge. Also for color images, the algorithm is not sensitive to different choices of L as long as it is large enough. One possible reason is that for online CDL algorithms on color images, the convolutional sparse coding step takes up most of the time, thus the time difference on running the FISTA update is not significant.

5.3.5 Image Inpainting

This part evaluates the performance of the proposed algorithm on masked data. The primary goal is not to devise novel inpainting algorithms to achieve high performances, but rather to show the applicabilities of the proposed SOCDL combined with existing techniques (5.40) on the inpainting problem. In this sense the comparison is carried out with the first order SGD-based masked OCDL algorithm (OCDL-SGD-Mask) [61] which also handles masked data in an online fashion.

The first experiment is conducted on images from VOC2007. At each iteration, it randomly draws an image from the training set and resize to 256×256 . A binary mask is then applied corrupting 50% of pixels with padding on the boundary. The corrupted images are preprocessed by ℓ_2 total-variation denoising similar to [61]. A dictionary is trained with number of atoms $m = 64$ and patch size $n = 8 \times 8$, and $\lambda = 0.1$. Both algorithms are iterated for 50 iterations. The reconstruction functionals are computed on 20 *noise-free* test images drawn from VOC2007 test split.

The functionals of the two algorithms on test set are given in Figure 5.7. The proposed surrogate based method converges significantly faster than the first order counterpart on masked data, and results in much lower test objective as well. The learned dictionaries from both algorithms are illustrated in Figure 5.8. It can be seen that both algorithms learn valid features, but for OCDL-SGD-Mask, more than half of dictionary atoms are random and not updated while the dictionary

FIGURE 5.7: Test set objective of different algorithms learned from masked data.

FIGURE 5.8: Dictionary learned from masked VOC07 dataset images by different online CSC algorithms.

learned by SOCDL-Mask exhibits more regular patterns. Two reasons are possible behind this large performance gap. First, second order surrogate based algorithm encodes more history information. Therefore it is more stable for increasing data. SGD based algorithm updates the dictionary based solely on current data point with a heuristic step size, which is far more unstable. Second, the masked SOCDL implementation relies on the *additive mask simulation* (AMS) (5.40) while the implementation of OCDL-SGD-Mask uses mask decoupling (MD). It has been shown that AMS has slightly faster convergence performance than MD for sparse coding and dictionary learning [89].

The second experiment is conducted on ten standard test images. Each of them is corrupted by 50% of noise. First, the dictionary *learned on VOC2007* (as in Figure 5.8) is used to reconstruct them. The performances are measured by PSNR and listed in Table 5.4. It can be seen that PSNR of the proposed algorithm is higher on all test images, indicating better performance of the learned dictionary within same iterations and same dataset. To validate the inpainting capability of the proposed SOCDL-Mask algorithm other than online learning performance, the dictionary is learned directly on each single corrupted test image. For comparison the masked version of Batch-CDL algorithm [36] is also included. The reconstruction results are presented in Table 5.4. It can be seen although the proposed algorithm does not yield superior results on all test images, it is comparable with other batch or online CSC based inpainting methods in terms of PSNR. Therefore combining SOCDL with existing boundary handling algorithm [89] is indeed applicable for learning from incomplete data.

Finally, inpainting results are demonstrated on images (*Parrots* and *Sails*) contaminated by 70% of corruption in Figure 5.9 and 5.10. It can be seen that the proposed masked SOCDL algorithm can restore the corrupted images to visually good quality. Also, it can be observed there still exists small artifacts on restoration (e.g. stripes around the eye in Figure 5.9c) and a moderate loss of sharp features.

5.4 Summary

This chapter proposes a novel online algorithm for convolutional dictionary learning called slice-based online convolutional dictionary learning (SOCDL). Specifically,

(A) Original image (*Parrots*).

(B) Damaged image with 70% of data removed.

(C) Restored image (PSNR: 31.49 dB).

FIGURE 5.9: Inpainting illustrations on *Parrots* using masked SOCDL. The dictionary is learn on each corrupted image with a size of $n = 8 \times 8$ and $m = 64$ atoms. λ is set to 0.05.

(A) Original image (*Sails*).

(B) Damaged image with 70% of data removed.

(C) Restored image (PSNR: 26.91 dB).

FIGURE 5.10: Inpainting illustrations on *Sails* using masked SOCDL. The dictionary is learn on each corrupted image with a size of $n = 8 \times 8$ and $m = 64$ atoms. λ is set to 0.05.

Section 5.2 shows the difference between the proposed algorithm and previous ones by formulating a locally defined sparse surrogate objective, and shows that such formulation achieves lower time complexity in terms of online dictionary learning compared to its counterpart with same performance. The proposed SOCDL algorithm can also be regraded as extensions of traditional online dictionary learning method based on image patches. The advantages of the proposed algorithm is empirically demonstrated by experiments in Section 5.3 on both image reconstruction and image inpainting tasks, which show that comparable or superior performance can be obtained by the proposed algorithm compared to related online CDL methods.

Chapter 6

Conclusions and Future Works

Chapter 6 concludes the thesis in Section 6.1 regarding structured sparse representations for supervised and unsupervised learning, followed by Section 6.2 on discussions on future work directions.

6.1 Conclusions

This thesis investigates different supervised and unsupervised algorithms for exploiting the sparse structures for efficient representations, which are utilized to solve problems of image classification, clustering and image inpainting. Specifically, Chapter 1 and Chapter 2 introduce background and related algorithms regarding the topic of sparse representations. Chapter 3 proposes ELM-DDL as a novel combination of ELM embeddings and support discrimination dictionary learning for image classification. Chapter 4 proposes ELM-ALC to utilize the local and sparse LLC representations with adaptive graph regularizations for graph learning and clustering. Chapter 5 proposes SOCDL as a new online convolutional dictionary learning algorithm to exploit the local structures of convolutional sparse features for image reconstruction and image inpainting. The major conclusions of the thesis can be summarized as follows:

- (1) *Exploiting structural information in sparse representations is beneficial for learning a discriminative dictionary which helps image classification.*

Learning a discriminative dictionary often involves incorporating available label information so that the dictionary atoms can be grouped to produce different activations to different classes. As discussed in Section 3.1, previous approaches for dictionary learning often involves learning sub-dictionaries specific to each class. By contrast, Chapter 3 investigates embedding class information in the sparse coding vectors themselves. This is achieved by a weighted class-specific $\ell_{1,2}$ norm proposed in Section 3.2.3, injecting structural information on each group of sparse coding vectors belonging to the same class, i.e. vectors from the same class share the same sparse supports and vectors from distinct classes have minimal overlap of supports. The resulting features enjoy two advantages: 1) Feature representations of objects from different classes can be easily distinguished from each other, providing a highly separated feature space, as demonstrated in Figure 3.4c. 2) The non-overlapping support partitions the dictionary automatically which enables the use of sparse representation classifier (SRC). SRC measures reconstruction errors on each classes and empirically provides better image classification performance, as shown in Section 3.4.5. The effectiveness of the dictionary learned by structured sparse representations for image classification is validated

via illustrations of feature activations in Figure 3.3 and via superior performance of ELM-DDL compared to other dictionary learning algorithms.

(2) *The localized LLC representations combined with adaptive graph regularizations can produce a sparse graph with clear structures for clustering.*

As discussed in Section 4.1, previous approaches often enforce a global sparse or low-rank constraint on the reconstruction representations to achieve self-expressiveness and to produce a data graph for clustering. Inspired by previous research on unsupervised feature learning, this thesis utilizes the locality-constrained linear coding (LLC) as regularized reconstruction representations to construct the data graph. Specifically in Chapter 4, LLC is combined with adaptive graph regularizations and manifold regularizations to learn an effective data graph for clustering. The proposed ALC graph learning algorithm has four characteristics: 1) The obtained graph is sparse, since locality enforced by LLC leads to sparsity. 2) The LLC graph extends self-expressiveness of subspace clustering with local similarities by regularizing on local distances with nearest neighbors. 3) The adaptive graph regularizations enforce the learned graph to incorporate cluster structures. 4) The manifold regularizations enforce the adjacent connectivities to be smooth on the intrinsic manifold structure. As demonstrated in Figure 4.1, the graph learned by ALC exhibits clear structures compared with heuristically constructed one. Experiments on a wide range of benchmark datasets in Section 4.4.3 also validates the effectiveness of the learned graph for clustering.

(3) *The slice-based representations provide a local and structural view on the convolutional sparse features which improves online convolutional dictionary learning.*

As discussed in Section 5.1, slice-based representations are utilized first to explore locally defined algorithms for batch-mode convolutional dictionary learning (CDL). This thesis demonstrates that the representations can be utilized as the basic component to a novel online learning algorithm for convolutional dictionary learning. Specifically, Section 5.2.3 proposes to minimize a surrogate function w.r.t. the representations, and the dictionary is updated via FISTA. The slice-based representations provide four advantages to the proposed SOCDL algorithm: 1) The update given by the local slices is similar to that of the online method for patch-based dictionary learning as demonstrated in Section 5.2.4, which provides insights into connections between patch-based sparse coding and image-based convolutional

sparse coding. 2) SOCDL is faster in terms of time complexity and is more memory efficient, as shown in complexity analysis in Section 5.2.5 and empirically demonstrated in Figure 5.2 which compares SOCDL with OCDL counterpart without utilizing such representations. 3) The convergence of SOCDL is easily analyzed and proved using the slice-based representations as shown in Theorem 5.1. 4) As discussed in Section 5.2.7, SOCDL can be combined with boundary handling techniques to address the problem of image inpainting, either in batch-mode or online, as validated in Section 5.3.5. Experiments on a series of datasets in Section 5.3.2 and Section 5.3.3 show empirically that the online CDL algorithm based on slice-based representation provides comparable or better image reconstruction performance.

