Semi-Supervised Learning for Scalable and Robust Visual Search

Jun Wang

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2011

ABSTRACT

Semi-Supervised Learning for Scalable and Robust Visual Search

Jun Wang

Unlike textual document retrieval, searching of visual data is still far from satisfactory. There exist major gaps between the available solutions and practical needs in both accuracy and computational cost. This thesis aims at the development of robust and scalable solutions for visual search and retrieval. Specifically, we investigate two classes of approaches: graph-based semi-supervised learning and hashing techniques. The graph-based approaches are used to improve accuracy, while hashing approaches are used to improve efficiency and cope with large-scale applications. A common theme shared between these two subareas of our work is the focus on semi-supervised learning paradigm, in which a small set of labeled data is complemented with large unlabeled datasets.

Graph-based approaches have emerged as methods of choice for general semi-supervised tasks when no parametric information is available about the data distribution. It treats both labeled and unlabeled samples as vertices in a graph and then instantiates pairwise edges between these vertices to capture affinity between the corresponding samples. A quadratic regularization framework has been widely used for label prediction over such graphs. However, most of the existing graph-based semi-supervised learning methods are sensitive to the graph construction process and the initial labels. We propose a new bivariate graph transduction formulation and an efficient solution via an alternating minimization procedure. Based on this bivariate framework, we also develop new methods to filter unreliable and noisy labels. Extensive experiments over diverse benchmark datasets demonstrate the superior performance of our proposed methods.

However, graph-based approaches suffer from the critical bottleneck in scalability since graph construction requires a quadratic complexity and the inference procedure costs even more. The widely used graph construction method relies on nearest neighbor search, which is prohibitive for large-scale applications. In addition, most large-scale visual search problems involve handling high-

dimensional visual descriptors, thereby causing another challenge in excessive storage requirement. To handle the scalability issue of both computation and storage, the second part of the thesis focuses on efficient techniques for conducting approximate nearest neighbor (ANN) search, which is key to many machine learning algorithms, including graph-based semi-supervised learning and clustering. Specifically, we propose Semi-Supervised Hashing (*SSH*) methods that leverage semantic similarity over a small set of labeled data while preventing overfitting. We derive a rigorous formulation in which a supervised term minimizes the empirical errors on the labeled data and an unsupervised term provides effective regularization by maximizing variance and independence of individual bits. Experiments on several large datasets demonstrate the clear performance gain over several state-of-the-art methods without significant increase of the computational cost.

The main contributions of the thesis include the following.

1. **Bivariate graph transduction**: a) a bivariate formulation for graph-based semi-supervised learning with an efficient solution by alternating optimization; b) theoretic analysis from the view of graph cut for the bivariate optimization procedure; c) novel applications of the proposed techniques, such as interactive image retrieval, automatic re-ranking for text based image search, and a brain computer interface (BCI) for image retrieval.

2. **Semi-supervised hashing**: a) a rigorous semi-supervised paradigm for hash functions learning with a tradeoff between empirical fitness on pair-wise label consistence and an information-theoretic regularizer; b) several efficient solutions for deriving semi-supervised hash functions, including an orthogonal solution using eigen-decomposition, a revised strategy for learning non-orthogonal hash functions, a sequential learning algorithm to derive boosted hash functions, and an extension to unsupervised cases by using pseudo labels.

Two parts of the thesis - bivariate graph transduction and semi-supervised hashing - are complimentary and can be combined to achieve significant performance improvement in both speed and accuracy. Hash methods can help build sparse graphs in a linear time fashion and greatly reduce the data size, but they lack sufficient accuracy. Graph-based methods provide unique capabilities to handle non-linear data structures with noisy labels but suffer from high computational complexity. The synergistic combination of the two offers great potential for advancing the state-of-the-art in large-scale visual search and many other applications.

Table of Contents

List of Figures

List of Tables

Glossary

SSL - Semi-Supervised Learning

GSSL - Graph-Based Semi-Supervised Learning

CBIR - Content-Based Image Retrieval

BM - b-Matching

LGC - Local and Global Consistency

GFHF - Gaussian Fields and Harmonic Functions

S3VMs - Semi-Supervised Support Vector Machines

TSVM - Transductive Support Vector Machines

LapSVM - Laplacian Support Vector Machines

GTAM - Graph Transduction via Alternating Minimization

GGMC - Greedy Gradient Max-Cut

LDST - Label Diagnosis though Self Tuning

CGL - Combined Graph Laplacian

BIP - Binary Integer Programming

MG-GTAM - Multi-Graph GTAM

ANN - Approximate Nearest Neighbors

LSH - Locality Sensitive Hashing

PCAH - Principal Component Analysis Based Hashing

SIKH - Shift Invariant Kernel based Hashing

BSSC - Boosting Similarity Sensitive Coding

RBMs - Restricted Boltzmann Machines

DBN - Deep Belief Networks

BRE - Binary Reconstruction Embedding

LAMP - Label-regularized Max-margin Partition

SH - Spectral Hashing

SSH - Semi-Supervised Hashing

SPLH - Sequential Projection Learning Based Hashing

USPLH - Unsupervised Sequential Projection Learning Based Hashing

TAG - Transductive Annotation by Graph

RNAi - RNA interference

HCS - High-Content Screening

LR-LGC - Label Regularized LGC

LR-GFHF - Label Regularized GFHF

BoW - Bag-of-Visual-Words

OOI - Objects of Interests

EEG - Electroencephalography

RSVP - Rapid Serial Visual Presentation

BCI - Brain Computer Interface

C3Vision - Cortically-Couple Computer Vision

HAC - Human Aided Computing

BCI-VPM - Brain Computer Interface and Visual Pattern Mining

Acknowledgments

A dissertation does not materialize on its own. I could not have completed this work without the crucial help of many individuals, all of whom have left a lasting imprint on me. I would like to express my sincere gratitude to my thesis advisor, Professor Shih-Fu Chang, who guided me in the right direction with generosity and helped me see my own perpetual insufficiency. I am also grateful for the precious freedom he granted me to work with different collaborators from Harvard University, Cornell University and Google Research.

I would like to thank Professor Tony Jebara, who not only taught me the fundamental knowledge of machine learning and also inspired me to identify and solve new problems. I would also like to thank Professor Paul Sajda and the members in his team, especially Eric Pohlmeyer and Barbara Hanna, for their great support and invaluable collaborations. I would like to thank Professor Stephen Wong and Professor Xiaobo Zhou, who presented me invaluable opportunities in biomedical research. Thanks also to my mentor from my internship at Google Research, Dr. Sanjiv Kumar, who always encouraged and supported me whenever there were seemingly insurmountable difficulties. I also would like to thank Professor Shree Nayar, whose computer vision classes gave me fruitful insights and motivations. A special thank you to all the members of my defense committee, Professor Xiaodong Wang, Professor Dan Ellis, and Professor David Waltz, who ungrudgingly sacrificed their time and effort to put the finishing touch on this thesis.

Thanks also to all DVMM folks (current and previous), especially Tian-Tsong Ng, Lexing Xie, Jessie Hsu, Yu-Gang Jiang, Lyndon Kennedy, Winston Hsu, Eric Zavesky, Wei Jiang and Zhenguo Li, who constantly reminded me that I am never alone. I am also very grateful to Shouzheng Liu, Yongtao Su, Shiqian Ma, and Wenjia Jing for their invaluable discussions and suggestions.

I would like to thank my parents and my brother for their generous and everlasting support. Finally, I am deeply grateful to my fiancée, Jennifer Chu, for her love, understanding, encouragement, and sharing in my Ph.D. journey.

To my parents

Chapter 1

Introduction

1.1 Motivation

Due to the advance of modern imaging techniques, rich media, e.g., images and videos, overflow in everyday life. For example, there is explosive growth of online media data. The photo sharing website, *Flickr*, has over 5 billion images [1] and is being uploaded at the rate of over 3000 images per minute. Another media sharing website, *YouTube*, receives more than 24 hours of uploaded videos per minute [2]. Besides the impact on the average person's everyday life, modern imaging technique also provides an unprecedented ability to record phenomena in the micro-world for scientific research, such as microscopic imaging for biomolecular research. For example, genome-wide high content screening has recently created an enormous amount of microscopic images that provide important visual clues for accessing gene functions and designing drugs [93][178].

There is an emerging need to search and retrieve relevant content from such massive visual databases. The widely-used commercial search engines, like *Google* and *Bing*, heavily rely on keyword matching techniques, and their search performance is often unsatisfactory due to erroneous textural tag information. Content based image retrieval (CBIR) has attracted substantial attention over the past decade [38][123][141][142]. Among the current CBIR research, the supervised methods, such as concept detection, have been extensively explored and applied for visual

[1] The 5 billionth image was uploaded on September 19th, 2010, and it can be viewed at http://blog.flickr.net/en/2010/09/19/5000000000/.

