CONVEX AND NON-CONVEX OPTIMIZATION METHODS FOR MACHINE LEARNING

by

FARIBA ZOHRIZADEH

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON
August 2019

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my supervisor Professor Farhad Kamangar and my co-supervisor Professor Ramtin Madani for their endless support, invaluable advice and immense knowledge during the course of my doctoral studies. I am extremely thankful of them for making the Ph.D. program a superb experience for me. I am also grateful to my committee members Professors Vassilis Athitsos, Gergely Zaruba, and Manfred Huber for providing me insightful suggestions and valuable feedback.

I cannot be grateful enough to my parents and my brothers who sacrificed their lives for my success and have always encouraged and inspired me. Foremost, I would like to thank my best friend and my husband Mohsen. It has been a privilege and truly an honor to collaborate with him. Without his love, support, and encouragement, I would not have finished this journey.

August 2019

ABSTRACT

CONVEX AND NON-CONVEX OPTIMIZATION METHODS FOR MACHINE LEARNING

Fariba Zohrizadeh, Ph.D.

The University of Texas at Arlington, 2019

This dissertation is concerned with modeling fundamental and challenging machine learning tasks as convex/non-convex optimization problems and designing a mechanism that could solve them in a cost and time-effective manner. Extensive theoretical and practical studies are carried out to give deeper insights into the robustness and effectiveness of the formulated problems. In what follows, we investigate some well-known tasks that frequently arise in machine learning applications.

Image Segmentation: Image segmentation is a fundamental and challenging task in computer vision with diverse applications in various areas. One of the major challenges in image segmentation is to determine the optimal number of coherent regions. This dissertation develops a novel and highly parallelizable convex model which takes into account the spatial relationship between the image features and simultaneously determines the number of clusters and their associated members. To solve the model, a computationally efficient algorithm is presented based on the alternating direction method of multiplier. Extensive experiments on benchmark image segmentation datasets demonstrate that the proposed method can provide high quality and competitive results compared to the existing state-of-the-art methods.

Convex Relaxation for Solving Optimization Problems with Orthogonality Constraints: A class of optimization problems with orthogonality constraints has been used to model various applications in machine learning such as discriminative dimensionality reduction, graph matching, dictionary learning, etc. Such optimization problems include nonconvex nonlinear equations, that substantively increase the computational complexity of the problems. In this dissertation, we develop a sequential approach based on *parabolic* relaxation which finds an orthogonal matrix that minimizes a non-convex and non-smooth objective function subject to additional quadratic constraints. We prove that under very mild assumptions, the proposed approach is guaranteed to provide a feasible solution for the original non-convex problem. The effectiveness of the proposed scheme is corroborated for the problem of discriminative dimensionality reduction and graph clustering.

Convex Relaxation for Training Neural Networks: Training of a neural network is formulated as a complex optimization problem which is non-convex and inherently hard to solve. In this dissertation, we propose a novel convexification approach that reduces the training problem into solving a sequence of polynomial-time solvable convex surrogates. The proposed approach, called *convexified neural network* (Convex-NN), jointly estimates the network parameters of all layers and can admit a wide range of additional convex constraints. We theoretically prove that Convex-NN is guaranteed to converge under mild conditions and perform empirical experiments to corroborate the effectiveness of the method.

Class Subset Selection for Partial Domain Adaptation: Deep neural networks have demonstrated superior performance in a variety of machine learning problems. These impressive achievements often rely on the availability of large amounts of labeled training data. However, in many applications, the acquisition of sufficient labeled data is difficult and time-consuming. This provides a strong motivation to reduce the labeling cost and effort by learning effective predictive models using richly-annotated datasets and transferring the knowledge to unlabeled datasets from different but related domains. This dissertation proposes an adversarial-based method for the problem of *partial domain adaptation* (PDA) in which the source label space is a subset of the target label space. Empirical results demonstrate the high potential of the proposed approach in addressing different partial domain adaptation tasks.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

Introduction

Mathematical models are ubiquitous in many areas of science and engineering including machine learning, computer vision, and pattern recognition. These models provide quantitative frameworks for characterizing and understanding many complex machine learning problems such as clustering, regression, dimensionality reduction, and segmentation. The mathematical models are often formulated as optimization problems with many variables and constraints. In this regard, extensive research studies are carried out to develop robust and computationally tractable algorithms for solving different classes of optimization problems.

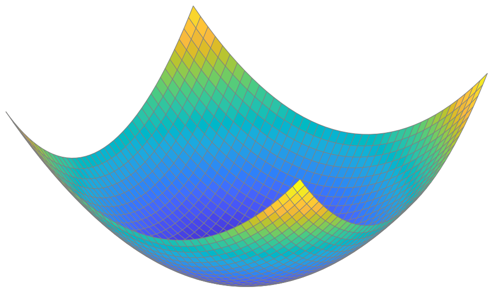
A mathematical optimization problem has the generic form

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad f(\boldsymbol{x}) \tag{1.1a}$$

$$\text{subject to} \quad \boldsymbol{x}\in\mathcal{C}, \tag{1.1b}$$

where vector $\boldsymbol{x}\in\mathbb{R}^n$ is the optimization variable, function $f:\mathbb{R}^n\to\mathbb{R}$ is the objective function and $\mathcal{C}$ is the cone of feasible solutions. A vector $\boldsymbol{x}^*$ is called an *optimal* solution if it has the smallest objective value among all $\boldsymbol{x}\in\mathcal{C}$. The optimization problem (1.1a) – (1.1b) is called a *convex* problem if (i) $f(\boldsymbol{x})$ is convex and (ii) $\mathcal{C}$ is a convex set, and *non-convex* otherwise. Figure 1.1 depicts a simple example of convex and nonconvex problem with objective functions $f(x_1,x_2)=x_1^2+x_2^2$ and $f(x_1,x_2)=(x_1^2+x_2-11)^2+(x_1+x_2^2-7)^2$, and convex cone $\mathcal{C}=\{\boldsymbol{x}\in\mathbb{R}^2|\, x_i\in[-4,4],\,\forall i\}$.

Convex optimization has attracted intense research attention due to its theoretical guarantees and computational tractability [13]. Many fundamental problems in machine


(a)

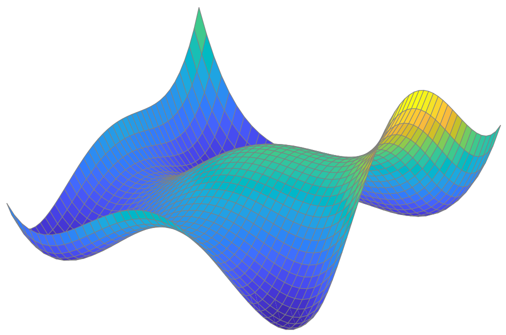

(b)

Figure 1.1: Example of a convex and a nonconvex problem. (a) convex problem with $f(x_1, x_2) = x_1^2 + x_2^2$ and $x_1, x_2 \in [-4, 4]$ (b) non-convex problem with $f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$ and $x_1, x_2 \in [-4, 4]$.

learning and computer vision such as clustering and classification [14, 15, 16, 17, 18, 19, 20, 21, 22], matrix completion [23], image segmentation [24, 25], object tracking [26, 27, 28], video summarization [29, 30], etc, can be expressed in the framework of convex optimization. Since these problems often involve highly complex and high-dimensional datasets, it is essential to be able to solve them efficiently. In this regard, there exists a variety of successful algorithms such as gradient methods [31, 32, 33], proximal methods [34, 13], Bregman iterative methods [35, 36], alternating direction method of multipliers (ADMM) [37], interior-point methods [38, 39, 40], etc, which are mostly used to solve convex problems.

Although in the past several years, a great research effort has been devoted to modeling real-world problems as convex optimization problems and developing efficient numerical algorithms to solve them, these models are unable to capture the underlying complexity of the data for sophisticated applications in domains such as machine learning and computer vision. In such cases, additional sophisticated constraints are required to enrich the models in terms of robustness and accuracy. These constraints can naturally introduce non-convexities into the enriched models. Examples of applications that benefit from non-convex optimization techniques include discriminative dimensionality reduction

[41, 4], outlier and anomaly detection [42, 43], graph clustering [41, 44] and graph matching [45, 46, 47].

The absence of convexity introduces several difficulties into the optimization problem, including the distinction between the local and global minima, lack of global optimality criteria, etc, that leads to a substantial computational complexity while passing from convex to nonconvex programming. The existing methods based on heuristics or local-search algorithms such as proximal gradient descent, conditional gradient descent [48], and ADMM produce a candidate solution without being able to measure its closeness to a global minimum. To tackle this issue, one of the most promising directions is to adopt convex relaxation techniques such as linear programming (LP), semidefinite programming (SDP) [49, 50], and second-order cone programming (SOCP) [51, 52]. The main goal of these techniques is to relax the non-convex problem to a convex surrogate (shown in Figure 1.2) which is computationally tractable to solve and provides solutions with small optimally gap. These approaches have been widely investigated and improved in terms of computational complexity and solution quality, nevertheless, developing an efficient approach for solving a large-scale nonconvex problem is still an intriguing open problem.

This dissertation is concerned with modeling computer vision and machine learning applications via convex and non-convex optimization problems and designing a mechanism that could solve them in a cost and time effective manner. In Chapter 2, we present an effective convex model for the problem of unsupervised image segmentation. Chapter 3 develops an efficient approach to solve nonconvex problems with orthogonality constraints. These problems are briefly discussed in the following sections.

1.1 Image Segmentation using Sparse Subset Selection

Image segmentation is a fundamental and challenging task in computer vision. One of the major challenges in image segmentation is to determine the optimal number of coherent regions. This parameter can be calculated based on the distribution of image features, given as a prior knowledge or set to a constant value, depending on the segmentation methodology. The performance of segmentation methods heavily depends on the right choice of this parameter. In the case that the number of coherent regions is overestimated, each coherent region may be divided into many separate segments. To merge these segments, many time-consuming and complicated steps are required which in turn increase the computational complexity and decrease the performance of the algorithm. On the other hand, when the number of coherent regions is underestimated, some coherent regions are forced to be merged together which in turn leads to a poor segmentation quality.

Generally, determination of the number of coherent regions in image segmentation is similar to the problem of finding the optimal number of clusters in unsupervised clustering [53, 54, 55]. These problems are similar in a sense that they both seek to find the optimal number of groups. However, they may be different depending on the nature and intrinsic properties of their input data. For instance, in image segmentation, the spatial relationship among the image features can be perceived as an important property of data. These properties need to be taken into account in determining the optimal number of coherent regions.

In this dissertation, we transform the problem of image segmentation to an unsupervised clustering problem in which the coherent regions and pixels are respectively corresponding to the clusters and cluster members. This problem is modeled by a highly parallelizable convex model which takes into account the spatial relationship between the image features and simultaneously determines the number of clusters and their associated members. To solve the model, a computationally efficient and parallelizable numerical al-

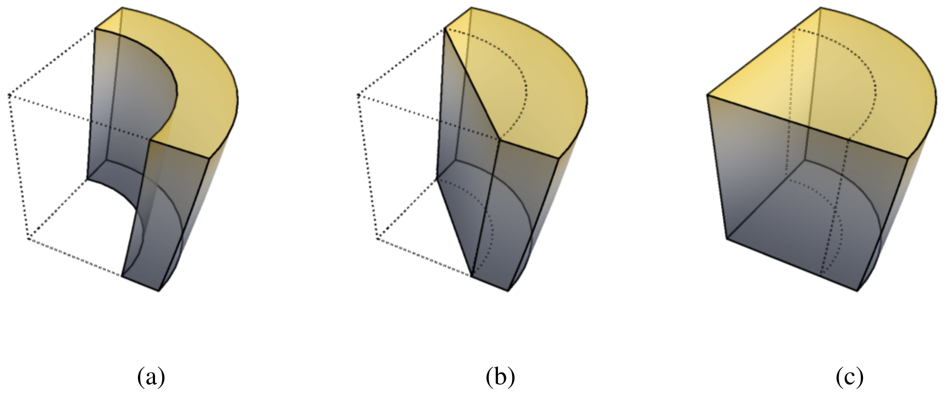


Figure 1.2: Convex relaxation of a non-convex set. (a) Non-convex set; (b) The tightest convex relaxation (convex hull); (c) A standard convex relaxation.

gorithm is presented based on ADMM. Due to the generalization power of the model, it can be simply employed in several other contexts such as clustering, data summarization, etc.

1.2 Convex Relaxations for Optimization Under Orthogonality Constraints

A class of optimization problems with orthogonality constraints has been used to model various applications in machine learning and computer vision such as discriminative dimensionality reduction [4], graph matching [56], feature selection [57, 58], dictionary learning [59, 60], etc. Such optimization problems include nonconvex nonlinear equations, that substantively increase the computational complexity of the problems.

There has been an explosion of research in developing efficient algorithms to find an orthogonal matrix that minimizes a non-convex and smooth objective function. The papers [61, 62, 63, 64, 65] proposed local search algorithms which preserve the manifold structure of orthogonality constraint during iterations via geodesics or retractions. Although these algorithms show satisfactory performance in dealing with orthogonality constraints, they mostly restrict the objective function to the class of smooth functions. Such limitations can considerably reduce the extent of their applicability in many domains including machine learning. To tackle this limitation, a series of splitting techniques are proposed in [66, 67, 68, 69] to deal with non-smooth objective functions. In these papers, the problem is divided into multiple sub-problems with analytical solutions that are solvable using Bregman iterations [36] or its variant [37]. The aforementioned methods mostly lack guaranteed convergence and theoretical analysis or admit no additional constraints.

In this dissertation, we develop an efficient approach to find an orthogonal matrix that minimizes a non-convex and non-smooth objective function subject to additional quadratic constraints. The proposed approach is based on convex relaxation, which transforms the problem into polynomial-time solvable convex surrogates. As an alternative to the common practice semidefinite programming (SDP) relaxation, we propose a novel *parabolic* relaxation, which only relies on convex quadratic constraints. In order to obtain feasible and near-globally optimal solutions, a penalization technique is developed, which is compatible with both SDP and parabolic relaxations. We prove that given an initial point, the penalized relaxation is guaranteed to provide a feasible solution for the original non-convex problem if the point lies within an analytical bound from its feasible set. Given that, we propose a constraint-preserving sequential scheme which solves a sequence penalized relaxations to obtain near-globally optimal solutions. Experimental results on synthetic and real datasets demonstrate the effectiveness of the proposed scheme for the problem of discriminative dimensionality reduction and graph clustering.

1.3 Convex Relaxation for Training Neural Networks

Deep neural networks have been demonstrated special ability in extracting sophisticated information from the raw data. This renders them suitable tools for a wide variety of applications in artificial intelligence and machine learning including classification

[70, 71, 72, 73], depth estimation [74, 75], speech recognition [76, 77], etc. Despite the impressive empirical success, a complete theoretical understanding about such extraordinary performance is still lacking. Great efforts have been devoted to providing theoretical insights into training neural networks with special architectures [78, 79, 80, 81, 82, 83]. However, there still exist several unexplored avenues along this new line of research.

Training of a neural network is formulated as a complex optimization problem which is non-convex and inherently hard to solve. In this dissertation, we aim to establish a bridge between the areas of artificial neural networks and convex optimization by developing a powerful and flexible training approach. To serve this purpose, we first transform the training problem into an equivalent constrained optimization problem. Then, we convexify this constrained optimization by means of a novel convex quadratic relaxation and the well-known difference of convex programming technique. To ensure that the convexified problem provides a feasible point to the training problem, a novel regularization term is incorporated into the objective of the relaxed problems.

On the theoretical front, we derive certain conditions under which the regularized relaxation is guaranteed to provide feasible points for the training problem. Moreover, we theoretically prove that, if certain assumptions are met, solving a sequence of the regularized problem results in a convergent sequence of feasible points whose objective values monotonically improve. The proposed approach offers various theoretical and practical advantages: it jointly estimates the network parameters, admits additional convex constraints, and provides a flexible framework for further study and exploration. The potential of the proposed approach is corroborated on the problem of imbalanced classification.

1.4 Class Subset Selection for Partial Domain Adaptation

Deep neural networks have demonstrated superior performance in a variety of machine learning problems such as semantic image segmentation [84, 85, 86], object detection and classification [87, 70, 88], etc. These impressive achievements often rely on the availability of large amounts of labeled training data. However, in many applications, the acquisition of sufficient labeled data is difficult and time consuming. This provides a strong motivation to reduce the labeling cost and effort by learning effective predictive models using richly-annotated datasets and transferring the knowledge to unlabeled datasets from different but related domains. This paradigm generally suffers from the domain shift between the distributions of the source and the target datasets. Unsupervised Domain Adaptation (UDA) is a common mechanism which aims to reduce the gap between the source and target domains by learning discriminative predictors that generalize well on the target domain with no labeled data [89, 90, 91]. Despite the advantages offered by the UDA problem, its applicability is mainly limited to special cases in which the source and the target domains possess the same set of classes. With the goal of considering a more realistic practical cases, [92] introduced partial domain adaptation (PDA) which assumes the source label space is a subset of the target label space. Hence, the primary challenge in PDA is to identify and reject the *outlier classes*, i.e. those classes in source domain that do not appear in the target domain, because they have adverse effect on discriminating the target classes [93]. Addressing this challenge enables PDA to transfer models learned from large-scale datasets (e.g. ImageNet) to unlabeled small-scale datasets from different but related domains. In this dissertation, we propose an adversarial-based method for the problem of partial domain adaptation. The proposed approach is inspired by the concept of subset selection.

CHAPTER 2

Image Segmentation using Sparse Subset Selection

In this chapter, we present a new image segmentation method based on the concept of sparse subset selection. Starting with an over-segmentation, we adopt local spectral histogram features to encode the visual information of the small segments into high-dimensional vectors, called superpixel features. Then, superpixel features are fed into a novel convex model which efficiently leverages the features to group the superpixels into a proper number of coherent regions. Our model automatically determines the optimal number of coherent regions and superpixels assignment to shape final segments. To solve our model, we propose a numerical algorithm based on the alternating direction method of multipliers (ADMM), whose iterations consists of two highly parallelizable sub-problems. We show each sub-problem enjoys closed-form solution which makes the ADMM iterations computationally very cheap. Extensive experiments on benchmark image segmentation datasets demonstrate that our proposed method in combination with an over-segmentation can provide high quality and competitive results compared to the existing state-of-the-art methods.

2.1 Introduction

Image segmentation is a fundamental and challenging task in computer vision with diverse applications in various areas, such as video segmentation [94, 95], object segmentation [96, 97, 98], and scene understanding [99]. The primary challenges of image segmentation are rooted in the diversity and ambiguity of visual textures encountered in input

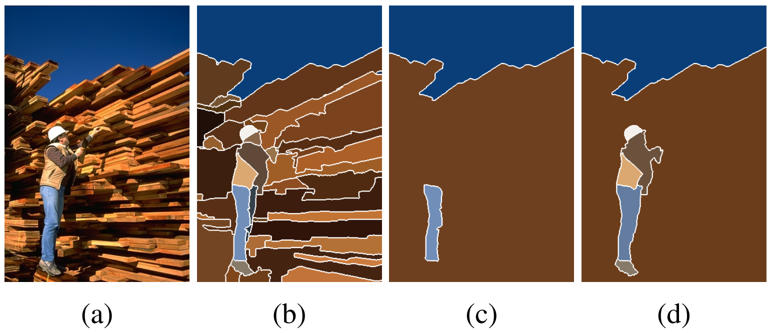


Figure 2.1: Segmenting image pixels into multiple coherent regions. (a) Input image. (b-d) Segmentation results when the number of coherent regions is overestimated (b), underestimated (c), and properly determined (our method) (d).

images. The solution to these challenges has been the subject of some research studies in the recent years [100, 1, 101].

One of the major challenges in image segmentation is to determine the optimal number of coherent regions. This parameter can be calculated based on the distribution of image features [101], given as a prior knowledge [102, 103, 24] or set to a constant value [100], depending on the segmentation methodology. The performance of segmentation methods heavily depends on the right choice of this parameter, denoted by K. Figure 2.1 illustrates the segmentation results obtained for various choices of K. In the case that K is overestimated (shown in Figure 2.1b), each coherent region may be divided into many separate segments. To merge these segments, many time-consuming and complicated steps are required which in turn increase the computational complexity and decrease the performance of the algorithm. On the other hand, when K is underestimated (shown in Figure 2.1c), some coherent regions are forced to be merged together which in turn leads to a poor segmentation quality. Figure 2.1d shows our method has achieved a high quality segmentation by properly determining parameter K.

Generally, determination of the number of coherent regions in image segmentation is nearly similar to the problem of finding the optimal number of clusters in other areas

[53, 54, 55]. These problems are similar in a sense that they both seek to find the optimal number of groups. However, they may be different depending on the nature and intrinsic properties of their input data. For instance, in image segmentation, the spatial relationship among the image features can be perceived as an important property of data. Features may also have more specific properties depending on the feature extraction procedure. These properties need to be taken into account in determining the optimal number of coherent regions.

In this work, we adopt local spectral histogram (LSH) features [104] to model the input image. These features are computed by averaging the distribution of visual properties (such as color, texture, etc.) over a local patch centered at each pixel. Therefore, they can be considered as powerful tools to encode the local texture information. As LSH features are computed by averaging distributions in a local neighborhood, it can be concluded that they are always nonnegative and the features belonging to the same coherent region are linearly dependent to each other. Our method leverages these properties to develop a convex model based on the concept of sparse subset selection. The main contributions of this work can be summarized as follows:

I: We design an effective convex model based on the properties of LSH features which automatically determines the optimal number of coherent regions and pixels assignment.

II: We develop a parallel numerical algorithm based on the alternating direction method of multiplier [37, 105] whose iterations consists of two sub-problems with closed-form solutions. We show the proposed algorithm can solve our model significantly faster than the standard convex solvers [8, 6, 7] while maintaining a high accuracy.

III: We conduct extensive experiments on three commonly used datasets, BSD300 [106], BSD500 [100], and MSRC [107] to show our results are competitive comparing to the results of state-of-the-arts methods.

The remainder of this work is structured as follows: Section 2, shortly reviews related works; Section 3, explains our method in detail; Section 4, provides experimental results; Section 5, draws a conclusion about this work.

Notation: Throughout this work, matrices, vectors, and scalars are denoted by boldface uppercase, boldface lowercase, and italic lowercase letters, respectively. For a given matrix $\boldsymbol{A}$, symbol $\boldsymbol{A}_{i,j}$ denotes the element at i^{th} row and j^{th} column, $\|\boldsymbol{A}\|_F$ indicates the Frobenius norm, and $\|\boldsymbol{A}\|_{p,q}$ is the ℓ_p norm of the ℓ_q norm of the rows in $\boldsymbol{A}$. For a given vector $\boldsymbol{A}$, symbols $\|\boldsymbol{A}\|_p$, $diag(\boldsymbol{A})$, and a_i denote standard ℓ_p norm, a diagonal matrix formed by the elements of $\boldsymbol{A}$, and the i^{th} element of $\boldsymbol{A}$, respectively. Symbol $tr(.)$ stands for the trace operator, $\mathbb{R}_+$ indicates the set of positive real numbers, and $\mathbf{1}$ is a column vector of all ones of appropriate dimension.

2.2 Related Work

Over-segmentation is obtained by partitioning the input image into multiple small homogeneous regions, called superpixels. Recent segmentation algorithms usually utilize an over-segmentation and merge the similar superpixels to shape final segments [1, 103, 24]. Fu [1] proposed a pipeline of three effective post-processing steps which are applied on an over-segmentation to shape final segments. Li [103] suggested to construct a bipartite graph over multiple over-segmentations provided by [108] and [109]. Then, the spectral clustering is applied on the graph to form final segments. Ren [2] presented a method which constructs a cascade of boundary classifiers and iteratively merges the superpixels of an over-segmentation to build final segments.

One notable segmentation method is presented by Arbelaez in [100], which reduces the problem of image segmentation to a contour detection. The method combines multiple contour cues of different image layers to create a contour map, called gPb. The contour map and its corresponding hierarchical segmentation further utilized by some algorithms as an initial over-segmentation [110, 111, 112, 113]. Liu [113] trained a classifier over the gPb results to construct a hierarchical region merging tree. Then, the classifier iteratively merges the most similar regions to shape final segments. Gao [111] proposed to construct a graph over the gPb results based on the spatial and visual information of superpixels. Then, a model is proposed to partition the graph into multiple components where each one is corresponding to a final segment. Recently, a widely-used extension of gPb is proposed by Arbelaez, called Multiscale Combinatorial Grouping (MCG) [3]. The method combines the information obtained from multiple image scales to generate a hierarchical segmentation. Yu [114] proposed a nonlinear embedding method based on a ℓ_1-regularized objective which is integrated into MCG framework to provide better local distances among the superpixels. Chen [115] realigned the MCG results by modifying the depth of segments in its hierarchical structure.

There are some recently-developed methods based on deep learning in many related tasks such as contour detection [116, 117] and semantic image segmentation [118, 119, 120, 121]. Although these methods are able to exploit more sophisticated and complex representative features, they are often highly demanding in terms of training data and training time. Therefore, these methods may not be the most appropriate choice in some applications. To illustrate, consider natural image segmentation in which many unknown and diverse patterns are likely to be presented in each single image. This implies that we may have insufficient number of training samples per each pattern. Motivated by this, we propose a new image segmentation method based on the concept of sparse subset selection [122, 123]. The method starts with an over-segmentation (e.g., MCG) and uses an effective convex model to group the superpixels into a proper number of coherent regions. Our work is roughly similar to the factorization-based segmentation (Fact) algorithm [101] in the sense that both use local spectral histogram (LSH) features to model the input image and seek to estimate the optimal number of coherent regions. However, our method differs from Fact in two major ways: (1) Fact determines the optimal number of coherent regions and pixels assignment in two consecutive steps which may lead to error propagation, but our model simultaneously determines the optimal number of coherent regions and pixels assignment in an effective manner. (2) Fact does not take advantage of the spatial information among pixels, but we incorporate this information as a Laplacian regularization term in our convex model. Moreover, we propose a parallel numerical algorithm based on the alternating direction method of multipliers [37, 105] to solve our model and obtain final segments. Note that the model can be easily utilized in some other applications such as video summarization [122, 124] and dimensionality reduction [123].

2.3 Proposed Method

This section describes our segmentation method in two phases: problem formulation and numerical algorithm. The first phase formulates a convex model based on the properties of local spectral histogram (LSH) features and the second phase presents our solution to the model in details.

2.3.1 Problem Formulation

Given an input image $\mathbf{I}$, we start with an over-segmentation consisting n superpixels. We form feature matrix $\mathbf{X} = [\mathbf{x}_1|\mathbf{x}_2|\dots|\mathbf{x}_n] \in \mathbb{R}_+^{d\times n}$ by averaging the LSH features of pixels within each superpixel. Hence, each $\mathbf{x}_i$ is considered as a d-dimensional feature corresponding to the i^{th} superpixel. Under the assumption of linear dependence among the LSH features, we model the feature matrix $\mathbf{X}$ as,

$$\mathbf{X} = \mathbf{D}\mathbf{U} + \mathbf{E}, \tag{2.1}$$

where $\mathbf{D} = [\mathbf{d}_1|\mathbf{d}_2|\dots|\mathbf{d}_l] \in \mathbb{R}_{+}^{d\times l}$ is a dictionary of l words inferred from the superpixels features, $\mathbf{U} = [\mathbf{u}_1|\mathbf{u}_2|\dots|\mathbf{u}_n] \in \mathbb{R}^{l\times n}$ denotes a coefficient matrix whose rows indicate the contribution of each word in reconstructing $\mathbf{X}$, and $\mathbf{E} \in \mathbb{R}^{d\times n}$ indicates the model error. The goal is to design a model which takes into account the linear dependence and spatial relationship among the features to compute an optimal matrix $\mathbf{U}$.

In order to incorporate the linear dependence among the features into the model, we adopt a non-negative matrix factorization framework to construct $\mathbf{D}$ over the feature matrix $\mathbf{X}$. Let $\mathbf{R} \in \mathbb{R}^{l\times n}$ be a dissimilarity matrix where $\mathbf{R}_{j,i}$ indicates the dissimilarity between $\mathbf{d}_j$ and $\mathbf{x}_i$. We define the elements of this matrix as $\mathbf{R}_{j,i} = \|\mathbf{d}_j - \mathbf{x}_i\|_2^2$. In the case of normalized features and visual words, the dissimilarity only depends on the inner product between $\mathbf{d}_j$ and $\mathbf{x}_i$. In other words, it shows how well $\mathbf{x}_i$ is expressible by $\mathbf{d}_j$ which is a reasonable dissimilarity measure according to the linear dependence among the features. Since the superpixels are not necessarily of the same size, we define a diagonal regularization matrix $\mathbf{P} \in \mathbb{R}^{n\times n}$ whose diagonal elements show the portion of overall pixels lie within the superpixels. The elements of $\mathbf{P}$ scale each $\mathbf{R}_{j,i}$ by the size of the i^{th} superpixel.

In order to embed the spatial relationship among the superpixels into the model, we construct a graph over the initial over-segmentation. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ be the graph where nodes are superpixels and edges connect every pairs of adjacent superpixels with a weight specified by $\mathbf{W} \in \mathbb{R}^{n\times n}$. The edge weight between the adjacent superpixels i and j indicates their similarity and is defined as:

$$\mathbf{W}_{i,j} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{\sigma_x} - \bar{b}}, \tag{2.2}$$

where $\bar{b}$ is the average strength of their common boundary and σ_x controls the effect of feature distances on their similarity weight. Given such graph $\mathcal{G}$, we define Laplacian matrix $\mathbf{L} \in \mathbb{R}^{n\times n}$ as $\mathbf{L} = diag(\mathbf{W1}) - \mathbf{W}$.

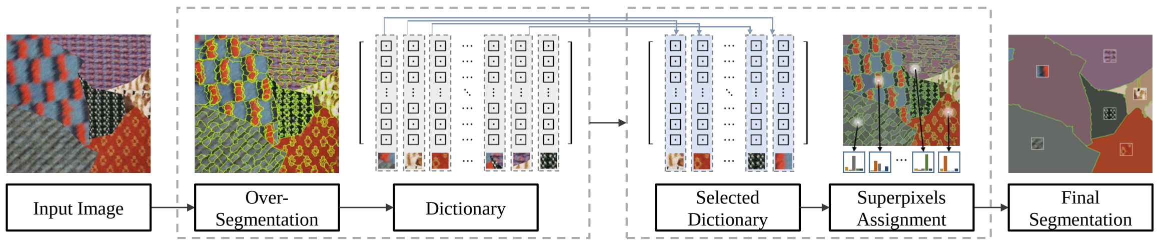


Figure 2.2: The pipeline of our proposed algorithm. Given an input image, we adopt an algorithm to generate a super-pixel segmentation layer. Then, we compute the superpixels features and learn a dictionary of words over all superpixels. Our convex model efficiently selects a small subset of informative words and softly assigns superpixels to the selected words. The neighboring superpixels which are assigned to the same selected words are merged to shape final segmentation.

Once the Laplacian matrix $\mathbf{L}$, the dissimilarity matrix $\mathbf{R}$, and the regularization matrix $\mathbf{P}$ are computed, we seek to find a small subset of the dictionary words that well represents feature matrix $\mathbf{X}$. To do so, a model is required which satisfies the following requirements:

- minimizes the number of selected words. In the ideal case, we are interested to have a single word corresponding to each coherent region.
- ensures each feature $\{\mathbf{x}_i\}_{i=1}^n$ is well expressible as a nonnegative linear combination of the selected words. The coefficients of such linear combination indicate the contribution of each selected word in reconstructing the feature.
- ensures each feature $\{\mathbf{x}_i\}_{i=1}^n$ is expressed by at least one selected word. To do so, we impose a constraint on the sum of the linear combination coefficients.
- takes advantage of the spatial relationship and linear dependence of the features.