(4) *ELM as feature representations is an efficient approach with high discrimination capabilities for both supervised and unsupervised learning.* As reviewed in Section 2.4, Extreme Learning Machine (ELM) is an efficient learning approach for regression and classification. ELM also has feature learning capabilities achieved via ELM based autoencoders (ELM-AE) in Section 2.4.2. This thesis demonstrates that ELM itself can be utilized as an effective and efficient feature representation approach. Specifically, Chapter 3 combines ELM embeddings with dictionary learning as a unified framework, i.e. ELM-DDL, for image classification. In Section 3.2.2, ELM with orthogonal output weights regularized via the class-aware maximum margin criterion (MMC) is used as a simple feature transformation of the input feature space. The discrimination of ELM embeddings is illustrated in Figure 3.4b, and the effect of ELM embeddings on the improvement of image classification is validated in Section 3.4.3. In Chapter 4, the ELM embeddings are used on the learned graph to perform spectral clustering, leading to the proposed ELM-ALC algorithm. Section 4.4.3 empirically demonstrates that the use of ELM embeddings for final stage spectral clustering provides comparable or better performance on benchmark datasets.

6.2 Future Works

In light of investigations of different aspects of sparse coding in this thesis, some interesting research topics can be further explored:

- As discussed in Section 3.2.4 regarding discriminative regularizations for dictionary learning, this thesis explores structured sparsity-inducing norm on the coding vectors to encourage discrimination for the learned dictionary. It has also been argued that enforcing structural information on the dictionary is beneficial and is orthogonal to the proposed ELM-DDL. Existing research has tried to enforce incoherence or low rank constraints on the sub-dictionaries for each class (see Section 2.2.2 for reviews). Therefore, it is interesting to explore effective structured dictionary regularizations to improve discrimination capabilities of ELM-DDL.
- ELM-ALC is an effective graph-based clustering approach and has outperformed ELM-CLR on many benchmark datasets (see Section 4.4.3). On the other hand, ELM-ALC regularizes Euclidean distances between samples on the original feature space ($\sum_i \|\mathbf{d}_{xi} \odot \mathbf{s}_i\|_2^2$ in (4.3)), while ELM-CLR utilizes dual data representations (original feature space and ELM feature space in (2.81)) for regularizations. As noted in the aforementioned conclusions, ELM is discriminative for feature representations. A direct extension of ELM-ALC is to leverage ELM feature representations directly for learning the data graph for clustering as ELM-CLR does.
- Convolutional dictionary learning has been applied for visual recognition and other vision tasks, but the results are not scalable to more complex datasets, partly due to the fact that solving CDL is slow, and partly because the obtained features are “low-level” and is not discriminative enough. Therefore, two directions can be further explored for CDL: 1) Designing faster online learning or approximation approaches for convolutional dictionary, and 2) designing structured sparse regularizations (as done for patch-based dictionary learning) to improve discriminations of learned sparse feature maps.
- As discussed above, ELM provide efficient and effective feature representations for supervised and unsupervised learning tasks. However the hidden space is random and does not provide structural information. One direction for extending plain ELM for feature representations, inspired by the investigations of this thesis, is to enforce structured regularizations on the output space $\mathbf{H}\beta$ to incorporate prior knowledge of the problem.
- This thesis focuses on supervised and unsupervised learning tasks with sparse representations for exploiting structural information. Particularly, Chapter 3

focuses on improving image classification with $\ell_{1,2}$ constrained sparsity, and Chapter 4 targets on general clustering problem with locality constrained linear coding. Apart from sparsity-based methods, these tasks by itself remain important topics with several challenges. The general image classification problem is considered *solved* in the sense that ImageNet classification has reached 85% top 1 accuracy [104], and has surpassed human-level performance since 2015 [266]. On the other hand, in many specialized areas that require domain knowledge such as medical images, it is expensive or prohibitive to obtain large numbers of annotated images to train an image classifier. This poses a huge challenge on building an effective classification system. This difficulty gives rise to a series of learning paradigms to effectively utilize different levels of available information to boost classification performance, including semi-supervised learning [107], self-supervised learning [110]–[112], weakly supervised learning [267], one-shot/few-shot learning [268], [269], meta learning [270], [271], federated learning [272], etc. Therefore, a potential future direction of this thesis, on the one hand, is to explore the effectiveness of sparse representations on low data regime (which is also discussed in Section 3.4.6) and/or their transferability, and on the other, to utilize sparsity to design downstream pretext tasks as self-supervisions for unsupervised feature learning.

- Compared with supervised image classification, there are more challenges in unsupervised problems and specifically clustering problems. For example, the definition of *clusters* depends on similarities between pair of samples, and similarity can be defined in various aspects. Different similarity definitions can lead to different clusters. In this sense, there is no universal algorithm that can perform well on *all* clustering datasets. Therefore, a more plausible setting to clustering is a shift from unsupervised clustering problem to *semi-supervised* clustering problem, where the unlabeled data define data distributions of the problem, and the labeled data provide implicit similarities definition, which can be utilized for understanding the target domain. In the scope of this thesis, a possible extension is to modify ALC so that it can incorporate (a small number of) labeled data specifically to each dataset which may significantly boost clustering performance.

List of Publications¹

Journal Articles

- **Yijie Zeng**, Jichao Chen, and Guang-Bin Huang, “Slice-based Online Convolutional Dictionary Learning,” *IEEE Transactions on Cybernetics*, 2019. DOI: 10.1109/TCYB.2019.2931914.
- **Yijie Zeng**, Yue Li, Jichao Chen, Xiaofan Jia, and Guang-Bin Huang, “ELM embedded discriminative dictionary learning for image classification,” *Neural Networks*, Mar. 2020. DOI: 10.1016/j.neunet.2019.11.015.
- **Yijie Zeng**, Jichao Chen, Yue Li, Yuanyuan Qing, and Guang-Bin Huang, “Clustering via Adaptive and Locality-constrained Graph Learning and Unsupervised ELM,” *Neurocomputing*, Mar. 2020. DOI: 10.1016/j.neucom.2020.03.045.
- Jichao Chen, **Yijie Zeng**, Yue Li, and Guang-Bin Huang, “Unsupervised Feature Selection based Extreme Learning Machine for Clustering,” *Neurocomputing*, Dec. 2019. DOI: 10.1016/j.neucom.2019.12.065.
- Yue Li, **Yijie Zeng**, Tianchi Liu, Xiaofan Jia, and Guang-Bin Huang, “Simultaneously learning affinity matrix and data representations for machine fault diagnosis,” *Neural Networks*, Feb. 2020. DOI: 10.1016/j.neunet.2019.11.007.
- Yue Li, **Yijie Zeng**, Yuanyuan Qing, and Guang-Bin Huang, “Learning Local Discriminative Representations via Extreme Learning Machine for Machine Fault Diagnosis,” *Neurocomputing*, May 2020. DOI: 10.1016/j.neucom.2020.05.021.

---

¹The superscript * indicates joint first authors.

- Chule Yang, Danwei Wang, **Yijie Zeng**, Yufeng Yue, and Prarinya Siritanawan, “Knowledge-based multimodal information fusion for role recognition and situation assessment by using mobile robot,” *Information Fusion*, Oct. 2019. DOI: 10.1016/j.inffus.2018.10.007.

Conference Proceedings

- **Yijie Zeng**, Jichao Chen, and Guang-Bin Huang, “Data Driven Convolutional Sparse Coding for Visual Recognition,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2018. DOI: 10.1109/ICASSP.2018.8462526.
- Siqi Wang*, **Yijie Zeng***, Xinwang Liu, En Zhu, Jianping Yin, Chuanfu Xu, and Marius Kloft, “Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network,” in *Advances in Neural Information Processing Systems*, 2019.
- Siqi Wang*, **Yijie Zeng***, Qiang Liu, Chengzhang Zhu, En Zhu, and Jianping Yin, “Detecting Abnormality Without Knowing Normality: A Two-stage Approach for Unsupervised Video Abnormal Event Detection,” in *Proceedings of the 26th ACM International Conference on Multimedia*, 2018. DOI: 10.1145/3240508.3240615.
- Jichao Chen, **Yijie Zeng**, Siqi Wang, Ling Min Soh, and Guang-Bin Huang, “Octree-based Convolutional Autoencoder Extreme Learning Machine for 3D Shape Classification,” in *2018 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2018. DOI: 10.1109/IJCNN.2018.8489409.
- Chule Yang, **Yijie Zeng**, Yufeng Yue, Prarinya Siritanawan, Jun Zhang, and Danwei Wang, “Knowledge-based role recognition by using human-object interaction and spatio-temporal analysis,” in *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec. 2017. DOI: 10.1109/ROBIO.2017.8324411.