[2] This statistic was recorded on March 17, 2010.

search [75][76][84][170]. Briefly speaking, these methods first define some concept categories, including objects, scenes, events and so on, and then train classifiers for each category. These trained classifiers are used to classify and index query images as well as the images in the databases to generate search results in response to user queries.

However, there are some critical challenges in applying such supervised methods to deal with applications over web-scale image databases. The first is the scalability issue. Since real large-scale image databases often contain millions, or even billions, of samples and the visual descriptors commonly contain hundreds or thousands of dimensions, there result both computational and storage bottlenecks. Second, usually only a small portion of the samples in the real image databases are associated with class labels. The available label information, such as the tags associated with the photos on *Flickr*, is often noisy and unreliable. All these challenging conditions make the supervised methods, like concept detection, infeasible for large-scale applications in the real world. Therefore, in this thesis, we focus on semi-supervised learning techniques and develop scalable and robust solutions for visual search applications. The two major technical components of this thesis, *bivariate graph transduction* and *semi-supervised hashing*, are summarized below.

1.2 Methodology

1.2.1 Bivariate Graph Transduction

Since labeled samples are scarce but unlabeled samples are abundant in reality, the learning paradigm considering both labeled and unlabeled data, i.e., semi-supervised learning (*SSL*), has proliferated in applied machine learning systems. Among various *SSL* methods, graph-based approaches have become popular due to its high accuracy and computational efficiency. Graph-based *SSL* (*GSSL*) first uses the input data to construct a weighted graph, where both labeled and unlabeled samples are treated as graph vertices, and weighted edges between these vertices indicate affinity between the corresponding samples. Then the small portion of labeled vertices is used to perform propagation or diffusion on the graph which provides unlabeled nodes with predicted labels. Designing a robust label diffusion algorithm for such graphs is a widely studied problem and many recent methods are reported, as surveyed in [27][180][181].

When applying the existing *GSSL* approaches to the visual search system, there remain critical

challenges in both robustness and scalability, two highly desired properties for practical applications. These limitations result from the heuristic process used in graph construction and the strong dependency of results on the labels. Therefore, existing approaches are extremely sensitive to the pre-computed graph and initial given labels [74][156]. Many practical situations, such as those associated with uneven sampling [74], imbalanced labels [155], noninformative labels, and erroneous labels [157], could ruin the label propagation process and generate unsatisfactory predictions. To handle these challenging situations, this thesis proposes a bivariate formulation for graph-based semi-supervised learning, where both the prediction function and the label matrix are treated as optimization variables. Though the final formulation, turning to be a mixed integer problem, is NP-hard, we design a very efficient solution via an alternating optimization procedure [156]. We further prove the equivalence between this bivariate formulation and a constrained Max-Cut problem, and provide a solution based on gradient greedy graph cut. Two advanced extensions of the bivariate framework, including a label tuning strategy for handling problematic labels and multi-graph based transduction, are also developed in this thesis. We validate the effectiveness of the proposed bivariate graph transduction methods through extensive experiments, and compare with state-of-the-art techniques on both artificial data and real applications from different domains.

1.2.2 Semi-Supervised Hashing

Note that all graph-based techniques require at least a quadratic complexity in computation since graph construction costs $\mathcal{O}(n^2)$ and the label diffusion procedure needs even more computational cost. Besides computational constraints, construction of the full adjacency graph is infeasible due to memory cost. Therefore, many *GSSL* techniques use sparse graphs, like KNN and b-matched graphs [74], in which each vertex is connected with a small number of neighbors. However, finding nearest neighbors in a large dataset is challenging. Nowadays, image datasets containing millions or billions of samples are quite common, with the data dimensionality exceeding hundreds or thousands. An exhaustive linear search to find nearest neighbors is infeasible for such gigantic datasets. Fortunately, in many applications, it is sufficient to return approximate nearest neighbors (ANN) instead of exact nearest neighbors. Conventional tree-based approximate nearest neighbor search techniques cannot deal with high dimensional data and also require large memory. On the contrary, hashing approaches achieve fast query time and require substantially reduced storage by indexing

data with compact hash codes.

Among various hashing techniques, linear projection based hashing functions remain popular due to their simplicity and efficiency. There are mainly two categories of linear projection based hashing. The first category relies on random projections, such as locality sensitive hashing (*LSH*). The second category of methods learn data-driven projections. For example, spectral hashing (*SH*) uses principal projections and usually performs better than *LSH*. However, for typical CBIR applications, even returning the exact nearest neighbors does not guarantee search quality. This is due to the well-known problem called *semantic gap*, where the high level semantic description of visual content often differs from the low level visual descriptors [141]. In summary, due to the lack of appropriate semantic visual descriptors and associated similarity measures, existing hashing based ANN searches do not yield satisfactory results.

To handle the above challenging issues, we propose a novel hashing method, called semi-supervised hashing (*SSH*), to learn efficient hash codes. Different from existing data-driven hash techniques, *SSH* uses both semantic similarity/dissimilarity information associated with a small set of sample pairs and the distribution properties of the whole dataset (both labeled and unlabeled) to obtain compact and robust binary codes. The formulation of *SSH* contains two components: a supervised term measuring empirical consistency over pair-wise labeled data and an unsupervised term reflecting desirable properties, e.g., maximizing the entropy of each hash bit. Starting from this formulation and using some relaxation processes, we derive three different versions of semi-supervised hashing methods, i.e., orthogonal, non-orthogonal, and sequential hashing. Finally, we propose an unsupervised extension of the sequential hashing method, where a set of pseudo-labels are generated and incorporated with the sequential learning procedure. Experiments on several large datasets, with up to 80 million points, clearly demonstrate the performance gain over several state-of-the-art methods without significant increase of computational cost.

1.3 Evaluation and Applications

Besides the theoretical formulation and validation of the proposed approaches, empirical evaluations, including both quantitative and qualitative testing, are also extensively investigated. We first evaluate the performance of the proposed techniques with widely used artificial data, e.g., two-moon

and two-circle point clouds, and standard benchmark data, e.g., UCI Machine Learning Repository datasets [52] and *Caltech 101* dataset [46]. In addition, we also apply our proposed methods to real image data from different domains, such as the high-content microscopic imagery [161], satellite imagery [160], and *Flickr* data collections [157]. Several different evaluation metrics, including Mean Average Precision (MAP) [149], are used in our experiments. For the largest image collection, which includes around 80 million tiny images, the qualitative evaluation is also provided.

Based on the proposed techniques, we also develop several visual search systems for different applications.

1. **Interactive Image Search**: We first explore an interactive search mode that keeps users in the loop and incorporates relevance feedback. A practical system, named Columbia TAG (Transductive Annotation by Graph) system [3] [154], is designed to facilitate both rapid retrieval and exploration of large image and video collections. It incorporates novel graph-based label propagation methods and intuitive graphic user interfaces (GUI) for relevance feedback. The system allows additional positive images/videos matching the user's interest to be quickly discovered. It can be used as a fast search system alone, or as a bootstrapping system for developing additional target recognition tools needed in critical application domains such as intelligence, surveillance, consumer media, biomedical informatics [155], and the Web.

2. **Automatic Image Search**: We design an automatic mode for web image search through refining the keyword based search results. In this framework, the textual tags are treated as potentially incorrect labels and a label refinement method, named *label diagnosis through self tuning* (*LDST*), is applied to correct labeling mistakes and to propagate label information over the entire collection. The experimental results over a collection of *Flickr* images confirm the significant performance gain over both text search baselines and state-of-the-art re-ranking methods [157].

3. **Hybrid Image Search with Brain Computer Interface**: Through integration with an Electroencephalogram (EEG)-based brain computer interface (BCI), we propose a hybrid paradigm that marries the strengths of human vision and graph-based learning systems in a unique and synergistic way: human vision for its superb capability in detecting general objects in diverse

[3] Demo: http://www.ee.columbia.edu/ln/dvmm/researchProjects/CTAG/tag.htm

and complex conditions, and content analytics for automatic processing of large volumes of data [127] [160].

4. **Large-Scale Visual Indexing and Search**: Finally, we apply the proposed semi-supervised hashing techniques [158] [159] to indexing and searching large-scale image databases, including a *Flickr* image collection with tens of thousands of samples [33] and the largest available image collection with around 80 million tiny images [147].