Motivated by [122], we formulate the following convex model which fulfills the requirements.

$$\underset{\mathbf{U}\in\mathbb{R}^{\mathbf{l}\times\mathbf{n}}}{\text{minimize}}\ \mathrm{tr}(\mathbf{P}\mathbf{R}^\top\mathbf{U})+\gamma\mathrm{tr}(\mathbf{U}\mathbf{L}\mathbf{U}^\top)+\lambda\|\mathbf{U}\|_{1,\infty} \tag{2.3a}$$

$$\text{subject to } \mathbf{U}\geq 0, \tag{2.3b}$$

$$\mathbf{1}^\top\mathbf{U}=\mathbf{1}^\top, \tag{2.3c}$$

where $\gamma > 0$ and $\lambda > 0$ are regularization parameters. The first term in (2.3) is corresponding to the cost of representing feature matrix $\mathbf{X}$ using dictionary $\mathbf{D}$ proportional to the size of superpixels. The Laplacian regularization term incorporates the spatial relation of superpixels into the objective and the last term is a row sparsity regularization term which penalizes the objective in proportion to the number of selected words. Note that although $\mathbf{D}$ does not directly appear in (2.3), the rows of $\mathbf{U}$ are constructed based on the contribution of the dictionary words, $\{\mathbf{d}_j\}_{j=1}^{l}$, in reconstructing $\mathbf{X}$.

The optimal solution of problem (2.3) is $\mathbf{U}^* \in [0,1]^{l\times n}$ whose nonzero rows are corresponding to the selected words. Note that $\mathbf{U}^*$ not only determines the selected words but also shows the contribution of selected words in reconstructing the superpixel features $\{\mathbf{x}_i\}_{i=1}^{n}$. Hence, the elements of $\mathbf{U}^*$ can be interpreted as a soft assignment of the superpixels to the selected words. In this case, the i^{th} superpixel is assigned to the selected word which has the largest contribution in the reconstruction of $\mathbf{x}_i$. Final segmentation is obtained by merging the neighboring superpixels which are assigned to the same selected word. Figure 2.2 illustrates our segmentation pipeline in details.

2.3.2 Numerical Algorithm

This section presents a numerical algorithm based on the alternating direction method of multipliers (ADMM) to solve our model. Let define $\boldsymbol{m} \in \mathbb{R}^l$ such that $\mathrm{m}_j = \arg\,\max_i |\mathbf{U}_{j,i}|$ and reformulate (2.3) as follows:

$$\underset{\substack{\mathbf{U}\in\mathbb{R}^{l\times n}\\ \boldsymbol{m}\in\mathbb{R}^{l}}}{\text{minimize}}\ \operatorname{tr}(\mathbf{P}\mathbf{R}^\top\mathbf{U}) + \gamma\operatorname{tr}(\mathbf{U}\mathbf{L}\mathbf{U}^\top) + \lambda\mathbf{1}^\top\boldsymbol{m} \tag{2.4a}$$

$$\text{subject to } \mathbf{U} \geq 0, \tag{2.4b}$$

$$\mathbf{1}^\top\mathbf{U} = \mathbf{1}^\top, \tag{2.4c}$$

$$\boldsymbol{m}\mathbf{1}^\top \geq \mathbf{U}. \tag{2.4d}$$

Note that (2.4d) is imposed to ensure the equivalence of (2.3) and (2.4). This inequality constraint can be transformed into an equality constraint by introducing a slack variable $\mathbf{V} = [\mathbf{v}_1|\dots|\mathbf{v}_n] \in \mathbb{R}^{l\times n}$. Therefore, (2.4) is rewritten as:

$$\underset{\substack{\mathbf{U},\mathbf{V}\in\mathbb{R}^{l\times n}\\ \boldsymbol{m}\in\mathbb{R}^{l}}}{\text{minimize}}\ \operatorname{tr}(\mathbf{P}\mathbf{R}^\top\mathbf{U}) + \gamma\operatorname{tr}(\mathbf{U}\mathbf{L}\mathbf{U}^\top) + \lambda\mathbf{1}^\top\boldsymbol{m} \tag{2.5a}$$

$$\text{subject to } \mathbf{U} \geq 0, \tag{2.5b}$$

$$\mathbf{1}^\top\mathbf{U} = \mathbf{1}^\top, \tag{2.5c}$$

$$\boldsymbol{m}\mathbf{1}^\top = \mathbf{V} + \mathbf{U}, \tag{2.5d}$$

$$\mathbf{V} \geq 0. \tag{2.5e}$$

As $\mathbf{1}^\top\boldsymbol{m}\mathbf{1}^\top\mathbf{1} = \mathbf{1}^\top(\mathbf{V}+\mathbf{U})\mathbf{1}$, the third term of (2.5a) can be equivalently written as $\frac{\lambda}{n}\mathbf{1}^\top(\mathbf{V}+\mathbf{U})\mathbf{1}$. Hence, (2.5) can be reformulated independent of $\boldsymbol{m}$ as:

$$\underset{\mathbf{U},\mathbf{V}\in\mathbb{R}^{l\times n}}{\text{minimize}}\quad \operatorname{tr}(\mathbf{P}\mathbf{R}^\top\mathbf{U}) + \gamma\operatorname{tr}(\mathbf{U}\mathbf{L}\mathbf{U}^\top) + \frac{\lambda}{n}\mathbf{1}^\top(\mathbf{V}+\mathbf{U})\mathbf{1} \tag{2.6a}$$

$$\text{subject to}\quad \mathbf{U} \geq 0, \tag{2.6b}$$

$$\mathbf{1}^\top\mathbf{U} = \mathbf{1}^\top, \tag{2.6c}$$

$$\mathbf{v}_{i-1} + \mathbf{u}_{i-1} = \mathbf{v}_i + \mathbf{u}_i, \quad {\scriptstyle i=2,\dots,n}, \tag{2.6d}$$

$$\mathbf{V} \geq 0, \tag{2.6e}$$

where (2.6d) is obtained by removing $\boldsymbol{m}$ from (2.5d). In order to derive an ADMM formulation with subproblems possessing explicit formulas, we introduce auxiliary matrices $\hat{\mathbf{U}} \in \mathbb{R}^{l\times n}$, $\hat{\mathbf{V}} \in \mathbb{R}^{l\times n}$ and reformulate (2.6) as:

$$\begin{aligned}
\underset{\mathbf{U},\mathbf{V},\hat{\mathbf{U}},\hat{\mathbf{V}}\in\mathbb{R}^{l\times n}}{\text{minimize}} \quad & \operatorname{tr}(\mathbf{P}\mathbf{R}^\top\hat{\mathbf{U}}) + \gamma\operatorname{tr}(\mathbf{U}\mathbf{L}\mathbf{U}^\top) + \frac{\lambda}{n}\mathbf{1}^\top(\mathbf{V}+\mathbf{U})\mathbf{1} \\
& + \frac{\mu_1}{2}\left\|\mathbf{U}-\hat{\mathbf{U}}\right\|_F^2 + \frac{\mu_2}{2}\left\|\mathbf{V}-\hat{\mathbf{V}}\right\|_F^2 && (2.7a) \\
\text{subject to} \quad & \hat{\mathbf{U}} \geq 0, && (2.7b) \\
& \mathbf{1}^\top\hat{\mathbf{U}} = \mathbf{1}^\top, && (2.7c) \\
& \mathbf{v}_{i-1} + \mathbf{u}_{i-1} = \mathbf{v}_i + \mathbf{u}_i, \quad {\scriptstyle i=2,\ldots,n}, && (2.7d) \\
& \hat{\mathbf{V}} \geq 0, && (2.7e) \\
& \mathbf{U} = \hat{\mathbf{U}}, && (2.7f) \\
& \mathbf{V} = \hat{\mathbf{V}}, && (2.7g)
\end{aligned}$$

where $\mu_1 > 0$ and $\mu_2 > 0$ are the augmented Lagrangian parameters. As it is suggested in [37], we can set $\mu_1 = \mu_2 = \mu$. Note that (2.7) is equivalent to (2.6), because the additional terms in (2.7a) vanish for any feasible solution. To solve (2.7), augmented Lagrangian function is formed as:

$$\begin{aligned}
\mathcal{L}_\mu(\mathbf{U},\mathbf{V},\hat{\mathbf{U}},\hat{\mathbf{V}},\boldsymbol{\Lambda}_1,\boldsymbol{\Lambda}_2) &= \operatorname{tr}(\mathbf{P}\mathbf{R}^\top\hat{\mathbf{U}}) + \gamma\operatorname{tr}(\mathbf{U}\mathbf{L}\mathbf{U}^\top) \\
&+ \frac{\lambda}{n}\mathbf{1}^\top(\mathbf{V}+\mathbf{U})\mathbf{1} + \frac{\mu}{2}\left\|\mathbf{U}-\hat{\mathbf{U}}+\frac{\boldsymbol{\Lambda}_1}{\mu}\right\|_F^2 + \frac{\mu}{2}\left\|\mathbf{V}-\hat{\mathbf{V}}+\frac{\boldsymbol{\Lambda}_2}{\mu}\right\|_F^2
\end{aligned} \tag{2.8}$$

where $\boldsymbol{\Lambda}_1 \in \mathbb{R}^{l\times n}$ and $\boldsymbol{\Lambda}_2 \in \mathbb{R}^{l\times n}$ are Lagrange multipliers associated with the equality constraints (2.7f) and (2.7g).

Given initial values for $\hat{\mathbf{U}}$, $\hat{\mathbf{V}}$, $\boldsymbol{\Lambda}_1$, and $\boldsymbol{\Lambda}_2$, the ADMM iterations to solve (2.7) are summarized as follow:

$$\begin{aligned}
(\mathbf{U}^{k+1},\mathbf{V}^{k+1}) := \underset{\mathbf{U},\mathbf{V}\in\mathbb{R}^{l\times n}}{\operatorname{argmin}} \quad & \mathcal{L}_\mu(\mathbf{U},\mathbf{V},\hat{\mathbf{U}}^k,\hat{\mathbf{V}}^k,\boldsymbol{\Lambda}_1^k,\boldsymbol{\Lambda}_2^k) \\
\text{subject to} \quad & \mathbf{v}_{i-1}+\mathbf{u}_{i-1} = \mathbf{v}_i + \mathbf{u}_i, \quad {\scriptstyle i=2,\ldots,n}.
\end{aligned} \tag{2.9}$$

$$(\hat{\mathbf{U}}^{k+1}, \hat{\mathbf{V}}^{k+1}) := \underset{\hat{\mathbf{U}}, \hat{\mathbf{V}} \in \mathbb{R}^{l \times n}}{\operatorname{argmin}} \quad \mathcal{L}_\mu(\mathbf{U}^{k+1}, \mathbf{V}^{k+1}, \hat{\mathbf{U}}, \hat{\mathbf{V}}, \mathbf{\Lambda}_1^k, \mathbf{\Lambda}_2^k)$$

$$\text{subject to } \hat{\mathbf{U}} \geq 0,\ \mathbf{1}^\top \hat{\mathbf{U}} = \mathbf{1}^\top, \qquad (2.10)$$

$$\hat{\mathbf{V}} \geq 0.$$

$$\begin{aligned} \mathbf{\Lambda}_1^{k+1} &= \mathbf{\Lambda}_1^k + \mu(\mathbf{U}^{k+1} - \hat{\mathbf{U}}^{k+1}) \\ \mathbf{\Lambda}_2^{k+1} &= \mathbf{\Lambda}_2^k + \mu(\mathbf{V}^{k+1} - \hat{\mathbf{V}}^{k+1}) \end{aligned} \qquad (2.11)$$

To solve (2.9), let form $\mathbf{y}_j \in \mathbb{R}^{2n}$ by concatenating the j^{th} rows of $\mathbf{U}$ and $\mathbf{V}$. Then, (2.9) can be divided into l equality constrained quadratic programs as follows:

$$\underset{\mathbf{y}_j \in \mathbb{R}^{2n}}{\text{minimize}} \ \frac{1}{2} {\mathbf{y}_j}^\top \mathbf{B} \mathbf{y}_j + {\mathbf{y}_j}^\top \mathbf{b}_j \qquad (2.12a)$$

$$\text{subject to } \boldsymbol{A} \mathbf{y}_j = \mathbf{c}, \qquad (2.12b)$$

where $\mathbf{B} \in \mathbb{R}^{2n \times 2n}$ is a block diagonal positive semi-definite matrix, $\boldsymbol{A} \in \mathbb{R}^{n \times 2n}$ is a sparse matrix corresponding to the constraint (2.7d), and $\mathbf{c} \in \mathbb{R}^n$ is a vector of all zeros.

Problem (2.10) can be split into two separate sub-problems with closed-form solutions as follows:

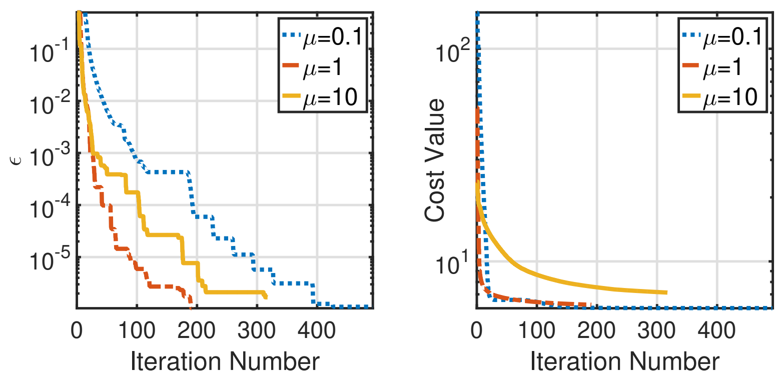
$$\underset{\hat{\mathbf{U}} \in \mathbb{R}^{l \times n}}{\text{minimize}} \qquad \left\| \hat{\mathbf{U}} - (\mathbf{U} + \frac{\mathbf{\Lambda}_1 + \mathbf{R}\mathbf{P}^\top}{\mu}) \right\|_F^2 \qquad (2.13a)$$

$$\text{subject to} \qquad \hat{\mathbf{U}} \geq 0,\ \ \mathbf{1}^\top \hat{\mathbf{U}} = \mathbf{1}^\top, \qquad (2.13b)$$

$$\underset{\hat{\mathbf{V}} \in \mathbb{R}^{l \times n}}{\text{minimize}} \qquad \left\| \hat{\mathbf{V}} - (\mathbf{V} + \frac{\mathbf{\Lambda}_2}{\mu}) \right\|_F^2 \qquad (2.14a)$$

$$\text{subject to} \qquad \hat{\mathbf{V}} \geq 0, \qquad (2.14b)$$

where each one consists of n computationally cheap parallel programs. Sub-problem (2.13) can be divided into n parallel programs over the columns of $\hat{\mathbf{U}}$ where each one is a Euclidean norm projection onto the probability simplex constraints. These programs enjoy



(a) (b)

Figure 2.3: Our convergence behavior for different choices of μ. (a) Combined residual. (b) Cost function.

closed-form solutions as presented in [125]. Sub-problem (2.14) consists of n small parallel programs over the columns of $\hat{\mathbf{V}}$, where each program is a minimization of the Euclidean norm projection onto the nonnegative orthant and admits closed-form solution.

Problem (2.11) can be split into two sub-problems over $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$ where each sub-problem consists of n parallel updates over the columns of corresponding matrix.

Our numerical algorithm consists of two sub-problems with closed-form solutions, which makes the iterations computationally cheap. To evaluate our convergence behavior, we adopt combined residual presented in [126] as:

$$\begin{aligned}\epsilon^{k+1} = &\frac{1}{\mu}\big\|\mathbf{\Lambda}_1^{k+1} - \mathbf{\Lambda}_1^{k}\big\|_F^2 + \mu\big\|\mathbf{U}^{k+1} - \mathbf{U}^{k}\big\|_F^2 \\ &+ \frac{1}{\mu}\big\|\mathbf{\Lambda}_2^{k+1} - \mathbf{\Lambda}_2^{k}\big\|_F^2 + \mu\big\|\mathbf{V}^{k+1} - \mathbf{V}^{k}\big\|_F^2\end{aligned} \tag{2.15}$$

Figure 2.3 demonstrate the convergence behavior of our algorithm in terms of combined residual and cost function. We solve (2.3) for three choices of μ to show the sensitivity of our numerical algorithm with respect to μ. Figure 2.3 indicates our algorithm converges in a reasonable number of iterations for a wide range of μ.

2.4 Experiments

We perform multiple experiments on benchmark image segmentation datasets to evaluate the performance of our method (termed IS4). The first part of this section gives information about the benchmarking datasets, evaluation measures, and parameter settings. The second part compares our results with state-of-the-art methods to demonstrate the effectiveness of IS4.

2.4.1 Settings

Datasets: We adopt three commonly used datasets in image segmentation: (1) BSD300 [106] containing 300 images (200 training and 100 validation) of size 321×481, where each one has in average 5 ground-truths manually drawn by human; (2) BSD500 [100] is an extension of BSD300 with 200 new testing images; (3) MSRC [107] containing 591 images of size 320×213, where each one has a single ground-truth. It should be mentioned that we use the cleaned version of MSRC [127] in our experiments.

Measures: We adopt three widely accepted measures in image segmentation: (1) segmentation Covering (Cov) [127], which measures the overlapping between two segmentations; (2) probability Rand Index (PRI) [128], which measures the probability that a pair of pixels is consistently labeled in two segmentations; (3) variation of Information (VoI) [129], which measures the distance between two segmentations as the average of their conditional entropy.

Parameters: Given an over-segmentation, IS4 computes the superpixels features $\{\mathbf{x}_i\}_{i=1}^n$ by averaging the local spectral histogram (LSH) [104] features of pixels within each superpixel. We use the algorithm and parameters presented in [101] to extract features and build a dictionary of size l. Parameter l should be chosen sufficiently large (larger than the number of coherent regions) to ensure each superpixel feature is well expressible as a nonnegative linear combination of the dictionary words. We set $l = 20$ which is normally much larger than the number of coherent regions in BSD and MSRC images. Our proposed model in (2.3) has two parameters γ and λ, where γ controls the effect of spatial relationship among superpixels and λ controls the number of selected words. As γ increases, the neighboring regions are more likely to be merged together and as λ increases, the number of selected words reduces. We optimize γ on the training set of BSD by applying grid search and use the optimized γ in our experiments on BSD300, BSD500, and MSRC datasets. Parameter λ is set to $\alpha\lambda_{max}$, where $\alpha \in [0, 1]$ and λ_{max} is a constant computed based on $\mathbf{PR}^{\top}$, γ, and $\mathbf{L}$ using [122]. If α is greater than 1 (which means $\lambda > \lambda_{max}$), only a single word is selected to represent the whole features. We follow [100, 3, 111, 2] to present our results as a family of segmentations which share the same parameter settings except for α that varies from 0 to 1. The evaluation measures are also reported at Optimal Dataset Scale (ODS) and Optimal Image Scale (OIS).

2.4.2 Results

Segmentation quality: We run IS4 on the benchmark datasets and report the results in tables 2.1, 2.2, and 2.3, to make a comparison with recent methods such as, Normalized cut (Ncut) [130], Multi-scale Normalized cut (MNcut)[131], gPb-Ultametric contour map (gPb) [100], Image Segmentation by Cascade Region Agglomeration (ISCRA) [2], Reverse Image Segmentation with High/Low-level pairwise potentials (RIS-HL) [132], Multiscale Combinatorial Grouping (MCG) [3], Contour-guided Color Palletes (CCP-2) [1], Piecewise Flat Embedding (PFE) [114], Discrete-Continuous Gradient Orientation Estimation for Segmentation (DC-Seg) [110], Graph Without Cut (GWC) [111], and Aligned hierarchical segmentation (MCG-Aligned) [115]. All scores are collected from [100, 3, 115, 110, 111] except the MCG on MSRC and CCP-2 on BSD500 which are obtained by running the implementations provided by the respective authors.

	Cov (↑)		PRI (↑)		VoI (↓)	
Methods	ODS	OIS	ODS	OIS	ODS	OIS
MNcut[131]	0.44	0.53	0.75	0.79	2.18	1.84
gPb-UCM [100]	0.59	0.65	0.81	0.85	1.65	1.47
ISCRA [2]	0.60	0.67	0.81	**0.86**	1.61	1.40
RIS+HL[132]	0.59	0.65	**0.82**	**0.86**	1.71	1.53
MCG [3]	**0.61**	0.67	0.81	**0.86**	1.55	**1.37**
GWC [111]	**0.61**	**0.68**	**0.82**	**0.86**	1.60	1.42
IS4(MCG)	**0.61**	0.65	0.81	0.83	**1.54**	1.40

Table 2.1: Quantitative comparisons on BSD300 val set.

Parameter sensitivity: To evaluate the role played by an initial over-segmentation, we run IS4 in combination with three segmentation methods CCP-2, ISRA, and MCG. In CCP-2, we use the same parameter settings as suggested by the respective author. In MCG and ISRA we respectively adopted the segmentations at scale 0.39 and 44 as the over-segmentations. In average, the over-segmentation layers provided by CCP-2, ISRA, and MCG have 120, 45, and 37 superpixels, respectively. Figure 2.4 and table 2.4 respectively show the qualitative and quantitative results of these combinations. Moreover, we run IS4 for different γ to assess our robustness with respect to the variations of γ. The results are reported in table 2.5 in terms of segmentation measures. As tables 2.4 and 2.5 indicate, IS4 not only achieves satisfactory results for a wide range of γ but also improves the quality of initial over-segmentations on most of the segmentation measures. It is worth pointing out that IS4 can be applied on the result of any segmentation method. The result may either be directly generated by a segmentation algorithm (e.g., CCP-2) or obtained from a specific level of a hierarchical segmentation (e.g., MCG).

Tables 2.1, 2.2, and 2.3 show that IS4 in combination with MCG generates a high quality segmentation. The scores indicate that IS4(MCG) outperforms all competitor methods on BSD300 (ODS: VoI) and MSCRC (ODS: Cov, PRI, VoI and OIS: Cov, PRI, VoI). Other scores obtained by IS4(MCG) are also on par or in close proximity of the best com-

	Cov (↑)		PRI (↑)		VoI (↓)	
Methods	ODS	OIS	ODS	OIS	ODS	OIS
Ncut [130]	0.45	0.53	0.78	0.80	2.23	1.89
gPb-UCM [100]	0.59	0.65	0.83	0.86	1.69	1.48
DC-Seg [110]	0.59	0.64	0.82	0.85	1.68	1.54
ISCRA [2]	0.59	0.66	0.82	0.85	1.60	1.42
RIS+HL[132]	0.57	0.66	**0.84**	0.86	1.73	1.55
MCG [3]	0.61	0.66	0.83	0.86	1.57	1.39
PFE+MCG [114]	0.62	**0.68**	**0.84**	**0.87**	1.56	1.36
MCG-Aligned [115]	**0.63**	**0.68**	0.83	0.86	**1.53**	1.38
GWC [111]	0.61	0.66	0.83	**0.87**	1.62	1.41
IS4(MCG)	**0.63**	0.66	0.83	0.85	1.55	**1.35**

Table 2.2: Quantitative comparisons on BSD500 test set.

	Cov (↑)		PRI (↑)		VoI (↓)	
Methods	ODS	OIS	ODS	OIS	ODS	OIS
gPb-UCM [100]	0.65	0.75	0.78	0.85	1.28	0.99
ISCRA [2]	0.67	0.75	0.77	0.85	1.18	1.02
GWC [111]	0.68	0.76	0.78	0.85	1.24	0.98
MCG [3]	0.66	0.72	0.78	0.83	1.23	1.14
IS4(MCG)	**0.69**	**0.77**	**0.80**	**0.86**	**1.15**	**0.91**

Table 2.3: Quantitative comparisons on MSRC dataset.

	Cov (↑)		PRI (↑)		VoI (↓)	
Methods	ODS	OIS	ODS	OIS	ODS	OIS
ISCRA [2]	0.59	0.66	0.82	0.85	1.60	1.42
IS4 (ISCRA) [108]	0.61	0.65	0.82	0.85	1.58	1.38
CCP-2 [1]	0.45	–	0.79	–	3.1	–
IS4 (CCP-2)	0.58	0.64	0.81	0.85	1.78	1.52
MCG [3]	0.61	0.66	0.83	0.86	1.57	1.39
IS4(MCG)	0.63	0.66	0.83	0.85	1.55	1.35

Table 2.4: Sensitivity of our method with respect to the different initial over-segmentations on BSD500 test set.

	Cov (↑)		PRI (↑)		VoI (↓)	
γ	ODS	OIS	ODS	OIS	ODS	OIS
10^{-2}	0.62	0.65	0.83	0.85	1.56	1.39
10^{-1}	0.62	0.65	0.83	0.85	1.55	1.39
10^{0}	0.62	0.66	0.83	0.85	1.54	1.38
10^{1}	0.63	0.66	0.83	0.85	1.54	1.37
10^{2}	0.62	0.64	0.81	0.83	1.53	1.41

Table 2.5: Sensitivity of our method with respect to the parameter variations on BSD500 test set.

petitors except for BSD300 (OIS: Cov, PRI) and BSD500 (OIS: PRI). In comparison with MCG, IS4(MCG) achieves better results on BSD300 (ODS: 0.01 on VoI), BSD500 (ODS: 0.02 on Cov, 0.02 on VoI and OIS: 0.04 on VoI), and MSRC (ODS: 0.03 on Cov, 0.02 on PRI, 0.08 on VoI and OIS: 0.05 on Cov, 0.03 on PRI, 0.23 on VoI). Our method is also on par with MCG on BSD300 (ODS: Cov, PRI) and BSD500 (ODS: PRI). Moreover, the tables indicate that IS4(MCG) achieves lower score than MCG in BSD300 (OIS: Cov, PRI, VoI) and BSD500 (OIS: PRI). It may seem reasonable to state that IS4(MCG) should consistently improve the MCG measures. However, this is not a fairly accurate statement. The reason is that IS4 does not directly apply on the MCG hierarchical segmentation to improve or degrade the MCG results. It just adopts an over-segmentation by simply thresholding the MCG hierarchical segmentation and generates a family of segmentations. These segmentations differ from the ones which may be obtained at different thresholds of MCG.

MCG usually provides a high-quality hierarchical segmentation, but its performance is sometimes unsatisfactory, especially in textured images. Our method in combination with MCG improves the segmentation quality of these images (shown in Figure 2.5) by adopting superpixels features as an informative representation of small regions. Despite the advantages of IS4, it may fail to correctly segment the pixels belonging to an elongated coherent region. The reason is that the local neighborhood around these pixels contains

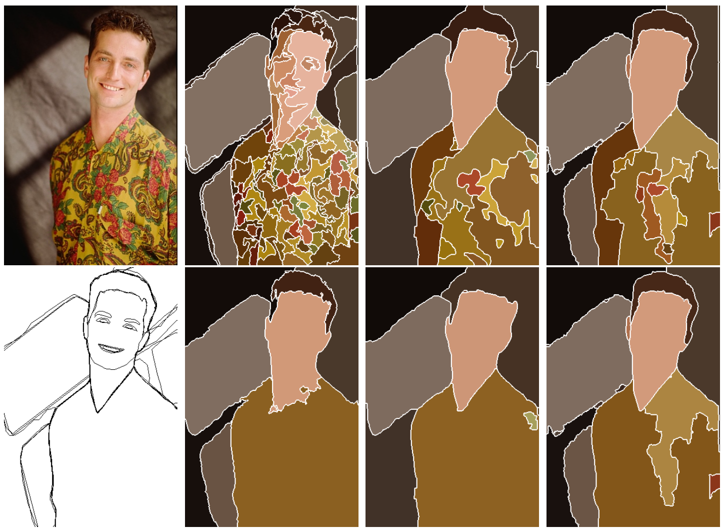


Figure 2.4: IS4 in combination with different over-segmentation. From left to right, top row: image, CCP-2 [1], ISCRA [2], and MCG [3]; bottom row: groundtruth, IS4(CCP-2), IS4(ISCRA), and IS4(MCG); The top row shows the initial over-segmentation provided by these methods and the bottom column shows our final segmentation with these initial over-segmentations.

visual information of neighboring coherent regions. Hence, their LSH features are inaccurate, which may cause a wrong assignment to the neighboring coherent regions.

Algorithm complexity: We decomposed model (2.3) into three sub-problems (2.9), (2.10), and (2.11) which are considered as the steps of the alternating direction method of multipliers (ADMM). To investigate the complexity of solving these sub-problems, we discuss about each one separately. Subproblem (2.9) is cast as l parallel equality constrained quadratic programs of form (2.12) where each can be efficiently solved by solving a system of linear equations [13] in $O(n^{2.8})$ [133]. Since $\boldsymbol{A}$ and B are the same in all programs, the left-hand side of such systems are similar. Hence, one program just need to be solved and then back-solves can be performed for the right-hand sides of other programs. Therefore, the complexity of solving (2.9) is $O(n^{2.8}) + O((l-1)n^2)$. It is worth mentioning the complexity would much lower than this amount, because $\boldsymbol{A}$ and B are highly sparse and structured. Subproblem (2.10) is split into two parallel problems (2.13), (2.14). Problem

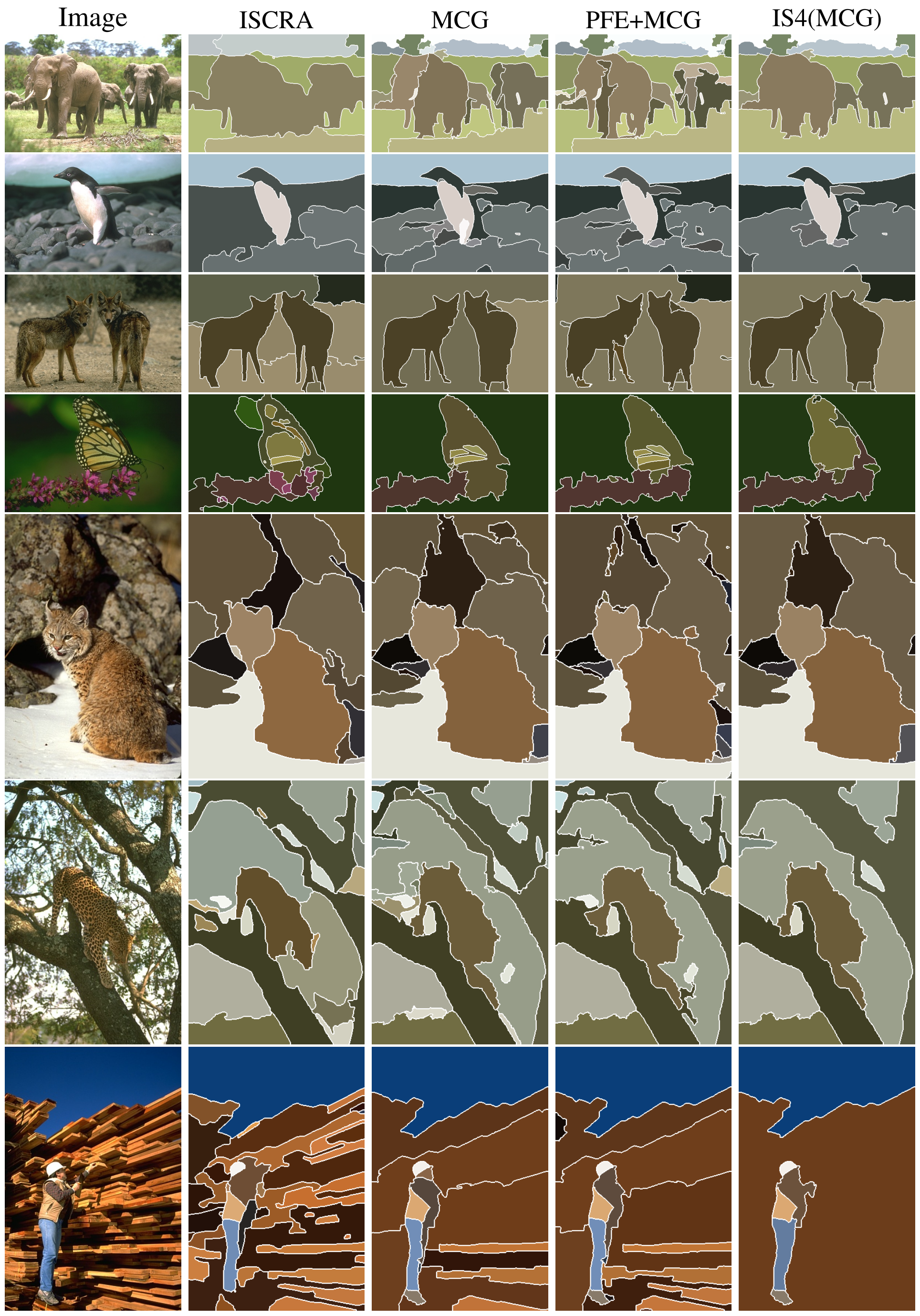


Figure 2.5: Qualitative comparison of segmentations. All segmentation results are shown at Optimal Dataset Scale (ODS).