Bibliography

- [4] B. A. Olshausen and D. J. Field, “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” English, *Nature; London*, vol. 381, no. 6583, pp. 607–9, Jun. 1996.
- [5] B. A. Olshausen and D. J. Field, “Sparse coding with an overcomplete basis set: A strategy employed by V1?” eng, *Vision Research*, vol. 37, no. 23, pp. 3311–3325, Dec. 1997.
- [6] R. Tibshirani, “Regression Shrinkage and Selection via the Lasso,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
- [7] D. L. Donoho, “Compressed sensing,” *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006. DOI: 10.1109/TIT.2006.871582.
- [8] E. J. Candes and M. B. Wakin, “An Introduction To Compressive Sampling,” *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, Mar. 2008. DOI: 10.1109/MSP.2007.914731.
- [9] S. Mallat, *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*, 3rd. Orlando, FL, USA: Academic Press, Inc., 2008.
- [10] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, “Robust Face Recognition via Sparse Representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, Feb. 2009. DOI: 10.1109/TPAMI.2008.79.
- [11] M. Elad and M. Aharon, “Image Denoising Via Sparse and Redundant Representations Over Learned Dictionaries,” *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006. DOI: 10.1109/TIP.2006.881969.

- [12] J. Mairal, M. Elad, and G. Sapiro, “Sparse Representation for Color Image Restoration,” *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 53–69, Jan. 2008. DOI: 10.1109/TIP.2007.911828.
- [13] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, “Non-local sparse models for image restoration,” in *2009 IEEE 12th International Conference on Computer Vision*, Sep. 2009, pp. 2272–2279. DOI: 10.1109/ICCV.2009.5459452.
- [14] J. Yang, J. Wright, T. S. Huang, and Y. Ma, “Image Super-Resolution Via Sparse Representation,” *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010. DOI: 10.1109/TIP.2010.2050625.
- [15] J. Yang, K. Yu, Y. Gong, and T. Huang, “Linear spatial pyramid matching using sparse coding for image classification,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 1794–1801. DOI: 10.1109/CVPR.2009.5206757.
- [16] A. Coates, A. Ng, and H. Lee, “An Analysis of Single-Layer Networks in Unsupervised Feature Learning,” en, in *PMLR*, Jun. 2011, pp. 215–223.
- [17] J. Mairal, F. Bach, and J. Ponce, “Task-Driven Dictionary Learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 791–804, Apr. 2012. DOI: 10.1109/TPAMI.2011.156.
- [18] Z. Jiang, Z. Lin, and L. S. Davis, “Label Consistent K-SVD: Learning a Discriminative Dictionary for Recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2651–2664, Nov. 2013. DOI: 10.1109/TPAMI.2013.88.
- [19] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan, “Sparse Representation for Computer Vision and Pattern Recognition,” *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1031–1044, Jun. 2010. DOI: 10.1109/JPROC.2010.2044470.
- [20] Q. Zhang and B. Li, “Discriminative K-SVD for dictionary learning in face recognition,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2010, pp. 2691–2698. DOI: 10.1109/CVPR.2010.5539989.

- [21] M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, and D. Cai, “Graph Regularized Sparse Coding for Image Representation,” *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1327–1336, May 2011. DOI: 10.1109/TIP.2010.2090535.
- [22] L. Li, S. Li, and Y. Fu, “Discriminative dictionary learning with low-rank regularization for face recognition,” in *2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, Apr. 2013, pp. 1–6. DOI: 10.1109/FG.2013.6553696.
- [23] Y. Yankelevsky and M. Elad, “Graph-constrained supervised dictionary learning for multi-label classification,” in *2016 IEEE International Conference on the Science of Electrical Engineering (ICSEE)*, Nov. 2016, pp. 1–5. DOI: 10.1109/ICSEE.2016.7806064.
- [24] S. Cai, W. Zuo, L. Zhang, X. Feng, and P. Wang, “Support Vector Guided Dictionary Learning,” en, in *Computer Vision – ECCV 2014*, ser. Lecture Notes in Computer Science, Springer, Cham, Sep. 2014, pp. 624–639. DOI: 10.1007/978-3-319-10593-2_41.
- [25] Y. Liu, W. Chen, Q. Chen, and I. Wassell, “Support Discrimination Dictionary Learning for Image Classification,” en, in *Computer Vision – ECCV 2016*, ser. Lecture Notes in Computer Science, Springer, Cham, Oct. 2016, pp. 375–390. DOI: 10.1007/978-3-319-46475-6_24.
- [26] M. Yuan and Y. Lin, “Model selection and estimation in regression with grouped variables,” en, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, Feb. 2006. DOI: 10.1111/j.1467-9868.2005.00532.x.
- [27] L. Jacob, G. Obozinski, and J.-P. Vert, “Group Lasso with Overlap and Graph Lasso,” in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML ’09, New York, NY, USA: ACM, 2009, pp. 433–440. DOI: 10.1145/1553374.1553431.
- [28] J. Huang, T. Zhang, and D. Metaxas, “Learning with Structured Sparsity,” *Journal of Machine Learning Research*, vol. 12, no. Nov, pp. 3371–3412, 2011.

- [29] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach, “Network Flow Algorithms for Structured Sparsity,” in *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS’10, USA: Curran Associates Inc., 2010, pp. 1558–1566.
- [30] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998. DOI: 10.1109/5.726791.
- [31] H. Bristow, A. Eriksson, and S. Lucey, “Fast Convolutional Sparse Coding,” in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2013, pp. 391–398. DOI: 10.1109/CVPR.2013.57.
- [32] B. Wohlberg, “Efficient convolutional sparse coding,” in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 7173–7177. DOI: 10.1109/ICASSP.2014.6854992.
- [33] F. Heide, W. Heidrich, and G. Wetzstein, “Fast and flexible convolutional sparse coding,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 5135–5143. DOI: 10.1109/CVPR.2015.7299149.
- [34] M. Šorel and F. Šroubek, “Fast convolutional sparse coding using matrix inversion lemma,” *Digital Signal Processing*, vol. 55, no. Supplement C, pp. 44–51, Aug. 2016. DOI: 10.1016/j.dsp.2016.04.012.
- [35] V. Pappyan, Y. Romano, J. Sulam, and M. Elad, “Convolutional Dictionary Learning via Local Processing,” *arXiv:1705.03239 [cs]*, May 2017. arXiv: 1705.03239 [cs].
- [36] 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, Sep. 2018. DOI: 10.1109/TCI.2018.2840334.
- [37] Y. Zhu and S. Lucey, “Convolutional Sparse Coding for Trajectory Reconstruction,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 529–540, Mar. 2015. DOI: 10.1109/TPAMI.2013.2295311.

- [38] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang, “Convolutional Sparse Coding for Image Super-Resolution,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015, pp. 1823–1831. DOI: 10.1109/ICCV.2015.212.
- [39] D. Carrera, G. Boracchi, A. Foi, and B. Wohlberg, “Detecting anomalous structures by convolutional sparse models,” in *2015 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2015, pp. 1–8. DOI: 10.1109/IJCNN.2015.7280790.
- [40] Y. Liu, X. Chen, R. K. Ward, and Z. J. Wang, “Image Fusion With Convolutional Sparse Representation,” *IEEE Signal Processing Letters*, vol. 23, no. 12, pp. 1882–1886, Dec. 2016. DOI: 10.1109/LSP.2016.2618776.
- [41] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, “Structured Sparsity through Convex Optimization,” *EN, Statistical Science*, vol. 27, no. 4, pp. 450–468, Nov. 2012. DOI: 10.1214/12-STS394.
- [42] F. Nie, H. Huang, X. Cai, and C. H. Ding, “Efficient and Robust Feature Selection via Joint ℓ_2, ℓ_1 -Norms Minimization,” in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds., Curran Associates, Inc., 2010, pp. 1813–1821.
- [43] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, “L2,1-norm Regularized Discriminative Feature Selection for Unsupervised Learning,” in *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, ser. IJCAI’11, AAAI Press, 2011, pp. 1589–1594. DOI: 10.5591/978-1-57735-516-8/IJCAI11-267.
- [44] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach, “Proximal Methods for Sparse Hierarchical Dictionary Learning,” in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML’10, USA: Omnipress, 2010, pp. 487–494.
- [45] —, “Proximal Methods for Hierarchical Sparse Coding,” *J. Mach. Learn. Res.*, vol. 12, pp. 2297–2334, Jul. 2011.
- [46] I. Ramirez, P. Sprechmann, and G. Sapiro, “Classification and clustering via dictionary learning with structured incoherence and shared features,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2010, pp. 3501–3508. DOI: 10.1109/CVPR.2010.5539964.