1.4 Thesis Overview

The following describes the organization of the remainder of the thesis. In Chapter 2, we start with a brief survey of semi-supervised learning, especially focusing on the existing graph-based *SSL* techniques, then present the background of graph construction and label propagation over graphs, and finally introduce some challenging issues, e.g., label dependency and heuristic graph construction. Chapter 3 presents the bivariate graph transduction framework and related theoretical proofs, followed by two advanced extensions, including one for handling mislabeled instances and the other for multi-graph based transductive learning. Chapter 4 presents the semi-supervised hashing techniques for approximate nearest neighbor search. In this chapter, we provide a brief survey of the related work, detailed theoretical formulation of our hashing approaches, including three different semi-supervised hashing methods and an unsupervised extension, followed by extensive experiments and comparison studies with emerging approaches on several large datasets. In Chapter 5, we present several real image search applications using the proposed techniques, including an interactive search mode with user feedback and an automatic search mode through refining keyword-based search results. Chapter 6 describes a very unique image retrieval system via integrating machine learning approaches with a Brain Machine Interface. Finally, in Chapter 7, we conclude the thesis and discuss future work.

Chapter 2

Graph-Based Semi-Supervised Learning: Background & Issues

2.1 Introduction

In many real applications, labeled samples are scarce but unlabeled samples are abundant. The learning paradigm that considers both labeled and unlabeled data, i.e., semi-supervised learning (*SSL*), has been extensively explored for practical machine learning systems. Of various *SSL* methods, graph-based approaches have become popular due to their high accuracy and computational efficiency. Graph-based semi-supervised learning (*GSSL*) treats both labeled and unlabeled samples from the dataset as vertices in a graph and builds pairwise edges between these vertices which are weighted by the affinity between the corresponding samples. The small portion of vertices with labels are then used by *SSL* methods to perform graph partition or information propagation to predict labels for unlabeled vertices. For instance, the graph mincuts approach formulates the label prediction as a graph cut problem [20][21]. Other *GSSL* methods, like graph transductive learning, formulate the problem as a regularized function estimation over an undirected weighted graph. Specifically, they try to optimize the tradeoff between the accuracy of the classification function on labeled samples and a regularization term that favors a smooth function. The weighted graph and the optimal function ultimately propagate label information from labeled data to unlabeled data and thus accomplish transductive predictions. Popular algorithms for *GSSL* include graph cuts [20][21][80][92], graph random walks [5][143], manifold regularization [14][15][136][138],

and graph regularization [176][182]. Comprehensive surveys can be found in [27][180][181]. Here, we provide a brief overview of the fundamental components and open issues of graph-based *SSL*.

2.2 Notations for Graph Representation

We define the notations of graph representations, which will be used throughout the thesis. Assume we are given *iid* (independent and identically distributed) labeled samples $\{(\mathbf{x}_1, z_1), \ldots, (\mathbf{x}_l, z_l)\}$ as well as unlabeled samples $\{\mathbf{x}_{l+1}, \ldots, \mathbf{x}_{l+u}\}$ drawn from a distribution $p(\mathbf{x}, z)$. Define the set of labeled inputs as $\mathbf{X}_l = \{\mathbf{x}_1, \ldots, \mathbf{x}_l\}$ with cardinality $|\mathbf{X}_l| = l$ and the set of unlabeled inputs $\mathbf{X}_u = \{\mathbf{x}_{l+1}, \ldots, \mathbf{x}_{l+u}\}$ with cardinality $|\mathbf{X}_u| = u$. The labeled set $\mathbf{X}_l$ is associated with labels $\mathcal{Z}_l = \{z_1, \cdots, z_l\}$, where $z_i \in \{1, \cdots, c\}, i = 1, 2, \cdots, l$. The goal of semi-supervised learning is to infer the missing labels $\{z_{l+1}, \cdots, z_n\}$ corresponding to the unlabeled data $\{\mathbf{x}_{l+1}, \cdots, \mathbf{x}_n\}$, where typically $l << n$ ($l + u = n$). A crucial component of *GSSL* is the estimation of a weighted sparse graph $\mathcal{G}$ from the input data $\mathbf{X} = \mathbf{X}_l \cup \mathbf{X}_u$. Subsequently, a labeling algorithm uses $\mathcal{G}$ and the known labels $\mathcal{Z}_l = \{z_1, \ldots, z_l\}$ to provide estimates $\hat{\mathcal{Z}}_u = \{\hat{z}_{l+1}, \ldots, \hat{z}_{l+u}\}$ which try to approximate the true labels $\mathcal{Z}_u = \{z_{l+1}, \ldots, z_{l+u}\}$ as measured by an appropriately chosen loss function.

In this thesis, assume the undirected graph converted from the data $\mathbf{X}$ is represented by $\mathcal{G} = \{\mathbf{X}, \mathbf{E}\}$, where the set of vertices is $\mathbf{X} = \{\mathbf{x}_i\}$ and the set of edges is $\mathbf{E} = \{e_{ij}\}$. Each sample $\mathbf{x}_i$ is treated as a vertex and the weight of edge e_{ij} is w_{ij}. Typically, one uses a kernel function $k(\cdot)$ over pairs of points to compute weights. For instance, $w_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ with the RBF kernel has been a popular choice. The weights for edges are used to build a weight matrix which is denoted by $\mathbf{W} = \{w_{ij}\}$. Similarly, the vertex degree matrix $\mathbf{D} = diag([d_1, \cdots, d_n])$ is defined as $d_i = \sum_{j=1}^{n} w_{ij}$. The Laplacian of graphs is defined as:

$$\mathbf{\Delta} = \mathbf{D} - \mathbf{W}. \tag{2.1}$$

The graph Laplacian can be viewed as an operator on the space of functions f which can be used to define a regularization measure of smoothness over strongly-connected regions in a graph [34]:

$$\langle f, \mathbf{\Delta} \rangle = \sum_i \sum_j w_{ij} \left\| f(\mathbf{x}_i) - f(\mathbf{x}_j) \right\|^2. \tag{2.2}$$

In the literature, there are different strategies to normalize the graph Laplacians [152]. The two typical versions are

$$\mathbf{L}_{sym} = \mathbf{D}^{-1/2}\mathbf{\Delta}\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2} \tag{2.3}$$

$$\mathbf{L}_{rw} = \mathbf{D}^{-1}\mathbf{\Delta} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}, \tag{2.4}$$

where $\mathbf{L}_{sym}$ denotes a symmetric version of the graph Laplacian and $\mathbf{L}_{rw}$ is closely related to a random walk since $\mathbf{D}^{-1}\mathbf{W}$ can be viewed as a transition matrix [1]. Although the author of [152] advocates using $\mathbf{L}_{rw}$ in spectral clustering tasks, we follow the work in [174] and use $\mathbf{L}_{sym}$ in this thesis. Without specific declaration, we use $\mathbf{L}$ to represent the symmetrically normalized version $\mathbf{L}_{sym}$ and define the corresponding smoothness measurement of functions f over a graph as

$$\langle f, \mathbf{L} \rangle = \sum_i \sum_j w_{ij} \left\| \frac{f(\mathbf{x}_i)}{\sqrt{d_i}} - \frac{f(\mathbf{x}_j)}{\sqrt{d_j}} \right\|^2 . \tag{2.5}$$

Finally, the label information is formulated as a label matrix $\mathbf{Y} = \{y_{ij}\} \in \mathbb{B}^{n \times c}$, where $y_{ij} = 1$ if sample $\mathbf{x}_i$ is associated with label j for $j \in \{1, 2, \cdots, c\}$, i.e., $z_i = j$, and $y_{ij} = 0$ otherwise. For single label problems (as opposed to multi-label problems), the constraints $\sum_{j=1}^{c} y_{ij} = 1$ are also imposed. Moreover, we will often refer to row and column vectors of such matrices. For instance, the ith row and jth column vectors of $\mathbf{Y}$ are denoted as $\mathbf{Y}_{i\cdot}$ and $\mathbf{Y}_{\cdot j}$, respectively. Let $\mathbf{F} = f(\mathbf{X})$ be the values of classification function over the dataset $\mathbf{X}$. Most of the *GSSL* methods then utilize the graph quantity $\mathbf{W}$ as well as the known labels to recover a continuous classification function $\mathbf{F} \in \mathbb{R}^{n \times c}$ by minimizing a predefined cost on the graph.