	SeDuMi	SDPT3	MOSEK	IS4(MCG)
Run-Time (sec.)	1.4×10^{2}	9.8×10^{0}	2.7×10^{0}	6.1×10^{-1}

(a)

	SeDuMi	SDPT3	MOSEK
Relative Error	1.7×10^{-2}	1.1×10^{-2}	9.0×10^{-3}

(b)

Table 2.6: Performance comparison with convex solvers, SeDuMi [6], SDPT3 [7], and MOSEK [8].

(2.13) consists of n parallel programs where each is solvable in $O(l\log(l))$ [125]. Problem (2.14) consists of n parallel programs where each is solved in $O(l)$ [37]. Therefore, (2.10) can be efficiently solved in $O(nl\log(l))$. Subproblem (2.11) consists of n parallel updates over the columns of $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$ which can be performed in $O(nl)$. In total, the complexity of our numerical is $O(n^{2.8}) + O((l-1)n^2)$ in the first iteration and $O(ln^2)$ in subsequent iterations. Since all steps of our ADMM are highly parallelizable, the complexity can be significantly reduced using parallel computation. In the case of having p parallel processing resources (assumption $n > l$), the complexity of (2.9), (2.10), and (2.11) are $O(n^{2.8})+O(\lceil\frac{l-1}{p}\rceil n^2)$, $O(\lceil\frac{n}{p}\rceil l\log(l))$, and $O(\lceil\frac{n}{p}\rceil l)$, respectively.

To compare our numerical algorithm with other solvers, we solve (2.3) for 20 randomly chosen images in BSD500 using three standard convex solvers with CVX [134, 135]. The results are reported in tables 2.6a and 2.6b in terms of the average running times and relative errors, respectively. In this case, our relative error is computed as $\frac{\|\mathbf{U}^*_{solver}-\mathbf{U}^*_{our}\|_F}{\|\mathbf{U}^*_{solver}\|_F}$. It should be noted that our average running time is computed without considering any parallelization.

Table 2.6 indicates that our numerical algorithm is not only significantly faster than SeDuMi [6], SDPT3 [7], and MOSEK [8] in solving (2.3), but also offers an optimal solution extremely close to their solutions.

The running time of IS4 heavily depends on the initial over-segmentation. IS4(MCG) takes 6.01 seconds per image in average, where 2.9 seconds takes for feature extraction, 2.5 seconds takes for learning the dictionary, and the remaining is taken by our numerical algorithm. All the experiments are performed on an Intel Core i5 quad-core 3.20GHz CPU and 16 GB RAM.

2.5 Conclusions

This work presented a novel segmentation model based on the concept of sparse subset selection. The model automatically estimates the optimal number of coherent regions and pixel assignments to form final segments. Moreover, we presented a parallel numerical algorithm based on the alternating direction method of multipliers (ADMM) to solve our model. The main advantages of this work are as follow: (1) does not require time for training over different datasets and works well in combination with various segmentation methods; (2) consists of three steps: extracting features, learning dictionary, and solving model where each one can be implemented in parallel; (3) contains ADMM steps with closed-form solutions which make the iterations computationally very cheap; (4) does not restricted to the segmentation problem and can be easily extended to other applications, such as video summarization, dimensionality reduction, etc.

CHAPTER 3

Sequential Convex Relaxations for Optimization Under Orthogonality Constraints

This chapter is concerned with the class of non-convex optimization problems with orthogonality constraints. We develop computationally efficient relaxations that transform non-convex orthogonality constrained problems into polynomial-time solvable surrogates. A novel penalization technique is used to enforce feasibility and derive certain conditions under which the constraints of the original non-convex problem are guaranteed to be satisfied. Moreover, we extend our approach to a feasibility-preserving sequential scheme that solves penalized relaxation to obtain near-globally optimal points. Experimental results on synthetic and real datasets demonstrate the effectiveness of the proposed approach on two practical applications in machine learning.

3.1 Introduction

Consider the following optimization problem

$$\underset{\boldsymbol{P}\in\mathbb{R}^{n\times m}}{\text{minimize}} \quad \bar{f}_0(\boldsymbol{P}) + g_0(\boldsymbol{P}) \tag{3.1a}$$

$$\text{subject to} \quad \bar{f}_k(\boldsymbol{P}) \leq 0, \qquad k \in \{1,\dots,p\}, \tag{3.1b}$$

$$\boldsymbol{P}^\top\boldsymbol{P} = \boldsymbol{I}_m, \tag{3.1c}$$

where $g_0\colon \mathbb{R}^{n\times m} \to \mathbb{R}$ is a convex piecewise linear function and $\bar{f}_k : \mathbb{R}^{n\times m} \to \mathbb{R}$ is an arbitrary quadratic function of the form $\bar{f}_k(\boldsymbol{P}) \triangleq \langle \boldsymbol{M}_k, \boldsymbol{P}\boldsymbol{P}^\top\rangle + \langle \boldsymbol{N}_k, \boldsymbol{P}\rangle + q_k$, for every $k \in \{0,1,\dots,p\}$, and $\{\boldsymbol{M}_k \in \mathbb{S}_n\}_{k=0}^p$, $\{\boldsymbol{N}_k \in \mathbb{R}^{n\times m}\}_{k=0}^p$ and $\{q_k \in \mathbb{R}\}_{k=0}^p$ are given. With no loss of generality, we assume that $q_0 = 0$ and write g_0 in the form of $g_0(\boldsymbol{P}) = \|\alpha(\boldsymbol{P}) + \boldsymbol{b}\|_1$, where $\boldsymbol{b} \in \mathbb{R}^w$ is a given vector, $\alpha : \mathbb{R}^{n\times m} \to \mathbb{R}^w$ is a linear matrix function defined as $\alpha(\boldsymbol{Y}) \triangleq \sum_{i=1}^{w} \langle \boldsymbol{A}_i, \boldsymbol{Y} \rangle \boldsymbol{e}_i$, the matrices $\{\boldsymbol{A}_i \in \mathbb{R}^{n \times m}\}_{i=1}^{w}$ are given, and $\{\boldsymbol{e}_i \in \mathbb{R}^{w}\}_{i=1}^{w}$ represent the standard basis for $\mathbb{R}^{w}$. The formulation (3.1a) – (3.1c) encompasses a broad class of computationally-hard optimization problems with a variety of practical applications in discriminative dimensionality reduction [4], graph matching [56], feature selection [57, 58], compressed modes [66, 136], among other areas of machine learning.

The majority of methods in the literature are focused on a special case of (3.1a) – (3.1c) that involves the minimization of a convex and smooth objective function over non-convex sets of the form $\mathcal{S}_{n,m} \triangleq \{\boldsymbol{P} \in \mathbb{R}^{n \times m} \mid \boldsymbol{P}^{\top}\boldsymbol{P} = \boldsymbol{I}_m\}$, known as the Stiefel manifolds. There are various iterative local search algorithms which preserve the structure of Stiefel manifolds via geodesics steps [61, 62] or retractions [64, 65]. Although these algorithms exhibit satisfactory performance in dealing with orthogonality constraints, they mostly restrict the objective function to the class of smooth functions and are not compatible with additional constraints [137]. To overcome these limitations, general algorithms are proposed that work with either smooth or non-smooth objective functions [4, 136]. The paper [4] uses a family of semidefinite programming (SDP) problems to generate a converging sequence of points Stiefel manifolds. The paper [136] introduces an inner-outer iteration scheme for solving ℓ_1-regularized optimization problems with orthogonality constraints based on the augmented Lagrangian method from [138] and the proximal alternating minimization technique from [139]. Moreover, a series of splitting techniques are proposed in [66, 67, 69] that can efficiently handle non-smooth objective functions. They partition the problem into multiple sub-problems with analytical solutions and employ Bregman iterations [36] or its variants [37] to obtain optimal solutions for orthogonality-constrained problems. In the more recent paper [140], an extended proximal alternating linearized minimization method is introduced to minimize convex functions subject to linear constraints and generalized orthogonality constraints.

Differentiated from the existing literature, we propose a computational method for solving problems of form (3.1a) – (3.1c) with theoretical analysis that provides guarantee to obtain near-optimal solutions. The main contributions of this work are outlined in the following subsection. Differentiated from the existing literature, we propose a computational approach with theoretical analysis for solving problems of the form (3.1a) – (3.1c), that guarantees the recovery of feasible points. The proposed approach generalizes the existing literature by including additional quadratic inequality constraints. The core of our approach is based on a novel and computationally efficient convex relaxation which transforms the non-convex problem (3.1a) – (3.1c) into a convex quadratically-constrained quadratic program (QCQP). To ensure that the solution of the relaxed problem is feasible for (3.1a) – (3.1c), we incorporate a penalty term into the objective function and derive certain conditions that guarantee the recovery of feasible points. Moreover, under certain conditions, we prove that by starting from any arbitrary initial point on a Stiefel manifold (not necessarily feasible), a sequence of penalized relaxations can be solved to find a feasible and near-optimal point. Unlike the existing algorithms, if mild assumptions are satisfied, the proposed sequential scheme is feasibility-preserving and improves the objective monotonically at every step. To corroborate the effectiveness of our method, we perform experiments on two practical applications with both synthetic and real datasets. The experimental results demonstrate that the proposed approach exhibits comparable results for both applications.

3.1.1 Notation

Throughout this work, the scalars, vectors, and matrices are shown by italic, bold lower-case and bold upper-case letters, respectively. The symbols $\mathbb{R}^n$, $\mathbb{R}^{n\times m}$, $\mathbb{S}_n$, and $\mathbb{S}_n^+$

denote the set of real n-dimensional vectors, real $n \times m$ matrices, real symmetric $n \times n$ matrices, and real positive semidefinite matrices, respectively. The symbols $\mathrm{tr}\{.\}$ and $(.)^\top$ are indicative of the trace and transpose operators, respectively. Given a vector $\boldsymbol{a}$ and a matrix $\boldsymbol{A}$, the symbols a_i and A_{ij}, respectively, refer to the i^{th} element of $\boldsymbol{a}$ and the $(i,j)^{th}$ element of $\boldsymbol{A}$. The notation $\boldsymbol{A} \succeq 0$ states that $\boldsymbol{A}$ is symmetric positive semidefinite. Given matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ of the same size, $\langle \boldsymbol{A}, \boldsymbol{B} \rangle \triangleq \mathrm{tr}\{\boldsymbol{A}^\top \boldsymbol{B}\}$ and $\mathrm{A} \circ \mathrm{B}$, respectively, denote the Frobenius inner-product and the Hadamard product of $\boldsymbol{A}$ and $\boldsymbol{B}$. The operator $\mathrm{diag}(.)$ gets a vector and forms a diagonal matrix with its input on the diagonal elements. The notation $\|.\|_p$ refers to either matrix norm or vector norm depending on the context, $\|.\|_{\mathrm{F}}$ shows the Frobenius norm, and $|.|$ indicates the absolute value or the cardinality of a set depending on the context. The symbol $\boldsymbol{I}_m$ denotes the identity matrix of size m and the letter $\mathcal{K}$ is used as a shorthand for the set $\{1, \ldots, p\}$. The symbol $\mathcal{S}_{n,m}$ as the set of real $n \times m$ matrices with orthonormal columns, i.e., $\mathcal{S}_{n,m} \triangleq \{\boldsymbol{P} \in \mathbb{R}^{n \times m} \mid \boldsymbol{P}^\top \boldsymbol{P} = \boldsymbol{I}_m\}$. The projection operator $\mathrm{proj}_{\mathcal{S}_{n,m}} : \mathbb{R}^{n \times m} \rightarrow \mathcal{S}_{n,m}$ is defined as $\mathrm{proj}_{\mathcal{S}_{n,m}} \boldsymbol{H} = \arg\min\{\|\boldsymbol{P} - \boldsymbol{H}\|_{\mathrm{F}} \mid \boldsymbol{P} \in \mathcal{S}_{n,m}\}$.

3.2 Problem Formulation

Optimization problems of the form (3.1a) – (3.1c) can be computationally challenging due to the non-convexities of the objective function and constraints. In order to derive convex relaxations, we first lift the problem into a higher dimensional space by introducing an auxiliary variable $\boldsymbol{X} \in \mathbb{S}_n$, accounting for the quadratic term $\boldsymbol{P}\boldsymbol{P}^\top$. For every $k \in \{0\} \cup \mathcal{K}$, define $f_k : \mathbb{R}^{n \times m} \times \mathbb{S}_n \rightarrow \mathbb{R}$ as:

$$f_k(\boldsymbol{P}, \boldsymbol{X}) \triangleq \langle \boldsymbol{M}_k, \boldsymbol{X} \rangle + \langle \boldsymbol{N}_k, \boldsymbol{P} \rangle + q_k. \tag{3.2}$$

Using the auxiliary variable $\boldsymbol{X}$, the optimization problem (3.1a) – (3.1c) can be equivalently reformulated as

$$
\begin{aligned}
\underset{\substack{\boldsymbol{P}\in\mathbb{R}^{n\times m}\\ \boldsymbol{X}\in\mathbb{S}_n}}{\text{minimize}} \quad & f_0(\boldsymbol{P},\boldsymbol{X}) + g_0(\boldsymbol{P}) && && (3.3\text{a})\\
\text{subject to} \quad & f_k(\boldsymbol{P},\boldsymbol{X}) \leq 0 && k \in \mathcal{K}, && (3.3\text{b})\\
& \boldsymbol{P}^{\top}\boldsymbol{P} = \boldsymbol{I}_m, && && (3.3\text{c})\\
& \boldsymbol{P}\ \boldsymbol{P}^{\top} = \boldsymbol{X}\ , && && (3.3\text{d})
\end{aligned}
$$

with a convex objective function and convex linear inequality constraints (3.3b). The above formulation is still not convex due to the presence of the constraints (3.3c) and (3.3d) that capture all non-convexities of the problem.

3.2.1 Convex Relaxation

In order to convexify the lifted problem (3.3a) – (3.3d), we relax the constraints (3.3c) and (3.3d) to

$$
\boldsymbol{I}_m - \boldsymbol{P}^{\top}\boldsymbol{P} \in \mathcal{C} \quad \wedge \quad \boldsymbol{X} - \boldsymbol{P}\boldsymbol{P}^{\top} \in \mathcal{D} \quad \wedge \quad \operatorname{tr}\{\boldsymbol{X}\} = m, \tag{3.4}
$$

where $\mathcal{C} \subseteq \mathbb{S}_m$ and $\mathcal{D} \subseteq \mathbb{S}_n$ are convex cones to be defined. In this work, we consider the common-practice semidefinite programming (SDP) relaxation and introduce a novel convex relaxation that transforms the problem (3.3a) – (3.3d) into a convex quadratically-constrained quadratic program (QCQP).

3.2.1.1 Semidefinite Programming Relaxation

This relaxation provides a powerful method for tackling non-convex polynomial optimization problems [13]. The SDP relaxation of the problem (3.3a) – (3.3d) can be derived by having $\mathcal{C} = \mathbb{S}_m^{+}$ and $\mathcal{D} = \mathbb{S}_n^{+}$. Despite the effectiveness of this relaxation in providing high-quality solutions, its applicability is limited to the problems of moderate size due to the computational cost of imposing high-dimensional conic constraints.

3.2.1.2 Convex Quadratic Relaxation

We propose a computationally efficient convex relaxation as an alternative to the SDP relaxation. In order to formulate the proposed relaxation for the problem (3.3a) – (3.3d), we need to have $\mathcal{C} = \mathcal{V}_m$ and $\mathcal{D} = \mathcal{V}_n$, where for every positive integer o, set $\mathcal{V}_o \subseteq \mathbb{S}_o$ is defined as follows

$$\mathcal{V}_o \triangleq \left\{ \boldsymbol{H} \in \mathbb{S}_o \,\middle|\, H_{ii} + H_{jj} \geq 2|H_{ij}|,\ \forall i,j \in \{1,\ldots,o\} \right\}.$$

Remark 3.1. *It can be easily observed that if* $(\mathcal{C},\mathcal{D}) = (\mathcal{V}_m,\mathcal{V}_n)$*, the constraints* (3.3c) *and* (3.3d) *boil down to convex quadratic inequalities. Hence, the proposed relaxation reduces (3.3a) – (3.3d) to a convex QCQP.*

Notice that either of the aforementioned relaxations may fail to produce a feasible point for (3.1a) – (3.1c), because in general, an optimal solution to a convex relaxation does not necessarily satisfy the constraints (3.3c) and (3.3d). In what follows, we propose a penalization technique that guarantees the recovery of feasible points for (3.1a) – (3.1c) under certain conditions.

3.2.2 Penalization

In this section, we show that by including a penalty term in the objective, one can obtain feasible points for the non-convex problem (3.3a) – (3.3d). Given an arbitrary initial point $\check{\boldsymbol{P}} \in \mathcal{S}_{n,m}$, that is not necessarily feasible, we transform the problem (3.3a) – (3.3d) into the following convex relaxation with revised objective function:

$$
\begin{aligned}
&\underset{\substack{\boldsymbol{P}\in\mathbb{R}^{n\times m}\\ \boldsymbol{X}\in\mathbb{S}_n}}{\text{minimize}} && f_0(\boldsymbol{P},\boldsymbol{X}) + g_0(\boldsymbol{P}) - \mu\langle \boldsymbol{P}, \check{\boldsymbol{P}}\rangle && && (3.5\text{a})\\
&\text{subject to} && f_k(\boldsymbol{P},\boldsymbol{X}) \leq 0 && k \in \mathcal{K}, && (3.5\text{b})\\
& && \boldsymbol{I}_m - \boldsymbol{P}^\top \boldsymbol{P} \in \mathcal{C}, && && (3.5\text{c})\\
& && \boldsymbol{X} - \boldsymbol{P}\,\boldsymbol{P}^\top \in \mathcal{D}, && && (3.5\text{d})\\
& && \operatorname{tr}\{\boldsymbol{X}\} = m, && && (3.5\text{e})
\end{aligned}
$$

where $(\mathcal{C},\mathcal{D}) \in \{(\mathbb{S}_m^+,\mathbb{S}_n^+),(\mathcal{V}_m,\mathcal{V}_n)\}$, and the fixed parameter $\mu > 0$ sets a trade-off between the original objective function and the linear penalty term $\langle \boldsymbol{P}, \check{\boldsymbol{P}}\rangle$.

Remark 3.2. *If an optimal solution* $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{X}})$ *of the problem (3.5a) – (3.5e) satisfies the constraints* (3.3c) *and* (3.3d)*, then* $\overset{*}{\boldsymbol{P}}$ *is feasible for (3.1a) – (3.1c).*

In the remainder of this section, certain conditions are introduced to guarantee that the penalized relaxation (3.5a) – (3.5e) produces feasible points for the non-convex problem (3.3a) – (3.3d).

Definition 3.1. *Define feasibility distance* $d_{\mathcal{F}} : \mathbb{R}^{n\times m} \rightarrow \mathbb{R}$ *as*

$$
d_{\mathcal{F}}(\boldsymbol{P}) \triangleq \inf\{\|\boldsymbol{C} - \boldsymbol{P}\|_{\mathrm{F}} \mid \boldsymbol{C} \in \mathcal{F}\}, \tag{3.6}
$$

where $\mathcal{F}$ *denotes the feasible set of the problem (3.1a) – (3.1c).*

Definition 3.2. *Define the singularity function* $s : \mathcal{S}_{n,m} \rightarrow \mathbb{R}$ *as:*

$$
s(\boldsymbol{P}) \triangleq \sup_{\boldsymbol{D}\in\mathcal{Z}_{\boldsymbol{P}}} \Big\{ \min_{k\in\mathcal{K}} \big\{ -\langle 2\boldsymbol{M}_k\boldsymbol{P} + \boldsymbol{N}_k, \boldsymbol{D}\rangle \big\} \Big\}, \tag{3.7}
$$

where $\mathcal{Z}_{\boldsymbol{P}} \triangleq \{\boldsymbol{D} \in \mathbb{R}^{n\times m} \mid \boldsymbol{P}^\top\boldsymbol{D} = \boldsymbol{0} \,\wedge\, \|\boldsymbol{D}\|_{\mathrm{F}} \leq 1\}$*. A point* $\boldsymbol{P} \in \mathcal{S}_{n,m}$ *is said to satisfy the Mangasarian-Fromovitz constraint qualification (MFCQ) condition if it is feasible for the problem (3.1a) – (3.1c) and* $s(\boldsymbol{P}) > 0$.

Theorem 3.1. *Define the constants*

$$\beta \triangleq \max_{\boldsymbol{P}\in\mathcal{S}_{m,n}} \left\{ \left| g_0(\boldsymbol{P}) + \langle \boldsymbol{M}_0, \boldsymbol{P}\boldsymbol{P}^\top \rangle + \langle \boldsymbol{N}_0, \boldsymbol{P} \rangle \right| \right\}, \tag{3.8a}$$

$$\psi \triangleq 2\|\boldsymbol{M}_0\|_{\mathrm{F}} + \|\boldsymbol{N}_0\|_{\mathrm{F}} + \sum_{i=1}^{w} \|\boldsymbol{A}_i\|_{\mathrm{F}}, \tag{3.8b}$$

$$\kappa \triangleq 4 \max_{k\in\mathcal{K}} \{\|\boldsymbol{M}_k\|_{\mathrm{F}}\} + \max_{k\in\mathcal{K}} \{\|\boldsymbol{N}_k\|_{\mathrm{F}}\} \tag{3.8c}$$

and let $\check{\boldsymbol{P}} \in \mathcal{F}$ be a feasible point for the problem (3.1a) – (3.1c) that satisfies the MFCQ condition. If

$$\mu > \max\{\beta^{-1}\psi^2,\ \beta(26\kappa)^2 s(\check{\boldsymbol{P}})^{-2},\ 144\beta\}, \tag{3.9}$$

then the penalized relaxation (3.5a) – (3.5e) has a unique optimal solution $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{X}})$, that satisfies (3.3c) *and* (3.3d). *Moreover, $\overset{*}{\boldsymbol{P}}$ is feasible for (3.1a) – (3.1c) and $\bar{f}_0(\overset{*}{\boldsymbol{P}}) + g_0(\overset{*}{\boldsymbol{P}}) \leq \bar{f}_0(\check{\boldsymbol{P}}) + g_0(\check{\boldsymbol{P}})$.*

Proof. See Section 3.4 for the proof. □

Remark 3.3. *For every point $\boldsymbol{P} \in \mathcal{S}_{m,n}$, it is straightforward to calculate $s(\boldsymbol{P})$ by solving the following convex problem:*

$$\begin{aligned} &\underset{t\in\mathbb{R}, \boldsymbol{D}\in\mathcal{Z}_{\mathcal{P}}}{\text{maximize}} && t \\ &\text{subject to} && t \leq -\langle 2\boldsymbol{M}_k\boldsymbol{P} + \boldsymbol{N}_k, \boldsymbol{D} \rangle, \quad k \in \mathcal{K}. \end{aligned}$$

Notice that β can be simply lower- and upper-bounded by any arbitrary member of the set $\mathcal{S}_{m,n}$ and the constant Ψ, respectively. This certifies the existence of a bounded μ that satisfies (3.9). In practice, there is no need to compute $s(\boldsymbol{P})$ for fine-tuning parameter μ, since (3.9) offers a conservative sufficient condition and there mostly exist smaller μ that satisfies Theorem 3.1. In Section 3.3, we assess the sensitivity of our approach with respect to different choices of μ.

Theorem 3.1 is concerned with the case where the initial point $\check{\boldsymbol{P}}$ is feasible for the original problem (3.1a) – (3.1c). However, finding a feasible starting point can be difficult due to the presence of the non-convex quadratic inequality constraints (3.1b). The next theorem states that even if $\check{\boldsymbol{P}}$ is not feasible, the proposed penalized relaxation can still result in a feasible point for the non-convex problem (3.1a) – (3.1c).

Theorem 3.2. *Consider an initial* $\check{\boldsymbol{P}} \in \mathcal{S}_{n,m}$ *that satisfies*

$$d_{\mathcal{F}}(\check{\boldsymbol{P}}) < 1, \tag{3.11a}$$

$$s(\check{\boldsymbol{P}}) > \kappa \, d_{\mathcal{F}}(\check{\boldsymbol{P}}) \left[1 + (1 - d_{\mathcal{F}}(\check{\boldsymbol{P}}))^{-1}\right], \tag{3.11b}$$

where κ *is defined in* (3.8c). *If* μ *is sufficiently large, then the penalized convex relaxation (3.5a) – (3.5e) has a unique optimal solution* $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{X}})$ *that satisfies* (3.3c) *and* (3.3d). *Moreover,* $\overset{*}{\boldsymbol{P}}$ *is feasible for (3.1a) – (3.1c).*

Proof. See Section 3.4 for the proof. □

3.2.3 Sequential Penalized Relaxation

Motivated by Theorems 3.1 and 3.2, this section presents a sequential approach that solves a sequence of penalized relaxations of the form (3.5a) – (3.5e) to infer high-quality feasible points for the non-convex problem (3.1a) – (3.1c). The proposed scheme starts from an initial point $\check{\boldsymbol{P}}$ on the Stiefel manifold. In each round, the solution of the penalized relaxation (3.5a) – (3.5e) is projected onto the Stiefel manifold and then the projected point is employed as an initialization for the next round. Once a feasible point for (3.1a) – (3.1c) is obtained, according to Theorem 3.1, the proposed scheme preserves feasibility and generates a sequence of points whose objective values monotonically improves. The details of the sequential scheme are delineated in Algorithm 1.

Algorithm 1 Sequential Penalized Relaxation

```
Input: $\check{\boldsymbol{P}} \in \mathcal{S}_{n,m}$, a fixed parameter $\mu > 0$, and $k = 0$,
1: repeat
2:     $k \leftarrow k + 1$
3:     $\boldsymbol{P}^k \leftarrow$ solve (3.5a) – (3.5e) with the penalty $\mu\langle \boldsymbol{P}, \check{\boldsymbol{P}} \rangle$
4:     $\check{\boldsymbol{P}} \leftarrow \mathrm{proj}_{\mathcal{S}_{n,m}} \boldsymbol{P}^k$
5: until stopping criteria is met
Output: $\boldsymbol{P}^k$
```

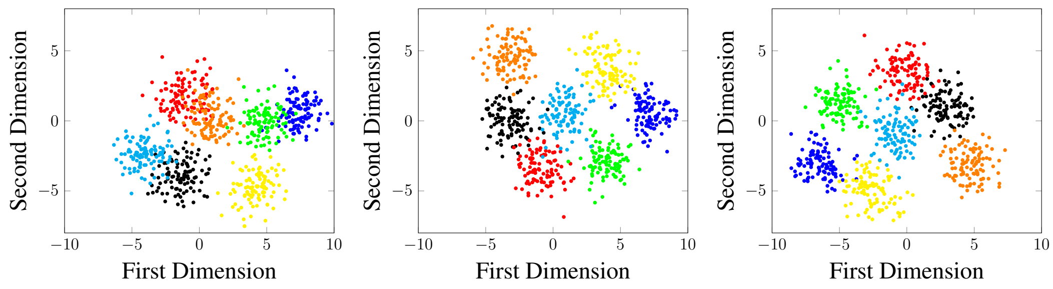


Figure 3.1: Two dimensional data representation on a training set from the synthetic data set. Left: MMDA [4], middle: SPR-S, right: SPR-Q. The results show that the SPR-S and SPR-Q algorithms have provided more discriminative 2D representations compared to the MMDA method.

3.3 Experimental Results

In this section, we conduct numerical experiments on real and synthetic datasets to verify the effectiveness of the proposed sequential approach, termed SPR, in solving non-convex optimization problems with orthogonality constraints. In Subsections 3.3.1 and 3.3.2, we apply SPR on two practical problems involving orthogonality constraints. We use SPR-S and SPR-Q to refer to the combination of Algorithm 1 with the SDP relaxation and the proposed convex quadratic relaxation, respectively. To solve the penalized relaxations in each round of the algorithm, we use MOSEK version 7.0 [8]. Through the experiments, we leverage the inherent sparsity patterns of the problems to reduce the computational cost of solving large-scale semidefinite programs. This enables us to break down large-scale conic constraints into a set of smaller ones [141]. Since finding a feasible point for (3.1a) – (3.1c) can be computationally demanding, we initialize Algorithm 1 with an arbitrary starting point on the Stiefel manifold and aim to improve the quality of the point. If the algorithm can recover a feasible point for (3.1a) – (3.1c), according to Theorem 3.1, it can generate a sequence of feasible points whose objective values monotonically improve. To measure the level of infeasibility, define $\mathrm{tr}\{\bar{\boldsymbol{X}} - \bar{\boldsymbol{P}}\bar{\boldsymbol{P}}^\top\}$ as the feasibility violation of an arbitrary feasible point $(\bar{\boldsymbol{P}}, \bar{\boldsymbol{X}})$ of the problem (3.5a) – (3.5e). We terminate the sequential algorithm once the feasibility violation and objective value improvement are less than 10^{-5} or if the round number exceeds 100. Notice that the Nesterov acceleration technique can be employed to improve the convergence behaviour of the SPR algorithm. However, in this case, the algorithm may fail to preserve the monotonically decreasing order of the objective values even if the initial point is feasible.

We apply the sequential algorithm on two fundamental machine learning problems of discriminative dimensionality reduction and graph clustering. Notice that each of these problems are well-studied in the literature and several approaches have been developed to efficiently target these applications. Therefore, it is not the intent of this work to compete with these state-of-the-art problem-specific approaches, but rather to demonstrate the potential of Algorithm 1 in solving the problems of form (3.1a) – (3.1c) that widely arise in different areas of machine learning.

3.3.1 Experiment I: Discriminative Dimensionality Reduction

Given a collection of high-dimensional data points from c different classes, the problem of discriminative dimensionality reduction aims to learn a low-dimensional subspace on which the projection of different classes are well-separated. To this end, [4] proposed a max-min distance analysis (MMDA) that maximizes the minimum distance between all class pairs. This problem can be cast as a non-convex and non-smooth optimization problem of form

$$\underset{\boldsymbol{P}\in\mathbb{R}^{n\times m}}{\text{maximize}} \quad \min_{1\leq i<j\leq c} \langle \boldsymbol{A}^{ij}, \boldsymbol{P}\boldsymbol{P}^{\top}\rangle \tag{3.12a}$$

$$\text{subject to} \quad \boldsymbol{P}^{\top}\boldsymbol{P} = \boldsymbol{I}_m, \tag{3.12b}$$

where each $\boldsymbol{A}^{ij} \in \mathbb{S}_n$ is a given weighted distance matrix between the i^{th} and j^{th} classes. In this experiment, we evaluate the performance of the SPR algorithm for solving the problems of form (3.12a) – (3.12b). Closely related to our work, [4] uses a sequence of local SDP relaxations to find the solution of problem (3.12a) – (3.12b). We benchmark the SPR method against the MMDA on both real and synthetic datasets. To ensure the comparison is fair, both methods use the same initial point and the same distance matrices $\boldsymbol{A}^{ij}$ which are computed based on [4]. Other parameter settings of the MMDA are set to their default values. Following [4], we conduct 100 independent experiments on 10-dimensional synthetic data from seven classes. For each class i, a mean vector $\boldsymbol{\eta}_i \in \mathbb{R}^{10}$ is sampled from 10-dimensional zero mean Gaussian distribution with co-variance matrix $2\boldsymbol{I}_{10}$ and then a pair of training and testing sets, each with 100 members, is generated based on the Gaussian distribution $\mathcal{N}(\boldsymbol{\eta}_i, \boldsymbol{I}_{10})$.