- [47] M. Yang, L. Zhang, X. Feng, and D. Zhang, “Fisher Discrimination Dictionary Learning for sparse representation,” in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 543–550. DOI: 10.1109/ICCV.2011.6126286.
- [48] L. Ma, C. Wang, B. Xiao, and W. Zhou, “Sparse representation for face recognition based on discriminative low-rank dictionary learning,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012, pp. 2586–2593. DOI: 10.1109/CVPR.2012.6247977.
- [49] T. H. Vu and V. Monga, “Fast Low-Rank Shared Dictionary Learning for Image Classification,” *IEEE Transactions on Image Processing*, vol. 26, no. 11, pp. 5160–5175, Nov. 2017. DOI: 10.1109/TIP.2017.2729885.
- [50] H. Foroughi, N. Ray, and H. Zhang, “Object Classification With Joint Projection and Low-Rank Dictionary Learning,” *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 806–821, Feb. 2018. DOI: 10.1109/TIP.2017.2766446.
- [51] Y. Quan, Y. Xu, Y. Sun, Y. Huang, and H. Ji, “Sparse Coding for Classification via Discrimination Ensemble,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5839–5847.
- [52] Y. Suo, M. Dao, T. Tran, H. Mousavi, U. Srinivas, and V. Monga, “Group structured dirty dictionary learning for classification,” in *2014 IEEE International Conference on Image Processing (ICIP)*, Oct. 2014, pp. 150–154. DOI: 10.1109/ICIP.2014.7025029.
- [53] J. Qi and W. Chen, “Learning a discriminative dictionary for classification with outliers,” *Signal Processing*, vol. 152, pp. 255–264, Nov. 2018. DOI: 10.1016/j.sigpro.2018.06.005.
- [54] C. Cortes and V. Vapnik, “Support-vector networks,” en, *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995. DOI: 10.1007/BF00994018.
- [55] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: Theory and applications,” *Neurocomputing, Neural Networks*, vol. 70, no. 1, pp. 489–501, Dec. 2006. DOI: 10.1016/j.neucom.2005.12.126.
- [56] Jianbo Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, Aug. 2000. DOI: 10.1109/34.868688.

- [57] E. Elhamifar and R. Vidal, “Sparse Subspace Clustering: Algorithm, Theory, and Applications,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2765–2781, Nov. 2013. DOI: 10.1109/TPAMI.2013.57.
- [58] K. Degraux, U. S. Kamilov, P. T. Boufounos, and D. Liu, “Online Convolutional Dictionary Learning for Multimodal Imaging,” *arXiv:1706.04256 [cs]*, Jun. 2017. arXiv: 1706.04256 [cs].
- [59] J. Liu, C. Garcia-Cardona, B. Wohlberg, and W. Yin, “Online Convolutional Dictionary Learning,” *arXiv:1706.09563 [cs, eess]*, pp. 1707–1711, Sep. 2017. DOI: 10.1109/ICIP.2017.8296573. arXiv: 1706.09563 [cs, eess].
- [60] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, “Scalable Online Convolutional Sparse Coding,” *arXiv:1706.06972 [cs]*, Jun. 2017. arXiv: 1706.06972 [cs].
- [61] J. Liu, C. Garcia-Cardona, B. Wohlberg, and W. Yin, “First and Second Order Methods for Online Convolutional Dictionary Learning,” *arXiv:1709.00106 [cs, eess, math, stat]*, Aug. 2017. arXiv: 1709.00106 [cs, eess, math, stat].
- [62] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, “Online Learning for Matrix Factorization and Sparse Coding,” *J. Mach. Learn. Res.*, vol. 11, pp. 19–60, Mar. 2010.
- [63] J. Wang, J. Yang, F. Lv, T. Huang, and Y. Gong, “Locality-constrained linear coding for image classification,” in *In CVPR*, 2010.
- [64] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*, 1st. Springer Publishing Company, Incorporated, 2010.
- [65] S. G. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993. DOI: 10.1109/78.258082.
- [66] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, “Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition,” in *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, Nov. 1993, 40–44 vol.1. DOI: 10.1109/ACSSC.1993.342465.

- [67] T. Blumensath and M. E. Davies, “Iterative hard thresholding for compressed sensing,” *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, Nov. 2009. DOI: 10.1016/j.acha.2009.04.002.
- [68] E. J. Candes, “The restricted isometry property and its implications for compressed sensing,” *Comptes Rendus Mathematique*, vol. 346, no. 9-10, pp. 589–592, 2008.
- [69] S. Chen, D. Donoho, and M. Saunders, “Atomic Decomposition by Basis Pursuit,” *SIAM Review*, vol. 43, no. 1, pp. 129–159, Jan. 2001. DOI: 10.1137/S003614450037906X.
- [70] Y. Li and S. Osher, “Coordinate descent optimization for l1 minimization with application to compressed sensing; a greedy algorithm,” *Inverse Problems and Imaging*, vol. 3, no. 3, pp. 487–503, 2009.
- [71] I. Daubechies, M. Defrise, and C. De Mol, “An iterative thresholding algorithm for linear inverse problems with a sparsity constraint,” en, *Communications on Pure and Applied Mathematics*, vol. 57, no. 11, pp. 1413–1457, Nov. 2004. DOI: 10.1002/cpa.20042.
- [72] A. Beck and M. Teboulle, “A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems,” *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, Jan. 2009. DOI: 10.1137/080716542.
- [73] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, “Least angle regression,” *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004. DOI: 10.1214/0090536040000000067.
- [74] J. Yang and Y. Zhang, “Alternating Direction Algorithms for ℓ_1 -Problems in Compressive Sensing,” *SIAM Journal on Scientific Computing*, vol. 33, no. 1, pp. 250–278, Jan. 2011. DOI: 10.1137/090777761.
- [75] H. Zou and T. Hastie, “Regularization and variable selection via the elastic net,” en, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, Apr. 2005. DOI: 10.1111/j.1467-9868.2005.00503.x.
- [76] D. D. Lee and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization,” en, *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999. DOI: 10.1038/44565.

- [77] K. Engan, S. O. Aase, and J. H. Husoy, “Method of optimal directions for frame design,” in *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, vol. 5, 1999, 2443–2446 vol.5. DOI: 10.1109/ICASSP.1999.760624.
- [78] H. Lee, A. Battle, R. Raina, and A. Y. Ng, “Efficient sparse coding algorithms,” in *In NIPS*, NIPS, 2007, pp. 801–808.
- [79] M. Aharon, M. Elad, and A. Bruckstein, “K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation,” *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006. DOI: 10.1109/TSP.2006.881199.
- [80] J. Mairal, F. Bach, and J. Ponce, “Sparse Modeling for Image and Vision Processing,” English, *Foundations and Trends® in Computer Graphics and Vision*, vol. 8, no. 2-3, pp. 85–283, Dec. 2014. DOI: 10.1561/06000000058.
- [81] L. Bottou and O. Bousquet, “The Tradeoffs of Large Scale Learning,” in *Advances in Neural Information Processing Systems 20*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds., Curran Associates, Inc., 2008, pp. 161–168.
- [82] M. Aharon and M. Elad, “Sparse and Redundant Modeling of Image Content Using an Image-Signature-Dictionary,” *SIAM Journal on Imaging Sciences*, vol. 1, no. 3, pp. 228–247, Jan. 2008. DOI: 10.1137/07070156X.
- [83] K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “Fast inference in sparse coding algorithms with applications to object recognition,” *arXiv preprint arXiv:1010.3467*, 2010.
- [84] M. S. Lewicki and T. J. Sejnowski, “Coding Time-Varying Signals Using Sparse, Shift-Invariant Representations,” in *Advances in Neural Information Processing Systems 11*, M. J. Kearns, S. A. Solla, and D. A. Cohn, Eds., MIT Press, 1999, pp. 730–736.
- [85] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, “Deconvolutional networks,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2010, pp. 2528–2535. DOI: 10.1109/CVPR.2010.5539957.

- [86] M. D. Zeiler, G. W. Taylor, and R. Fergus, “Adaptive deconvolutional networks for mid and high level feature learning,” in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2018–2025. DOI: 10.1109/ICCV.2011.6126474.
- [87] V. Pappyan, J. Sulam, and M. Elad, “Working Locally Thinking Globally: Theoretical Guarantees for Convolutional Sparse Coding,” *IEEE Transactions on Signal Processing*, vol. PP, no. 99, pp. 1–1, 2017. DOI: 10.1109/TSP.2017.2733447.
- [88] R. Chalasani, J. C. Principe, and N. Ramakrishnan, “A fast proximal method for convolutional sparse coding,” in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, Aug. 2013, pp. 1–5. DOI: 10.1109/IJCNN.2013.6706854.
- [89] B. Wohlberg, “Boundary handling for convolutional sparse representations,” in *2016 IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 1833–1837. DOI: 10.1109/ICIP.2016.7532675.
- [90] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers,” *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011. DOI: 10.1561/22000000016.
- [91] H. Zhang and V. M. Patel, “Convolutional Sparse and Low-Rank Coding-Based Image Decomposition,” *IEEE Transactions on Image Processing*, vol. 27, no. 5, pp. 2121–2133, May 2018. DOI: 10.1109/TIP.2017.2786469.
- [92] B. Choudhury, R. Swanson, F. Heide, G. Wetzstein, and W. Heidrich, “Consensus Convolutional Sparse Coding,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 4290–4298. DOI: 10.1109/ICCV.2017.459.
- [93] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. LeCun, “Learning Convolutional Feature Hierarchies for Visual Recognition,” in *Proceedings of the 23rd International Conference on Neural Information Processing Systems*, ser. NIPS’10, USA: Curran Associates Inc., 2010, pp. 1090–1098.