2.3 Graph Construction for Semi-Supervised Learning

To estimate $\hat{\mathcal{Z}}_u = \{\hat{z}_{l+1}, \ldots, \hat{z}_{l+u}\}$ using $\mathcal{G}$ and the known labels $\mathcal{Z}_l = \{z_1, \ldots, z_l\}$, the first step is to convert the data points $\mathbf{X} = \mathbf{X}_l \cup \mathbf{X}_u$ to a graph $\mathcal{G} = \{\mathbf{X}, \mathbf{E}\}$. In this section, the graph construction $\mathbf{X} \rightarrow \mathcal{G}$ is addressed. Given input data $\mathbf{X}$ with cardinality $|\mathbf{X}| = l + u$, graph construction produces a graph $\mathcal{G} = (\mathbf{X}, \mathbf{E})$ consisting of $n = l + u$ vertices where each vertex is associated with the sample $\mathbf{x}_i$. Furthermore, take $\mathbf{E}$ to be the set of undirected edges between pairs of vertices. It is common to also associate a weighted symmetric adjacency matrix $\mathbf{W}$ with the edges $\mathbf{E}$ in $\mathcal{G}$ where $\mathbf{W} = \{w_{ij}\} \in \mathbb{R}^{n \times n}$ has zeros on its diagonal and each scalar w_{ij} represents

the edge weight between vertex $\mathbf{x}_i$ and vertex $\mathbf{x}_j$. The estimation of $\mathcal{G}$ from $\mathbf{X}$ usually proceeds in two steps.

The first step is to compute a similarity score between all pairs of vertices using a similarity function. This creates a full adjacency matrix, $\mathbf{K} \in \mathbb{R}^{n \times n}$, where $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ is computed using kernel function $k(\cdot)$ to measure sample similarity. Subsequently, in the second step of graph construction, the matrix $\mathbf{K}$ is sparsified and reweighted to produce the final matrix $\mathbf{W}$. Sparsification is important since it leads to improved efficiency, better accuracy, and robustness to noise in the label inference stage. Furthermore, the kernel function $k(\cdot)$ is often only locally useful as a similarity and does not recover reliable weights between pairs of samples that are relatively far apart.

2.3.1 Graph Sparsification

Starting with the fully connected matrix $\mathbf{K}$, sparsification removes edges by recovering a binary matrix $\mathbf{B} \in \mathbb{B}^{n \times n}$ where $\mathbf{B}_{ij} = 1$ indicates that an edge is present between sample $\mathbf{x}_i$ and $\mathbf{x}_j$, and $\mathbf{B}_{ij} = 0$ indicates the edge is absent (assume $\mathbf{B}_{ii} = 0$ unless otherwise noted). Here we will primarily investigate two graph sparsification algorithms: neighborhood approaches including the k-nearest and ϵ neighbors algorithms, and matching approaches such as b-matching (BM) [44]. All such methods operate on the matrix $\mathbf{K}$ or, equivalently, the distance matrix $\mathcal{D} \in \mathbb{R}^{n \times n}$ obtained from $\mathbf{K}$ element-wise as $\mathcal{D}_{ij} = \sqrt{\mathbf{K}_{ii} + \mathbf{K}_{jj} - 2\mathbf{K}_{ij}}$ since it is possible to convert a similarity function $k(\cdot)$ into a distance function via

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{k(\mathbf{x}_i, \mathbf{x}_i) + k(\mathbf{x}_j, \mathbf{x}_j) - 2k(\mathbf{x}_i, \mathbf{x}_j)}\ . \tag{2.6}$$

Sparsification via Neighborhood Methods: There are two typical ways to build a neighborhood graph: the ϵ-neighborhood graph connecting samples within a distance of ϵ, and the kNN graph connecting k closest samples. Recent studies show the dramatic influences that different neighborhood methods have on clustering techniques [25][102]. In practice, the kNN graph remains a more common approach since it is more adaptive to scale variation and data density anomalies while an improper threshold value in the ϵ-neighborhood graph may result in disconnected components or subgraphs in the dataset or even isolated singleton vertices, as shown in Figure 2.1(a) and Figure 2.2(d). In this thesis, we often use kNN neighborhood graphs since the ϵ-neighborhood

graphs provide consistently weaker performance. In the remainder of this thesis, we will use neighborhood and kNN neighborhood graph interchangeably without specific declaration.

More specifically, the k-nearest neighbor graph is a graph in which two vertices $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected by an edge if the distance $\mathcal{D}_{ij}$ between $\mathbf{x}_i$ and $\mathbf{x}_j$ is within or equal k-th smallest among the distances from $\mathbf{x}_i$ to other samples in $\mathbf{X}$. Roughly speaking, the k-nearest neighbors algorithm starts with a matrix $\hat{\mathbf{B}}$ of all zeros and for each point, searches for the k closest points to it (without considering itself). If a point j is one of the k closest neighbors to i, then we set $\hat{\mathbf{B}}_{ij} = 1$. It is straightforward to show that the k-nearest neighbors search solves the following optimization problem:

$$\begin{aligned} &\min_{\hat{\mathbf{B}} \in \mathbb{B}} \sum\nolimits_{ij} \hat{\mathbf{B}}_{ij} \mathcal{D}_{ij} \\ &s.t. \ \sum\nolimits_{j} \hat{\mathbf{B}}_{ij} = k, \hat{\mathbf{B}}_{ii} = 0, \ \forall i, j \ \in 1, \ldots, n. \end{aligned} \tag{2.7}$$

Then the final solution is produced by symmetrizing $\hat{\mathbf{B}}$ as $\mathbf{B}_{ij} = \max(\hat{\mathbf{B}}_{ij}, \hat{\mathbf{B}}_{ji})$ [4]. This greedy algorithm is in fact not solving a well defined optimization problem over symmetric binary matrices. In addition, since it produces a symmetric matrix only via the *ad hoc* maximization over $\hat{\mathbf{B}}$ and its transpose, the solution $\mathbf{B}$ it produces does not satisfy the equality $\sum_k \mathbf{B}_{ij} = k$, but, rather, only satisfies the inequality $\sum_j \mathbf{B}_{ij} \geq k$. Ironically, despite conventional wisdom and the nomenclature, the k-nearest neighbors algorithm is producing an undirected subgraph with more than k neighbors for each vertex. This motivates researchers to investigate the b-matching algorithm which actually achieves the desired output.

Sparsification via b-Matching: The b-matching problem generalizes maximum weight matching, i.e., linear assignment problem, where the objective is to find the binary matrix to minimize the optimization problem

$$\begin{aligned} &\min_{\mathbf{B} \in \mathbb{B}} \sum\nolimits_{ij} \mathbf{B}_{ij} \mathcal{D}_{ij} \\ &s.t. \ \sum\nolimits_{j} \mathbf{B}_{ij} = b, \mathbf{B}_{ii} = 0, \mathbf{B}_{ij} = \mathbf{B}_{ji}, \ \forall i, j \ \in 1, \ldots, n, \end{aligned} \tag{2.8}$$

achieving symmetry directly without post-processing. Here, the symmetric solution is recovered up-front by enforcing the additional constraints $\mathbf{B}_{ij} = \mathbf{B}_{ji}$. The matrix then satisfies the equality

[4] It is possible to replace the maximization operator with minimization to produce a symmetric matrix, yet in the setting $\mathbf{B} = \min(\hat{\mathbf{B}}, \hat{\mathbf{B}}^\top)$, the solution $\mathbf{B}$ only satisfies the inequality $\sum_j \mathbf{B}_{ij} \leq k$ and not the desired equality.

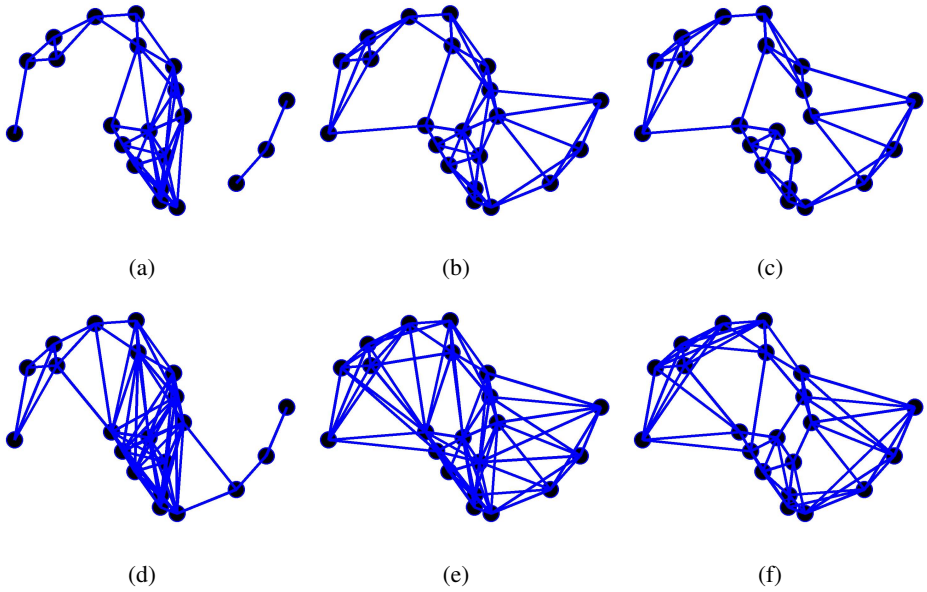
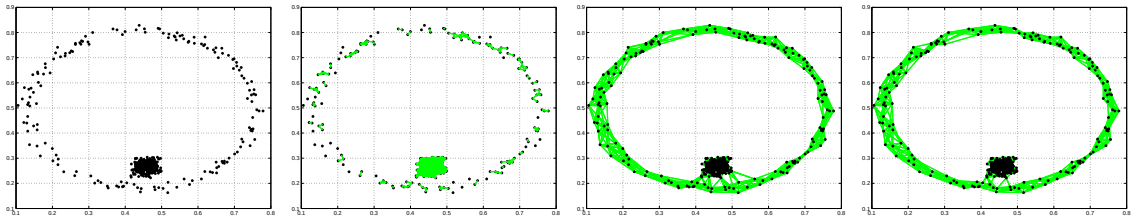


Figure 2.1: A comparison of the neighborhood algorithms and the b-matching algorithm for graph construction. a) & d) ϵ-neighborhood connectivity; b) & e) kNN connectivity for $k = 4$ and $k = 6$, respectively; c) & f) b-matching connectivity for $b = 4$ and $b = 6$, respectively. It is easy to verify that b-matching technique generates a more balanced (or regular) graph with equal vertex degree.