To compare the classification error rate, we project each test set into subspaces with varying dimensions, learned on its corresponding training set. The projected instances are then classified using the nearest mean classifier. Figure 3.2 (left) shows the average classification error rate with respect to the reduced dimensionality on the synthetic datasets. To run the experiment on the synthetic datasets, we set μ to 100 and 200 for SPR-S and SPR-Q, respectively. Moreover, we conduct this experiment on the YALE dataset consisting of 165 frontal face images of 15 individuals under different illumination and lightening conditions [142]. Each image is of size 32×32 pixels. The results of this experiment are illustrated in Figure 3.2 (right). According to Figure 3.2, SPR-S and SPR-Q perform on

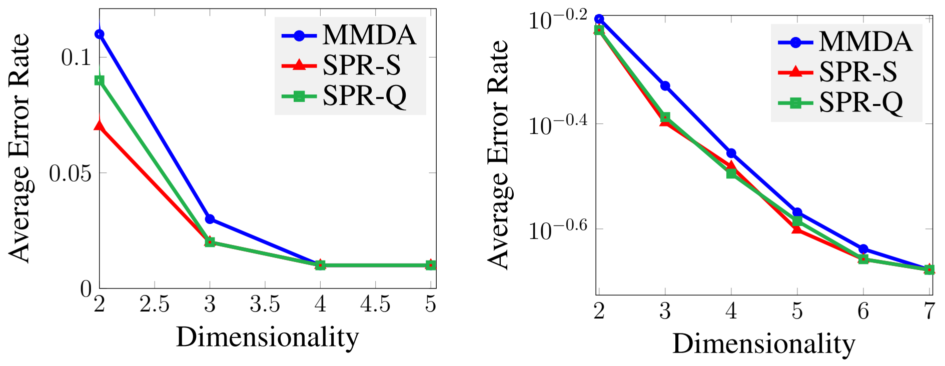
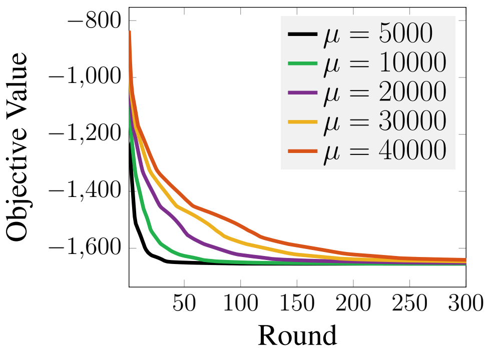


Figure 3.2: Performance of SPR comparing to MMDA [4] on left: synthetic dataset, right: YALE dataset [5]. *Best viewed in color.*

par or better than the MMDA algorithm on both real and synthetic datasets in the problem of discriminative dimensionality reduction. In the experiment on the YALE dataset, we set μ to 5000 and 10000 for SPR-S and SPR-Q, respectively. To qualitatively compare the methods, Figure 3.1 visualizes the results of projecting a randomly chosen training set from the synthetic dataset on the 2D space. Observe that comparing to the MMDA method, the SPR-based algorithms learn more discriminative 2D representations that are suitable for classification tasks.

To assess the sensitivity of the SPR algorithm with respect to the parameter μ, we perform the discriminative dimensionality reduction experiment with $m = 2$ on YALE dataset and report the results in Figure 3.3 for various choices of μ. Observe that the final solution obtained by the proposed algorithm is not very sensitive to the choice of μ. According to the figure, the SPR-S requires smaller values of μ to recover feasible points, e.g. $\mu = 5000$, while SPR-Q fails to find feasible points for such choice of μ. Moreover, it can be seen that if μ exceeds a certain threshold, both SPR-S and SPR-Q provide the same sequence of feasible points.



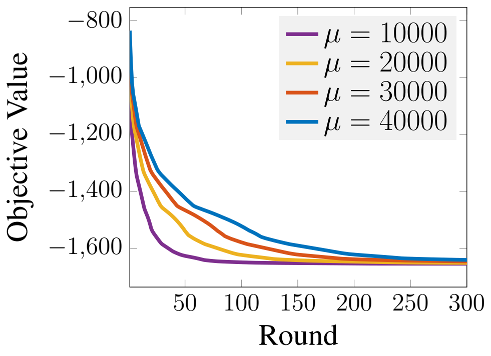


Figure 3.3: Sensitivity analysis of SPR-S (left) and SPR-Q (right) with respect to different choices of parameter μ for the discriminative dimensionality reduction problem, where $m\!=\!2$. This experiment is performed on the YALE dataset. *Best viewed in color.*

3.3.2 Experiment II: Graph Clustering

Given a weighted graph $\mathcal{G}$ with n vertices, the graph clustering problem aims to partition $\mathcal{G}$ into a set of sub-graphs such that the vertices within each one are more densely connected to each other than those belonging to different sub-graphs. Inspired by the well-known spectral clustering technique [143], this experiment incorporates a set of non-negative constraints to formulate the graph clustering problem as the following optimization [44]:

$$\underset{\boldsymbol{P}\in\mathbb{R}^{n\times m}}{\text{minimize}} \quad \langle \boldsymbol{L}, \boldsymbol{P}\boldsymbol{P}^\top \rangle \tag{3.13a}$$

$$\text{subject to} \quad \boldsymbol{P}^\top \boldsymbol{P} = \boldsymbol{I}_m, \tag{3.13b}$$

$$\boldsymbol{P} \geq 0, \tag{3.13c}$$

where $\boldsymbol{L}$ denotes the Laplacian matrix of the weighted graph $\mathcal{G}$ and $\geq$ is the element-wise inequality operator. Comparing to the spectral clustering, formulation (3.13a) – (3.13c) offers a more interpretable clustering framework which requires no further post-processing steps to identify the cluster members. Given $\overset{*}{\boldsymbol{P}}$, the optimal solution of the above problem, each vertex i is assigned to a cluster with label $\operatorname{argmax}_j \overset{*}{P}_{ij}$. [44] proposed a fast and scalable heuristic, denoted by ONGR, to solve large-scale instances of the form (3.13a) –

Dataset	n	Dim.	m	ONGR	SPR-S	SPR-Q
Iris	150	4	3	79.84	**86.71**	81.23
Spiral	312	2	3	87.44	**95.76**	94.15
Jain	373	2	2	88.42	**92.33**	90.26
Compound	399	2	6	74.57	74.25	**76.48**
R15	600	2	15	86.07	85.36	**86.94**
Aggregation	788	2	7	**87.84**	86.39	84.66

Table 3.1: Clustering performance (%) on the UCI datasets [9] and shape sets [10, 11, 12].

(3.13c). Due to the fact that this problem is a special case of (3.1a) – (3.1c), we apply the SPR algorithm to find the solution of (3.13a) – (3.13c) and use the same procedure as [44] to create the Laplacian matrix $\boldsymbol{L}$. To make a fair comparison between the ONGR and SPR, we use the same initialization for both methods. Table 3.1 reports the clustering performance of the SPR against [44] on well-known datasets from the UCI machine learning repository [9] and shape sets [10, 12]. For each dataset, n, Dim, and m refer to the number of sample points, dimension of each point, and the number of classes, respectively. The scores for each method is computed by averaging over 30 independent runs for each dataset. As the results indicate, SPR-S and SPR-Q exhibit better performance compared to [44] on most of the datasets. Through this experiment, we set $\mu = 1000$ in the SPR algorithm and use the default parameter settings for the ONGR algorithm.

3.4 Proofs

This section presents the proof of Theorems 3.1 and 3.2. Before proceeding with the proofs, we provide some prerequisite lemmas.

Using the well-known epigraph technique [13], the non-smooth term $g_0(\boldsymbol{P})$ in (3.3a) can be removed by adding a pair of linear constraints and incorporating an additional term into the objective function. This reformulation of (3.3a) – (3.3d) leads to the following penalized non-convex problem:

$$\begin{aligned}
&\underset{\substack{\boldsymbol{P}\in\mathbb{R}^{n\times m}\\ \boldsymbol{t}\in\mathbb{R}^{w}}}{\text{minimize}} && \mathbf{1}^{\top}\boldsymbol{t}+\langle\boldsymbol{M}_0,\boldsymbol{P}\boldsymbol{P}^{\top}\rangle+\langle\boldsymbol{N}_0,\boldsymbol{P}\rangle-\mu\langle\check{\boldsymbol{P}},\boldsymbol{P}\rangle && (3.14a)\\
&\text{subject to} && \bar{\boldsymbol{\gamma}}:+\alpha(\boldsymbol{P})+\boldsymbol{b}\leq\boldsymbol{t}, && (3.14b)\\
& && \underline{\boldsymbol{\gamma}}:-\alpha(\boldsymbol{P})-\boldsymbol{b}\leq\boldsymbol{t}, && (3.14c)\\
& && \boldsymbol{\lambda}:\langle\boldsymbol{M}_k,\boldsymbol{P}\boldsymbol{P}^{\top}\rangle+\langle\boldsymbol{N}_k,\boldsymbol{P}\rangle+q_k\leq 0,\ k\in\mathcal{K}, && (3.14d)\\
& && \boldsymbol{\Omega}:\boldsymbol{P}^{\top}\boldsymbol{P}=\boldsymbol{I}_m, && (3.14e)
\end{aligned}$$

with $\bar{\boldsymbol{\gamma}}\in\mathbb{R}^{w}$, $\underline{\boldsymbol{\gamma}}\in\mathbb{R}^{w}$, $\boldsymbol{\lambda}\in\mathbb{R}^{|\mathcal{K}|}$, and $\boldsymbol{\Omega}\in\mathbb{S}_m$ as the dual variables associated with the constraints (3.14b), (3.14c), (3.14d), and (3.14e), respectively. Observe that the problems (3.14a) – (3.14e) and (3.1a) – (3.1c) are equivalent, if $\mu = 0$. In what follows, we show that under certain conditions, the optimal solution of (3.14a) – (3.14e) can be obtained in polynomial time via convex relaxation.

The next lemma guarantees the existence of Lagrange multipliers corresponding optimal solutions of the problem (3.14a) – (3.14e).

Lemma 3.1. *Consider an arbitrary* $\check{\boldsymbol{P}}\in\mathcal{S}_{n,m}$ *that satisfies*

$$s(\check{\boldsymbol{P}})-\kappa\, d_{\mathcal{F}}(\check{\boldsymbol{P}})>0. \tag{3.15}$$

If the following inequality holds true,

$$\mu>4\beta[\kappa^{-1}s(\check{\boldsymbol{P}})-d_{\mathcal{F}}(\check{\boldsymbol{P}})]^{-2}, \tag{3.16}$$

then for every primal optimal pair $(\overset{*}{\boldsymbol{P}},\overset{*}{\boldsymbol{t}})$ *of (3.14a) – (3.14e), there exists Lagrange multipliers* $(\overset{*}{\bar{\gamma}},\overset{*}{\underline{\gamma}},\overset{*}{\boldsymbol{\lambda}},\overset{*}{\boldsymbol{\Omega}}) \in \mathbb{R}^w \times \mathbb{R}^w \times \mathbb{R}^{|\mathcal{K}|} \times \mathbb{S}_m$ *that satisfy the following Karush–Kuhn–Tucker (KKT) conditions*

$$\nabla_{\scriptscriptstyle P}\mathcal{L}(\overset{*}{\boldsymbol{P}},\overset{*}{\boldsymbol{t}},\overset{*}{\bar{\gamma}},\overset{*}{\underline{\gamma}},\overset{*}{\boldsymbol{\lambda}},\overset{*}{\boldsymbol{\Omega}}) = \boldsymbol{0}, \tag{3.17a}$$

$$\boldsymbol{1} + \overset{*}{\bar{\gamma}} + \overset{*}{\underline{\gamma}} = \boldsymbol{0}, \tag{3.17b}$$

$$\overset{*}{\bar{\gamma}} \circ (+\alpha(\overset{*}{\boldsymbol{P}}) + \boldsymbol{b} - \overset{*}{\boldsymbol{t}}) = \boldsymbol{0}, \tag{3.17c}$$

$$\overset{*}{\underline{\gamma}} \circ (-\alpha(\overset{*}{\boldsymbol{P}}) - \boldsymbol{b} - \overset{*}{\boldsymbol{t}}) = \boldsymbol{0}, \tag{3.17d}$$

$$\overset{*}{\lambda}_k(\langle \boldsymbol{M}_k, \overset{*}{\boldsymbol{P}}\overset{*}{\boldsymbol{P}}^{\top}\rangle + \langle \boldsymbol{N}_k, \overset{*}{\boldsymbol{P}}\rangle + q_k) = 0 \qquad k \in \mathcal{K}, \tag{3.17e}$$

$$\overset{*}{\bar{\gamma}} \; \leq \boldsymbol{0}, \qquad \overset{*}{\underline{\gamma}} \; \leq \boldsymbol{0}, \qquad \overset{*}{\boldsymbol{\lambda}} \leq 0, \tag{3.17f}$$

where $\mathcal{L}(\boldsymbol{P},\boldsymbol{t},\bar{\gamma},\underline{\gamma},\boldsymbol{\lambda},\boldsymbol{\Omega})$ *represents the Lagrangian function of (3.14a) – (3.14e), defined as*

$$\begin{aligned}\mathcal{L}(\boldsymbol{P},\boldsymbol{t},\bar{\gamma},\underline{\gamma},\boldsymbol{\lambda},\boldsymbol{\Omega}) \triangleq\; & \boldsymbol{1}^{\top}\boldsymbol{t} + \langle \boldsymbol{M}_0, \boldsymbol{P}\boldsymbol{P}^{\top}\rangle + \langle \boldsymbol{N}_0, \boldsymbol{P}\rangle - \mu\langle \check{\boldsymbol{P}}, \boldsymbol{P}\rangle \\ & - \bar{\gamma}^{\top}(\alpha(\boldsymbol{P}) + \boldsymbol{b} - \boldsymbol{t}) + \underline{\gamma}^{\top}(\alpha(\boldsymbol{P}) + \boldsymbol{b} + \boldsymbol{t}) \\ & - \sum_{k\in\mathcal{K}} \lambda_k(\langle \boldsymbol{M}_k, \boldsymbol{P}\boldsymbol{P}^{\top}\rangle + \langle \boldsymbol{N}_k, \boldsymbol{P}\rangle + q_k) - \langle \boldsymbol{\Omega}, \boldsymbol{P}^{\top}\boldsymbol{P} - \boldsymbol{I}_m\rangle.\end{aligned}$$

and, β *and* κ *are defined in* (3.8a) *and* (3.8c).

The next lemma provides an upper bound on the Lagrange multipliers of the problem (3.14a) – (3.14e), that will be used to show that this problem can be relaxed to (3.5a) – (3.5e) with no effect on the solution.

Lemma 3.2. *Consider an arbitrary* $\check{\boldsymbol{P}} \in \mathcal{S}_{n,m}$ *that satisfies* (3.15) *and let* μ *satisfy* (3.16). *For every solution* $(\overset{*}{\boldsymbol{P}},\overset{*}{\boldsymbol{t}})$ *of (3.14a) – (3.14e), there exist Lagrange multipliers* $(\overset{*}{\bar{\gamma}},\overset{*}{\underline{\gamma}},\overset{*}{\boldsymbol{\lambda}},\overset{*}{\boldsymbol{\Omega}}) \in \mathbb{R}^w \times \mathbb{R}^w \times \mathbb{R}^p \times \mathbb{S}_m$ *that satisfy the KKT conditions (3.17a) – (3.17f) as well as the inequalities:*

$$-\frac{\boldsymbol{1}^{\top}\overset{*}{\boldsymbol{\lambda}}}{\mu} \leq \frac{d_{\mathcal{F}}(\check{\boldsymbol{P}}) + \mu^{-1}\psi + 2\sqrt{\beta\mu^{-1}}}{s(\check{\boldsymbol{P}}) - \kappa(d_{\mathcal{F}}(\check{\boldsymbol{P}}) + 2\sqrt{\beta\mu^{-1}})} \tag{3.18a}$$

$$\Big\|\frac{2}{\mu}\overset{*}{\boldsymbol{\Omega}} + \boldsymbol{I}_m\Big\|_{\mathrm{F}} \leq \kappa_2\Big(-\frac{\boldsymbol{1}^{\top}\overset{*}{\boldsymbol{\lambda}}}{\mu}\Big) + d_{\mathcal{F}}(\check{\boldsymbol{P}}) + \mu^{-1}\psi + 2\sqrt{\beta\mu^{-1}} \tag{3.18b}$$

where constant κ_2 is given by

$$\kappa_2 \triangleq 2 \max_{k\in\mathcal{K}}\{\|\boldsymbol{M}_k\|_{\mathrm{F}}\} + \max_{k\in\mathcal{K}}\{\|\boldsymbol{N}_k\|_{\mathrm{F}}\}, \tag{3.19}$$

and β, ψ and κ are defined in (3.8a) – (3.8c).

Using Lemma 3.2, the next lemma offers conditions to guarantee that penalized relaxations give feasible points for (3.14a) – (3.14e).

Lemma 3.3. *Consider an initial point $\check{\boldsymbol{P}} \in \mathcal{S}_{n,m}$ and $\mu > 0$. Let $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{t}})$ be a primal optimal solution of (3.14a) – (3.14e) with the corresponding Lagrange multipliers $(\overset{*}{\bar{\gamma}}, \overset{*}{\underline{\gamma}}, \overset{*}{\boldsymbol{\lambda}}, \overset{*}{\boldsymbol{\Omega}})$ that satisfy the KKT conditions (3.17a) – (3.17e). Define*

$$\varepsilon \triangleq \frac{1}{4}\Big(1 - d_{\mathcal{F}}(\check{\boldsymbol{P}}) - \frac{\kappa\, d_{\mathcal{F}}(\check{\boldsymbol{P}})}{s(\check{\boldsymbol{P}}) - \kappa\, d_{\mathcal{F}}(\check{\boldsymbol{P}})}\Big). \tag{3.20}$$

If the following inequalities hold true

$$2\mu^{-1}\|\boldsymbol{M}_0\| \leq \varepsilon, \tag{3.21a}$$

$$-\frac{\boldsymbol{1}^{\top}\overset{*}{\boldsymbol{\lambda}}}{\mu} \leq \frac{d_{\mathcal{F}}(\check{\boldsymbol{P}})}{s(\check{\boldsymbol{P}}) - \kappa\, d_{\mathcal{F}}(\check{\boldsymbol{P}})} + \frac{\varepsilon}{2\kappa_2}, \tag{3.21b}$$

$$\Big\|\frac{2}{\mu}\overset{*}{\boldsymbol{\Omega}} + \boldsymbol{I}_m\Big\|_{\mathrm{F}} \leq \kappa_2\Big(-\frac{\boldsymbol{1}^{\top}\overset{*}{\boldsymbol{\lambda}}}{\mu}\Big) + d_{\mathcal{F}}(\check{\boldsymbol{P}}) + \varepsilon, \tag{3.21c}$$

then the pair $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{P}}\overset{*}{\boldsymbol{P}}^{\top})$ is the unique primal solution of the penalized convex relaxation (3.5a) – (3.5d), where κ and κ_2 are defined in (3.8c) *and* (3.19)*, respectively.*

Proof of Theorem 3.1. Due to the main assumption, it is straightforward to verify the following three inequalities:

$$\mu^{-1}\psi < \sqrt{\beta\mu^{-1}}, \tag{3.22a}$$

$$2\kappa\sqrt{\beta\mu^{-1}} < 13^{-1}s(\check{\boldsymbol{P}}), \tag{3.22b}$$

$$\sqrt{\beta\mu^{-1}} < 12^{-1}. \tag{3.22c}$$

Consider an arbitrary optimal solution $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{t}})$ of (3.14a) – (3.14e). The point $\overset{*}{\boldsymbol{P}}$ is consequently feasible for (3.1a) – (3.1c). Therefore $d_{\mathcal{F}}(\check{\boldsymbol{P}}) = 0$ and the inequalities (3.15) and (3.16) are satisfied. According to Lemma 3.2, there exist Lagrange multipliers $(\overset{*}{\bar{\boldsymbol{\gamma}}}, \overset{*}{\underline{\boldsymbol{\gamma}}}, \overset{*}{\boldsymbol{\lambda}}, \overset{*}{\boldsymbol{\Omega}}) \in \mathbb{R}^w \times \mathbb{R}^w \times \mathbb{R}^p \times \mathbb{S}_m$ corresponding to $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{t}})$ that satisfy the KKT conditions (3.17a) – (3.17f) as well as the inequalities (3.18a) and (3.18b). Based on Lemma 3.3 and since $d_{\mathcal{F}}(\check{\boldsymbol{P}}) = 0$, in order to prove the theorem, it suffices to show that:

$$2\mu^{-1}\|\boldsymbol{M}_0\| \leq 4^{-1} \tag{3.23a}$$

$$-\frac{\mathbf{1}^\top \overset{*}{\boldsymbol{\lambda}}}{\mu} \leq \frac{4^{-1}}{2\kappa_2} \tag{3.23b}$$

$$\Big\|\frac{2}{\mu}\overset{*}{\boldsymbol{\Omega}} + \boldsymbol{I}_m\Big\|_{\mathrm{F}} \leq \kappa_2\Big(-\frac{\mathbf{1}^\top \overset{*}{\boldsymbol{\lambda}}}{\mu}\Big) + 4^{-1}. \tag{3.23c}$$

- (3.23a) is the direct consequence of (3.22a):

$$2\mu^{-1}\|\boldsymbol{M}_0\| \leq \mu^{-1}\psi \leq \sqrt{\beta\mu^{-1}} \leq 12^{-1} < 4^{-1}. \tag{3.24}$$

- (3.23b) is the direct consequence of (3.18a), (3.22b), and (3.22c):

$$\begin{aligned}
-\frac{\mathbf{1}^\top \overset{*}{\boldsymbol{\lambda}}}{\mu} &\leq \frac{\mu^{-1}\psi + 2\sqrt{\beta\mu^{-1}}}{s(\check{\boldsymbol{P}}) - 2\kappa\sqrt{\beta\mu^{-1}}} \leq \frac{\sqrt{\beta\mu^{-1}} + 2\sqrt{\beta\mu^{-1}}}{s(\check{\boldsymbol{P}}) - 2\kappa\sqrt{\beta\mu^{-1}}} && (3.25\text{a})\\
&\leq \frac{\sqrt{\beta\mu^{-1}} + 2\sqrt{\beta\mu^{-1}}}{s(\check{\boldsymbol{P}}) - 13^{-1}s(\check{\boldsymbol{P}})} = \frac{3\sqrt{\beta\mu^{-1}}}{(1 - 13^{-1})s(\check{\boldsymbol{P}})} && (3.25\text{b})\\
&\leq \frac{3 \times 13^{-1}(2\kappa)^{-1}s(\check{\boldsymbol{P}})}{(1 - 13^{-1})s(\check{\boldsymbol{P}})} = \frac{4^{-1}}{2\kappa} < \frac{4^{-1}}{2\kappa_2}. && (3.25\text{c})
\end{aligned}$$

- (3.23c) can be concluded from (3.18b), (3.22a), and (3.22c):

$$\begin{aligned}
\Big\|\frac{2}{\mu}\overset{*}{\boldsymbol{\Omega}} + \boldsymbol{I}_m\Big\|_{\mathrm{F}} &\leq \kappa_2\Big(-\frac{\mathbf{1}^\top \overset{*}{\boldsymbol{\lambda}}}{\mu}\Big) + \mu^{-1}\psi + 2\sqrt{\beta\mu^{-1}} && (3.26\text{a})\\
&\leq \kappa_2\Big(-\frac{\mathbf{1}^\top \overset{*}{\boldsymbol{\lambda}}}{\mu}\Big) + 3\sqrt{\beta\mu^{-1}} && (3.26\text{b})\\
&\leq \kappa_2\Big(-\frac{\mathbf{1}^\top \overset{*}{\boldsymbol{\lambda}}}{\mu}\Big) + 4^{-1}. && (3.26\text{c})
\end{aligned}$$

Hence, according to Lemma 3.3, the point $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{P}}\overset{*}{\boldsymbol{P}}^\top)$ is the unique optimal solution for the penalized relaxation (3.5a) – (3.5e), for which the relaxed constraints (3.3c) and (3.3d) are satisfied. Finally, due to the feasibility of pair $(\check{\boldsymbol{P}}, \check{\boldsymbol{P}}\check{\boldsymbol{P}}^\top)$, we have:

$$\begin{aligned} \bar{f}_0(\check{\boldsymbol{P}})+g_0(\check{\boldsymbol{P}})-\mu m &= f_0(\check{\boldsymbol{P}},\check{\boldsymbol{P}}\check{\boldsymbol{P}}^\top)+g_0(\check{\boldsymbol{P}})-\mu\langle\check{\boldsymbol{P}},\check{\boldsymbol{P}}\rangle && (3.27a)\\ &\geq f_0(\overset{*}{\boldsymbol{P}},\overset{*}{\boldsymbol{P}}\overset{*}{\boldsymbol{P}}^\top)+g_0(\overset{*}{\boldsymbol{P}})-\mu\langle\overset{*}{\boldsymbol{P}},\check{\boldsymbol{P}}\rangle && (3.27b)\\ &\geq \bar{f}_0(\check{\boldsymbol{P}})+g_0(\check{\boldsymbol{P}})-\mu m && (3.27c)\end{aligned}$$

and the proof is completed. □

Proof of Theorem 3.2. Consider an arbitrary optimal solution $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{t}})$ of (3.14a) – (3.14e). Due to the main assumption, (3.15) is satisfied and if μ is large, then (3.16) is satisfied as well. Moreover, according to Lemma 3.2, there exist Lagrange multipliers $(\overset{*}{\bar{\gamma}}, \overset{*}{\underline{\gamma}}, \overset{*}{\boldsymbol{\lambda}}, \overset{*}{\boldsymbol{\Omega}}) \in \mathbb{R}^w \times \mathbb{R}^w \times \mathbb{R}^p \times \mathbb{S}_m$ corresponding to $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{t}})$ that satisfy the KKT conditions (3.17a) – (3.17f) as well as the inequalities (3.18a) and (3.18b). According to Lemma 3.3, the proof follows directly from the fact that

$$\varepsilon = \frac{1}{4}\Big(1 - d_{\mathcal{F}}(\check{\boldsymbol{P}}) - \frac{\kappa\ d_{\mathcal{F}}(\check{\boldsymbol{P}})}{s(\check{\boldsymbol{P}}) - \kappa\ d_{\mathcal{F}}(\check{\boldsymbol{P}})}\Big) > 0, \qquad (3.28)$$

and therefore, if μ is sufficiently large, the inequalities (3.18a) and (3.18b) conclude (3.21a) – (3.21c). As a result, if μ is large, $(\overset{*}{\boldsymbol{P}}, \overset{*}{\boldsymbol{P}}\overset{*}{\boldsymbol{P}}^\top)$ is the unique primal solution of the penalized convex relaxation (3.5a) – (3.5d). □

3.5 Conclusions

This work introduces convex relaxations for solving a broad class of non-convex and non-smooth optimization problems involving orthogonality constraints. The proposed approach relies on solving a sequence of penalized convex relaxations to find feasible and near-globally optimal points for a given non-convex orthogonality-constrained problem.

Experimental results on two fundamental problems in machine learning demonstrate the potential and effectiveness of the proposed approach in solving practical problems.

CHAPTER 4

Convex Relaxations for Training Neural Networks

This chapter investigates the application of convex optimization in training neural networks (NNs). Our main contribution is a convex relaxation tailored for NNs that can serve as an alternative to the existing relaxations from the area of nonlinear optimization. We prove that by incorporating a family of regularization terms the proposed relaxation is guaranteed to be exact, using which the training task can be cast as a convergent sequence of convex problems. This approach improves upon the common-practice gradient-based methods by enabling the incorporation of hard constraints. Lastly, the potential of the proposed approach is corroborated on the problem of imbalanced classification.

4.1 Introduction

Neural networks (NNs) have been demonstrated to have special abilities in extracting sophisticated information from raw data. This renders them as suitable tools for a wide variety of applications in artificial intelligence and machine learning including classification [70, 71, 72, 73] and depth estimation [74, 75], image-to-image translation [144, 145] among many. Despite the widespread use of NNs, a complete understanding of their success is still lacking and theoretical studies are mainly limited to the networks with special architectures [80, 83].

The primary challenge in training NNs arises from the non-convexity of the training problems. Gradient-based approaches such as stochastic gradient descent, conjugate gradient, and Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) are among the most popular numerical methods for training NNs [146, 147, 32, 33, 148, 149]. These approaches rely descent directions obtained via recursively calculation of gradient with respect to network parameter. There is considerable theoretical and empirical evidence indicating the effectiveness of the gradient-based methods in converging to global optimality (or satisfactory local optimality), under various assumptions [150, 151, 152]. However, for general network architectures, gradient-based methods often suffer from several problems such as the vanishing/exploding gradient phenomena. Moreover, they cannot incorporate hard constraints which have been proven to be advantageous in many applications [153, 154].

In the light of empirical success of NNs, a question arises as to whether more sophisticated optimization techniques can be leveraged to train these networks. In response to this question, we investigate a well-studied technique in nonlinear optimization literature, namely convex relaxation, which reduces non-convex problems into convex surrogate. Among various relaxation approaches, semidefinite programming (SDP) [49] stands out for offering high-quality solutions. However, its applicability is limited to moderate size problems since they increase the problem size quadratically and require computationally expensive conic inequalities. Moreover, in the presence of constraints, the optimal solutions of these relaxations are not necessarily feasible for the original non-convex problem [155].

4.1.1 Contributions

In this work, we aim to establish a bridge between the areas of artificial neural networks and convex optimization by developing a powerful and flexible training approach. To serve this purpose, we first transform the training problem into an equivalent constrained optimization. Then, we convexify this constrained optimization by means of a novel convex quadratic relaxation and the well-known difference of convex programming technique.

To ensure that the convexified problem provides a feasible point to the training problem, a novel regularization term is incorporated into the objective of the relaxed problems.

On the theoretical front, we derive certain conditions under which the regularized relaxation is guaranteed to provide feasible points for the training problem. Moreover, we theoretically prove that, if certain assumptions are met, solving a sequence of the regularized problem results in a convergent sequence of feasible points whose objective values monotonically improve. The proposed approach, called Convex-NN, offers various theoretical and practical advantages: it jointly estimates the network parameters, admits additional convex constraints, and provides a flexible framework for further study and exploration. The potential of the proposed approach is corroborated on the problem of imbalanced classification.