- [94] R. Rigamonti, V. Lepetit, G. González, E. Türetken, F. Benmansour, M. Brown, and P. Fua, “On the relevance of sparsity for image classification,” *Computer Vision and Image Understanding*, vol. 125, pp. 115–127, Aug. 2014. DOI: 10.1016/j.cviu.2014.03.009.
- [95] C. Garcia-Cardona and B. Wohlberg, “Subproblem coupling in convolutional dictionary learning,” in *2017 IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, pp. 1697–1701. DOI: 10.1109/ICIP.2017.8296571.
- [96] B. Wohlberg, “SPORCO: A Python package for standard and convolutional sparse representations,” *Proceedings of the 16th Python in Science Conference*, pp. 1–8, 2017. DOI: 10.25080/shinma-7f4c6e7-001.
- [97] M. S. C. Almeida and M. Figueiredo, “Deconvolving Images With Unknown Boundaries Using the Alternating Direction Method of Multipliers,” *IEEE Transactions on Image Processing*, vol. 22, no. 8, pp. 3074–3086, Aug. 2013. DOI: 10.1109/TIP.2013.2258354.
- [98] Y. Zeng, J. Chen, and G.-B. Huang, “Data Driven Convolutional Sparse Coding for Visual Recognition,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2018, pp. 2736–2740. DOI: 10.1109/ICASSP.2018.8462526.
- [99] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” en, *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. DOI: 10.1038/nature14539.
- [100] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS’12, USA: Curran Associates Inc., 2012, pp. 1097–1105.
- [101] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.
- [102] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *arXiv:1409.1556 [cs]*, Apr. 2015. arXiv: 1409.1556 [cs].
- [103] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *arXiv:1512.03385 [cs]*, Dec. 2015. arXiv: 1512.03385 [cs].

- [104] M. Tan and Q. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” en, in *International Conference on Machine Learning*, May 2019, ch. Machine Learning, pp. 6105–6114.
- [105] A. Tarvainen and H. Valpola, “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results,” *arXiv:1703.01780 [cs, stat]*, Apr. 2018. arXiv: 1703.01780 [cs, stat].
- [106] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, “Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning,” *arXiv:1704.03976 [cs, stat]*, Jun. 2018. arXiv: 1704.03976 [cs, stat].
- [107] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, “MixMatch: A Holistic Approach to Semi-Supervised Learning,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d\ textquotesingle Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 5049–5059.
- [108] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, “Discriminative Unsupervised Feature Learning with Convolutional Neural Networks,” in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2014, pp. 766–774.
- [109] R. Zhang, P. Isola, and A. A. Efros, “Colorful Image Colorization,” en, in *Computer Vision – ECCV 2016*, Springer, Cham, Oct. 2016, pp. 649–666. DOI: 10.1007/978-3-319-46487-9_40.
- [110] S. Gidaris, P. Singh, and N. Komodakis, “Unsupervised Representation Learning by Predicting Image Rotations,” Feb. 2018.
- [111] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, “Momentum Contrast for Unsupervised Visual Representation Learning,” *arXiv:1911.05722 [cs]*, Nov. 2019. arXiv: 1911.05722 [cs].
- [112] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A Simple Framework for Contrastive Learning of Visual Representations,” *arXiv:2002.05709 [cs, stat]*, Mar. 2020. arXiv: 2002.05709 [cs, stat].

- [113] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, “Mastering the game of Go with deep neural networks and tree search,” en, *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016. DOI: 10.1038/nature16961.
- [114] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, “Mastering the game of Go without human knowledge,” en, *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017. DOI: 10.1038/nature24270.
- [115] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017. DOI: 10.1109/TPAMI.2016.2577031.
- [116] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 3431–3440. DOI: 10.1109/CVPR.2015.7298965.
- [117] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” *arXiv:1703.06870 [cs]*, Mar. 2017. arXiv: 1703.06870 [cs].
- [118] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, “Fully-Convolutional Siamese Networks for Object Tracking,” en, in *Computer Vision – ECCV 2016 Workshops*, G. Hua and H. Jégou, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2016, pp. 850–865. DOI: 10.1007/978-3-319-48881-3_56.
- [119] R. Alp Güler, N. Neverova, and I. Kokkinos, “DensePose: Dense Human Pose Estimation in the Wild,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7297–7306.
- [120] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local Neural Networks,” *arXiv:1711.07971 [cs]*, Nov. 2017. arXiv: 1711.07971 [cs].
- [121] C. Feichtenhofer, H. Fan, J. Malik, and K. He, “SlowFast Networks for Video Recognition,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6202–6211.

- [122] F. Radenović, G. Tolias, and O. Chum, “Fine-tuning CNN Image Retrieval with No Human Annotation,” *arXiv:1711.02512 [cs]*, Jul. 2018. arXiv: 1711.02512 [cs].
- [123] D. Yi, Z. Lei, S. Liao, and S. Z. Li, “Deep Metric Learning for Person Re-identification,” in *2014 22nd International Conference on Pattern Recognition*, Aug. 2014, pp. 34–39. DOI: 10.1109/ICPR.2014.16.
- [124] Y. Bengio, A. Courville, and P. Vincent, “Representation Learning: A Review and New Perspectives,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013. DOI: 10.1109/TPAMI.2013.50.
- [125] A. Coates and A. Y. Ng, “The importance of encoding versus training with sparse coding and vector quantization,” in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 921–928.
- [126] T.-H. Lin and H. T. Kung, “Stable and Efficient Representation Learning with Nonnegativity Constraints,” in *PMLR*, Jan. 2014, pp. 1323–1331.
- [127] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, “Greedy layer-wise training of deep networks,” in *Proceedings of the 19th International Conference on Neural Information Processing Systems*, ser. NIPS’06, Canada: MIT Press, Dec. 2006, pp. 153–160.
- [128] M. t. aurelio Ranzato, C. Poultney, S. Chopra, and Y. L. Cun, “Efficient Learning of Sparse Representations with an Energy-Based Model,” in *Advances in Neural Information Processing Systems 19*, P. B. Schölkopf, J. C. Platt, and T. Hoffman, Eds., MIT Press, 2007, pp. 1137–1144.
- [129] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML ’08, Helsinki, Finland: Association for Computing Machinery, Jul. 2008, pp. 1096–1103. DOI: 10.1145/1390156.1390294.
- [130] Y. Bengio, “Learning Deep Architectures for AI,” *Foundations and Trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, Jan. 2009. DOI: 10.1561/2200000006.

- [131] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion,” *Journal of Machine Learning Research*, vol. 11, no. Dec, pp. 3371–3408, 2010.
- [132] H. Bourlard and Y. Kamp, “Auto-association by multilayer perceptrons and singular value decomposition,” en, *Biological Cybernetics*, vol. 59, no. 4, pp. 291–294, Sep. 1988. DOI: 10.1007/BF00332918.
- [133] M. E. Tipping and C. M. Bishop, “Probabilistic Principal Component Analysis,” *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.
- [134] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv:1810.04805 [cs]*, May 2019. arXiv: 1810.04805 [cs].
- [135] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” Nov. 2016.
- [136] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and Harnessing Adversarial Examples,” *arXiv:1412.6572 [cs, stat]*, Mar. 2015. arXiv: 1412.6572 [cs, stat].
- [137] A. Nguyen, J. Yosinski, and J. Clune, “Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 427–436.
- [138] S. Zhou, S. Zhang, and J. Wang, “Deep Sparse Coding Network for Image Classification,” in *Proceedings of the 7th International Conference on Internet Multimedia Computing and Service*, ser. ICIMCS ’15, New York, NY, USA: ACM, 2015, 24:1–24:4. DOI: 10.1145/2808492.2808516.
- [139] S. Zhang, J. Wang, X. Tao, Y. Gong, and N. Zheng, “Constructing Deep Sparse Coding Network for image classification,” *Pattern Recognition*, vol. 64, pp. 130–140, Apr. 2017. DOI: 10.1016/j.patcog.2016.10.032.
- [140] X. Sun, N. M. Nasrabadi, and T. D. Tran, “Supervised Deep Sparse Coding Networks,” in *2018 25th IEEE International Conference on Image Processing (ICIP)*, Oct. 2018, pp. 346–350. DOI: 10.1109/ICIP.2018.8451701.

- [141] V. Pappyan, Y. Romano, J. Sulam, and M. Elad, “Theoretical Foundations of Deep Learning via Sparse Representations: A Multilayer Sparse Model and Its Connection to Convolutional Neural Networks,” *IEEE Signal Processing Magazine*, vol. 35, no. 4, pp. 72–89, Jul. 2018. DOI: 10.1109/MSP.2018.2820224.
- [142] K. Gregor and Y. LeCun, “Learning fast approximations of sparse coding,” in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 399–406.
- [143] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang, “Deep Networks for Image Super-Resolution With Sparse Prior,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 370–378.
- [144] J. Nocedal and S. Wright, *Numerical Optimization*, en, 2nd ed., ser. Springer Series in Operations Research and Financial Engineering. New York: Springer-Verlag, 2006.
- [145] B. Wohlberg, “ADMM Penalty Parameter Selection by Residual Balancing,” *arXiv:1704.06209 [cs, eess, math]*, Apr. 2017. arXiv: 1704.06209 [cs, eess, math].
- [146] M. Benning, F. Knoll, C.-B. Schönlieb, and T. Valkonen, “Preconditioned ADMM with Nonlinear Operator Constraint,” en, in *System Modeling and Optimization*, Springer, Cham, Jun. 2015, pp. 117–126. DOI: 10.1007/978-3-319-55795-3_10.
- [147] J. Wang and L. Zhao, “Nonconvex Generalization of ADMM for Nonlinear Equality Constrained Problems,” *arXiv:1705.03412 [cs, math]*, Jul. 2019. arXiv: 1705.03412 [cs, math].
- [148] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, “Compressed sensing using generative models,” in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML’17, Sydney, NSW, Australia: JMLR.org, Aug. 2017, pp. 537–546.
- [149] F. Latorre, A. eftekhari, and V. Cevher, “Fast and Provable ADMM for Learning with Generative Priors,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d\ tex-tquotesingle Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 12 027–12 039.