$\sum_j \mathbf{B}_{ij} = \sum_i \mathbf{B}_{ij} = b$ strictly. The solution to Eq. (2.8) is not quite as straightforward or efficient as the greedy k-nearest neighbors algorithm. A polynomial time $\mathcal{O}(bn^3)$ solution has been known, yet recent advances show that much faster alternatives are possible via (guaranteed) loopy belief propagation [67].

In Figure 2.1, an intuitive demonstration of neighborhood graphs and b-matching graphs is shown, where appropriate values of ϵ were chosen to make the total number of edges in ϵ-neighborhood graphs comparable to those in other graphs. Compared with the neighborhood graphs, the b-matching graph is balanced or b-regular. In other words, each vertex in the b-matched graph has exactly b edges connecting it to other vertices. This advantage plays a key role when conducting label propagation on typical samples $\mathbf{X}$ that are unevenly and non-uniformly distributed. An efficient implementation for the b-matching problem is available online from the authors of the loopy belief propagation method [67], which handles both unipartite and bipartite graphs. In addition, the faster belief propagation implementation has been developed, as described in Appendix A. It is possible to apply the method to large scale problems without significantly increasing the runtime of the overall



(a) (b) (c) (d)

Figure 2.2: The synthetic dataset used for demonstrating different graph construction approaches. a) The synthetic data; b) The ϵ-nearest neighbor graph; c) The k-nearest neighbor graph; d) The b-matched graph.

semi-supervised learning procedure.

Previous work also applied b-matching for spectral clustering through removing spurious edges and sparsifying a graph from the original fully connected adjacency matrix [73]. For example, in Figure 2.2, this dataset clearly contains two clusters of points, a dense Gaussian cluster surrounded by a ring cluster. Furthermore, the cluster data is unevenly sampled; one cluster is dense and the other is fairly sparse. In this example, the k-nearest neighbor graph constantly generates many cross-cluster edges while b-matching efficiently alleviates this problem by removing most of the improper edges. The example clearly shows that the b-matching technique produces regular graphs which could overcome the drawback of cross-structure linkages often generated by nearest neighbor methods. This intuitive study confirms the importance of graph construction methods and advocates b-matching as a valuable alternative to k-nearest neighbors, a method that many practitioners expect to produce regular undirected graphs, though in practice often generates irregular graphs.

2.3.2 Graph Edge Re-Weighting

Once a graph has been sparsified and a binary matrix $\mathbf{B}$ is computed and used to delete unwanted edges, several procedures can then be used to update the weights in the matrix $\mathbf{K}$ to produce a final set of edge weights $\mathbf{W}$. Specifically, whenever $\mathbf{B}_{ij} = 0$, the edge weight is also $\mathbf{W}_{ij} = 0$; however, $\mathbf{B}_{ij} = 1$ implies that $\mathbf{W}_{ij} \geq 0$. Three possible approaches are considered here for estimating the non-zero components of $\mathbf{W}$.

Binary Weighting: The simplest approach for building the weighted graph is the *binary* weighting approach, where all the linked edges in the graph are given the weight 1 and the edge weights of disconnected vertices are given the weight 0. In other words, this setting simply uses $\mathbf{W} = \mathbf{B}$. However, this uniform weight on graph edges can be sensitive, particularly if some of the graph vertices were improperly connected by the sparsification procedure (either the neighborhood based procedures or the b-matching procedure).

Gaussian Kernel Weighting: An alternative approach is *Gaussian kernel* weighting which is often applied to modulate sample similarity. Therein, the edge weight between two connected samples $\mathbf{x}_i$ and $\mathbf{x}_j$ is computed as:

$$w_{ij} = \mathbf{B}_{ij} \exp\left(-\frac{d(\mathbf{x}_i, \mathbf{x}_j)}{2\sigma^2}\right), \tag{2.9}$$

where the function $d(\mathbf{x}_i, \mathbf{x}_j)$ evaluates the dissimilarity of samples $\mathbf{x}_i$ and $\mathbf{x}_j$, and σ is the kernel bandwidth parameter. There are many choices for the distance function $d(\cdot)$ including any ℓ_p distance and the χ^2 distance, as listed below:

$$d_{\ell_1}(\mathbf{x}_i, \mathbf{x}_j) = |\mathbf{x}_i - \mathbf{x}_j| \tag{2.10}$$

$$d_{\ell_2}(\mathbf{x}_i, \mathbf{x}_j) = \| \mathbf{x}_i - \mathbf{x}_j \| \tag{2.11}$$

$$d_{\chi^2}(\mathbf{x}_i, \mathbf{x}_j) = \sum_k \frac{(\mathbf{x}_{i,k} - \mathbf{x}_{j,k})^2}{\mathbf{x}_{i,k} + \mathbf{x}_{j,k}}. \tag{2.12}$$

The ℓ_2 distance has been widely used in previous research [180], while χ^2 distance is useful for histograms [172]. Moreover, the cosine distance is another straightforward way to compute scale invariant sample similarity and is commonly used for text classification [14].

Locally Linear Reconstruction Based Weighting: Another way of estimating edge weight for the connected edges is motivated by the locally linear embedding technique presented in [122]. In this procedure, a novel edge weighting model using the non-negative coefficients of the *locally linear reconstruction* is performed as described in [153]. Given the sparse connectivity matrix $\mathcal{P}$, the error of a locally linear reconstruction of the sample $\mathbf{x}_i$, given its (sparse) connectivity information (or neighborhood information, equivalently) is defined as:

$$\varepsilon_i = \left\| \mathbf{x}_i - \sum_{j=1}^{n} \mathbf{B}_{ij} w_{ij} \mathbf{x}_j \right\|^2. \tag{2.13}$$

The best local linear reconstruction can be achieved by minimizing the above reconstruction error. Since direct optimization on Eq. (2.13) could generate negative coefficients for w_{ij}, constraints are imposed on the optimization such that the weights are non-negative and normalize to unity as follows:

$$\min_{W} \sum_{i} \left\| \mathbf{x}_i - \sum_{j=1}^{n} \mathbf{B}_{ij} w_{ij} \mathbf{x}_j \right\|^2 \qquad (2.14)$$
$$s.t. \sum\nolimits_j w_{ij} = 1, w_{ij} \geq 0.$$

The above can be solved as standard quadratic programming problems, as described in [153]. By considering all neighborhoods for each point as determined by $\mathbf{B}_{ij}$, it is possible to reconstruct an entire set of edge weights. This gives another procedure for setting the edge weights $\mathbf{W}$ from the sparsified binary connectivity $\mathbf{B}$.

This final step in the graph construction procedure ensures that the unlabeled data $\mathbf{X}$ has now been converted into a graph $\mathcal{G}$ with a weighted and undirected sparse matrix $\mathbf{W}$. Given this graph and some initial label information $\mathbf{Y}_l$, any of the current popular algorithms for graph-based *SSL* can be used to solve the labeling problem.

2.4 Label Diffusion and Inference over Graphs

Given the constructed graph $\mathcal{G} = \{\mathbf{X}, \mathbf{E}\}$, whose geometric structure is represented by the weight matrix $\mathbf{W}$, the label inference task is to diffuse the known labels $\mathcal{Z}_l$ to all the unlabeled vertices $\mathbf{X}_u$ in the graph and estimate $\hat{\mathcal{Z}}_u$. Designing a robust label diffusion algorithm for such graphs is a widely studied problem and many recent methods are surveyed in [27][180][181].