4.1.2 Notation

Throughout this work, symbols $\mathbb{R}^n$, $\mathbb{R}^{n\times m}$, and $\mathbb{S}_n$ denote the set of real n-dimensional vectors, real $n\times m$ matrices, and $n\times n$ real symmetric matrices, respectively. For a given matrix $\boldsymbol{a}$, the notations a_{ij} and $\boldsymbol{a}_{i*}$ respectively refer to the $(i,j)^{th}$ element and the i^{th} row of $\boldsymbol{a}$. In addition, given matrices $\boldsymbol{a}$ and $\boldsymbol{b}$ of the same size, symbols $\boldsymbol{a}\circ\boldsymbol{b}$ and $\langle\boldsymbol{a},\boldsymbol{b}\rangle$ stand for their Hadamard product and inner product. Notation $\|.\|_p$ refers to either the matrix norm or the vector norm depending on the context, $\|.\|_{\mathrm{F}}$ shows the Frobenius norm, and $|.|$ indicates the absolute value. Symbols $\mathrm{tr}\{.\}$ and $(.)^\top$, respectively, denote the trace operator and transpose operator. The notation $\mathcal{I}_n$ represents the index set $\{1,\ldots,n\}$. Notations $\mathbf{1}_n$ and $\boldsymbol{I}_n$ refer to $n\times 1$ column vector of all ones and identity matrix of size $n\times n$.

4.2 Problem Formulation

Consider a general neural network consists of L layers, each of which is defined by a linear function with weights $\boldsymbol{w}^l\!\in\!\mathbb{R}^{n_{l-1}\times n_l}$ and biases $\boldsymbol{b}^l\!\in\!\mathbb{R}^{n_l}$, and nonlinear activation function

h^l: $\mathbb{R} \rightarrow \mathbb{R}$, where n_l denotes the number of hidden units in layer l. This network can be seen as a mapping function $f(\,\cdot\,; \boldsymbol{w}, \boldsymbol{b})$: $\mathbb{R}^{n_0} \rightarrow \mathbb{R}^{n_L}$ that maps the input vector $\boldsymbol{x} \in \mathbb{R}^{n_0}$ into the following output:

$$f(\boldsymbol{x}; \boldsymbol{w}, \boldsymbol{b}) = \boldsymbol{w}^{L^\top} h^{L\text{-}1}(...\boldsymbol{w}^{2^\top} h^1(\boldsymbol{w}^{1^\top} \boldsymbol{x} + \boldsymbol{b}^1)...) + \boldsymbol{b}^L, \tag{4.1}$$

where $\boldsymbol{b} = \{\boldsymbol{b}^l\}_{l \in \mathcal{I}_L}$ and $\boldsymbol{w} = \{\boldsymbol{w}^l\}_{l \in \mathcal{I}_L}$. Following the common practice, we assume that h^L is the identity function [156, 157]. Let $\{(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathbb{R}^{n_0} \times \mathbb{R}^{n_L}\}_{i \in \mathcal{I}_m}$ be a set of m data points where $\boldsymbol{x}_i \in \mathbb{R}^{n_0}$ and $\boldsymbol{y}_i \in \mathbb{R}^{n_L}$ denote the i^{th} feature data and its corresponding label, respectively. Training the neural network (4.1) is tantamount to learning parameters $\boldsymbol{w}$ and $\boldsymbol{b}$ such that the misfit between the predicted output and the true labels is minimized in terms of a desired loss function. Using the ℓ_2-norm loss function, this problem can be formulated as:

$$\underset{\boldsymbol{w}, \boldsymbol{b}}{\text{minimize}} \quad C(\boldsymbol{w}, \boldsymbol{b}), \tag{4.2}$$

where $C(\boldsymbol{w}, \boldsymbol{b}) = \sum_{i \in \mathcal{I}_m} \|f(\boldsymbol{x}_i; \boldsymbol{w}, \boldsymbol{b}) - \boldsymbol{y}_i\|_2^2$. The training problem (4.2) is non-convex and computationally challenging, due to nonlinearity of the nested function f [152, 158, 159]. Despite the efficiency of common-practice local search algorithms for solving the problem (4.2), they suffer from several issues including vanishing gradients, wherein the gradient elements corresponding to the network parameters in early layers become considerably small and provide little information about the loss function, resulting in slow convergence [78, 160, 161, 156].

To break the functional dependencies of function f, define auxiliary variables $\boldsymbol{z} = \{\boldsymbol{z}^l \in \mathbb{R}^{n_l \times m}\}_{l \in \mathcal{I}_L}$ and $\boldsymbol{a} = \{\boldsymbol{a}^l \in \mathbb{R}^{n_l \times m}\}_{l \in \mathcal{I}_{L\text{-}1}}$, where $\boldsymbol{z}^l$ and $\boldsymbol{a}^l$, respectively, account for the output of layer l linear transformation and nonlinear activation function. Using the auxiliary variables, the problem (4.2) can be equivalently written as:

$$\begin{aligned}
&\underset{\boldsymbol{w},\boldsymbol{b},\boldsymbol{z},\boldsymbol{a}}{\text{minimize}} && \|\boldsymbol{z}^L - \boldsymbol{y}\|_{\mathrm{F}}^2 && && \text{(4.3a)}\\
&\text{subject to} && \boldsymbol{z}^l = \boldsymbol{w}^{l\top}\boldsymbol{a}^{l-1} + \boldsymbol{b}^l\mathbf{1}_m^\top, && l \in \mathcal{I}_L, && \text{(4.3b)}\\
& && \boldsymbol{a}^l = h^l(\boldsymbol{z}^l), && l \in \mathcal{I}_{L-1}, && \text{(4.3c)}
\end{aligned}$$

where $\boldsymbol{a}^0 \in \mathbb{R}^{n_0 \times m}$ and $\boldsymbol{y} \in \mathbb{R}^{n_L \times m}$ are fixed matrices obtained by concatenating column vectors $\{\boldsymbol{x}_i\}_{i\in\mathcal{I}_m}$ and $\{\boldsymbol{y}_i\}_{i\in\mathcal{I}_m}$, respectively. Notice that the problem (4.3a) – (4.3c) offers a more interpretable reformulation of (4.2) in which bilinear terms $\{\boldsymbol{w}^{l\top}\boldsymbol{a}^{l-1}\}_{l\in\mathcal{I}_L}$ and nonlinear activation functions $\{h^l\}_{l\in\mathcal{I}_{L-1}}$ are the sources of nonconvexity. In this work, we transform (4.3a) – (4.3c) into a class of computationally tractable convex programs by means of a novel relaxation and the classical difference of convex programming technique. The following section offers brief survey of the relevant literature.

4.3 Related Work

There has been a recent surge of interest in developing convex formulations of neural network and its variant [162, 163, 164, 161, 165]. [162] showed that training a neural network can be seen as a convex optimization problem involving an infinite number of parameters. [164] developed a convex formulation of multi-layer learning using normalized kernels. [81] cast the problem of training a convolutional neural network as a low-rank minimization problem which is further relaxed to obtain a convex formulation.

From a different viewpoint, an alternative line of research has considered replacing the training problem (4.2) with a constrained non-convex problem of form (4.3a) – (4.3c) or its variants [161, 156, 166]. [161] proposed the method of auxiliary coordinates (MAC) which introduces auxiliary variables as a proxy of the network activations and then incorporate a quadratic penalty term into the objective function to approximately impose the non-convex relation between them. Closely related to MAC, [156] proposed to solve the problem (4.3a) – (4.3c) using a highly parallelizable approach based on the alternating direction method of multipliers. The proposed approach alternatively solves a sequence of minimization sub-steps that each enjoys a closed-form solution. However, their proposed approach is not necessary compatible with additional hard constraints.

4.4 Convexified Neural Networks

This section describes our proposed approach, called *convexified neural network* (Convex-NN). We first introduce a computationally tractable convex relaxation for the problem (4.3a) – (4.3c). Then, we present a regularization method to ensure that the solution of the relaxed problem satisfies the constraints of the original problem. Finally, we devise a sequential approach to obtain high-quality feasible points.

4.4.1 Convex Relaxation

Problem (4.3a) – (4.3c) is non-convex and possibly intractable. The first source of non-convexity arises from the presence of bilinear products $\boldsymbol{w}^{l}\boldsymbol{a}^{l\text{-}1}$. In order to tackle the non-convexity, we propose a novel convex relaxation which transforms the constraint (4.3b) to a set of convex quadratic inequalities. For every $l\in\mathcal{I}_L$, let $\boldsymbol{W}^{l}\in\mathbb{R}^{n_l\times n_l}$ and $\boldsymbol{A}^{l\text{-}1}\in\mathbb{R}^{m\times m}$ account for the quadratic terms $\boldsymbol{w}^{l\top}\boldsymbol{w}^{l}$ and $\boldsymbol{a}^{l\text{-}1\top}\boldsymbol{a}^{l\text{-}1}$, respectively. The constraint (4.3b) can be equivalently reformulated as:

$$\begin{bmatrix} \boldsymbol{W}^{l} & (\boldsymbol{z}^{l}-\boldsymbol{b}^{l}\boldsymbol{1}_m^{\top}) \\ (\boldsymbol{z}^{l}-\boldsymbol{b}^{l}\boldsymbol{1}_m^{\top})^{\top} & \boldsymbol{A}^{l\text{-}1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{w}^{l\top} \\ \boldsymbol{a}^{l\text{-}1\top} \end{bmatrix} \begin{bmatrix} \boldsymbol{w}^{l} & \boldsymbol{a}^{l\text{-}1} \end{bmatrix}, \tag{4.4}$$

for every $l\in\mathcal{I}_L$. To obtain a computationally tractable surrogate, we relax the constraint (4.4) to the following quadratic constraints:

$$W^l_{ii} + A^{l\text{-}1}_{jj} + 2\,(z^l_{ij} - b^l_i) \geq \|\boldsymbol{w}^l_{*i} + \boldsymbol{a}^{l\text{-}1}_{*j}\|^2, \qquad i,j \in \mathcal{I}_{n_l}, \mathcal{I}_m \tag{4.5a}$$

$$W^l_{ii} + A^{l\text{-}1}_{jj} - 2\,(z^l_{ij} - b^l_i) \geq \|\boldsymbol{w}^l_{*i} - \boldsymbol{a}^{l\text{-}1}_{*j}\|^2, \qquad i,j \in \mathcal{I}_{n_l}, \mathcal{I}_m \tag{4.5b}$$

$$W^l_{ii} \geq \|\boldsymbol{w}^l_{*i}\|^2, \qquad i \in \mathcal{I}_{n_l} \tag{4.5c}$$

$$A^{l\text{-}1}_{jj} \geq \|\boldsymbol{a}^{l\text{-}1}_{*j}\|^2, \qquad j \in \mathcal{I}_m \tag{4.5d}$$

Notice that the off-diagonal elements of $\boldsymbol{W}^l$ and $\boldsymbol{A}^{l\text{-}1}$ do not appear in the relaxed inequalities (4.5a) – (4.5d). Hence, the proposed relaxation requires a single auxiliary variable W^l_{ii} for every hidden unit $i \in \mathcal{I}_{n_l}$ and a single auxiliary variable $A^{l\text{-}1}_{jj}$ for each data point $j \in \mathcal{I}_m$. As a result, the problem size grows linearly with respect to the data points and hidden units. This is the primary strength of the proposed relaxation compared to the common-practice methods such as the semidefinite programming relaxations [167].

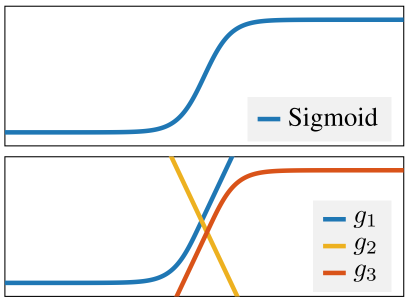
Definition 4.1. *For every $l \in \mathcal{I}_L$, define $\mathcal{Q}^l$ to be the set of all quadruplets $(\boldsymbol{w}^l, \boldsymbol{a}^{l\text{-}1}, \boldsymbol{W}^l, \boldsymbol{A}^{l\text{-}1})$ that satisfy (4.5a) – (4.5d).*

The second source of non-convexity is induced by the non-linearity of the activation functions $\{h^l\}_{l \in \mathcal{I}_{L-1}}$, for which we employ the difference of convex (DC) programming method.

Proposition 4.1. *Any energy function with bounded Hessian can be decomposed as a difference of convex functions [168].*

Proposition 4.1 states that every activation function h^l with bounded Hessian can be decomposed as $h^l(\boldsymbol{z}^l) = p^l(\boldsymbol{z}^l) – q^l(\boldsymbol{z}^l)$ where $p^l : \mathbb{R} \rightarrow \mathbb{R}$ and $q^l : \mathbb{R} \rightarrow \mathbb{R}$ are convex functions. To illustrate, as shown in Figure 4.1, the widely-used Sigmoid activation function $h_{\text{sig}}(z) = \frac{1}{1+e^{-z}}$ can be written as the difference of convex functions $p_{\text{sig}}(z) = g(z) - \frac{z}{4} - \frac{1}{2}$ and $q_{\text{sig}}(z) = g(-z) - 1$, where

$$g(z) = \begin{cases} \frac{1}{1+e^{-z}} & z \leq 0 \\ \frac{z}{4} + \frac{1}{2} & z > 0 \end{cases}.$$



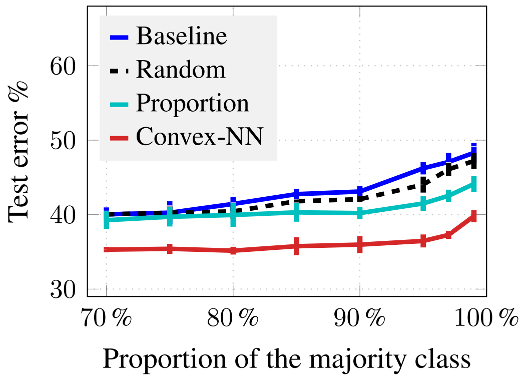


Figure 4.1: Left: Convex-decomposition of Sigmoid activation function. Sigmoid function can be cast as difference of convex functions $g_1 + g_2$ and $-g_3$. Right: Comparison of Convex-NN with hard constraint (4.9) to the existing tricks for the imbalance classification problem.

Similarly, the Softplus activation function $h_{\text{soft}}(z) = \frac{1}{\tau}\log(1+e^{\tau z})$ [169] which is a smoothed version of the ReLU function, can be decomposed as $p_{\text{soft}}(z) - q_{\text{soft}}(z)$ where $p_{\text{soft}}(z) = \frac{1}{\tau}\log(1+e^{\tau z})$ and $q_{\text{soft}}(z) = 0$.

Given the decompositions $h^l(\boldsymbol{z}^l) = p^l(\boldsymbol{z}^l) - q^l(\boldsymbol{z}^l)$, for every $l \in \mathcal{I}_{L\text{-}1}$, we can cast the constraint (4.3c) into the following equations:

$$\frac{\boldsymbol{u}^l + \boldsymbol{a}^l}{2} = p^l(\boldsymbol{z}^l) \;\wedge\; \frac{\boldsymbol{u}^l - \boldsymbol{a}^l}{2} = q^l(\boldsymbol{z}^l), \qquad \forall l \in \mathcal{I}_{L\text{-}1},$$

where for every $l \in \mathcal{I}_{L\text{-}1}$, the auxiliary variable $\boldsymbol{u}^l \in \mathbb{R}^{n_l \times m}$ accounts for $p^l(\boldsymbol{z}^l) + q^l(\boldsymbol{z}^l)$. Next, we relax the above equations to the following convex inequalities:

$$\frac{\boldsymbol{u}^l + \boldsymbol{a}^l}{2} \geq p^l(\boldsymbol{z}^l) \;\wedge\; \frac{\boldsymbol{u}^l - \boldsymbol{a}^l}{2} \geq q^l(\boldsymbol{z}^l), \qquad \forall l \in \mathcal{I}_{L\text{-}1}. \tag{4.6}$$

Definition 4.2. *For every $l \in \mathcal{I}_{L\text{-}1}$, define $\mathcal{U}^l$ to be the set of all triplets $(\boldsymbol{z}^l, \boldsymbol{a}^l, \boldsymbol{u}^l)$ that satisfy equations* (4.6).

Given the proposed convex relaxations, the relaxed problem can be formulated as an optimization problem with objective function (4.3a) and convex constraints (4.5a) – (4.5c) and (4.6). In practice, the optimal solution obtained by solving the relaxed problem may not be feasible for the original non-convex problem (4.3a) – (4.3c). To provide insight into the extent of this issue, consider a trivial in-feasible point $(\boldsymbol{w},\boldsymbol{b},\boldsymbol{z},\boldsymbol{a})$ for the problem (4.3a) – (4.3c), in which all variables except $\boldsymbol{z}^L$ are equal to zero, and $\boldsymbol{z}^L = \boldsymbol{y}$. While this solution minimizes objective function (4.3a), it cannot be considered as a meaningful solution for the original training problem; since all the weight and bias parameters are equal to zero. Next, we address this issue by incorporating a regularization term into the objective function of the relaxation.

4.4.2 Regularization

In order to enforce the equality constraints (4.3b) – (4.3c), we introduce a regularization technique that promotes feasible and meaningful solutions for the problem (4.3a) – (4.3c). To this end, consider arbitrary weights and biases $\check{\boldsymbol{w}} = \{\check{\boldsymbol{w}}^l \in \mathbb{R}^{n_{l-1}\times n_l}\}_{l\in\mathcal{I}_L}$ and $\check{\boldsymbol{b}} = \{\check{\boldsymbol{b}}^l \in \mathbb{R}^{n_l}\}_{l\in\mathcal{I}_L}$. Let $\check{\boldsymbol{a}}^0 \triangleq \boldsymbol{a}^0$ and define:

$$\check{\boldsymbol{z}}^l \triangleq \check{\boldsymbol{w}}^{l\top}\check{\boldsymbol{a}}^{l-1} + \check{\boldsymbol{b}}^l\boldsymbol{1}_m^\top, \qquad l\in\mathcal{I}_L, \tag{4.7a}$$

$$\check{\boldsymbol{a}}^l \triangleq p^l(\check{\boldsymbol{z}}^l) - q^l(\check{\boldsymbol{z}}^l), \quad \check{\boldsymbol{u}}^l \triangleq p^l(\check{\boldsymbol{z}}^l) + q^l(\check{\boldsymbol{z}}^l), \qquad l\in\mathcal{I}_{L\text{-}1}. \tag{4.7b}$$

We employ regularization terms of the form

$$\bar{R}(\boldsymbol{w},\boldsymbol{b},\boldsymbol{a},\boldsymbol{u},\boldsymbol{z},\boldsymbol{W},\boldsymbol{A}) \triangleq \sum_{l\in\mathcal{I}_L}\bar{R}_1^l(\boldsymbol{w}^l,\boldsymbol{b}^l,\boldsymbol{a}^{l-1},\boldsymbol{W}^l,\boldsymbol{A}^{l-1}) + \sum_{l\in\mathcal{I}_{L\text{-}1}}\bar{R}_2^l(\boldsymbol{a}^l,\boldsymbol{u}^l,\boldsymbol{z}^l).$$

whose first and second terms enforce the equations (4.3b) and (4.3c), respectively. Define

$$\begin{aligned}\bar{R}_1^l(\boldsymbol{w}^l,\boldsymbol{b}^l,\boldsymbol{a}^{l-1},\boldsymbol{W}^l,\boldsymbol{A}^{l-1}) &\triangleq \mu_w^l \times \operatorname{tr}\{\boldsymbol{W}^l - 2\check{\boldsymbol{w}}^l\boldsymbol{w}^{l\top} + \check{\boldsymbol{w}}^l\check{\boldsymbol{w}}^{l\top}\}\\ &+ \mu_h^l \times \operatorname{tr}\{\boldsymbol{A}^{l-1} - 2\check{\boldsymbol{a}}^{l-1}\boldsymbol{a}^{l-1\top} + \check{\boldsymbol{a}}^{l-1}\check{\boldsymbol{a}}^{l-1\top}\} + \mu_b^l \times \|\boldsymbol{b}^l - \check{\boldsymbol{b}}^l\|^2,\end{aligned}$$

for every $l\in\mathcal{I}_L$ and define

$$\begin{aligned}\bar{R}_2^l(\boldsymbol{a}^l,\boldsymbol{u}^l,\boldsymbol{z}^l) &\triangleq \mu_p^l \times \operatorname{tr}\Big\{\frac{(\boldsymbol{u}^l - \check{\boldsymbol{u}}^l) + (\boldsymbol{a}^l - \check{\boldsymbol{a}}^l)}{2}\boldsymbol{1}_{m\times n_l} - \dot{p}(\check{\boldsymbol{z}}^l)(\boldsymbol{z}^l - \check{\boldsymbol{z}}^l)\Big\}\\ &+ \mu_q^l \times \operatorname{tr}\Big\{\frac{(\boldsymbol{u}^l - \check{\boldsymbol{u}}^l) - (\boldsymbol{a}^l - \check{\boldsymbol{a}}^l)}{2}\boldsymbol{1}_{m\times n_l} - \dot{q}(\check{\boldsymbol{z}}^l)(\boldsymbol{z}^l - \check{\boldsymbol{z}}^l)\Big\},\end{aligned}$$

for every $l\in\mathcal{I}_{L\text{-}1}$, where μ_w^l, μ_h^l, μ_b^l, μ_p^l and μ_q^l are positive constants. Using the introduced regularization functions, the *regularized relaxation* problem can be formulated as:

$$\underset{\substack{\boldsymbol{w},\boldsymbol{b},\boldsymbol{z},\boldsymbol{a},\boldsymbol{u}\\ \boldsymbol{W},\boldsymbol{A}}}{\text{minimize}} \quad \|\boldsymbol{z}^{L}-\boldsymbol{y}\|_F^2+\eta\times\bar{R}(\boldsymbol{w},\boldsymbol{b},\boldsymbol{a},\boldsymbol{u},\boldsymbol{z},\boldsymbol{W},\boldsymbol{A}) \tag{4.8a}$$

$$\text{subject to} \quad (\boldsymbol{w}^{l},\boldsymbol{a}^{l\text{-}1},\boldsymbol{W}^{l},\boldsymbol{A}^{l\text{-}1})\in\mathcal{Q}^{l}, \qquad l\in\mathcal{I}_{L}, \tag{4.8b}$$

$$(\boldsymbol{z}^{l},\boldsymbol{a}^{l},\boldsymbol{u}^{l})\in\mathcal{U}^{l}, \qquad l\in\mathcal{I}_{L\text{-}1}, \tag{4.8c}$$

where the parameter $\eta > 0$ controls the importance of the regularization term. Observe that the problem (4.8a) – (4.8c) is convex and can be solved effectively using standard solvers.

The following theorem provides a sufficient condition to guarantee that the proposed regularized relaxations produce feasible points for the original non-convex problem (4.3a) – (4.3c).

Theorem 4.1. *If $\check{\eta}$ is sufficiently large, then the optimal solution $(\overset{*}{\boldsymbol{w}},\overset{*}{\boldsymbol{b}},\overset{*}{\boldsymbol{z}},\overset{*}{\boldsymbol{a}},\overset{*}{\boldsymbol{u}},\overset{*}{\boldsymbol{W}},\overset{*}{\boldsymbol{A}})$ of the convex problem (4.8a) – (4.8c) satisfies the non-convex constraints* (4.3b) *and* (4.3c)*. Moreover, $C(\overset{*}{\boldsymbol{w}},\overset{*}{\boldsymbol{b}})\leq C(\check{\boldsymbol{w}},\check{\boldsymbol{b}})$.*

Proof. See Section 4.7 for the proof. □

In the light of Theorem 4.1, the relaxation of non-convex constraints (4.3b) – (4.3c) to (4.8b) – (4.8c) is lossless if η is sufficiently large.

4.4.3 Sequential Regularized Relaxation

In this section, we propose Algorithm 2 to infer feasible and high-quality points for the non-convex problem (4.3a) – (4.3c) by solving the regularized relaxation problem (4.8a) – (4.8c) sequentially. The algorithm is terminated when the improvement of the objective value, between two consecutive round, is less than a positive value ε (e.g. $\varepsilon = 10^{-7}$). Notice that the main purpose of this work is to study and investigate the applications of convex relaxation in training neural networks and it is not our intention to compete with the gradient-based approaches.

Algorithm 2 Sequential Regularized Relaxation.

```
Input: $\check{w}$, $\check{b}$, and a fixed $\eta > 0$
1: repeat
2:     Obtain $\overset{*}{w}$, $\overset{*}{b}$, $\overset{*}{z}$, $\overset{*}{a}$, and $\overset{*}{u}$ by solving the optimization problem (4.8a) – (4.8c).
3:     $(\check{w}, \check{b}) \leftarrow (\overset{*}{w}, \overset{*}{b})$.
4: until stopping criteria is met.
Output: $\check{w}$, $\check{b}$
```

The following theorem offers a sufficient condition for the convergence of Algorithm 2 to a minimizer of the original training problem (4.2).

Theorem 4.2. *Let* $(\boldsymbol{w}_{\text{opt}}, \boldsymbol{b}_{\text{opt}})$ *be an isolated minimizer of* (4.2). *There exists* $\bar{\varepsilon} > 0$ *such that if* $\|\boldsymbol{w}_{\text{opt}} - \check{\boldsymbol{w}}\|_{\mathrm{F}} + \|\boldsymbol{b}_{\text{opt}} - \check{\boldsymbol{b}}\|_{\mathrm{F}} < \bar{\varepsilon}$ *and* η *is sufficiently large, then the sequence of weight and bias parameters generated by Algorithm 2 converges to* $(\boldsymbol{w}_{\text{opt}}, \boldsymbol{b}_{\text{opt}})$.

Proof. See Section 4.7 for the proof. □

4.5 Experiments

This section demonstrates the potential of sequential convex relaxation in imposing hard constraints on training problems. There has been a recent surge of interest in imposing task-specific constraints on the output of neural networks for improving the quality of predictions [153, 170, 154, 171]. Such constraints are often incorporated as penalty terms into the loss function. Despite the simplicity and empirical success, this strategy suffers from two major drawbacks. Firstly, adjusting relative weights between the loss function and penalty terms can be extremely challenging. Secondly, there is no guarantee that the resulting solution satisfies the hard constraints. Recently, [170] has demonstrated that replacing the soft constraints by the hard ones improves the behavioral performance of neural networks and can effectively address the aforementioned drawbacks.

We consider the problem of imbalanced classification [172, 173, 174, 175] in which the training samples are distributed highly unbalanced in different classes. The traditional classifiers exhibit poor performance in solving this problem as they tend to misclassify the minority class instances as the majority. In what follows, we present experimental results that compare the performance of Convex-NN against two commonly used tricks for the problem of imbalanced classification. Notice that, it is not the intention of this work to compete with the existing algorithms that specifically target the imbalanced classification problem, but rather to show the potential and applicability of Convex-NN on real-world problems. We use the Yeast dataset from the UCI machine learning repository [176] and subsample classes 2 and 3 to create an imbalanced binary classification problem. We train a simple neural network architecture with two fully-connected hidden layers of 6 nodes each, and Sigmoid activation function. Constraints of the form:

$$\sum_{i \in \mathcal{C}_j} \| \boldsymbol{z}^{L}_{*i} - \boldsymbol{y}_i \|_1 \leq \kappa_j, \tag{4.9}$$

are imposed to reduce the effect of unbalanced training samples, where $\boldsymbol{z}^{L}_{*i}$ denotes the i^{th} column of $\boldsymbol{z}^{L}$ and index set $\mathcal{C}_j \subseteq \mathcal{I}_m$ refers to the training samples belonging to the j^{th} class. Intuitively, the constraint (4.9) ensures that the misfit between predicted outputs and the true labels for class j does not exceed κ_j. We impose the constraint (4.9) on both majority and minority classes with $\kappa_j = 0.5|\mathcal{C}_j|$. This enables Convex-NN to automatically assign the relative weights and avoid generating a biased classifier.

Following [175], we compare our results with conventional techniques for the imbalanced classification problem: 1) Proportion method weights the training samples of each class in proportion to the inverse of the class size; 2) Random approach weights the samples based on a rectified Gaussian distribution. We run these methods on the same network architecture and employ the Adam optimizer [33] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and learning rate 0.005 to train the network. Note that these parameters are well-tuned to obtain the best performance of the Adam optimizer. The results of this experiment are demonstrated in Figure 4.1 as the value of test classification error across various imbalance ratios. The scores in Figure 4.1 are obtained by averaging 10 independent runs with random splits and random starting points, where the weights are randomly initialized using He's initializer [177] and the bias parameters are all set to zero. Observe that imposing hard constraints using Convex-NN has resulted in better test accuracy.

4.6 Conclusion

In this work, we presented a novel convexification approach, called Convex-NN, which reduces the problem of training neural networks into solving a sequence of convex programs. We theoretically proved that under certain assumptions, the proposed approach results in a convergent sequence of feasible points whose objective values monotonically improve. Convex-NN improves upon the common-practice gradient-based methods by jointly estimating the network parameters and admitting additional convex constraints. Numerical results corroborated the potential of the proposed approach on the problem of imbalanced classification.

4.7 Proofs

This section presents detailed proofs of Theorem 4.1 and 4.2. Before proceeding to the proofs, we introduce some prerequisite definitions and notations.

Definition 4.3. *For every $r \in \mathcal{I}_L$, define function s^r as*

$$\boldsymbol{s}^r(\{\boldsymbol{w}^l,\boldsymbol{b}^l\}_{l=1}^r) \triangleq \boldsymbol{w}^{r\top} h^{r\text{-}1}(\cdots h^1(\boldsymbol{w}^{1\top}\boldsymbol{x}+\boldsymbol{b}^1\mathbf{1}_m^\top)\cdots)+\boldsymbol{b}^r\mathbf{1}_m^\top,$$

which denotes the output of layer l linear transformation.

Definition 4.4. *Define function R as:*

$$R(\boldsymbol{w},\boldsymbol{b}) \triangleq \sum_{r\in\mathcal{I}_L} R_1^r(\{\boldsymbol{w}^l,\boldsymbol{b}^l\}_{l=1}^r) + \sum_{l\in\mathcal{I}_{L\text{-}1}} R_2^r(\{\boldsymbol{w}^l,\boldsymbol{b}^l\}_{l=1}^r),$$

where R_1^r for every $r \in \mathcal{I}_L$ is defined as

$$R_1^r(\{\boldsymbol{w}^l,\boldsymbol{b}^l\}_{l=1}^r) \triangleq \mu_{\boldsymbol{w}}^r \times \|\boldsymbol{w}^r - \check{\boldsymbol{w}}^r\|_{\mathrm{F}}^2 + \mu_{\boldsymbol{b}}^r \times \|\boldsymbol{b}^r - \check{\boldsymbol{b}}^r\|_{\mathrm{F}}^2 + \mu_h^r \times \|h^r(\boldsymbol{s}^r) - h^r(\check{\boldsymbol{s}}^r)\|_{\mathrm{F}}^2,$$

and R_2^r for every $r \in \mathcal{I}_{L\text{-}1}$ is formulated as

$$\begin{aligned} R_2^r(\{\boldsymbol{w}^l,\boldsymbol{b}^l\}_{l=1}^r) \triangleq\, & \mu_p^r \mathrm{tr}\big\{\big(p^r(\boldsymbol{s}^r) - p^r(\check{\boldsymbol{s}}^r)\big)\mathbf{1}_{m\times n_r} - \dot{p}^r(\check{\boldsymbol{s}}^r)(\boldsymbol{s}^r - \check{\boldsymbol{s}}^r)^\top\big\} \\ & + \mu_q^r \mathrm{tr}\big\{\big(q^r(\boldsymbol{s}^r) - q^r(\check{\boldsymbol{s}}^r)\big)\mathbf{1}_{m\times n_r} - \dot{q}^r(\check{\boldsymbol{s}}^r)(\boldsymbol{s}^r - \check{\boldsymbol{s}}^r)^\top\big\}, \end{aligned}$$

where $\boldsymbol{s}^r$ and $\check{\boldsymbol{s}}^r$ refer to $\boldsymbol{s}^r(\{\boldsymbol{w}^l,\boldsymbol{b}^l\}_{l=1}^r)$ and $\boldsymbol{s}^r(\{\check{\boldsymbol{w}}^l,\check{\boldsymbol{b}}^l\}_{l=1}^r)$, respectively.