- [150] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative Adversarial Nets,” in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2014, pp. 2672–2680.
- [151] G. Li and T. K. Pong, “Global Convergence of Splitting Methods for Nonconvex Composite Optimization,” *SIAM Journal on Optimization*, vol. 25, no. 4, pp. 2434–2460, Jan. 2015. DOI: 10.1137/140998135.
- [152] X. Shen, L. Chen, Y. Gu, and H. C. So, “Square-Root Lasso With Nonconvex Regularization: An ADMM Approach,” *IEEE Signal Processing Letters*, vol. 23, no. 7, pp. 934–938, Jul. 2016. DOI: 10.1109/LSP.2016.2567482.
- [153] L. Yang, T. K. Pong, and X. Chen, “Alternating Direction Method of Multipliers for a Class of Nonconvex and Nonsmooth Problems with Applications to Background/Foreground Extraction,” *SIAM Journal on Imaging Sciences*, vol. 10, no. 1, pp. 74–110, Jan. 2017. DOI: 10.1137/15M1027528.
- [154] Q. Liu, X. Shen, and Y. Gu, “Linearized ADMM for Nonconvex Nonsmooth Optimization With Convergence Analysis,” *IEEE Access*, vol. 7, pp. 76 131–76 144, 2019. DOI: 10.1109/ACCESS.2019.2914461.
- [155] Y. Wang, W. Yin, and J. Zeng, “Global Convergence of ADMM in Nonconvex Nonsmooth Optimization,” in *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, Jan. 2019. DOI: 10.1007/s10915-018-0757-z.
- [156] M. F. Sahin, A. eftekhari, A. Alacaoglu, F. Latorre, and V. Cevher, “An Inexact Augmented Lagrangian Framework for Nonconvex Optimization with Nonlinear Constraints,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d\ textquotesingle Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 13 965–13 977.
- [157] B. He and X. Yuan, “On the $O(1/n)$ Convergence Rate of the Douglas–Rachford Alternating Direction Method,” *SIAM Journal on Numerical Analysis*, vol. 50, no. 2, pp. 700–709, Jan. 2012. DOI: 10.1137/110836936.
- [158] B. Wohlberg, “Efficient Algorithms for Convolutional Sparse Representations,” *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 301–315, Jan. 2016. DOI: 10.1109/TIP.2015.2495260.

- [159] N. Parikh and S. Boyd, “Proximal Algorithms,” *Found. Trends Optim.*, vol. 1, no. 3, pp. 127–239, Jan. 2014. DOI: 10.1561/24000000003.
- [160] B. Wohlberg, “Convolutional sparse representation of color images,” in *2016 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, Mar. 2016, pp. 57–60. DOI: 10.1109/SSIAI.2016.7459174.
- [161] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” in *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, Dec. 1989. DOI: 10.1007/BF02551274.
- [162] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986. DOI: 10.1038/323533a0.
- [163] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: A new learning scheme of feedforward neural networks,” in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, vol. 2, Jul. 2004, 985–990 vol.2. DOI: 10.1109/IJCNN.2004.1380068.
- [164] G.-B. Huang, L. Chen, and C.-K. Siew, “Universal Approximation Using Incremental Constructive Feedforward Networks with Random Hidden Nodes,” *Trans. Neur. Netw.*, vol. 17, no. 4, pp. 879–892, Jul. 2006. DOI: 10.1109/TNN.2006.875977.
- [165] G.-B. Huang and L. Chen, “Convex incremental extreme learning machine,” *Neurocomputing, Neural Network Applications in Electrical Engineering*, vol. 70, no. 16, pp. 3056–3062, Oct. 2007. DOI: 10.1016/j.neucom.2007.02.009.
- [166] G. B. Huang, H. Zhou, X. Ding, and R. Zhang, “Extreme Learning Machine for Regression and Multiclass Classification,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, Apr. 2012. DOI: 10.1109/TSMCB.2011.2168604.
- [167] G. B. Huang, Z. Bai, L. L. C. Kasun, and C. M. Vong, “Local Receptive Fields Based Extreme Learning Machine,” *IEEE Computational Intelligence Magazine*, vol. 10, no. 2, pp. 18–29, May 2015. DOI: 10.1109/MCI.2015.2405316.
- [168] J. Tang, C. Deng, and G. Huang, “Extreme Learning Machine for Multilayer Perceptron,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 809–821, Apr. 2016. DOI: 10.1109/TNNLS.2015.2424995.

- [169] L. Kasun, H. Zhou, G.-B. Huang, and C.-M. Vong, “Representational Learning with ELMs for Big Data,” *IEEE Intelligent Systems*, vol. 28, pp. 31–34, Nov. 2013.
- [170] D. Cui, G. Zhang, and W. Han, “Compact Feature Representation for Image Classification Using ELMs,” in *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, Oct. 2017, pp. 1015–1022. DOI: 10.1109/ICCVW.2017.124.
- [171] D. Cui, G. Huang, L. L. C. Kasun, G. Zhang, and W. Han, “ELMNET: Feature learning using extreme learning machines,” in *2017 IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, pp. 1857–1861. DOI: 10.1109/ICIP.2017.8296603.
- [172] G. Zhang, D. Cui, S. Mao, and G.-B. Huang, “Sparse Bayesian Learning for Extreme Learning Machine Auto-encoder,” en, in *Proceedings of ELM 2018*, J. Cao, C. M. Vong, Y. Miche, and A. Lendasse, Eds., ser. Proceedings in Adaptation, Learning and Optimization, Springer International Publishing, 2018, pp. 319–327.
- [173] Gao Huang, Shiji Song, J. N. D. Gupta, and Cheng Wu, “Semi-Supervised and Unsupervised Extreme Learning Machines,” *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2405–2417, Dec. 2014. DOI: 10.1109/TCYB.2014.2307349.
- [174] L. L. C. Kasun, Y. Yang, G. Huang, and Z. Zhang, “Dimension Reduction With Extreme Learning Machine,” *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3906–3918, Aug. 2016. DOI: 10.1109/TIP.2016.2570569.
- [175] T. Liu, C. K. Liyanarachchi Lekamalage, G.-B. Huang, and Z. Lin, “Extreme Learning Machine for Joint Embedding and Clustering,” *Neurocomputing, Hierarchical Extreme Learning Machines*, vol. 277, pp. 78–88, Feb. 2018. DOI: 10.1016/j.neucom.2017.01.115.
- [176] —, “An adaptive graph learning method based on dual data representations for clustering,” *Pattern Recognition*, vol. 77, pp. 126–139, May 2018. DOI: 10.1016/j.patcog.2017.12.001.

- [177] S. Wang, L. Zhao, E. Zhu, J. Yin, and H. Yang, “Ensemble One-Class Extreme Learning Machine Based on Overlapping Data Partition,” en, in *Cognitive Systems and Signal Processing*, F. Sun, H. Liu, and D. Hu, Eds., ser. Communications in Computer and Information Science, Springer Singapore, 2017, pp. 408–416.
- [178] S. Wang, E. Zhu, J. Yin, and F. Porikli, “Video anomaly detection and localization by local motion based joint video representation and OCELM,” *Neurocomputing*, Hierarchical Extreme Learning Machines, vol. 277, pp. 161–175, Feb. 2018. DOI: 10.1016/j.neucom.2016.08.156.
- [179] J. Chen, Y. Zeng, Y. Li, and G.-B. Huang, “Unsupervised Feature Selection based Extreme Learning Machine for Clustering,” en, *Neurocomputing*, Dec. 2019. DOI: 10.1016/j.neucom.2019.12.065.
- [180] L. An and B. Bhanu, “Image super-resolution by extreme learning machine,” in *2012 19th IEEE International Conference on Image Processing*, Sep. 2012, pp. 2209–2212. DOI: 10.1109/ICIP.2012.6467333.
- [181] W. Siqu, L. Qiang, G. Xifeng, Z. En, and Y. Jianping, “Fast and unsupervised outlier removal by recurrent adaptive reconstruction extreme learning machine,” en, *International Journal of Machine Learning and Cybernetics*, Mar. 2019. DOI: 10.1007/s13042-019-00943-4.
- [182] Y. Li, C. K. L. Lekamalage, T. Liu, P. Chen, and G. Huang, “Learning Representations with Local and Global Geometries Preserved for Machine Fault Diagnosis,” *IEEE Transactions on Industrial Electronics*, pp. 1–1, 2019. DOI: 10.1109/TIE.2019.2905830.
- [183] Y. Wang, Z. Xie, K. Xu, Y. Dou, and Y. Lei, “An efficient and effective convolutional auto-encoder extreme learning machine network for 3d feature learning,” *Neurocomputing*, vol. 174, no. Part B, pp. 988–998, Jan. 2016. DOI: 10.1016/j.neucom.2015.10.035.
- [184] J. Chen, Y. Zeng, S. Wang, L. M. Soh, and G.-B. Huang, “Octree-based Convolutional Autoencoder Extreme Learning Machine for 3D Shape Classification,” in *2018 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2018, pp. 1–7. DOI: 10.1109/IJCNN.2018.8489409.
- [185] G.-B. Huang, D. H. Wang, and Y. Lan, “Extreme learning machines: A survey,” en, *International Journal of Machine Learning and Cybernetics*, vol. 2, no. 2, pp. 107–122, Jun. 2011. DOI: 10.1007/s13042-011-0019-y.