Here we are particularly interested in a category of approaches that estimates the prediction function $\mathbf{F} \in \mathbb{R}^{n \times c}$ through minimizing a quadratic cost function defined over the graph. The cost function typically involves a tradeoff between the smoothness of the function over the graph of both labeled and unlabeled data, i.e., consistence of the predictions on closely connected vertices, and the accuracy of the function at fitting the label information on the labeled vertices. Many well-known methods like the Gaussian fields and harmonic functions (*GFHF*) method [182] and the local and global consistency (*LGC*) method [174] fall in this category, as does our proposed method of graph transduction via alternating minimization (*GTAM*) [156].

Both *LGC* and *GFHF* define a cost function $\mathcal{Q}$ that involves the combined contribution of two penalty terms: the global smoothness Q_{smooth} and local fitting accuracy Q_{fit}. The final prediction function $\mathbf{F}$ is obtained by minimizing the cost function as:

$$\mathbf{F}^* = \arg\min_{\mathbf{F}\in\mathbb{R}^{n\times c}} \mathcal{Q}(\mathbf{F}) = \arg\min_{\mathbf{F}\in\mathbb{R}^{n\times c}} \{Q_{smooth}(\mathbf{F}) + Q_{fit}(\mathbf{F})\}. \tag{2.15}$$

A very natural formulation of the above cost function was proposed as *LGC* in [174], where an elastic regularizer framework was used to formulate the following cost function:

$$\mathcal{Q}(\mathbf{F}) = \|\mathbf{F}\|_{\mathcal{G}}^2 + \frac{\mu}{2}\|\mathbf{F}-\mathbf{Y}\|^2. \tag{2.16}$$

The first term $\|\mathbf{F}\|_{\mathcal{G}}^2$ represents function smoothness over graph $\mathcal{G}$, and the second term $\|\mathbf{F}-\mathbf{Y}\|^2$ plays as a regularization term through measuring the empirical loss on given labeled samples. Specifically, in *LGC*, the function smoothness is defined in the form of semi-inner product as:

$$\mathcal{Q}_{smooth} = \|\mathbf{F}\|_{\mathcal{G}}^2 = \frac{1}{2}\langle \mathbf{F}, \mathbf{L}\mathbf{F}\rangle = \frac{1}{2}\mathrm{tr}(\mathbf{F}^\top \mathbf{L}\mathbf{F}). \tag{2.17}$$

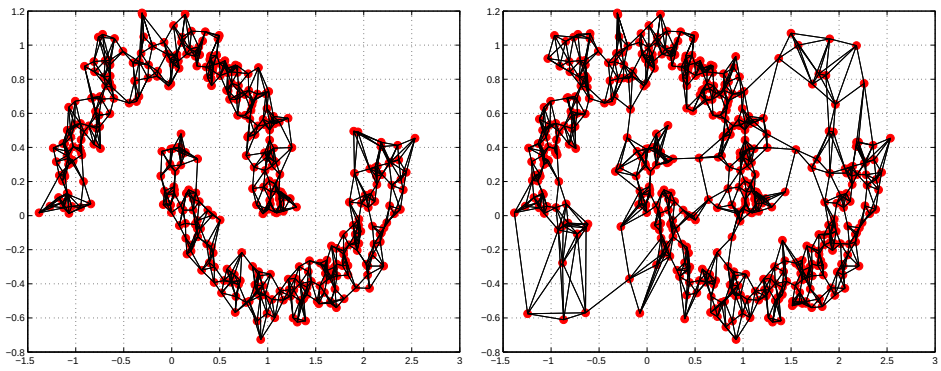
Note that the coefficient μ in Equation (2.16) balances global smoothness and local fitting penalty terms. If we set $\mu = \infty$ and use a standard graph Laplacian quantity $\mathbf{\Delta}$ (as defined in Eq. 2.1) for the smoothness term, the above framework reduces to the harmonic function formulation proposed in [182]. More precisely, the cost function only preserves the smoothness term as:

$$\mathcal{Q}(\mathbf{F}) = \mathrm{tr}(\mathbf{F}^\top \mathbf{\Delta}\mathbf{F}). \tag{2.18}$$

Meanwhile, the harmonic function $\mathbf{F}$, minimizing the above cost, also satisfies two conditions:

$$\begin{aligned} &\frac{\partial \mathcal{Q}}{\partial \mathbf{F}_u} = \mathbf{\Delta}\mathbf{F}_u = 0 \\ &\mathbf{F}_l = \mathbf{Y}_l, \end{aligned} \tag{2.19}$$

where $\mathbf{F}_l, \mathbf{F}_u$ are the function values of $f(\cdot)$ over labeled and unlabeled vertices, i.e., $\mathbf{F}_l = f(\mathbf{X}_l)$, $\mathbf{F}_u = f(\mathbf{X}_u)$, and $\mathbf{F} = [\mathbf{F}_l \;\; \mathbf{F}_u]^\top$. The first equation above denotes the zero derivative of the object function on the unlabeled data and the second equation requires clamping the function value on the given label value $\mathbf{Y}_l$. Both *LGC* and *GFHF* fit in a univariate regularization framework, where the continuous prediction function is treated as the only variable in the optimization procedure. Since the cost functions are convex, the optimal solutions for Eq. 2.16 and Eq. 2.18 are easily obtained through solving a linear system.



(a) (b)

Figure 2.3: Examples of constructed kNN ($k = 5$) graphs on the artificial two moon dataset: a) completely separable graph; b) non-separable graph due to noisy samples.

2.5 Open Issues

Existing graph-based *SSL* methods hinge on having good label information and an appropriately constructed graph [156]. But the heuristic design of the graph may result in suboptimal inference. In addition, the label propagation procedure can easily be misled if there exist excessive noise or outliers in the initial labeled set. Finally, in *iid* settings, the difference between empirically estimated class proportions and their true expected value is bounded [68]. However, practical annotation procedures are not necessarily *iid* and labeled data may have empirical class frequencies that deviate significantly from the expected class ratios. These degenerate situations seem to plague real world problems and compromise the performance of many state-of-the-art *SSL* algorithms. We next discuss some open issues which occur often in graph construction and label propagation, two critical components of all *GSSL* algorithms.

Sensitivity to Graph Construction:

As shown in Figure 2.3(a), a well-built graph with separable manifolds leads to simple cases for which most of the existing *GSSL* approaches work well. However, practical applications often produce non-separable graphs, as shown in Figure 2.3(b). In addition to the widely used kNN graph, we showed that b-matching could be used successfully for graph construction [74]. But both kNN graphs and b-matched graphs are heuristics and require the careful selection of the parameter k or b which controls the number of links incident to each vertex in the graph. Moreover, edge