To prove Theorem 4.1, consider a regularized formulation of the problem (4.2) as

$$\underset{\boldsymbol{w},\boldsymbol{b}}{\text{minimize}} \quad C(\boldsymbol{w},\boldsymbol{b}) + \eta \times R(\boldsymbol{w},\boldsymbol{b}). \tag{4.10}$$

It can be easily verified that problems (4.10) and (4.2) are equivalent if $\eta = 0$. In what follows, we state a lemma to show that for appropriate choices of η, the optimal solution of (4.10) can be obtained by solving convex program (4.8a) – (4.8c) whose solution approximates the optimal solution of (4.3a) – (4.3c).

Consider the following reformulation of (4.3a) – (4.3c):

$$\begin{aligned} \underset{\substack{\boldsymbol{w},\boldsymbol{b},\boldsymbol{u},\boldsymbol{a},\boldsymbol{z}\\ \boldsymbol{W},\boldsymbol{A}}}{\text{minimize}} \quad & \|\boldsymbol{z}^L - \boldsymbol{y}\|_{\mathrm{F}}^2 + \eta \times \bar{R}(\boldsymbol{w},\boldsymbol{b},\boldsymbol{a},\boldsymbol{u},\boldsymbol{z},\boldsymbol{W},\boldsymbol{A}) && && (4.11\text{a})\\ \text{subject to} \quad & \begin{bmatrix} \boldsymbol{W}^l & \boldsymbol{z}^l - \boldsymbol{b}^l\mathbf{1}_m^\top \\ (\boldsymbol{z}^l - \boldsymbol{b}^l\mathbf{1}_m^\top)^\top & \boldsymbol{A}^{l\text{-}1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{w}^{l\top} \\ \boldsymbol{a}^{l\text{-}1\top} \end{bmatrix} \begin{bmatrix} \boldsymbol{w}^l & \boldsymbol{a}^{l\text{-}1} \end{bmatrix} && l \in \mathcal{I}_L, && (4.11\text{b})\\ & \frac{\boldsymbol{u}^l + \boldsymbol{a}^l}{2} = p^l(\boldsymbol{z}^l) && l \in \mathcal{I}_L, && (4.11\text{c})\\ & \frac{\boldsymbol{u}^l - \boldsymbol{a}^l}{2} = q^l(\boldsymbol{z}^l) && l \in \mathcal{I}_L, && (4.11\text{d}) \end{aligned}$$

where matrices $\boldsymbol{W}^l$ and $\boldsymbol{A}^{l\text{-}1}$, respectively, account for $\boldsymbol{w}^{l\top}\boldsymbol{w}^l$ and $\boldsymbol{a}^{l\text{-}1\top}\boldsymbol{a}^{l\text{-}1}$, and functions p^L and q^L are chosen such that h^L becomes identity function. Notice that if $\eta = 0$, the problems

(4.3a) – (4.3c) and (4.11a) – (4.11d) are equivalent and have the same minimizers. For every $l \in \mathcal{I}_L$, define matrices

$$\boldsymbol{\Lambda}^l \triangleq \begin{bmatrix} \boldsymbol{\lambda}_{\boldsymbol{W}}^l & \boldsymbol{\lambda}_{\boldsymbol{z}}^l \\ \boldsymbol{\lambda}_{\boldsymbol{z}}^{l\top} & \boldsymbol{\lambda}_{\boldsymbol{A}}^{l\text{-}1} \end{bmatrix} \in \mathbb{S}_{n_l+m}, \;\; \boldsymbol{\lambda}_p^l \in \mathbb{R}^{n_l \times m}, \;\; \boldsymbol{\lambda}_q^l \in \mathbb{R}^{n_l \times m}$$

as the dual variables associated with (4.11b), (4.11c), and (4.11d), respectively. For ease of notation, let $\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{A}}$, $\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{W}}$, $\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{z}}$, $\overset{*}{\boldsymbol{\lambda}}_p$, and $\overset{*}{\boldsymbol{\lambda}}_q$ denote the set of all $\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{A}}^l$, $\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{W}}^l$, $\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{z}}^l$, $\overset{*}{\boldsymbol{\lambda}}_p^l$, $\overset{*}{\boldsymbol{\lambda}}_q^l$, for every $l \in \mathcal{I}_L$, respectively. By augmenting constraints (4.11b) – (4.11d) to the objective function (4.11a), we can verify that for every $l \in \mathcal{I}_L$, $\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{A}}^l = \eta\mu_h^l \boldsymbol{I}$, $\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{W}}^l = \eta\mu_w^l \boldsymbol{I}$, and the remaining dual variables for an arbitrary minimizer $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}, \overset{*}{\boldsymbol{u}}, \overset{*}{\boldsymbol{a}}, \overset{*}{\boldsymbol{z}}, \overset{*}{\boldsymbol{W}}, \overset{*}{\boldsymbol{A}})$ can be computed recursively as follows:

$$\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{z}}^l = \;\; \overset{*}{\boldsymbol{w}}^{l+1\top} \overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{z}}^{l+1} \circ \dot{h}^l(\overset{*}{\boldsymbol{z}}^l) + \overset{*}{\boldsymbol{c}}_{\dot{h}}^l + \overset{*}{\boldsymbol{c}}_p^l + \overset{*}{\boldsymbol{c}}_q^l, \tag{4.12a}$$

$$\overset{*}{\boldsymbol{\lambda}}_p^l = \;\; \overset{*}{\boldsymbol{w}}^{l+1\top} \overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{z}}^{l+1} + \overset{*}{\boldsymbol{c}}_h^l + \eta\mu_p^l, \tag{4.12b}$$

$$\overset{*}{\boldsymbol{\lambda}}_q^l = -\overset{*}{\boldsymbol{w}}^{l+1\top} \overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{z}}^{l+1} - \overset{*}{\boldsymbol{c}}_h^l + \eta\mu_q^l, \tag{4.12c}$$

$$\overset{*}{\boldsymbol{c}}_p^l = \;\; \eta\mu_p^l\big(\dot{p}^l(\overset{*}{\boldsymbol{z}}^l) - \dot{p}^l(\check{\boldsymbol{z}}^l)\big), \tag{4.12d}$$

$$\overset{*}{\boldsymbol{c}}_q^l = \;\; \eta\mu_q^l\big(\dot{q}^l(\overset{*}{\boldsymbol{z}}^l) - \dot{q}^l(\check{\boldsymbol{z}}^l)\big), \tag{4.12e}$$

$$\overset{*}{\boldsymbol{c}}_h^l = \; 2\eta\mu_h^l\big(h^l(\overset{*}{\boldsymbol{z}}^l) - h^l(\check{\boldsymbol{z}}^l)\big), \tag{4.12f}$$

$$\overset{*}{\boldsymbol{c}}_{\dot{h}}^l = \;\; \overset{*}{\boldsymbol{c}}_h^l \circ \dot{h}^l(\overset{*}{\boldsymbol{z}}^l), \tag{4.12g}$$

for every $l \in \mathcal{I}_{L\text{-}1}$ and $\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{z}}^L = 2(\overset{*}{\boldsymbol{z}}^L - \boldsymbol{y}) + \overset{*}{\boldsymbol{c}}_{\dot{h}}^L + \overset{*}{\boldsymbol{c}}_p^L + \overset{*}{\boldsymbol{c}}_q^L$, $\overset{*}{\boldsymbol{\lambda}}_p^L = \overset{*}{\boldsymbol{c}}_h^L + \eta\mu_p^L$, and $\overset{*}{\boldsymbol{\lambda}}_q^L = -\overset{*}{\boldsymbol{c}}_h^L + \eta\mu_q^L$. In what follows, we aim to show that the optimal dual solutions of (4.11a) – (4.11d) can serve as certificate of optimality for the solutions of relaxed problem (4.8a) – (4.8c).

Lemma 4.1. *Let $(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})$ be an initial point. For every layer $l \in \mathcal{I}_L$, there exist continuous functions $\theta_{\boldsymbol{z}}^l(\boldsymbol{w}, \boldsymbol{b})$ and $\theta_{pq}^l(\boldsymbol{w}, \boldsymbol{b})$ such that the inequalities*

$$\frac{1}{\eta}\|\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{z}}^l\|_{\mathrm{F}} \leq \frac{1}{\sqrt{\eta}}\,\theta_{\boldsymbol{z}}^l(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}), \qquad \forall l \in \mathcal{I}_L, \tag{4.13a}$$

$$\frac{1}{\eta}\|\overset{*}{\boldsymbol{w}}^{l+1\top}\overset{*}{\boldsymbol{\lambda}}_{\boldsymbol{z}}^{l+1} + \overset{*}{\boldsymbol{c}}_h^l\|_{\max} \leq \frac{1}{\sqrt{\eta}}\,\theta_{pq}^l(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}), \qquad \forall l \in \mathcal{I}_{L\text{-}1}. \tag{4.13b}$$

are satisfied for every minimizer $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}, \overset{*}{\boldsymbol{u}}, \overset{*}{\boldsymbol{a}}, \overset{*}{\boldsymbol{z}}, \overset{*}{\boldsymbol{W}}, \overset{*}{\boldsymbol{A}})$ *of (4.11a) – (4.11d) and every* $\eta > 1$.

Proof. Refer to the supplementary materials. $\square$

Proof of Theorem 4.1. Consider an arbitrary minimizer $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}, \overset{*}{\boldsymbol{u}}, \overset{*}{\boldsymbol{a}}, \overset{*}{\boldsymbol{z}}, \overset{*}{\boldsymbol{W}}, \overset{*}{\boldsymbol{A}})$ of the problem (4.11a) – (4.11d) with the corresponding Lagrange multiplier $(\overset{*}{\boldsymbol{\lambda}}_{W}, \overset{*}{\boldsymbol{\lambda}}_{A}, \overset{*}{\boldsymbol{\lambda}}_{z}, \overset{*}{\boldsymbol{\lambda}}_{p}, \overset{*}{\boldsymbol{\lambda}}_{q})$. If

$$\eta > (\mu^{l}_{p})^{-2}\theta^{l}_{pq}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})^{2}, \qquad \eta > (\mu^{l}_{q})^{-2}\theta^{l}_{pq}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})^{2},$$
$$\eta > m(\mu^{l}_{h})^{-2}\, \theta^{l}_{z}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})^{2}, \qquad \eta > n_{l}(\mu^{l}_{w})^{-2}\, \theta^{l}_{z}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})^{2},$$

are satisfied for every $l \in \mathcal{I}_{L}$, then equations (4.12b) – (4.12c) together with the results of Lemma 4.1 implies that

$$\mu^{l}_{p} - \frac{\overset{*}{\boldsymbol{\lambda}}{}^{l}_{p}}{\eta} \leq \frac{\|\overset{*}{\boldsymbol{w}}{}^{l+1^{\top}}\overset{*}{\boldsymbol{\lambda}}{}^{l+1}_{\boldsymbol{z}} + \overset{*}{\boldsymbol{c}}{}^{l}_{h}\|_{\max}}{\eta} \leq \frac{\theta^{r}_{pq}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})}{\sqrt{\eta}} < \mu^{l}_{p}, \tag{4.15a}$$

$$\mu^{l}_{q} - \frac{\overset{*}{\boldsymbol{\lambda}}{}^{l}_{q}}{\eta} \leq \frac{\|\overset{*}{\boldsymbol{w}}{}^{l+1^{\top}}\overset{*}{\boldsymbol{\lambda}}{}^{l+1}_{\boldsymbol{z}} + \overset{*}{\boldsymbol{c}}{}^{l}_{h}\|_{\max}}{\eta} \leq \frac{\theta^{r}_{pq}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})}{\sqrt{\eta}} < \mu^{l}_{q}, \tag{4.15b}$$

$$\frac{\|\overset{*}{\boldsymbol{\lambda}}{}^{l^{\top}}_{\boldsymbol{z}}\|_{1}}{\eta} \leq \frac{\|\overset{*}{\boldsymbol{\lambda}}{}^{l}_{\boldsymbol{z}}\|_{\mathrm{F}}\sqrt{m}}{\eta} \leq \sqrt{\frac{m}{\eta}}\; \theta^{l}_{\boldsymbol{z}}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) < \mu^{l}_{h}, \tag{4.15c}$$

$$\frac{\|\overset{*}{\boldsymbol{\lambda}}{}^{l}_{\boldsymbol{z}}\|_{1}}{\eta} \leq \frac{\|\overset{*}{\boldsymbol{\lambda}}{}^{l}_{\boldsymbol{z}}\|_{\mathrm{F}}\sqrt{n_{l}}}{\eta} \leq \sqrt{\frac{n_{l}}{\eta}}\; \theta^{l}_{\boldsymbol{z}}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) < \mu^{l}_{w}. \tag{4.15d}$$

According to the inequalities (4.15a) and (4.15b), we can conclude $\overset{*}{\boldsymbol{\lambda}}{}^{l}_{p} > 0$ and $\overset{*}{\boldsymbol{\lambda}}{}^{l}_{q} > 0$, for every $l \in \mathcal{I}_{L}$. This implies that the relaxation of the non-convex constraints (4.11c) and (4.11d) into the convex constraints (4.8c) is lossless. Moreover, according to (4.15c) and (4.15d), for every $l \in \mathcal{I}_{L}$, matrix $\overset{*}{\boldsymbol{\Lambda}}{}^{l}$ is diagonally-dominant and hence the non-convex constraint (4.11b) can be equivalently relaxed to

$$\begin{bmatrix} \boldsymbol{W}^{l} & \boldsymbol{z}^{l} - \boldsymbol{b}^{l}\boldsymbol{1}^{\top}_{m} \\ (\boldsymbol{z}^{l} - \boldsymbol{b}^{l}\boldsymbol{1}^{\top}_{m})^{\top} & \boldsymbol{A}^{l\text{-}1} \end{bmatrix} - \begin{bmatrix} \boldsymbol{w}^{l^{\top}} \\ \boldsymbol{a}^{l\text{-}1^{\top}} \end{bmatrix} \begin{bmatrix} \boldsymbol{w}^{l} & \boldsymbol{a}^{l\text{-}1} \end{bmatrix} \in \mathcal{DD}^{*}_{n_{l}+m}, \tag{4.16}$$

where $\mathcal{DD}^{*}_{n_{l}+m}$ denotes the dual cone of the symmetric diagonally-dominant matrices of size $n_{l} + m$. Observe that constraint (4.16) is equivalent to the convex constraint (4.8b). Therefore, optimal solution $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}, \overset{*}{\boldsymbol{u}}, \overset{*}{\boldsymbol{a}}, \overset{*}{\boldsymbol{z}}, \overset{*}{\boldsymbol{W}}, \overset{*}{\boldsymbol{A}})$ can be considered as a minimizer of the problem (4.8a) – (4.8c), as well. In addition, due to the convexity of (4.8a) – (4.8c), point $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}, \overset{*}{\boldsymbol{u}}, \overset{*}{\boldsymbol{a}}, \overset{*}{\boldsymbol{z}}, \overset{*}{\boldsymbol{W}}, \overset{*}{\boldsymbol{A}})$ is a globally optimal solution and the following relations are valid

$$\begin{aligned} C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) &\leq \|\overset{*}{\boldsymbol{z}}{}^{L} - \boldsymbol{y}\|_{\mathrm{F}}^{2} + \eta\, \bar{R}(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}, \overset{*}{\boldsymbol{z}}, \overset{*}{\boldsymbol{a}}, \overset{*}{\boldsymbol{u}}, \overset{*}{\boldsymbol{W}}, \overset{*}{\boldsymbol{A}}) \\ &\leq \|\check{\boldsymbol{z}}^{L} - \boldsymbol{y}\|_{\mathrm{F}}^{2} + \eta\, \bar{R}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}, \check{\boldsymbol{z}}, \check{\boldsymbol{a}}, \check{\boldsymbol{u}}, \check{\boldsymbol{W}}, \check{\boldsymbol{A}}) \\ &= \|\check{\boldsymbol{z}}^{L} - \boldsymbol{y}\|_{\mathrm{F}}^{2} = C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}), \end{aligned}$$

which completes the proof. $\square$

Proof of Theorem 4.2. For every δ, $\varepsilon > 0$, define set $\mathcal{A}_{\delta}$ as the connected component of $\{(\boldsymbol{w}, \boldsymbol{b}) \mid C(\boldsymbol{w}, \boldsymbol{b}) < C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) + \delta\}$ that contains $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})$ and define $\mathcal{B}_{\varepsilon} \triangleq \{(\boldsymbol{w}, \boldsymbol{b}) \mid \|\boldsymbol{w} - \overset{*}{\boldsymbol{w}}\|_{\mathrm{F}} + \|\boldsymbol{b} - \overset{*}{\boldsymbol{b}}\|_{\mathrm{F}} \leq \varepsilon\}$. Due to the fact that $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})$ is an isolated local minimizer, there exists a sufficiently small $\bar{\delta}$ such that $\mathcal{A}_{\bar{\delta}}$ contains no local minimizer other than $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})$. Since C is a continous function on its domain, $\mathcal{A}_{\bar{\delta}}$ is an open set and there exists $\bar{\varepsilon} > 0$ such that $\mathcal{B}_{\bar{\varepsilon}} \subseteq \mathcal{A}_{\bar{\delta}}$. Define $\bar{\eta}^{l}_{pq}(\boldsymbol{w}, \boldsymbol{b}) = \max\{(\mu^{l}_{p})^{-2}, (\mu^{l}_{q})^{-2}\} \theta^{l}_{pq}(\boldsymbol{w}, \boldsymbol{b})^{2}$, $\bar{\eta}^{l}_{z}(\boldsymbol{w}, \boldsymbol{b}) = \max\{m(\mu^{l}_{h})^{-2}, n_{l}(\mu^{l}_{w})^{-2}\} \theta^{l}_{\boldsymbol{z}}(\boldsymbol{w}, \boldsymbol{b})^{2}$, and

$$\eta^{l} \triangleq \max_{(\boldsymbol{w}, \boldsymbol{b}) \in \mathcal{B}_{\bar{\varepsilon}}} \{\bar{\eta}^{l}_{pq}(\boldsymbol{w}, \boldsymbol{b}), \bar{\eta}^{l}_{z}(\boldsymbol{w}, \boldsymbol{b})\},$$

whose existence is concluded from the compactness of set $\mathcal{B}_{\bar{\varepsilon}}$ and the continuity of functions θ^{l}_{pq} and $\theta^{l}_{\boldsymbol{z}}$ for every $l \in \mathcal{I}_{L}$. In addition, define $\eta^{\max} \triangleq \max\{1, \eta^{1}, \ldots, \eta^{L}\}$. According to Theorem 4.1, if $(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) \in \mathcal{B}_{\bar{\varepsilon}}$ and $\eta > \eta^{\max}$ then the optimal point $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}, \overset{*}{\boldsymbol{z}}, \overset{*}{\boldsymbol{a}}, \overset{*}{\boldsymbol{u}}, \overset{*}{\boldsymbol{W}}, \overset{*}{\boldsymbol{A}})$ of the convex problem (4.11a) – (4.11d) satisfies the non-convex constraints (4.3b) and (4.3c) and $C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) \leq C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})$ which implies that $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) \in \mathcal{B}_{\bar{\varepsilon}}$. Given that, if the initial point belongs to $\mathcal{B}_{\bar{\varepsilon}}$ and η is sufficiently large, Algorithm 2 generates a sequence of points whose objective values monotonically decreases and they are all within set $\mathcal{B}_{\bar{\varepsilon}}$. Since the sequence of objective values is lower-bounded by $C(\check{\boldsymbol{w}}_{\text{opt}}, \check{\boldsymbol{b}}_{\text{opt}})$, it is guaranteed to be convergent. Hence, the following relations are valid at every round of the algorithm:

$$\begin{aligned}\sum_{l\in\mathcal{I}_L} \mu_w^l \|\overset{*}{\boldsymbol{w}}^l - \check{\boldsymbol{w}}^l\|_{\text{F}}^2 + \mu_b^l \|\overset{*}{\boldsymbol{b}}^l - \check{\boldsymbol{b}}^l\|_{\text{F}}^2 &\le R(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) \\ &= \frac{C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) + \eta\, R(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) - C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})}{\eta} \\ &\le \frac{C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) + \eta\, R(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) - C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})}{\eta} \\ &= \frac{C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) - C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})}{\eta} \to 0\end{aligned}$$

which indicates that the sequence of weights and biases generated by Algorithm 2 is Cauchy and it converges to a point $(\bar{\boldsymbol{w}}, \bar{\boldsymbol{b}}) \in \mathcal{B}_{\bar{\varepsilon}}$. Moreover, due to optimality of every generated pair $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})$ for the problem (4.10), we have:

$$\begin{aligned}&\nabla C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) + \eta \nabla \sum_{l\in\mathcal{I}_L} \mu_w^l \|\overset{*}{\boldsymbol{w}}^l - \check{\boldsymbol{w}}^l\|_{\text{F}}^2 + \mu_b^l \|\overset{*}{\boldsymbol{b}}^l - \check{\boldsymbol{b}}^l\|_{\text{F}}^2 = \boldsymbol{0} \\ \Rightarrow &\left\|\nabla C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})\right\|_{\text{F}} \le 2\eta \sum_{l\in\mathcal{I}_L} \mu_w^l \|\overset{*}{\boldsymbol{w}}^l - \check{\boldsymbol{w}}^l\|_{\text{F}} + \mu_b^l \|\overset{*}{\boldsymbol{b}}^l - \check{\boldsymbol{b}}^l\|_{\text{F}}.\end{aligned}$$

As a result, the gradient of the original loss function C at $(\bar{\boldsymbol{w}}, \bar{\boldsymbol{b}})$ is equal to zero and hence it is a minimizer of the loss function C. Given the fact hat pointt$(\bar{\boldsymbol{w}}, \bar{\boldsymbol{b}})$ belongs to $\mathcal{A}_{\bar{\delta}}$, we can conclude that $\bar{\boldsymbol{w}} = \boldsymbol{w}_{\text{opt}}$ and $\bar{\boldsymbol{b}} = \boldsymbol{b}_{\text{opt}}$. □

Acknowledgments

The authors thank Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration in support of this work.

4.8 Supplementary Materials

This supplementary material gives a detailed proof of Lemma 4.1 which is discussed in Section 4.7. Before proceeding to the proofs, we introduce a prerequisite definition.

Definition 4.5. *For every layer $l \in \mathcal{I}_L$, let strongly convex and Lipschitz gradient functions $p^l : \mathbb{R} \to \mathbb{R}$ and $q^l : \mathbb{R} \to \mathbb{R}$ decompose the activation function h^l as $h^l(x) = p^l(x) - q^l(x)$. The convexity of p and q implies that there exist $\sigma_p^l > 0$ and $\sigma_q^l > 0$ such that*

$$\forall x,y \in \mathrm{dom}(p^l), \quad p^l(y) - p^l(x) - \dot{p}^l(x)(y-x) \geq \sigma_p^l (y-x)^2,$$

$$\forall x,y \in \mathrm{dom}(q^l), \quad q^l(y) - q^l(x) - \dot{q}^l(x)(y-x) \geq \sigma_q^l (y-x)^2,$$

and the Lipschitz continuous gradient assumption states that there are $\gamma_p^l > 0$ and $\gamma_q^l > 0$ such that the following inequalities hold:

$$\forall x,y \in \mathrm{dom}(\dot{p}^l), \quad |\dot{p}^l(x) - \dot{p}^l(y)| \leq \gamma_p^l |x-y|,$$

$$\forall x,y \in \mathrm{dom}(\dot{q}^l), \quad |\dot{q}^l(x) - \dot{q}^l(y)| \leq \gamma_q^l |x-y|.$$

Lemma 4.2. *Let $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})$ be an optimal solution of the problem* (4.10). *The following inequality holds:*

$$\sum_{l \in \mathcal{I}_L} \mu_w^l \times \|\overset{*}{\boldsymbol{w}}^l - \check{\boldsymbol{w}}^l\|_{\mathrm{F}}^2 + \mu_b^l \times \|\overset{*}{\boldsymbol{b}}^l - \check{\boldsymbol{b}}^l\|_{\mathrm{F}}^2 \leq \frac{C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})}{\eta}. \tag{4.18}$$

Proof. Follows from Definitions 4.4 and 4.5, for every $l \in \mathcal{I}_L$, function R is non-negative and $R(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) = 0$. Therefore, the optimality of $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})$ implies that

$$\begin{aligned} C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) &\geq C(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) + \eta\, R(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) \geq \eta\, R(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) \\ \Rightarrow \frac{C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})}{\eta} &\geq R(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}) \end{aligned} \tag{4.19}$$

Additionally, according to Definition 4.4 we have

$$\sum_{l \in \mathcal{I}_L} \mu_w^l \times \|\overset{*}{\boldsymbol{w}}^l - \check{\boldsymbol{w}}^l\|_{\mathrm{F}}^2 + \mu_b^l \times \|\overset{*}{\boldsymbol{b}}^l - \check{\boldsymbol{b}}^l\|_{\mathrm{F}}^2 \leq R(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}). \tag{4.20}$$

Given the relations (4.19) and (4.20), we can conclude that (4.18) holds true for point $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}})$. □

Lemma 4.3. *Let $(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})$ be an initial point. There exists a continuous function $\phi(\boldsymbol{w}, \boldsymbol{b})$ such that the following inequality holds true for every $\eta > 0$ and every minimizer of $(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}, \overset{*}{\boldsymbol{u}}, \overset{*}{\boldsymbol{a}}, \overset{*}{\boldsymbol{z}}, \overset{*}{\boldsymbol{W}}, \overset{*}{\boldsymbol{A}})$ of the problem (4.11a)–(4.11d):*

$$\frac{1}{\eta}\Big(\sum_{l \in \mathcal{I}_L} \|\overset{*}{\boldsymbol{c}}{}_p^l\|_{\mathrm{F}} + \|\overset{*}{\boldsymbol{c}}{}_q^l\|_{\mathrm{F}} + \|\overset{*}{\boldsymbol{c}}{}_h^l\|_{\mathrm{F}} + \|\overset{*}{\boldsymbol{c}}{}_{\hat{h}}^l\|_{\mathrm{F}}\Big) \leq \frac{\phi(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})}{\sqrt{\eta}}. \tag{4.21}$$

where $\{(\overset{*}{\boldsymbol{c}}_p, \overset{*}{\boldsymbol{c}}_q, \overset{*}{\boldsymbol{c}}_h, \overset{*}{\boldsymbol{c}}_{\dot{h}})\}_{l\in\mathcal{I}_L}$ *are given by* (4.12a) – (4.12g).

Proof. The Lipschitz continuity of function h^l implies that there exists constant $g_h^l \in \mathbb{R}$ such that $|\dot{h}^l(\boldsymbol{z}^l)| \le |g_h^l|$, for every $\boldsymbol{z}$ belonging to the feasible set of the problem (4.11a) – (4.11d). Given that, we can upper-bound $\|\overset{*}{\boldsymbol{c}}{}_h^l\|_{\mathrm{F}} + \|\overset{*}{\boldsymbol{c}}{}_{\dot{h}}^l\|_{\mathrm{F}}$ by $(1+|g_h^l|)\|\overset{*}{\boldsymbol{c}}{}_h^l\|_{\mathrm{F}}$. Define $\alpha_p^l \triangleq (\gamma_p^l)^{2}(\mu_p^l)^{-1}\sigma_p^l$, $\alpha_q^l \triangleq (\gamma_q^l)^{2}(\mu_q^l)^{-1}\sigma_q^l$, $\alpha_h^l \triangleq (\mu_h^l)^{-1}\big(2(1+|g_h^l|)\big)^{-2}$, and $\underline{\alpha} \triangleq \min\{\alpha_p^1,\cdots,\alpha_p^L,\alpha_q^1,\cdots,\alpha_q^L,(1+|g_h^1|)^{-1}\alpha_h^1,\cdots,(1+|g_h^L|)^{-1}\alpha_h^L\}$. Using the defined parameters, it can be observed that:

$$\begin{aligned}&\frac{\underline{\alpha}}{\eta}\Big(\sum_{l\in\mathcal{I}_L}\|\overset{*}{\boldsymbol{c}}{}_p^l\|_{\mathrm{F}} + \|\overset{*}{\boldsymbol{c}}{}_q^l\|_{\mathrm{F}} + (1+|g_h^l|)\|\overset{*}{\boldsymbol{c}}{}_h^l\|_{\mathrm{F}}\Big)\\ &\le \frac{1}{\eta}\Big(\sum_{l\in\mathcal{I}_L}\alpha_p^l\|\overset{*}{\boldsymbol{c}}{}_p^l\|_{\mathrm{F}} + \alpha_q^l\|\overset{*}{\boldsymbol{c}}{}_q^l\|_{\mathrm{F}} + \alpha_h^l\|\overset{*}{\boldsymbol{c}}{}_h^l\|_{\mathrm{F}}\Big).\end{aligned}$$

Given the above equation, we can use the Cauchy-Schwarz inequality to write the following relations:

$$\begin{aligned}&\frac{1}{\eta^2}\Big(\sum_{l\in\mathcal{I}_L}\alpha_p^l\|\overset{*}{\boldsymbol{c}}{}_p^l\|_{\mathrm{F}} + \alpha_q^l\|\overset{*}{\boldsymbol{c}}{}_q^l\|_{\mathrm{F}} + \alpha_h^l\|\overset{*}{\boldsymbol{c}}{}_h^l\|_{\mathrm{F}}\Big)^2\\ &\le \frac{1}{\eta^2}\Big(\sum_{l\in\mathcal{I}_L}\alpha_p^l\|\overset{*}{\boldsymbol{c}}{}_p^l\|_{\mathrm{F}}^2 + \alpha_q^l\|\overset{*}{\boldsymbol{c}}{}_q^l\|_{\mathrm{F}}^2 + \alpha_h^l\|\overset{*}{\boldsymbol{c}}{}_h^l\|_{\mathrm{F}}^2\Big)\Big(\sum_{l\in\mathcal{I}_L}\alpha_p^l + \alpha_q^l + \alpha_h^l\Big)\\ &\le \bar{R}\times\Big(\sum_{l\in\mathcal{I}_L}\alpha_p^l + \alpha_q^l + \alpha_h^l\Big),\end{aligned}$$

where $\bar{R}$ refers to $\bar{R}(\overset{*}{\boldsymbol{w}}, \overset{*}{\boldsymbol{b}}, \overset{*}{\boldsymbol{a}}, \overset{*}{\boldsymbol{u}}, \overset{*}{\boldsymbol{z}}, \overset{*}{\boldsymbol{W}}, \overset{*}{\boldsymbol{A}})$, and is upperbounded by

$$\begin{aligned}&\|\overset{*}{\boldsymbol{z}}{}^L - \boldsymbol{y}\|_{\mathrm{F}}^2 + \eta\,\bar{R} \le C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})\\ &\qquad\Rightarrow \bar{R} \le \frac{C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})}{\eta}.\end{aligned}$$

Using the above upperbound, we can conclude that (4.21) holds true

$$\phi(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) = \frac{C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})^{\frac{1}{2}}}{\underline{\alpha}}\Big(\sum_{l\in\mathcal{I}_L}\alpha_p^l + \alpha_q^l + \alpha_h^l\Big)^{\frac{1}{2}}.$$

This completes the proof. □

Proof of Lemma 4.1 – (4.13a). We use backward induction to prove the existence of function $\theta_{\boldsymbol{z}}^l(\boldsymbol{w}, \boldsymbol{b})$ for every $l \in \mathcal{I}_L$. For the case $l = L$, define function $\theta_{\boldsymbol{z}}^L(\boldsymbol{w}, \boldsymbol{b})$ as

$$\theta_{\boldsymbol{z}}^L(\boldsymbol{w}, \boldsymbol{b}) \triangleq 2\|\check{\boldsymbol{z}}^L - \boldsymbol{y}\|_{\mathrm{F}} + \phi(\boldsymbol{w}, \boldsymbol{b}). \tag{4.22}$$