- [186] G.-B. Huang, “An Insight into Extreme Learning Machines: Random Neurons, Random Features and Kernels,” en, *Cognitive Computation*, vol. 6, no. 3, pp. 376–390, Sep. 2014. DOI: 10.1007/s12559-014-9255-2.
- [187] —, “What are Extreme Learning Machines? Filling the Gap Between Frank Rosenblatt’s Dream and John von Neumann’s Puzzle,” en, *Cognitive Computation*, vol. 7, no. 3, pp. 263–278, Jun. 2015. DOI: 10.1007/s12559-015-9333-0.
- [188] G. Huang, G.-B. Huang, S. Song, and K. You, “Trends in extreme learning machines: A review,” *Neural Networks*, vol. 61, pp. 32–48, Jan. 2015. DOI: 10.1016/j.neunet.2014.10.001.
- [189] K. Hornik, “Approximation capabilities of multilayer feedforward networks,” en, *Neural Networks*, vol. 4, no. 2, pp. 251–257, Jan. 1991. DOI: 10.1016/0893-6080(91)90009-T.
- [190] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, “Multilayer feedforward networks with a nonpolynomial activation function can approximate any function,” en, *Neural Networks*, vol. 6, no. 6, pp. 861–867, Jan. 1993. DOI: 10.1016/S0893-6080(05)80131-5.
- [191] A. Rahimi and B. Recht, “Random Features for Large-Scale Kernel Machines,” in *Advances in Neural Information Processing Systems 20*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds., Curran Associates, Inc., 2008, pp. 1177–1184.
- [192] —, “Uniform approximation of functions with random bases,” in *2008 46th Annual Allerton Conference on Communication, Control, and Computing*, Sep. 2008, pp. 555–561. DOI: 10.1109/ALLERTON.2008.4797607.
- [193] —, “Weighted Sums of Random Kitchen Sinks: Replacing minimization with randomization in learning,” in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., Curran Associates, Inc., 2009, pp. 1313–1320.
- [194] G. Yehudai and O. Shamir, “On the Power and Limitations of Random Features for Understanding Neural Networks,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 6598–6608.

- [195] K. Kawaguchi, B. Xie, V. Verma, and L. Song, “Deep Semi-Random Features for Nonlinear Function Approximation,” *arXiv:1702.08882 [cs, stat]*, Feb. 2017. arXiv: 1702.08882 [cs, stat].
- [196] C. Louart and R. Couillet, “Harnessing neural networks: A random matrix approach,” in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2017, pp. 2282–2286. DOI: 10.1109/ICASSP.2017.7952563.
- [197] P. L. Bartlett, “The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network,” *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 525–536, Mar. 1998. DOI: 10.1109/18.661502.
- [198] M. Belkin, P. Niyogi, and V. Sindhwani, “Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples,” *Journal of Machine Learning Research*, vol. 7, no. Nov, pp. 2399–2434, 2006.
- [199] M. Belkin and P. Niyogi, “Laplacian Eigenmaps for Dimensionality Reduction and Data Representation,” *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, Jun. 2003. DOI: 10.1162/089976603321780317.
- [200] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” in *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. DOI: 10.1023/B:VISI.00000029664.99615.94.
- [201] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, vol. 1, Jun. 2005, 886–893 vol. 1. DOI: 10.1109/CVPR.2005.177.
- [202] L. Li, S. Li, and Y. Fu, “Learning low-rank and discriminative dictionary for image classification,” *Image and Vision Computing*, Best of Automatic Face and Gesture Recognition 2013, vol. 32, no. 10, pp. 814–823, Oct. 2014. DOI: 10.1016/j.imavis.2014.02.007.
- [203] D. Chen, X. Cao, F. Wen, and J. Sun, “Blessing of Dimensionality: High-Dimensional Feature and Its Efficient Compression for Face Verification,” in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, Portland, OR, USA: IEEE, Jun. 2013, pp. 3025–3032. DOI: 10.1109/CVPR.2013.389.

- [204] H. Zhang, Y. Zhang, and T. S. Huang, “Simultaneous discriminative projection and dictionary learning for sparse representation based classification,” *Pattern Recognition*, vol. 46, no. 1, pp. 346–354, Jan. 2013. DOI: 10.1016/j.patcog.2012.07.010.
- [205] Z. Feng, M. Yang, L. Zhang, Y. Liu, and D. Zhang, “Joint discriminative dimensionality reduction and dictionary learning for face recognition,” *Pattern Recognition*, vol. 46, no. 8, pp. 2134–2143, Aug. 2013. DOI: 10.1016/j.patcog.2013.01.016.
- [206] B.-Q. Yang, C.-C. Gu, K.-J. Wu, T. Zhang, and X.-P. Guan, “Simultaneous dimensionality reduction and dictionary learning for sparse representation based classification,” in *Multimedia Tools and Applications*, vol. 76, no. 6, pp. 8969–8990, Mar. 2017. DOI: 10.1007/s11042-016-3492-1.
- [207] R. Ptucha and A. Savakis, “LGE-KSVD: Flexible Dictionary Learning for Optimized Sparse Representation Classification,” in *2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Jun. 2013, pp. 854–861. DOI: 10.1109/CVPRW.2013.126.
- [208] B. Schölkopf, A. Smola, and K.-R. Müller, “Nonlinear Component Analysis as a Kernel Eigenvalue Problem,” *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998. DOI: 10.1162/089976698300017467.
- [209] M. Ranzato, F. J. Huang, Y. L. Boureau, and Y. LeCun, “Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition,” in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2007, pp. 1–8. DOI: 10.1109/CVPR.2007.383157.
- [210] H. Li, T. Jiang, and K. Zhang, “Efficient and robust feature extraction by maximum margin criterion,” *IEEE Transactions on Neural Networks*, vol. 17, no. 1, pp. 157–165, Jan. 2006. DOI: 10.1109/TNN.2005.860852.
- [211] A. S. Georgiades, P. N. Belhumeur, and D. J. Kriegman, “From few to many: Illumination cone models for face recognition under variable lighting and pose,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 643–660, Jun. 2001. DOI: 10.1109/34.927464.
- [212] Kuang-Chih Lee, J. Ho, and D. J. Kriegman, “Acquiring linear subspaces for face recognition under variable lighting,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 684–698, May 2005. DOI: 10.1109/TPAMI.2005.92.

- [213] A. M. Martinez and R. Benavente, “The ar face database,” *CVC Tech. Report*, vol. 24, Jun. 1998.
- [214] A. M. Martinez and A. C. Kak, “PCA versus LDA,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228–233, Feb. 2001. DOI: 10.1109/34.908974.
- [215] S. K. N. S. A. Nene and H. Murase, “Columbia object image library (coil-100),” *Technical Report CUCS-006-96*, Feb. 1996.
- [216] H. Xiao, K. Rasul, and R. Vollgraf. (Aug. 28, 2017). “Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms.” arXiv: [cs.LG/1708.07747](https://arxiv.org/abs/1708.07747) [cs.LG].
- [217] L. van der Maaten and G. Hinton, “Visualizing Data using t-SNE,” *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.
- [218] S. Zagoruyko and N. Komodakis, “Wide Residual Networks,” en, in *Proceedings of the British Machine Vision Conference 2016*, York, UK: British Machine Vision Association, 2016, pp. 87.1–87.12. DOI: 10.5244/C.30.87.
- [219] A. Makhzani and B. J. Frey, “Winner-Take-All Autoencoders,” in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., Curran Associates, Inc., 2015, pp. 2791–2799.
- [220] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” en, in *PMLR*, Jun. 2015, pp. 448–456.
- [221] J. Hu, L. Shen, and G. Sun, “Squeeze-and-Excitation Networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.
- [222] S. Lloyd, “Least squares quantization in PCM,” *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982. DOI: 10.1109/TIT.1982.1056489.
- [223] A. Y. Ng, M. I. Jordan, and Y. Weiss, “On Spectral Clustering: Analysis and an algorithm,” in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds., MIT Press, 2002, pp. 849–856.

- [224] U. von Luxburg, “A tutorial on spectral clustering,” en, *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, Dec. 2007. DOI: 10.1007/s11222-007-9033-z.
- [225] F. Murtagh and P. Contreras, “Algorithms for hierarchical clustering: An overview,” en, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 86–97, 2012. DOI: 10.1002/widm.53.
- [226] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD’96, AAAI Press, 1996, pp. 226–231.
- [227] Daxin Jiang, Chun Tang, and Aidong Zhang, “Cluster analysis for gene expression data: A survey,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 11, pp. 1370–1386, Nov. 2004. DOI: 10.1109/TKDE.2004.68.
- [228] Z. He, X. Xu, and S. Deng, “Discovering cluster-based local outliers,” *Pattern Recognition Letters*, vol. 24, no. 9, pp. 1641–1650, Jun. 2003. DOI: 10.1016/S0167-8655(03)00003-5.
- [229] A. D. Marco and R. Navigli, “Clustering and Diversifying Web Search Results with Graph-Based Word Sense Induction,” en-us, *Computational Linguistics*, vol. 39, no. 3, pp. 709–754, 2013. DOI: 10.1162/COLI_a_00148.
- [230] S. T. Roweis and L. K. Saul, “Nonlinear Dimensionality Reduction by Locally Linear Embedding,” en, *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000. DOI: 10.1126/science.290.5500.2323.
- [231] B. Nasihatkon and R. Hartley, “Graph connectivity in sparse subspace clustering,” in *CVPR 2011*, Jun. 2011, pp. 2137–2144. DOI: 10.1109/CVPR.2011.5995679.
- [232] X. Zhu, X. Li, S. Zhang, C. Ju, and X. Wu, “Robust Joint Graph Sparse Coding for Unsupervised Spectral Feature Selection,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 6, pp. 1263–1275, Jun. 2017. DOI: 10.1109/TNNLS.2016.2521602.
- [233] H. Hu, Z. Lin, J. Feng, and J. Zhou, “Smooth Representation Clustering,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2014, pp. 3834–3841. DOI: 10.1109/CVPR.2014.484.