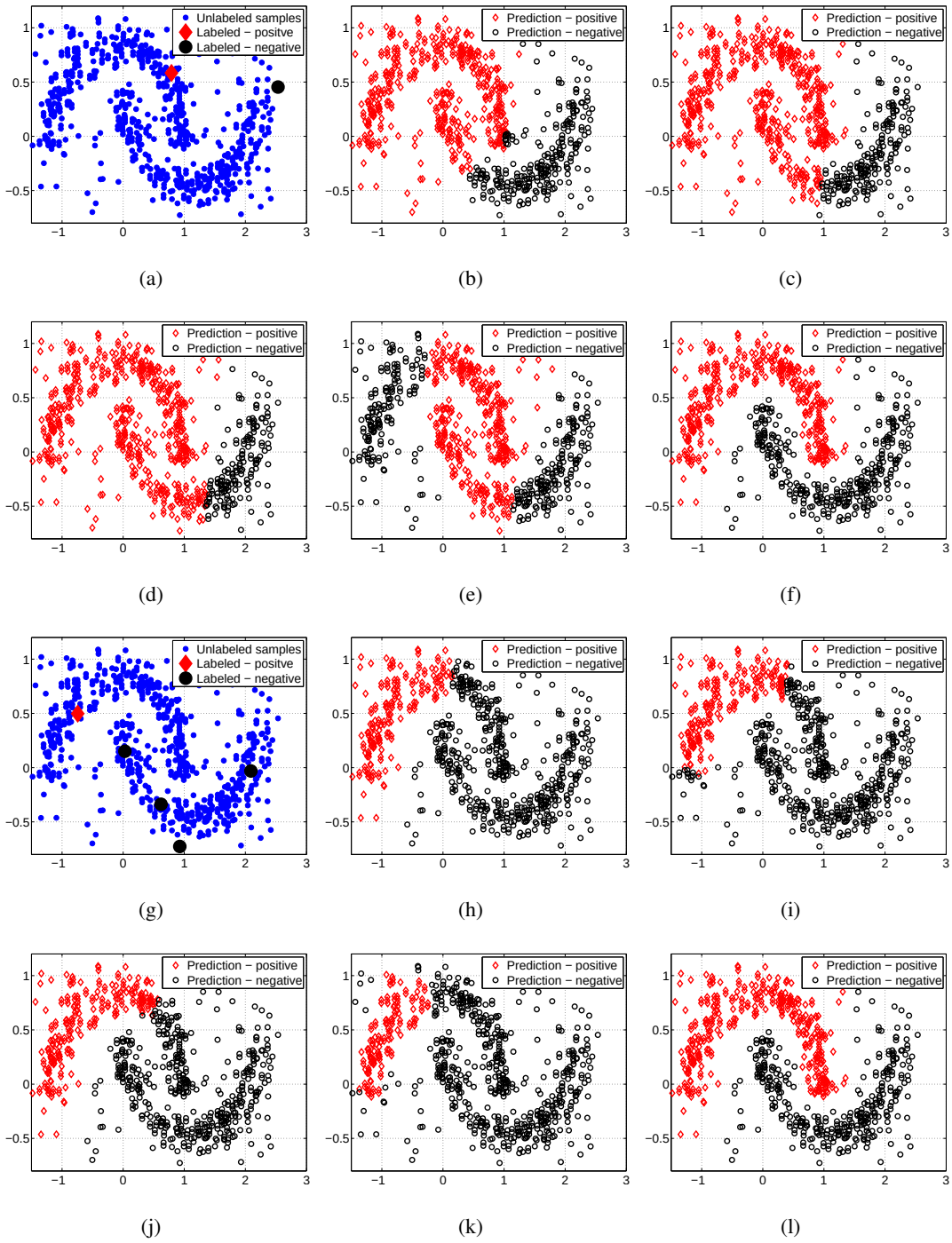


Figure 2.4: Examples illustrating the sensitivity of graph-based *SSL* to adverse labeling conditions. Particularly challenging conditions are shown in (a) where an uninformative label with on a outlier sample is the only negative label (denoted by a black circle) and in (g) where imbalanced labeling is involved. Prediction results are shown for the *GFHF* method [182] in (b) and (h), the *LGC* method [174] in (c) and (i), the *LapSVM* method [15] in (d) and (j), the *TSVM* method [79] in (e) and (k); our method in (f) and (l).

reweighing on the sparse graph often also requires exploration forcing the user to select try kernels and various kernel parameters. All these heuristic steps in graph design require extra effort from the user and demand some level of familiarity with the data domain.

Sensitivity to Label Noise:

Most of the existing methods *GSSL* methods are based on a univariate quadratic regularization framework which relies heavily on the quality of the initially assigned labels. For certain synthetic and real data problems, such graph transduction approaches achieve promising performance. However, several realistic labeling conditions produce unsatisfactory performance [156]. Even if the graph is perfectly constructed from the data, noisy initial labels can easily deteriorate the performance of *SSL* prediction. Figure 2.4 provides examples depicting imbalanced and noisy labels that lead to invalid graph transduction solutions for all the aforementioned algorithms. The first labeling problem involves uninformative labels (Figure 2.4(a)). The only negative label (dark circle) is located in an outlier region where the low density connectivity limits its diffusion to the rest of the graph. The leading *SSL* methods classify the majority of unlabeled nodes in the graph as positive (Figure 2.4(b)-Figure 2.4(e)). Such conditions are frequent in real problems like content-based image retrieval (CBIR) where the visual query example is not necessarily representative of the class. Another difficult case is due to imbalanced labeling. There, the ratio of training labels is disproportionate to the underlying class proportions. For example, Figure 2.4(g) depicts two half-circles with an almost equal number of samples. However, since the training labels contain three negative samples and only one positive example, the *SSL* predictions are strongly biased towards the negative class (see Figures 2.4(h) to 2.4(k)). This imbalanced labeling situation occurs frequently in realistic problems such as the annotation of microscopic images [155]. Therein, the human labeler favors certain cellular phenotypes due to domain-specific biological hypotheses. Other challenging cases, e.g., mislabeling due to uncontrolled labeling, can also generate undesirable label prediction results, as described in [157].

In summary, there exist several challenging issues, including heuristic setting of graph construction and sensitivity to label conditions, that create the need for a more robust label prediction mechanism. In the next Chapter, we will propose a novel graph-based *SSL* framework based on a bivariate optimization procedure and show its robustness to both graph construction and labeling conditions.

2.6 Related Work

Here we briefly go through several related topics about semi-supervised learning and graph-based learning problems.

Co-Training:

As one of the earliest cases of the empirical success of *SSL*, co-training was first developed for text mining [22], and was later extended in various forms to other applications [2][31][61][106][179]. Briefly speaking, multiple classifiers are first trained using conditionally independent feature sets of training data. The predictions with sufficient confidence of each classifier are then added to the label set and used to train new classifiers iteratively. Its performance highly relies on the existence of independent and complementary classifiers – some classifiers can correctly classify the samples that are mislabeled by other classifiers. Theoretical study shows that a weak assumption on the underlying data distribution is sufficient for co-training to work [8]. Another recent theoretical analysis treats co-training as a combinative label propagation over multiple views and provides a sufficient and necessary condition desired for co-training [163]. However, the performance could be dramatically degraded if the classifiers do not complement each other or the independency assumption does not hold [88]. Though co-training is conceptually treated as a semi-supervised learning paradigm due to the way unlabeled data is incorporated, the classifier training procedure is often supervised [22].

Semi-Supervised Support Vector Machines:

The extension of traditional supervised support vector machines (*SVMs*) to a semi-supervised scenario is intuitive and straightforward. Instead of maximizing separation (maximum-margin hyperplane) over training data in standard *SVMs*, semi-supervised *SVMs* (*S3VMs*) aims at estimating a hyperplane to balance maximum-margin partition over labeled data and separation through low-density regions of the data [151]. For example, in [79], transductive support vector machines (TSVMs) were developed as one of the earliest versions of semi-supervised *SVMs* [5]. Various optimization techniques have been applied to solve *S3VMs* [28], resulting in a wide range of methods, such as low density separation [30], semi-definite programming based methods [18][166], and a branch and bound based approach [29]. A thorough review of *S3VMs* methods can be found

[5] It is actually more appropriate to call this method semi-supervised *SVMs* since the learned classifier is indeed inductive [181].

in [181].

Graph-based Clustering:

As a natural representation of data, graphs have also been used for clustering since the early 1970s [40]. One of the most representative methods is *spectral clustering* [112][134]. Like *GSSL* methods, spectral clustering starts by representing the data with a similarity graph and then computes the eigenvectors of the graph Laplacian matrix. Using the theoretical foundation from the spectral graph theory [34], a number of spectral clustering algorithms have been designed for different applications with various graph designs and different normalization strategies of graph Laplacian [6] [32][73][167][171][175]. As an extension, *constrained clustering* uses side information or weak supervision, e.g., so-called *must-link* and *cannot-link* information, to achieve better clustering performance [11][97][164] [6]. For details about graph-based clustering, refer to [50][152] for additional survey.

[6]Some literatures categorize constrained clustering as a special case of semi-supervised learning.

Chapter 3

Bivariate Framework for Graph Transduction

3.1 Introduction

In order to handle the various challenging issues discussed in the previous chapter, we first extend the existing *GSSL* formulation by casting it as a *bivariate* optimization problem over the classification function *and* the labels. Then we demonstrate that minimizing the mixed bivariate cost function can be reduced to a pure integer programming problem that is equivalent to a constrained Max-Cut problem. Though semi-definite programming can be used to obtain approximate solutions, it is impractical due to the scalability issue. We propose an efficient greedy gradient-based solution, named *graph transduction via alternating minimization* (*GTAM*). *GTAM* alternates the steps to minimize the cost function over both the label matrix and the prediction function that leads to convergence to a local minimum. Moreover, to alleviate the dependency on initial labels, we propose using a label weight term to apply within-class normalization by vertex degree, as well as between-class normalization by class prior information. We demonstrate that the *GTAM* algorithm produces significantly better performance on both artificial and real datasets.

In addition, we further propose a greedy gradient-based Max-Cut solution that remedies the instability previous methods seem to have vis-a-vis the initial labeling. In our greedy solution, initial labels simply act as the starting value of the graph cut which is incrementally refined until convergence. During each iteration of the greedy search, the optimal unlabeled vertex is assigned

to the labeled subset with minimum connectivity to maximally preserve cross-subset edge weights. Finally, the overall Max-Cut is achieved after placing all the unlabeled vertices in the proper label sets. It is then straightforward to come up with the final label prediction based on the graph cut results. It is interesting that the greedy gradient-based Max-Cut solution can be used to interpret the alternative minimization procedure of *GTAM*.