For dual variable $\overset{*}{\boldsymbol{\lambda}}_z^L$, we have

$$\begin{aligned}\frac{1}{\eta}\|\overset{*}{\boldsymbol{\lambda}}_z^L\|_{\mathrm{F}} &= \frac{1}{\eta}\big\|2(\overset{*}{\boldsymbol{z}}^L - \boldsymbol{y}) + \overset{*}{\boldsymbol{c}}_p^L + \overset{*}{\boldsymbol{c}}_q^L + \overset{*}{\boldsymbol{c}}_h^L\big\|_{\mathrm{F}} \\ &\leq \frac{2}{\eta}\|\overset{*}{\boldsymbol{z}}^L - \boldsymbol{y}\|_{\mathrm{F}} + \frac{1}{\eta}\|\overset{*}{\boldsymbol{c}}_p^L + \overset{*}{\boldsymbol{c}}_q^L + \overset{*}{\boldsymbol{c}}_h^L\|_{\mathrm{F}}. \end{aligned} \tag{4.23}$$

Additionally, the optimality of $\overset{*}{\boldsymbol{z}}$ implies that

$$\|\overset{*}{\boldsymbol{z}}^L - \boldsymbol{y}\|_{\mathrm{F}}^2 \leq \|\overset{*}{\boldsymbol{z}}^L - \boldsymbol{y}\|_{\mathrm{F}}^2 + \eta\,\bar{R} \leq \|\check{\boldsymbol{z}}^L - \boldsymbol{y}\|_{\mathrm{F}}^2. \tag{4.24}$$

Putting inequalities (4.23), (4.24), and Lemma 4.3 together, we can conclude that

$$\begin{aligned}\frac{1}{\eta}\|\overset{*}{\boldsymbol{\lambda}}_z^L\|_{\mathrm{F}} &\leq \frac{2}{\eta}\|\check{\boldsymbol{z}}^L - \boldsymbol{y}\|_{\mathrm{F}} + \frac{1}{\sqrt{\eta}}\phi(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) \\ &\leq \frac{1}{\sqrt{\eta}}\big(2\|\check{\boldsymbol{z}}^L - \boldsymbol{y}\|_{\mathrm{F}} + \phi(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})\big) = \frac{1}{\sqrt{\eta}}\,\theta_z^L(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}).\end{aligned}$$

Next, for every $l \in \mathcal{I}_{L\text{-}1}$, define function $\theta_z^{l\text{-}1}(\boldsymbol{w}, \boldsymbol{b})$ as

$$\theta_z^{l\text{-}1}(\boldsymbol{w}, \boldsymbol{b}) \triangleq \big((\mu_w^l)^{\text{-}\frac{1}{2}} C(\boldsymbol{w}, \boldsymbol{b}) + \|\check{\boldsymbol{w}}^l\|_{\mathrm{F}}\big)|g_h^{l\text{-}1}|\theta_z^l(\boldsymbol{w}, \boldsymbol{b}) + \phi(\boldsymbol{w}, \boldsymbol{b}).$$

We can conclude the following relations from the equation (4.12a):

$$\begin{aligned}\frac{1}{\eta}\|\overset{*}{\boldsymbol{\lambda}}_z^{l\text{-}1}\|_{\mathrm{F}} &= \frac{1}{\eta}\|\overset{*}{\boldsymbol{w}}^{l\top}\overset{*}{\boldsymbol{\lambda}}_z^l \circ \dot{h}^{l\text{-}1}(\overset{*}{\boldsymbol{z}}^{l\text{-}1}) + \overset{*}{\boldsymbol{c}}_p^{l\text{-}1} + \overset{*}{\boldsymbol{c}}_q^{l\text{-}1} + \overset{*}{\boldsymbol{c}}_h^{l\text{-}1}\|_{\mathrm{F}} \\ &\leq \frac{1}{\eta}\|\overset{*}{\boldsymbol{w}}^{l\top}\overset{*}{\boldsymbol{\lambda}}_z^l \circ \dot{h}^{l\text{-}1}(\overset{*}{\boldsymbol{z}}^{l\text{-}1})\|_{\mathrm{F}} + \frac{1}{\eta}\|\overset{*}{\boldsymbol{c}}_p^{l\text{-}1} + \overset{*}{\boldsymbol{c}}_q^{l\text{-}1} + \overset{*}{\boldsymbol{c}}_h^{l\text{-}1}\|_{\mathrm{F}} \\ &\leq \frac{|g_h^{l\text{-}1}|}{\eta}\|\overset{*}{\boldsymbol{w}}^{l\top}\overset{*}{\boldsymbol{\lambda}}_z^l\|_{\mathrm{F}} + \frac{1}{\sqrt{\eta}}\phi(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) \\ &\leq \frac{|g_h^{l\text{-}1}|}{\eta}\big(\|\overset{*}{\boldsymbol{w}}^l - \check{\boldsymbol{w}}^l\|_{\mathrm{F}} + \|\check{\boldsymbol{w}}^l\|_{\mathrm{F}}\big)\|\overset{*}{\boldsymbol{\lambda}}_z^l\|_{\mathrm{F}} + \frac{1}{\sqrt{\eta}}\phi(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}),\end{aligned}$$

where $g_h^{l\text{-}1}$ denotes the Lipschitz constant of function $h^{l\text{-}1}$. Using the results of Lemma 4.2, we can easily verify that

$$\|\overset{*}{\boldsymbol{w}}^l - \check{\boldsymbol{w}}^l\|_{\mathrm{F}} \leq (\eta\mu_w^l)^{\text{-}\frac{1}{2}} C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) \leq (\mu_w^l)^{\text{-}\frac{1}{2}} C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}),$$

and consequently

$$\begin{aligned}\frac{\|\overset{*}{\boldsymbol{\lambda}}_z^{l\text{-}1}\|_{\mathrm{F}}}{\eta} &\leq \frac{\|\overset{*}{\boldsymbol{\lambda}}_z^l\|_{\mathrm{F}}}{\eta}\big((\mu_w^l)^{\text{-}\frac{1}{2}} C(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) + \|\check{\boldsymbol{w}}^l\|_{\mathrm{F}}\big)|g_h^{l\text{-}1}| + \frac{1}{\sqrt{\eta}}\phi(\check{\boldsymbol{w}}, \check{\boldsymbol{b}}) \\ &\leq \frac{\theta_z^{l\text{-}1}(\check{\boldsymbol{w}}, \check{\boldsymbol{b}})}{\sqrt{\eta}}.\end{aligned}$$

which completes the proof. □

Proof of Lemma 4.1 – (4.13b). Define function $\theta^{l}_{pq}(\boldsymbol{w},\boldsymbol{b})$ for every $l \in \mathcal{I}_{L\text{-}1}$ as

$$\theta^{l}_{pq}(\boldsymbol{w},\boldsymbol{b}) \triangleq \theta^{l+1}_{\boldsymbol{z}}(\boldsymbol{w},\boldsymbol{b})\big((\mu^{l+1}_{\boldsymbol{w}})^{-\frac{1}{2}}C(\boldsymbol{w},\boldsymbol{b})+\|\check{\boldsymbol{w}}^{l+1}\|_{\mathrm{F}}\big)+\phi(\boldsymbol{w},\boldsymbol{b}).$$

We use the triangle inequality to write the following relations

$$\begin{aligned}
&\frac{1}{\eta}\|\overset{*}{\check{\boldsymbol{w}}}{}^{l+1^{\top}}\overset{*}{\boldsymbol{\lambda}}{}^{l+1}_{\boldsymbol{z}}+\overset{*}{\check{\boldsymbol{c}}}{}^{l}_{h}\|_{\max}\\
&\quad\le\frac{1}{\eta}\|\overset{*}{\check{\boldsymbol{w}}}{}^{l+1^{\top}}\overset{*}{\boldsymbol{\lambda}}{}^{l+1}_{\boldsymbol{z}}\|_{\max}+\frac{1}{\eta}\|\overset{*}{\check{\boldsymbol{c}}}{}^{l}_{h}\|_{\max}\\
&\quad\le\frac{1}{\eta}\|\overset{*}{\check{\boldsymbol{w}}}{}^{l+1^{\top}}\overset{*}{\boldsymbol{\lambda}}{}^{l+1}_{\boldsymbol{z}}\|_{\mathrm{F}}+\frac{1}{\eta}\|\overset{*}{\check{\boldsymbol{c}}}{}^{l}_{h}\|_{\mathrm{F}}\\
&\quad\le\frac{\|\overset{*}{\boldsymbol{\lambda}}{}^{l+1}_{\boldsymbol{z}}\|_{\mathrm{F}}}{\eta}\big(\|\overset{*}{\check{\boldsymbol{w}}}{}^{l+1}-\check{\boldsymbol{w}}^{l+1}\|_{\mathrm{F}}+\|\check{\boldsymbol{w}}^{l+1}\|_{\mathrm{F}}\big)+\frac{1}{\sqrt{\eta}}\phi(\check{\boldsymbol{w}},\check{\boldsymbol{b}})\\
&\quad\le\frac{\theta^{l+1}_{\boldsymbol{z}}(\check{\boldsymbol{w}},\check{\boldsymbol{b}})}{\sqrt{\eta}}\big((\mu^{l+1}_{\boldsymbol{w}})^{-\frac{1}{2}}C(\check{\boldsymbol{w}},\check{\boldsymbol{b}})+\|\check{\boldsymbol{w}}^{l+1}\|_{\mathrm{F}}\big)+\frac{1}{\sqrt{\eta}}\phi(\check{\boldsymbol{w}},\check{\boldsymbol{b}})\\
&\quad=\frac{1}{\sqrt{\eta}}\,\theta^{l}_{pq}(\check{\boldsymbol{w}},\check{\boldsymbol{b}}).
\end{aligned}$$

This completes the proof. □

CHAPTER 5

Class Subset Selection for Partial Domain Adaptation

Domain adaptation is the task of transferring knowledge from a labeled source dataset to an unlabeled target dataset. Partial domain adaptation (PDA) investigates the scenarios in which the target label space is a subset of the source label space. The main purpose of the PDA is to identify the shared classes between the domains and promote learning transferable knowledge from these classes. Inspired by the idea of subset selection, we propose an adversarial PDA approach which aims to not only automatically select the most relevant subset of source domain classes but also ignore the samples that are less transferable across the domains. In the absence of target labels, the proposed approach is able to effectively learn domain-invariant feature representations, which in turn can facilitate and enhance the classification performance in the target domain. Empirical results on Office-31 and Office-Home datasets demonstrate the high potential of the proposed approach in addressing different partial domain adaptation tasks.

5.1 Introduction

Deep neural networks have demonstrated superior performance in a variety of machine learning problems such as semantic image segmentation [85, 84, 86], object detection, and classification [70, 88, 87], etc. These impressive achievements heavily depend on the availability of large amounts of labeled training data. However, in many applications, the acquisition of sufficient labeled data is difficult and time-consuming. One potential solution to reduce the labeling consumption is to build an effective predictive model using richly-annotated datasets from different but related domains. However, this paradigm generally suffers from the domain shift between the distributions of the source and the target datasets. As a result, deep networks trained on labeled source datasets often exhibit unsatisfactory performance on the target domain classification task. In the absence of target labels, unsupervised domain adaptation (UDA) seeks to bridge different domains by learning feature representations that are discriminative and domain-invariant [91, 90, 89].

Recently, various approaches have been proposed to combine both domain adaptation and deep feature learning in a unified framework for exploiting more transferable knowledge across domains [178, 179, 145, 180, 181, 182] (see [183] for a comprehensive survey on deep domain adaptation methods). A class of deep domain adaptation methods aims to reduce the misfit between the distributions of the source and target domains through minimizing discrepancy measures such as maximum mean discrepancy [145, 180], correlation distance [184, 185], etc. In this way, they map the domains into the same latent space, which results in learning feature representations that are domain-invariant. A new line of research has recently emerged which uses the concept of generative adversarial networks [186] to align feature distributions across the domains and learn discriminators that are able to predict the domain labels of different samples [187, 188, 189]. Specifically, these methods try to generate feature representations that are difficult for the discriminators to differentiate.

Despite the advantages offered by the existing UDA methods, they mostly exhibit superior performance in scenarios in which the source and target domains share the same label space. With the goal of considering more realistic cases, [92] introduced partial domain adaptation (PDA) as a new adaptation scenario which assumes the target domain label space is a subset of the source domain label space. The primary challenge in PDA is to identify and reject the source domain classes that do not appear in the target domain, known as *outlier classes*, since they may have negative impacts on the transfer performance [93, 190]. Addressing this challenge enables the PDA methods to effectively transfer models learned on large labeled datasets (e.g. ImageNet) to small-scale datasets from different but related domains.

In this work, we propose an adversarial approach for partial domain adaptation, which aims to not only automatically reject the outlier source classes, but also down-weight the relative importance of *irrelevant samples*, i.e. those samples that are highly dissimilar across different domains. Our method uses the same network architecture as partial adversarial domain adaptation (PADA) [92] and incorporates two additional regularization terms to boost the target domain classification performance. Inspired by the idea of subset selection, the first regularization is a row-sparsity term on the output of the classifier, which promotes the selection of a small subset of classes that are in common between the source and target domains. The second regularization is a minimum entropy term which utilizes the output of the discriminator to down-weight the relative importance of irrelevant samples from both domains. We empirically observe that our method can effectively enhance the target classification accuracy on different PDA tasks.

5.2 Related Work

To date, various deep unsupervised domain adaptation methods have been developed to extract domain-invariant feature representations from different domains. Some studies [191, 145, 180, 192, 193] have proposed to minimize the maximum mean discrepancy between the source and target distributions. In [194], a correlation alignment (CORAL) method is proposed that utilizes a linear transformation to match the second-order statistics between the domains. [185] presented an extension of the CORAL method that aligns correlations of layer activations in deep networks by learning a non-linear transformation. Despite the practical success of the aforementioned methods in aligning the domain distributions, it is shown that they are unable to completely eliminate the domain shift [179, 182].

Recently, adversarial learning has been widely employed to enhance the performance of UDA methods [195, 196, 197, 187, 198]. The basic idea behind the adversarial-based methods is to train a discriminator for predicting domain labels and a deep network for extracting features that are indistinguishable by the discriminator. By doing so, the discrepancy between the source and target domains can be efficiently eliminated, which results in significant improvement in the overall classification performance [196, 188, 198]. [199] developed an incremental adversarial scheme which gradually reduces the gap between the domain distributions by iteratively selecting the high confidence pseudo-labeled target samples to enlarge the training set.

Towards the task of PDA, great studies have been recently developed which simultaneously promote positive transfer from the common classes between the domains and alleviate the negative transfer from the outlier classes [93, 92, 200]. Selective adversarial network [93] trains separate domain discriminators for each source class to align the distributions of the source and target domains across the shared label space and to ignore the outlier classes. Partial adversarial domain adaptation (PADA) [92] proposed a new architecture which assigns a weight to each source domain class based on the target label prediction and automatically reduces the weights of the outlier classes. Importance weighted adversarial nets [200] develops a two-domain classifier strategy to estimate the relative importance of the source domain samples.

Closely related to our work, transferable attention for domain adaptation (TADA) [189] proposed an attention-based mechanism for UDA, which can highlight transferable regions or images. Unlike TADA, our method is focused on the PDA problem and utilizes a different network architecture with a novel loss function that efficiently assigns weights to both classes and samples. Our method differs from PADA [92] in the sense that we incorporate two novel regularization terms which not only able to discover and reject the

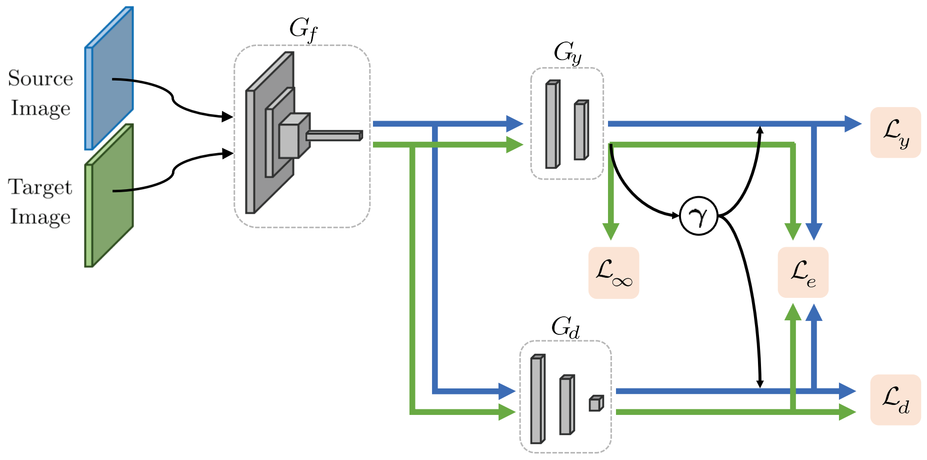


Figure 5.1: Overview of the proposed adversarial network for partial transfer learning. The network consists of a feature extractor, a classifier, and a domain discriminator, denoted by $\boldsymbol{G}_f$, $\boldsymbol{G}_y$, and $\boldsymbol{G}_d$, respectively. The blue and green arrows depict the source flow and target flow. Loss functions $\mathcal{L}_y$, $\mathcal{L}_d$, $\mathcal{L}_e$, and $\mathcal{L}_\infty$ denote the classification loss, the discriminative loss, the entropy loss, and the selection loss. *Best viewed in color.*

outlier classes more effectively but also down-weight the relative importance of the irrelevant samples in the training procedure.

5.3 Problem Formulation

This section briefly reviews two well-established domain adaptation methods and then provides a detailed explanation on how our proposed method relates to them. Let $\{(\boldsymbol{x}_s^i, \boldsymbol{y}_s^i)\}_{i=1}^{n_s}$ be a set of n_s sample points drawn $i.i.d$ from the source domain $\mathcal{D}_s$, where $\boldsymbol{x}_s^i$ denotes the i^{th} source image with label $\boldsymbol{y}_s^i$. Similarly, let $\{\boldsymbol{x}_t^i\}_{i=1}^{n_t}$ be a set of n_t sample points collected $i.i.d$ from the target domain $\mathcal{D}_t$, where $\boldsymbol{x}_t^i$ indicates the i^{th} target image. To clarify notation, let $\mathcal{X} = \mathcal{X}_s \cup \mathcal{X}_t$ be the set of entire images from both domains, where $\mathcal{X}_s = \{\boldsymbol{x}_s^i\}_{i=1}^{n_s}$ and $\mathcal{X}_t = \{\boldsymbol{x}_t^i\}_{i=1}^{n_t}$. The UDA methods assume that the source and target domains possess the same label space, denoted as $\mathcal{C}_s$ and $\mathcal{C}_t$, respectively. In the absence of target labels, the primary goal of the UDA is to learn domain-invariant feature representations that can reduce the domain shift. One promising direction to achieve this goal is to train a domain adversarial neural network [196] which consists of a discriminator G_d for predicting the domain labels, a feature extractor G_f for confusing the discriminator by learning transferable feature representations, and a classifier G_y that classifies the source domain samples. Training the adversarial network is equivalent to solve the following optimization problem

$$\max_{\boldsymbol{\theta}_d} \min_{\boldsymbol{\theta}_y, \boldsymbol{\theta}_f} \frac{\lambda}{n_s} \sum_{\boldsymbol{x}^i \in \mathcal{X}_s} L_y(G_y(G_f(\boldsymbol{x}^i; \boldsymbol{\theta}_f); \boldsymbol{\theta}_y), \boldsymbol{y}^i) - \frac{1}{n} \sum_{\boldsymbol{x}^i \in \mathcal{X}} L_d(G_d(G_f(\boldsymbol{x}^i; \boldsymbol{\theta}_f); \boldsymbol{\theta}_d), d^i),$$

where $n = n_s + n_t$, $\lambda > 0$ is a regularization parameter, $\boldsymbol{y}^i$ is the one-hot class label of image $\boldsymbol{x}^i$ and $d^i \in \{0, 1\}$ denotes its domain label; $d^i = 0$ if $\boldsymbol{x}^i$ belongs to the source domain and $d^i = 1$ otherwise. L_y and L_d are cross-entropy loss functions corresponding to the classifier G_y and the domain discriminator G_d, respectively. Moreover, variables $\boldsymbol{\theta}_f$, $\boldsymbol{\theta}_y$, and $\boldsymbol{\theta}_d$ are parameters associated with the networks G_f, G_y, and G_d, respectively. For the brevity of notation, we drop the reference to the parameters $\boldsymbol{\theta}_f$, $\boldsymbol{\theta}_y$, and $\boldsymbol{\theta}_d$ in the subsequent formulations.

As noted earlier, standard domain adaptation approaches assume that the source and target possess the same label space, i.e. $\mathcal{C}_s = \mathcal{C}_t$. This assumption may not be fulfilled in a wide range of practical applications in which $\mathcal{C}_s$ is large and diverse (e.g., ImageNet) and $\mathcal{C}_t$ only contains a small subset of source classes, i.e. $\mathcal{C}_t \subset \mathcal{C}_s$. Under this assumption, aligning the domain distributions may not necessarily facilitate the classification task in the target domain due to the adverse effect of transferring information from the outlier classes $\mathcal{C}_s \backslash \mathcal{C}_t$ [92, 93]. Hence, the primary goal in partial domain adaptation is to learn a feature extractor that can align the distributions of the source and target domains across the shared label space and simultaneously identify and reject the outlier classes. A classifier trained along such feature extractor can generalize well to the target domain. To this end, PADA

[92] proposed the following weighting procedure to highlight the shared classes and reduce the importance of outlier classes

$$\boldsymbol{\gamma} = \frac{1}{n_t} \sum_{i=1}^{n_t} \hat{\boldsymbol{y}}_t^i$$

where $\hat{\boldsymbol{y}}_t^i = G_y(G_f(\boldsymbol{x}_t^i))$ denotes the output of G_y to the target sample $\boldsymbol{x}_t^i$. The weighting vector $\boldsymbol{\gamma}$ is further normalized as $\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma} \backslash \max(\boldsymbol{\gamma})$ to show the relative weights of the classes.

The weights associated with the outlier classes are expected to be much smaller than that of the shared classes, mainly because the target samples are significantly dissimilar to the samples belonging to the outlier classes. Ideally, $\boldsymbol{\gamma}$ is expected to be a vector whose elements are non-zero except those corresponding to the outlier classes. Given that PADA proposes to train the adversarial network through solving the following minimax optimization problem

$$\begin{aligned} \max_{\boldsymbol{\theta}_d} \min_{\boldsymbol{\theta}_y, \boldsymbol{\theta}_f} \; & \frac{\lambda}{n_s} \sum_{\boldsymbol{x}^i \in \mathcal{X}_s} \gamma_{c_i} \, L_y(G_y(G_f(\boldsymbol{x}^i)), \boldsymbol{y}^i) \\ & - \frac{1}{n_s} \sum_{\boldsymbol{x}^i \in \mathcal{X}_s} \gamma_{c_i} \, L_d(G_d(G_f(\boldsymbol{x}^i)), d^i) \\ & - \frac{1}{n_t} \sum_{\boldsymbol{x}^i \in \mathcal{X}_t} L_d(G_d(G_f(\boldsymbol{x}^i)), d^i), \end{aligned}$$

where $c_i = \mathrm{argmax}_j \, y_j^i$ denotes the index of the largest element in $\boldsymbol{y}^i$.

Besides the outlier classes, the irrelevant samples are inherently less transferable and they may significantly degrade the target classification performance in different PDA tasks. In the next section, we present a novel algorithm to simultaneously identify and reject the outlier classes and down-weight the relative importance of the irrelevant samples.

5.4 Proposed Method

We adopt the same network architecture as PADA and employ two novel regularization terms to better align the source and target distributions across the shared classes and learn more transferable features.

The first regularization is a row-sparsity term which promotes the selection of a small subset of source domain classes that appear in the target domain. This, in turn, encourages γ to be a vector of zeros except for the elements corresponding to the shared classes. This selection regularization can be defined as follows

$$\mathcal{L}_\infty(\mathcal{X}_t, \boldsymbol{\theta}_f, \boldsymbol{\theta}_y) = \frac{\mu}{|\mathcal{C}_s|} \left\| G_y(G_f(\boldsymbol{x}_t^1)), \ldots, G_y(G_f(\boldsymbol{x}_t^{|\mathcal{X}_t|})) \right\|_{1,\infty},$$

where $|.|$ denotes the cardinality of its input set, $\|.\|_{1,\infty}$ computes the sum of the infinity norms of the rows of an input matrix, and μ is a regularization parameter. Imposing the above term takes into account the relation between the entire target samples and encourages the classifier to generate a sparse output vector with its non-zero entries located at certain indices correspond to the shared classes.

The second regularization term seeks to reduce the importance of irrelevant samples in the training procedure by leveraging the following entropy minimization term

$$\begin{aligned}\mathcal{L}_e(\mathcal{X}_s, \mathcal{X}_t, \boldsymbol{\theta}_f, \boldsymbol{\theta}_d, \boldsymbol{\theta}_y) = & \frac{1}{n_s} \sum_{\boldsymbol{x}^i \in \mathcal{X}_s} \gamma_{c_i} (1 + L_d^e(G_d(G_f(\boldsymbol{x}^i)))) \, L_y^e(G_y(G_f(\boldsymbol{x}^i))) \\ & + \frac{1}{n_t} \sum_{\boldsymbol{x}^i \in \mathcal{X}_t} (1 + L_d^e(G_d(G_f(\boldsymbol{x}^i)))) \, L_y^e(G_y(G_f(\boldsymbol{x}^i))),\end{aligned}$$

where L_y^e and L_d^e are the entropy loss functions corresponding to the classifier G_y and the domain discriminator G_d, respectively. The above regularization encourages assigning higher weights to those samples whose domain labels are confidently predicted by the discriminator. This, in turn, reduces the relative importance of the irrelevant samples and helps to learn more transferable features for classification.

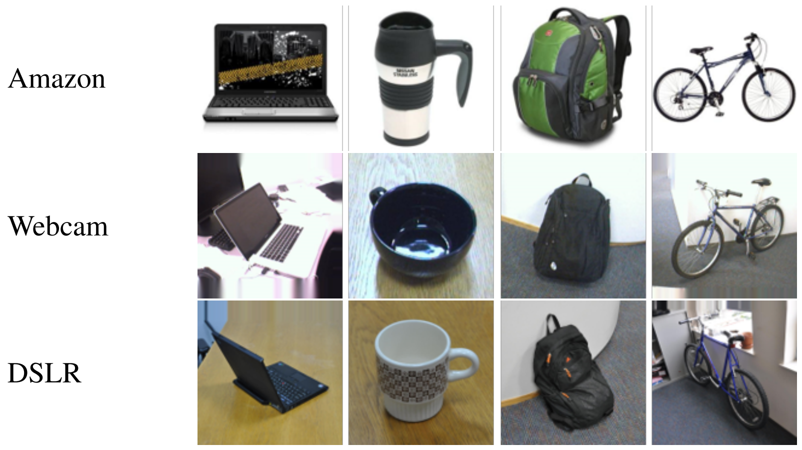


Figure 5.2: Example images of the Office-31 dataset.

By integrating both regularization terms into the total loss function, our method can not only automatically identify and reject the outlier classes, but also down-weight the irrelevant samples that are inherently not transferable across domains. Figure 5.1 illustrates the architecture of our proposed network in details.

5.5 Experiments

In this section, we conduct empirical experiments on two benchmark datasets to evaluate the efficacy of our approach, named SSPDA, for partial domain adaptation (PDA) across different tasks. The experiments are performed in an unsupervised setting, where the target labels are unknown. In what follows, we briefly explain the datasets, the PDA tasks, and the network hyperparameters used in the experiments.

5.5.1 Setup

Dataset: We evaluate the performance of SSPDA on two commonly used datasets for domain adaptation: Office-31 and Office-Home. The Office-31 object dataset [201] consists of $4,652$ images from 31 classes, where the images are collected from three different domains: *Amazon* (**A**), *Webcam* (**W**), and *DSLR* (**D**). We follow the procedure presented in

Method	**A → W**	**D → W**	**W → D**	**A → D**	**D → A**	**W → A**	**Avg**
ResNet	54.52	94.57	94.27	65.61	73.17	71.71	75.64
DAN	46.44	53.56	58.60	42.68	65.66	65.34	55.38
DANN	41.35	46.78	38.85	41.36	41.34	44.68	42.39
ADDA	43.65	46.48	40.12	43.66	42.76	45.95	43.77
RTN	75.25	97.12	98.32	66.88	85.59	85.70	84.81
SAN	80.02	98.64	100	81.28	80.58	83.09	87.27
IWAN	76.27	98.98	100	78.98	89.46	81.73	87.57
PADA	86.54	**99.32**	**100**	82.17	92.69	95.41	92.69
SSPDA-selection	87.45	95.31	98.48	82.25	91.89	95.34	91.79
SSPDA-entropy	90.51	96.59	97.45	89.08	92.38	95.30	93.55
SSPDA	**93.42**	97.62	**100**	**90.43**	**93.45**	**95.53**	**95.07**

Table 5.1: Accuracy of partial domain adaptation tasks on *Office-31* (ResNet-50).

Method	**Ar→Cl**	**Ar→Pr**	**Ar→Rw**	**Cl→Ar**	**Cl→Pr**	**Cl→Rw**	**Pr→Ar**	**Pr→Cl**	**Pr→Rw**	**Rw→Ar**	**Rw→Cl**	**Rw→Pr**	**Avg**
ResNet	38.57	60.78	75.21	39.94	48.12	52.90	49.68	30.91	70.79	65.38	41.79	70.42	53.71
DAN	44.36	61.79	74.49	41.78	45.21	54.11	46.92	38.14	68.42	64.37	45.37	68.85	54.48
DANN	44.89	54.06	68.97	36.27	34.34	45.22	44.08	38.03	68.69	52.98	34.68	46.50	47.39
RTN	49.37	64.33	76.19	47.56	51.74	57.67	50.38	41.45	75.53	70.17	51.82	74.78	59.25
PADA	51.95	67	78.74	**52.16**	**53.78**	59.03	52.61	**43.22**	78.79	**73.73**	**56.6**	77.09	62.06
SSPDA	**52.31**	**68.35**	**80.17**	50.79	51.29	**60.87**	**56.68**	42.53	**79.15**	70.94	56.43	**78.92**	**62.37**

Table 5.2: Accuracy of partial domain adaptation tasks on *Office-Home* (ResNet-50).

[92] to transfer knowledge from a source domain with 31 classes to a target domain with 10 classes. The results are provided as the target domain classification accuracy across six different PDA tasks: **A** $\rightarrow$ **W**, **W** $\rightarrow$ **A**, **D** $\rightarrow$ **W**, **W** $\rightarrow$ **D**, **A** $\rightarrow$ **D**, and **D** $\rightarrow$ **A**.

The Office-Home [202] is a more complex dataset consisting of around $15,500$ images collected from four distinct domains: *Art* (**Ar**), *Clipart* (**Cl**), *Product* (**Pr**), and *Real-World* (**Rw**), where each domain has 65 classes. Following the procedure presented in [92], we aim to transfer information from a source domain containing 65 classes to a target domain with 25 classes. The results on this dataset are also reported as the target classification accuracy on twelve pairs of source-target domains: **Ar** $\rightarrow$ **Cl**, **Ar** $\rightarrow$ **Pr**, **Ar** $\rightarrow$ **Rw**, **Cl** $\rightarrow$ **Ar**, **Cl** $\rightarrow$ **Pr**, **Cl** $\rightarrow$ **Rw**, **Pr** $\rightarrow$ **Ar**, **Pr** $\rightarrow$ **Cl**, **Pr** $\rightarrow$ **Rw**, **Rw** $\rightarrow$ **Ar**, **Rw** $\rightarrow$ **Cl**, and **Rw** $\rightarrow$ **Pr**.