- [234] F. Nie, X. Wang, and H. Huang, “Clustering and Projected Clustering with Adaptive Neighbors,” in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’14, New York, NY, USA: ACM, 2014, pp. 977–986. DOI: 10.1145/2623330.2623726.
- [235] F. Nie, X. Wang, M. I. Jordan, and H. Huang, “The Constrained Laplacian Rank Algorithm for Graph-Based Clustering,” en, in *Thirtieth AAAI Conference on Artificial Intelligence*, Mar. 2016.
- [236] J. Wen, X. Fang, Y. Xu, C. Tian, and L. Fei, “Low-rank representation with adaptive graph regularization,” *Neural Networks*, vol. 108, pp. 83–96, Dec. 2018. DOI: 10.1016/j.neunet.2018.08.007.
- [237] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, “Robust Recovery of Subspace Structures by Low-Rank Representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 171–184, Jan. 2013. DOI: 10.1109/TPAMI.2012.88.
- [238] Q. Wang, Z. Qin, F. Nie, and X. Li, “Spectral Embedded Adaptive Neighbors Clustering,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 4, pp. 1265–1271, Apr. 2019. DOI: 10.1109/TNNLS.2018.2861209.
- [239] F. Nie, Z. Zeng, I. W. Tsang, D. Xu, and C. Zhang, “Spectral Embedded Clustering: A Framework for In-Sample and Out-of-Sample Spectral Clustering,” *IEEE Transactions on Neural Networks*, vol. 22, no. 11, pp. 1796–1808, Nov. 2011. DOI: 10.1109/TNN.2011.2162000.
- [240] Q. He, X. Jin, C. Du, F. Zhuang, and Z. Shi, “Clustering in extreme learning machine feature space,” *Neurocomputing*, vol. 128, pp. 88–95, Mar. 2014. DOI: 10.1016/j.neucom.2012.12.063.
- [241] M. Liu, B. Liu, C. Zhang, and W. Sun, “Spectral Nonlinearly Embedded Clustering Algorithm,” en, *Mathematical Problems in Engineering*, 2016. DOI: 10.1155/2016/9264561.
- [242] L. Zelnik-Manor and P. Perona, “Self-tuning Spectral Clustering,” in *Proceedings of the 17th International Conference on Neural Information Processing Systems*, ser. NIPS’04, Cambridge, MA, USA: MIT Press, 2004, pp. 1601–1608.

- [243] K. Fan, “On a Theorem of Weyl Concerning Eigenvalues of Linear Transformations: II,” en, *Proceedings of the National Academy of Sciences*, vol. 36, no. 1, pp. 31–35, Jan. 1950. DOI: 10.1073/pnas.36.1.31.
- [244] D. B. Graham and N. M. Allinson, “Characterising Virtual Eigensignatures for General Purpose Face Recognition,” en, *Face Recognition*, pp. 446–456, 1998. DOI: 10.1007/978-3-642-72201-1_25.
- [245] F. S. Samaria and A. C. Harter, “Parameterisation of a stochastic model for human face identification,” in *Proceedings of 1994 IEEE Workshop on Applications of Computer Vision*, Dec. 1994, pp. 138–142. DOI: 10.1109/ACV.1994.341300.
- [246] A. Duller, G. Duller, I. France, and H. Lamb, “A pollen image database for evaluation of automated identification systems,” *Quaternary Newsletter*, vol. 89, pp. 4–9, Oct. 1999.
- [247] J. J. Hull, “A database for handwritten text recognition research,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, May 1994. DOI: 10.1109/34.291440.
- [248] B. Leibe and B. Schiele, “Analyzing appearance and contour based methods for object categorization,” in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, vol. 2, Jun. 2003, pp. II–409. DOI: 10.1109/CVPR.2003.1211497.
- [249] D. Dua and C. Graff, *UCI machine learning repository*, 2017.
- [250] M. Wu and B. Schölkopf, “A Local Learning Approach for Clustering,” in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. C. Platt, and T. Hoffman, Eds., MIT Press, 2007, pp. 1529–1536.
- [251] J. Ye, Z. Zhao, and M. Wu, “Discriminative K-means for Clustering,” in *Advances in Neural Information Processing Systems 20*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds., Curran Associates, Inc., 2008, pp. 1649–1656.
- [252] Y. Yang, D. Xu, F. Nie, S. Yan, and Y. Zhuang, “Image Clustering Using Local Discriminant Models and Global Integration,” *IEEE Transactions on Image Processing*, vol. 19, no. 10, pp. 2761–2773, Oct. 2010. DOI: 10.1109/TIP.2010.2049235.

- [253] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, “Deep Clustering for Unsupervised Learning of Visual Features,” en, in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., ser. Lecture Notes in Computer Science, Springer International Publishing, 2018, pp. 139–156.
- [254] C. You, D. Robinson, and R. Vidal, “Scalable Sparse Subspace Clustering by Orthogonal Matching Pursuit,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3918–3927.
- [255] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, “Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering,” *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007. DOI: 10.1109/TIP.2007.901238.
- [256] L. Xu, C. Lu, Y. Xu, and J. Jia, “Image smoothing via L 0 gradient minimization,” in *ACM Transactions on Graphics (TOG)*, vol. 30, ACM, 2011, p. 174.
- [257] W. Dong, L. Zhang, G. Shi, and X. Li, “Nonlocally Centralized Sparse Representation for Image Restoration,” *IEEE Transactions on Image Processing*, vol. 22, no. 4, pp. 1620–1630, Apr. 2013. DOI: 10.1109/TIP.2012.2235847.
- [258] B. Zhao, L. Fei-Fei, and E. P. Xing, “Online detection of unusual events in videos via dynamic sparse coding,” in *CVPR 2011*, Jun. 2011, pp. 3313–3320. DOI: 10.1109/CVPR.2011.5995524.
- [259] D. Batenkov, Y. Romano, and M. Elad, “On the Global-Local Dichotomy in Sparsity Modeling,” *arXiv:1702.03446 [cs, math, stat]*, Feb. 2017. arXiv: 1702.03446 [cs, math, stat].
- [260] V. Pappayan, Y. Romano, and M. Elad, “Convolutional Neural Networks Analyzed via Convolutional Sparse Coding,” *Journal of Machine Learning Research*, vol. 18, no. 83, pp. 1–52, 2017.
- [261] D. Davis and W. Yin, “Convergence Rate Analysis of Several Splitting Schemes,” en, in *Splitting Methods in Communication, Imaging, Science, and Engineering*, ser. Scientific Computation, R. Glowinski, S. J. Osher, and W. Yin, Eds., Cham: Springer International Publishing, 2016, pp. 115–163. DOI: 10.1007/978-3-319-41589-5_4.

- [262] J. F. Bonnans and A. Shapiro, “Optimization Problems with Perturbations: A Guided Tour,” *SIAM Review*, vol. 40, no. 2, pp. 228–264, Jan. 1998. DOI: 10.1137/S0036144596302644.
- [263] J. M. Danskin, *The Theory of Max-Min and Its Application to Weapons Allocation Problems*, en, ser. Ökonometrie Und Unternehmensforschung Econometrics and Operations Research. Berlin Heidelberg: Springer-Verlag, 1967. DOI: 10.1007/978-3-642-46092-0.
- [264] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” 2009.
- [265] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, *The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results*, <http://host.robots.ox.ac.uk/pascal/VOC/voc2007/>.
- [266] K. He, X. Zhang, S. Ren, and J. Sun, “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.
- [267] Z.-H. Zhou, “A brief introduction to weakly supervised learning,” en, *National Science Review*, vol. 5, no. 1, pp. 44–53, Jan. 2018. DOI: 10.1093/nsr/nwx106.
- [268] O. Vinyals, C. Blundell, T. Lillicrap, k. kavukcuoglu, and D. Wierstra, “Matching Networks for One Shot Learning,” in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds., Curran Associates, Inc., 2016, pp. 3630–3638.
- [269] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang, “A Closer Look at Few-shot Classification,” in *International Conference on Learning Representations*, Sep. 2018.
- [270] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. de Freitas, “Learning to learn by gradient descent by gradient descent,” *arXiv:1606.04474 [cs]*, Nov. 2016. arXiv: 1606.04474 [cs].
- [271] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, “Meta-Learning in Neural Networks: A Survey,” *arXiv:2004.05439 [cs, stat]*, Apr. 2020. arXiv: 2004.05439 [cs, stat].

- [272] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roseland, “Towards Federated Learning at Scale: System Design,” *arXiv:1902.01046 [cs, stat]*, Mar. 2019. arXiv: 1902.01046 [cs, stat].