For each of the aforementioned tasks, we report the average target classification accuracy of five independent runs with different initialization as generated in [92]. We compare the performance of SSPDA against several deep transfer learning methods: Deep Adaptation Network (DAN) [180], Domain Adversarial Neural Network (DANN) [196], Residual Transfer Networks (RTN) [192], Adversarial Discriminative Domain Adaptation (ADDA) [198], Importance Weighted Adversarial Nets (IWAN) [200], Selective Adversarial Network (SAN) [93], and Partial Adversarial Domain Adaptation (PADA) [92].

Parameter: We adopt ResNet-50 [73] pre-trained on ImageNet [203] as the backbone for the network G_f. Also, we fine-tune the entire feature layers and apply back-propagation to train the domain discriminator G_d and the classifier G_y. Through the experiments, parameter λ is set to 1.0 and 2.0 for the Office-31 dataset and Office-Home dataset, respectively. Also, we set $\mu = 0.1$ for both datasets. Notice that since the classifier is not appropriately trained in the first few epochs, we gradually increase parameter μ from 0 to 0.1. To minimize the loss function, we use mini-batch stochastic gradient descent (SGD) with a momentum of 0.95 and the learning rate is adjusted during SGD by: $\eta = \frac{\eta_0}{(1+\alpha\times\rho)^\beta}$ where $\eta_0 = 10^{-2}$, $\alpha = 10$, $\beta = 0.75$, and ρ is the training progress linearly changing from 0 to 1 [92, 196]. We use a batch size of 72 with 36 samples for each domain.

5.5.2 Results

Tables 5.1 and 5.2 show the target domain classification accuracy of various methods on different PDA tasks including 6 tasks of Office-31 dataset and 12 tasks of Office-Home dataset. All the results are reported based on the ResNet-50 and the scores of the competitor methods are directly obtained from [92, 93, 200].

Observe that some deep domain adaptation methods such as DAN and DANN have exhibited worse performance than the standard ResNet-50 on few PDA tasks in both datasets. This can be attributed to the fact that these methods aim to align the marginal distributions across the domains and hence are prone to the negative transfer resulted from the outlier classes. On the other hand, the PDA methods, such as PADA, SAN, and IWAN, achieve promising results on most of the PDA tasks since they leverage weighting mechanisms to highlight a subset of samples that are more transferable. By doing so, these methods can effectively mitigate transferring knowledge from the outlier source classes and promote learning from the shared classes between the domains, which in turn enhance the classification accuracy in the target domain.

Notice that SSPDA uses the same network architecture as PADA, but introduce a novel loss function to identify and reject the outlier classes and irrelevant samples. The results in Table 5.1 indicate that SSPDA performs better than or close to the state-of-the-art methods at all PDA tasks on Office-31 dataset. In particular, it achieves considerable improvement on $\mathbf{A} \rightarrow \mathbf{W}$ and $\mathbf{A} \rightarrow \mathbf{D}$, and generally increases the average accuracy of all tasks by almost 2.4%. Moreover, Table 5.2 shows that SSPDA maintains the performance of PADA and exhibits slight improvement in the average classification accuracy over all partial domain adaptation tasks on Office-Home dataset. The numerical results provided in the above tables imply that SSPDA has high potential in transferring semantic information and learning domain-invariant features in different tasks of partial domain adaptation.

Ablation Study: To demonstrate the improvements obtained by each of the proposed regularizations, in this part, we conduct an ablation study by discarding the selection regularization (SSPDA-selection) or the entropy minimization term (SSPDA-entropy). The results are reported in Table 5.1. It can be seen that both SSPDA-selection and SSPDA-entropy generally obtain better or close results than the baselines. In particular, SSPDA-entropy works better on some difficult tasks such as $\mathbf{A} \rightarrow \mathbf{W}$ and $\mathbf{A} \rightarrow \mathbf{D}$.

5.6 Conclusion

This work presented an adversarial approach for the task of partial domain adaptation. The proposed approach minimizes a novel loss function to reduce the effect of the outlier classes and the irrelevant samples, which results in learning more transferable feature representations for classification. The experiments conducted on standard benchmark datasets demonstrate the high potential of our approach for partial domain adaptation tasks.

CHAPTER 6

Conclusion

In this dissertation, we first developed a novel and highly parallelizable convex model which automatically estimates the optimal number of coherent regions and pixel assignments to form final segments. To solve the model, a computationally efficient algorithm is presented based on the alternating direction method of multiplier. Extensive experiments corroborate the robustness and effectiveness of the proposed methods compared to the other state-of-the-art approaches. Second, we developed a sequential approach based on parabolic relaxation which finds an orthogonal matrix that minimizes a non-convex and non-smooth objective function subject to additional quadratic constraints. We prove that under very mild assumptions, the proposed approach is guaranteed to provide a feasible solution for the original non-convex problem. The effectiveness of the proposed scheme is corroborated for the problem of discriminative dimensionality reduction and graph clustering. Third, we proposed a novel convexification approach that reduces the problem of training neural networks into solving a sequence of polynomial-time solvable convex surrogates. The proposed approach jointly estimates the network parameters of all layers and can admit a wide range of additional convex constraints. We theoretically prove that the proposed approach is guaranteed to converge under mild conditions and perform empirical experiments to corroborate the effectiveness of the method. Fourth, we presented an adversarial approach for the task of partial domain adaptation. The proposed approach minimizes a novel loss function to reduce the effect of the outlier classes and the irrelevant samples, which results in learning more transferable feature representations for classification. The experiments conducted on standard benchmark datasets demonstrate the high potential of our approach for partial domain adaptation tasks.

REFERENCES

[1] X. Fu, C.-Y. Wang, C. Chen, C. Wang, and C.-C. Jay Kuo, “Robust image segmentation using contour-guided color palettes,” in *ICCV*, 2015.

[2] Z. Ren and G. Shakhnarovich, “Image segmentation by cascaded region agglomeration,” in *CVPR*, 2013.

[3] P. Arbelaez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik, “Multiscale combinatorial grouping,” in *CVPR*, 2014.

[4] W. Bian and D. Tao, “Max-min distance analysis by using sequential SDP relaxation for dimension reduction,” *IEEE T PATTERN ANAL*, vol. 33, no. 5, pp. 1037–1050, 2011.

[5] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, “From few to many: Illumination cone models for face recognition under variable lighting and pose,” *IEEE T PATTERN ANAL*, vol. 23, no. 6, pp. 643–660, 2001.

[6] J. F. Sturm, “Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones,” *OPTIM METHOD SOFTW*, vol. 11, no. 1-4, pp. 625–653, 1999.

[7] K.-C. Toh, M. J. Todd, and R. H. Tütüncü, “SDPT3—a MATLAB software package for semidefinite programming, version 1.3,” *OPTIM METHOD SOFTW*, vol. 11, no. 1-4, pp. 545–581, 1999.

[8] A. Mosek, “The MOSEK optimization toolbox for MATLAB manual,” *Version 7.1*, 2015.

[9] D. Dua and C. Graff, “UCI machine learning repository,” 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[10] A. Gionis, H. Mannila, and P. Tsaparas, “Clustering aggregation,” *ACM T KNOWL DISCOV D*, vol. 1, no. 1, p. 4, 2007.

[11] H. Chang and D.-Y. Yeung, “Robust path-based spectral clustering,” *PATTERN RECOGN*, vol. 41, no. 1, pp. 191–203, 2008.

[12] A. K. Jain and M. H. Law, “Data clustering: A user’s dilemma,” in *ICPRAI*, 2005.

[13] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[14] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992, pp. 144–152.

[15] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[16] N. Cristianini, J. Shawe-Taylor, *et al.*, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[17] J. A. Suykens and J. Vandewalle, “Least squares support vector machine classifiers,” *NEURAL PROCESS LETT*, vol. 9, no. 3, pp. 293–300, 1999.

[18] M. C. Ferris and T. S. Munson, “Interior-point methods for massive support vector machines,” *SIAM J OPTIMIZ*, vol. 13, no. 3, pp. 783–804, 2002.

[19] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, “Pegasos: Primal estimated sub-gradient solver for svm,” *MATH PROGRAM*, vol. 127, no. 1, pp. 3–30, 2011.

[20] E. Elhamifar and R. Vidal, “Sparse subspace clustering: Algorithm, theory, and applications,” *IEEE T PATTERN ANAL*, vol. 35, no. 11, pp. 2765–2781, 2013.

[21] C. Zhu, H. Xu, C. Leng, and S. Yan, “Convex optimization procedure for clustering: theoretical revisit,” in *NIPS*, 2014.

[22] K. Pelckmans, J. De Brabanter, B. De Moor, and J. Suykens, “Convex clustering shrinkage,” in *Workshop on Statistics and optimization of clustering Workshop*, 2005.

[23] E. J. Candès and B. Recht, “Exact matrix completion via convex optimization,” *Foundations of Computational mathematics*, vol. 9, no. 6, p. 717, 2009.

[24] F. Zohrizadeh, M. Kheirandishfard, K. Ghasedidizaji, and F. Kamangar, “Reliability-based local features aggregation for image segmentation,” in *ISVC*, 2016.

[25] F. Zohrizadeh, M. Kheirandishfard, and F. Kamangar, “Image segmentation using sparse subset selection,” in *WACV*, 2018.

[26] Z. Hong, X. Mei, D. Prokhorov, and D. Tao, “Tracking via robust multi-task multi-view joint sparse representation,” in *ICCV*, 2013.

[27] T. Zhang, A. Bibi, and B. Ghanem, “In defense of sparse tracking: Circulant sparse tracker,” in *CVPR*, 2016.

[28] T. Zhang, C. Xu, and M.-H. Yang, “Robust structural sparse tracking,” *IEEE T PATTERN ANAL*, 2018.

[29] E. Elhamifar, G. Sapiro, and S. S. Sastry, “Dissimilarity-based sparse subset selection,” *IEEE T PATTERN ANAL*, vol. 38, no. 11, pp. 2182–2197, 2016.

[30] J. Meng, H. Wang, J. Yuan, and Y.-P. Tan, “From keyframes to key objects: Video summarization by representative object proposal selection,” in *CVPR*, 2016.

[31] J. Barzilai and J. M. Borwein, “Two-point step size gradient methods,” *IMA J NUMER ANAL*, vol. 8, no. 1, pp. 141–148, 1988.

[32] M. D. Zeiler, “ADADELTA: an adaptive learning rate method,” *arXiv preprint arXiv:1212.5701*, 2012.

[33] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *ICLR*, 2015.

[34] N. Parikh, S. Boyd, *et al.*, “Proximal algorithms,” *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[35] T. Goldstein and S. Osher, “The split bregman method for l1-regularized problems,” *SIAM J IMAGING SCI*, vol. 2, no. 2, pp. 323–343, 2009.

[36] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, “Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing,” *SIAM J IMAGING SCI*, vol. 1, no. 1, pp. 143–168, 2008.

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

[38] A. V. Fiacco and G. P. McCormick, *Nonlinear programming: sequential unconstrained minimization techniques*. Siam, 1990, vol. 4.

[39] P. E. Gill, W. Murray, M. A. Saunders, J. A. Tomlin, and M. H. Wright, “On projected newton barrier methods for linear programming and an equivalence to karmarkar’s projective method,” *MATH PROGRAM*, vol. 36, no. 2, pp. 183–209, 1986.

[40] Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming*. SIAM, 1994.

[41] F. Zohrizadeh, M. Kheirandishfard, F. Kamangar, and R. Madani, “Sequential convex relaxations for optimization under orthogonality constraints,” 2018.

[42] Y. She and A. B. Owen, “Outlier detection using nonconvex penalized regression,” *J AM STAT ASSOC*, vol. 106, no. 494, pp. 626–639, 2011.

[43] D. Kong, H. Bondell, and W. Shen, “Outlier detection and robust estimation in nonparametric regression,” in *International Conference on Artificial Intelligence and Statistics*, 2018, pp. 208–216.

[44] J. Han, K. Xiong, and F. Nie, “Orthogonal and nonnegative graph reconstruction for large scale clustering,” in *IJCAI*, 2017.

[45] F. Zhou and F. De la Torre, "Factorized graph matching," in *CVPR*, 2012.

[46] C. Schellewald and C. Schnörr, "Probabilistic subgraph matching based on convex relaxation," in *EMMCVPR*, 2005.

[47] F. Zhou and F. De la Torre, "Deformable graph matching," in *CVPR*, 2013.

[48] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval research logistics quarterly*, vol. 3, no. 1-2, pp. 95–110, 1956.

[49] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM review*, pp. 49–95, 1996.

[50] J. Lavaei and S. H. Low, "Zero duality gap in optimal power flow problem," *IEEE T POWER SYST*, vol. 1, no. 27, pp. 92–107, 2012.

[51] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear algebra and its applications*, vol. 284, no. 1-3, pp. 193–228, 1998.

[52] F. Alizadeh and D. Goldfarb, "Second-order cone programming," *MATH PROGRAM*, pp. 3–51, 2003.

[53] A. Ben-Hur, A. Elisseeff, and I. Guyon, "A stability based method for discovering structure in clustered data," in *PSB*, 2001.

[54] S. Still and W. Bialek, "How many clusters? an information-theoretic perspective," *Neural computation*, vol. 16, no. 12, pp. 2483–2506, 2004.

[55] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 411–423, 2001.

[56] B. Jiang, J. Tang, C. H. Ding, and B. Luo, "Nonnegative orthogonal graph matching." in *AAAI*, 2017.

[57] J. Tang and H. Liu, "Unsupervised feature selection for linked social media data," in *SIGKDD*, 2012.

[58] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, "$\ell_{2,1}$-norm regularized discriminative feature selection for unsupervised learning," in *IJCAI*, 2011.

[59] C. Bao, H. Ji, Y. Quan, and Z. Shen, "L0 norm based dictionary learning by proximal methods with global convergence," in *CVPR*, 2014.

[60] ——, "Dictionary learning for sparse coding: Algorithms and convergence analysis," *IEEE T PATTERN ANAL*, vol. 38, no. 7, pp. 1356–1369, 2016.

[61] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM J MATH ANAL*, vol. 20, no. 2, pp. 303–353, 1998.

[62] T. E. Abrudan, J. Eriksson, and V. Koivunen, "Steepest descent algorithms for optimization under unitary matrix constraint," *IEEE T SIGNAL PROCES*, vol. 56, no. 3, pp. 1134–1147, 2008.

[63] Y. Nishimori and S. Akaho, "Learning algorithms utilizing Quasi-Geodesic flows on the Stiefel manifold," *Neurocomputing*, vol. 67, pp. 106–135, 2005.

[64] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[65] Z. Wen and W. Yin, "A feasible method for optimization with orthogonality constraints," *MATH PROGRAM*, vol. 142, no. 1-2, pp. 397–434, 2013.

[66] V. Ozoliņš, R. Lai, R. Caflisch, and S. Osher, "Compressed modes for variational problems in mathematics and physics," *PNAS*, vol. 110, no. 46, pp. 18 368–18 373, 2013.

[67] R. Lai and S. Osher, "A splitting method for orthogonality constrained problems," *SIAM J SCI COMPUT*, vol. 58, no. 2, pp. 431–449, 2014.

[68] T. Neumann, K. Varanasi, C. Theobalt, M. Magnor, and M. Wacker, "Compressed manifold modes for mesh processing," in *COMPUT GRAPH FORUM*, vol. 33, no. 5. Wiley Online Library, 2014, pp. 35–44.

[69] A. Kovnatsky, K. Glashoff, and M. M. Bronstein, "MADMM: a generic algorithm for non-smooth optimization on manifolds," in *ECCV*, 2016.

[70] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[71] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*, 2014.

[72] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *ICLR*, 2014.

[73] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[74] Y. Zhang and T. Funkhouser, "Deep depth completion of a single RGB-D image," in *CVPR*, 2018.

[75] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *3DV*, 2016.

[76] J. R. Hershey, S. J. Rennie, P. A. Olsen, and T. T. Kristjansson, "Super-human multi-talker speech recognition: A graphical modeling approach," *Computer Speech & Language*, pp. 45–66, 2010.

[77] A. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE T AUDIO SPEECH*, pp. 14–22, 2012.

[78] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE T NEURAL NETWOR*, pp. 157–166, 1994.

[79] G. Swirszcz, W. M. Czarnecki, and R. Pascanu, "Local minima in training of neural networks," *arXiv preprint arXiv:1611.06310*, 2016.

[80] Y. Zhang, J. Lee, M. Wainwright, and M. Jordan, "On the learnability of fully-connected neural networks," in *AISTATS*, 2017.

[81] Y. Zhang, P. Liang, and M. J. Wainwright, "Convexified convolutional neural networks," in *ICML*, 2017.

[82] M. Soltanolkotabi, A. Javanmard, and J. D. Lee, "Theoretical insights into the optimization landscape of over-parameterized shallow neural networks," *IEEE Transactions on Information Theory*, 2018.

[83] S. S. Du, X. Zhai, B. Poczos, and A. Singh, "Gradient descent provably optimizes over-parameterized neural networks," in *ICLR*, 2019.

[84] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.

[85] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE T PATTERN ANAL*, vol. 40, no. 4, pp. 834–848, 2018.

[86] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang, "Semantic image segmentation via deep parsing network," in *ICCV*, 2015.

[87] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *NIPS*, 2013.

[88] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[89] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE T NEURAL NETWOR*, vol. 22, no. 2, pp. 199–210, 2011.

[90] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *CVPR*, 2012.

[91] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann, "Unsupervised domain adaptation by domain invariant projection," in *ICCV*, 2013.

[92] Z. Cao, L. Ma, M. Long, and J. Wang, "Partial adversarial domain adaptation," in *ECCV*, 2018.

[93] Z. Cao, M. Long, J. Wang, and M. I. Jordan, "Partial transfer learning with selective adversarial networks," in *CVPR*, 2018.

[94] A. Khoreva, F. Galasso, M. Hein, and B. Schiele, "Learning must-link constraints for video segmentation based on spectral clustering," in *GCPR*, 2014.

[95] ——, "Classifier based graph construction for video segmentation," in *CVPR*, 2015.

[96] S. Dutt Jain and K. Grauman, "Active image segmentation propagation," in *CVPR*, 2016.

[97] G. Ghiasi and C. C. Fowlkes, "Laplacian pyramid reconstruction and refinement for semantic segmentation," in *ECCV*, 2016.

[98] S. D. Jain, B. Xiong, and K. Grauman, "Fusionseg: Learning to combine motion and appearance for fully automatic segmention of generic objects in videos," *arXiv preprint arXiv:1701.05384*, 2017.

[99] R. Díaz, M. Lee, J. Schubert, and C. C. Fowlkes, "Lifting gis maps into strong geometric context for scene understanding," in *WACV*, 2016.

[100] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE T PATTERN ANAL*, vol. 33, no. 5, pp. 898–916, 2011.

[101] J. Yuan, D. Wang, and A. M. Cheriyadat, "Factorization-based texture segmentation," *IEEE T IMAGE PROCESS*, vol. 24, no. 11, pp. 3488–3497, 2015.

[102] T. H. Kim, K. M. Lee, and S. U. Lee, "Learning full pairwise affinities for spectral segmentation," *IEEE T PATTERN ANAL*, vol. 35, no. 7, pp. 1690–1703, 2013.

[103] Z. Li, X.-M. Wu, and S.-F. Chang, "Segmentation using superpixels: A bipartite graph partitioning approach," in *CVPR*, 2012.

[104] X. Liu and D. Wang, "Image and texture segmentation using local spectral histograms," *IEEE T IMAGE PROCESS*, vol. 15, no. 10, pp. 3066–3077, 2006.

[105] D. Gabay and B. Mercier, “A dual algorithm for the solution of nonlinear variational problems via finite element approximation,” *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976.

[106] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in *ICCV*, 2001.

[107] J. Shotton, J. Winn, C. Rother, and A. Criminisi, “Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context,” *INT J COMPUT VISION*, vol. 81, no. 1, pp. 2–23, 2009.

[108] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis,” *IEEE T PATTERN ANAL*, vol. 24, no. 5, pp. 603–619, 2002.

[109] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient graph-based image segmentation,” *INT J COMPUT VISION*, vol. 59, no. 2, pp. 167–181, 2004.

[110] M. Donoser and D. Schmalstieg, “Discrete-continuous gradient orientation estimation for faster image segmentation,” in *CVPR*, 2014.

[111] L. Gao, J. Song, F. Nie, F. Zou, N. Sebe, and H. T. Shen, “Graph-without-cut: An ideal graph learning for image segmentation,” in *AAAI*, 2016.

[112] J. Kim and K. Grauman, “Boundary preserving dense local regions,” in *CVPR*, 2011.

[113] T. Liu, M. Seyedhosseini, and T. Tasdizen, “Image segmentation using hierarchical merge tree,” *IEEE T IMAGE PROCESS*, vol. 25, no. 10, pp. 4596–4607, Oct 2016.

[114] Y. Yu, C. Fang, and Z. Liao, “Piecewise flat embedding for image segmentation,” in *ICCV*, 2015.

[115] Y. Chen, D. Dai, J. Pont-Tuset, and L. Van Gool, “Scale-aware alignment of hierarchical image segmentation,” in *CVPR*, 2016.

[116] I. Kokkinos, “Pushing the boundaries of boundary detection using deep learning,” *arXiv preprint arXiv:1511.07386*, 2015.

[117] K.-K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. Van Gool, “Convolutional oriented boundaries: From image segmentation to high-level tasks,” *IEEE T PATTERN ANAL*, 2017.

[118] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, “Simultaneous detection and segmentation,” in *ECCV*, 2014.

[119] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *CVPR*, 2015.

[120] H. Noh, S. Hong, and B. Han, “Learning deconvolution network for semantic segmentation,” in *ICCV*, 2015.

[121] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, “Conditional random fields as recurrent neural networks,” in *ICCV*, 2015.

[122] E. Elhamifar, G. Sapiro, and S. S. Sastry, “Dissimilarity-based sparse subset selection,” *IEEE T PATTERN ANAL*, vol. 38, no. 11, pp. 2182–2197, Nov 2016.

[123] E. Esser, M. Moller, S. Osher, G. Sapiro, and J. Xin, “A convex model for nonnegative matrix factorization and dimensionality reduction on physical space,” *IEEE T IMAGE PROCESS*, vol. 21, no. 7, pp. 3239–3252, 2012.

[124] E. Elhamifar, G. Sapiro, and R. Vidal, “See all by looking at a few: Sparse modeling for finding representative objects,” in *CVPR*, 2012.

[125] W. Wang and M. A. Carreira-Perpinán, “Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application,” *arXiv preprint arXiv:1309.1541*, 2013.

[126] T. Goldstein, B. O’Donoghue, S. Setzer, and R. Baraniuk, “Fast alternating direction optimization methods,” *SIAM J IMAGING SCI*, vol. 7, no. 3, pp. 1588–1623, 2014.

[127] T. Malisiewicz and A. A. Efros, “Improving spatial support for objects via multiple segmentations,” in *BMVC*, 2007.

[128] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J AM STAT ASSOC*, vol. 66, no. 336, pp. 846–850, 1971.

[129] M. Meila, "Comparing clusterings: an axiomatic view," in *ICML*, 2005.

[130] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE T PATTERN ANAL*, vol. 22, no. 8, pp. 888–905, 2000.

[131] T. Cour, F. Benezit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in *CVPR*, 2005.

[132] J. Wu, J. Zhu, and Z. Tu, "Reverse image segmentation: A high-level solution to a low-level task." in *BMVC*, 2014.

[133] V. Strassen, "Gaussian elimination is not optimal," *Numerische Mathematik*, vol. 13, no. 4, pp. 354–356, 1969.

[134] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences. Springer-Verlag Limited, 2008, pp. 95–110.

[135] I. CVX Research, "CVX: MATLAB software for disciplined convex programming, version 2.0," http://cvxr.com/cvx, Aug. 2012.

[136] W. Chen, H. Ji, and Y. You, "An augmented Lagrangian method for ℓ_1-regularized optimization problems with orthogonality constraints," *SIAM J SCI COMPUT*, vol. 38, no. 4, pp. B570–B592, 2016.

[137] B. Gao, X. Liu, X. Chen, and Y.-x. Yuan, "A new first-order algorithmic framework for optimization problems with orthogonality constraints," *SIAM J OPTIMIZ*, vol. 28, no. 1, pp. 302–332, 2018.

[138] M. Fortin and R. Glowinski, *Augmented Lagrangian methods: applications to the numerical solution of boundary-value problems*. Elsevier, 2000, vol. 15.

[139] H. Attouch, J. Bolte, and B. F. Svaiter, "Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and

regularized Gauss–Seidel methods," *MATH PROGRAM*, vol. 137, no. 1-2, pp. 91–129, 2013.

[140] H. Zhu, X. Zhang, D. Chu, and L.-Z. Liao, "Nonconvex and nonsmooth optimization with generalized orthogonality constraints: An approximate augmented Lagrangian method," *SIAM J SCI COMPUT*, pp. 1–42, 2017.

[141] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota, "Exploiting sparsity in semidefinite programming via matrix completion II: Implementation and numerical results," *MATH PROGRAM*, vol. 95, no. 2, pp. 303–327, 2003.

[142] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE T PATTERN ANAL*, no. 7, pp. 711–720, 1997.

[143] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *NIPS*, 2002, pp. 849–856.

[144] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *CVPR*, 2017.

[145] H. Kazemi, S. Soleymani, F. Taherkhani, S. Iranmanesh, and N. Nasrabadi, "Unsupervised image-to-image translation using domain-specific variational information bound," in *NIPS*, 2018.

[146] J. Nocedal, "Updating quasi-newton matrices with limited storage," *MATH COMP*, pp. 773–782, 1980.

[147] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *MATH PROGRAM*, pp. 503–528, 1989.

[148] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," in *ICLR*, 2018.

[149] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "Signsgd: Compressed optimisation for non-convex problems," in *ICML*, 2018.

[150] Q. Nguyen and M. Hein, "The loss surface of deep and wide neural networks," in *J MACH LEARN RES*, 2017, pp. 2603–2612.

[151] M. Hardt and T. Ma, "Identity matters in deep learning," in *ICLR*, 2017.

[152] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks," in *AISTATS*, 2015.

[153] D. Pathak, P. Krahenbuhl, and T. Darrell, "Constrained convolutional neural networks for weakly supervised segmentation," in *ICCV*, 2015.

[154] S. K. Roy, Z. Mhammedi, and M. Harandi, "Geometry aware constrained optimization techniques for deep learning," in *CVPR*, 2018.

[155] Z.-Q. Luo, W.-K. Ma, A. M.-C. So, Y. Ye, and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE SIGNAL PROC MAG*, pp. 20–34, 2010.

[156] G. Taylor, R. Burmeister, Z. Xu, B. Singh, A. Patel, and T. Goldstein, "Training neural networks without gradients: A scalable ADMM approach," in *ICML*, 2016.

[157] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *ICLR*, 2018.

[158] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in *NIPS*, 2014.

[159] M. Janzamin, H. Sedghi, and A. Anandkumar, "Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods," *arXiv preprint arXiv:1506.08473*, 2015.

[160] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, "Visualizing higher-layer features of a deep network," 2009.

[161] M. Carreira-Perpinan and W. Wang, "Distributed optimization of deeply nested systems," in *AISTATS*, 2014.

[162] Y. Bengio, N. L. Roux, P. Vincent, O. Delalleau, and P. Marcotte, “Convex neural networks,” in *NIPS*, 2006.

[163] Ö. Aslan, H. Cheng, X. Zhang, and D. Schuurmans, “Convex two-layer modeling,” in *NIPS*, 2013.

[164] Ö. Aslan, X. Zhang, and D. Schuurmans, “Convex deep learning via normalized kernels,” in *NIPS*, 2014.

[165] F. Bach, “Breaking the curse of dimensionality with convex neural networks,” *J MACH LEARN RES*, pp. 1–53, 2017.

[166] T. Frerix, T. Möllenhoff, M. Moeller, and D. Cremers, “Proximal backpropagation,” in *ICLR*, 2018.

[167] V. Ganapathiraman, X. Zhang, Y. Yu, and J. Wen, “Convex two-layer modeling with latent structure,” in *NIPS*, 2016.

[168] A. L. Yuille and A. Rangarajan, “The concave-convex procedure (CCCP),” in *NIPS*, 2002.

[169] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” in *AISTATS*, 2011.

[170] S. N. Ravi, T. Dinh, V. S. R. Lokhande, and V. Singh, “Constrained deep learning using conditional gradient and applications in computer vision,” *arXiv preprint arXiv:1803.06453*, 2018.

[171] P. Huang, E. Gonultas, S. Medjkouh, O. Castaneda, O. Tirkkonen, T. Goldstein, and C. Studer, “Representation-constrained autoencoders and an application to wireless positioning,” 2018.

[172] N. Japkowicz and S. Stephen, “The class imbalance problem: A systematic study,” *INTELL DATA ANAL*, pp. 429–449, 2002.

[173] Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang, “Cost-sensitive boosting for classification of imbalanced data,” *Pattern Recognition*, pp. 3358–3378, 2007.

[174] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data," *IEEE T NEUR NET LEAR*, pp. 3573–3587, 2018.

[175] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *ICML*, 2018.

[176] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[177] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *ICCV*, 2015.

[178] W. Deng, L. Zheng, Q. Ye, G. Kang, Y. Yang, and J. Jiao, "Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification," in *CVPR*, 2018.

[179] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *ICML*, 2014.

[180] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *ICML*, 2015.

[181] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, "Simultaneous deep transfer across domains and tasks," in *ICCV*, 2015.

[182] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *NIPS*, 2014.

[183] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, 2018.

[184] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *AAAI*, 2016.

[185] B. Sun and K. Saenko, “Deep coral: Correlation alignment for deep domain adaptation,” in *EECV*, 2016.

[186] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *NIPS*, 2014.

[187] M. Long, Z. Cao, J. Wang, and M. I. Jordan, “Conditional adversarial domain adaptation,” in *NIPS*, 2018.

[188] Z. Pei, Z. Cao, M. Long, and J. Wang, “Multi-adversarial domain adaptation,” in *AAAI*, 2018.

[189] X. Wang, L. Li, W. Ye, M. Long, and J. Wang, “Transferable attention for domain adaptation,” in *AAAI*, 2019.

[190] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE T KNOWL DATA EN*, vol. 22, no. 10, pp. 1345–1359, 2010.

[191] M. Ghifary, W. B. Kleijn, and M. Zhang, “Domain adaptive neural networks for object recognition,” in *PRICAI*. Springer, 2014, pp. 898–904.

[192] M. Long, H. Zhu, J. Wang, and M. I. Jordan, “Unsupervised domain adaptation with residual transfer networks,” in *NIPS*, 2016.

[193] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo, “Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation,” in *CVPR*, 2017.

[194] B. Sun, J. Feng, and K. Saenko, “Correlation alignment for unsupervised domain adaptation,” in *Domain Adaptation in Computer Vision Applications*. Springer, 2017, pp. 153–171.

[195] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, “Unsupervised pixel-level domain adaptation with generative adversarial networks,” in *CVPR*, 2017.

[196] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *J MACH LEARN RES*, vol. 17, no. 1, pp. 2096–2030, 2016.

[197] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, "Deep reconstruction-classification networks for unsupervised domain adaptation," in *ECCV*, 2016.

[198] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *CVPR*, 2017.

[199] W. Zhang, W. Ouyang, W. Li, and D. Xu, "Collaborative and adversarial network for unsupervised domain adaptation," in *CVPR*, 2018.

[200] J. Zhang, Z. Ding, W. Li, and P. Ogunbona, "Importance weighted adversarial nets for partial domain adaptation," in *CVPR*, 2018.

[201] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *ECCV*, 2010.

[202] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *CVPR*, 2017.

[203] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *INT J COMPUT VISION*, vol. 115, no. 3, pp. 211–252, 2015.