Based on the above bivariate framework of graph transduction, we further extend two advanced versions. First, we design a graph-based label tuning strategy, called *Label Diagnosis through Self Tuning*, to handle errors in the initial labeled set. Second, we propose multi-graph based transductive learning through conducting an alternating optimization strategy across multiple graphs. We empirically demonstrate its superiority in both prediction accuracy and computational efficiency, compared with prior work. Both these two extensions make the bivariate framework easily adaptable to practical applications, such as web image search and video retrieval.

The remainder of this chapter is organized as follows: In Section 3.2, we present our bivariate graph transduction framework and propose the alternating minimization solution. Section 3.3 provides several theoretic proofs of the equivalence between the bivariate framework and maximum cut problem and describes an efficient solution using greedy gradient Max-Cut. Two advanced extensions, including label tuning technique and multiple-graph transduction, are discussed in Sections 3.4 and 3.5. Section 3.6 includes the experimental validation and comparison with other leading *SSL* methods over both toy and real datasets. Concluding remarks and discussions are provided in Section 3.7.

3.2 Bivariate Graph Transduction

3.2.1 The Cost Function

Recall the univariate regularization formulation for graph-based *SSL* in Eq. (2.15). As discussed earlier in Section 2.5, the label propagation process is very sensitive to the choice of the initial labels and the pre-computed graph. First, the optimization procedure of the existing approaches, such as *LGC* and *GFHF*, can be broken up into separate parallel problems since the cost function decomposes into terms that only depend on individual columns of the prediction matrix $\mathbf{F}$. Such a decomposition reveals that biases may arise if the input labels are disproportionately imbalanced.

Second, when the graph contains background noise and makes class manifolds nonseparable (Figure 2.3(b)), these existing graph transduction approaches fail to output reasonable classification results.

Realizing that the univariate framework treating the initial label information as a constant serves as the main cause of the label sensitivity issue, we propose a novel bivariate optimization framework that explicitly optimizes over both the classification function $\mathbf{F}$ and the binary label matrix $\mathbf{Y}$:

$$(\mathbf{F}^*, \mathbf{Y}^*) = \arg\min_{\mathbf{F}\in\mathbb{R}^{n\times c}, \mathbf{Y}\in\mathbb{B}^{n\times c}} \mathcal{Q}(\mathbf{F}, \mathbf{Y}), \tag{3.1}$$

where $\mathbb{B}^{n\times c}$ is the set of all binary matrices $\mathbf{Y}$ of size $n \times c$ that satisfy $\sum_j \mathbf{Y}_{ij} = 1$, and for the labeled data $\mathbf{x}_i \in \mathbf{X}_l$, $\mathbf{Y}_{ij} = 1$ if $z_i = j$. Second, we specify the loss function as

$$\mathcal{Q}(\mathbf{F}, \mathbf{Y}) = \|\mathbf{F}_{\mathcal{G}}\|^2 + \frac{\mu}{2}\|\mathbf{F} - \mathbf{Y}\|^2 = \frac{1}{2}\mathrm{tr}\left(\mathbf{F}^\top \mathbf{L}\mathbf{F} + \mu(\mathbf{F} - \mathbf{Y})^\top(\mathbf{F} - \mathbf{Y})\right). \tag{3.2}$$

Finally, rewriting the cost as a summation [174] reveals a more intuitive formulation where

$$\mathcal{Q}(\mathbf{F}, \mathbf{Y}) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}\left\|\frac{\mathbf{F}_{i\cdot}}{\sqrt{d_i}} - \frac{\mathbf{F}_{j\cdot}}{\sqrt{d_j}}\right\|^2 + \frac{\mu}{2}\sum_{i=1}^{n}\|\mathbf{F}_{i\cdot} - \mathbf{Y}_{i\cdot}\|^2. \tag{3.3}$$

3.2.2 Reduction to a Univariate Problem

In the new graph regularization framework proposed above, the cost function involves two variables to be optimized. Simultaneously recovering both solutions is intractable due to the mixed integer programming problem over binary $\mathbf{Y}$ with constraints and continuous $\mathbf{F}$. To solve the issue, we first show how to reduce the original mixed problem to a univariate optimization problem with respect to the label variable $\mathbf{Y}$.

F optimization step:

In each loop with $\mathbf{Y}$ fixed, the classification function $\mathbf{F} \in \mathbb{R}^{n\times c}$ is continuous and the cost function is convex, allowing the minimum to be recovered by zeroing the partial derivative:

$$\begin{aligned}&\frac{\partial \mathcal{Q}}{\partial \mathbf{F}^*} = 0 \Longrightarrow \mathbf{L}\mathbf{F}^* + \mu(\mathbf{F}^* - \mathbf{Y}) = 0\\ &\Longrightarrow \mathbf{F}^* = (\mathbf{L}/\mu + \mathbf{I})^{-1}\mathbf{Y} = \mathbf{P}\mathbf{Y},\end{aligned} \tag{3.4}$$

where we denote the $\mathbf{P}$ matrix as

$$\mathbf{P} = (\mathbf{L}/\mu + \mathbf{I})^{-1}, \tag{3.5}$$

and name it the *propagation matrix* since it is used to derived a prediction function $\mathbf{F}$ given a label matrix $\mathbf{Y}$. Because the graph is often symmetrically built, it is easy to show that the graph Laplacian $\mathbf{L}$ and the propagation matrix $\mathbf{P}$ are both symmetric.

$\mathbf{Y}$ optimization step:

Next replace $\mathbf{F}$ in Eq. (3.2) by its optimal value $\mathbf{F}^*$ from the solution of Eq. (3.4). This yields

$$\begin{aligned}\mathcal{Q}(\mathbf{Y}) &= \frac{1}{2}\mathrm{tr}(\mathbf{Y}^\top\mathbf{P}^\top\mathbf{L}\mathbf{P}\mathbf{Y} + \mu(\mathbf{P}\mathbf{Y}-\mathbf{Y})^\top(\mathbf{P}\mathbf{Y}-\mathbf{Y})) \\ &= \frac{1}{2}\mathrm{tr}\left(\mathbf{Y}^\top\left[\mathbf{P}^\top\mathbf{L}\mathbf{P} + \mu(\mathbf{P}^\top-\mathbf{I})(\mathbf{P}-\mathbf{I})\right]\mathbf{Y}\right) = \frac{1}{2}\mathrm{tr}\left(\mathbf{Y}^\top\mathbf{A}\mathbf{Y}\right), \end{aligned} \tag{3.6}$$

where we group all the constant parts in the above equation and define

$$\mathbf{A} = \mathbf{P}^\top\mathbf{L}\mathbf{P} + \mu(\mathbf{P}^\top-\mathbf{I})(\mathbf{P}-\mathbf{I}) = \mathbf{P}^\top\mathbf{L}\mathbf{P} + \mu(\mathbf{P}-\mathbf{I})^2. \tag{3.7}$$

The final optimization problem becomes

$$\begin{aligned}&\mathbf{Y}^* = \arg\min\frac{1}{2}\mathrm{tr}\left(\mathbf{Y}^\top\mathbf{A}\mathbf{Y}\right) \\ &s.t. \quad y_{ij}\in\{0,1\}, \\ &\qquad \textstyle\sum_j y_{ij} = 1, \;\; j = 1,\cdots,c \\ &\qquad y_{ij} = 1, \text{for} \;\; z_i = j, \;\; j = 1,\cdots,c. \end{aligned} \tag{3.8}$$

The first constraint produces a binary integer problem and the second one $\sum_j y_{ij} = 1$ produces a single assignment constraint, i.e., each vertex can only be assigned one class label. The third group of constraints encode the initial label information in the variable $\mathbf{Y}$. Since the binary matrix $\mathbf{Y}\in\mathbb{B}^{n\times c}$ is subject to linear constraints of the form $\sum_j y_{ij} = 1$ and initial labeling conditions, the optimization in Eq. (3.8) requires solving a linearly constrained binary integer programming (BIP) problem which is NP hard [36][82].

3.2.3 Incorporating Label Normalization

To solve the minimization problem in Eq. (3.8), a straightforward way is to use gradient approach to greedily update label variable $\mathbf{Y}$. However, this will raise anther practical issue of biased classification since the class with more labels will be preferred to be assigned new labeled samples in each iteration. This has been seen in practice (see the examples in Figure 2.4) but can also be explained

mathematically by the fact that $\mathbf{Y}$ starts off extremely sparse and has many unknown terms. To handle this bias issue during label propagation, we propose using a normalized label variable $\tilde{\mathbf{Y}} = \mathbf{\Lambda Y}$ for computing the cost function in Eq. (3.2) as:

$$\begin{aligned} \mathcal{Q} &= \frac{1}{2}\mathrm{tr}\left(\mathbf{F}^\top \mathbf{L}\mathbf{F} + \mu(\mathbf{F}-\tilde{\mathbf{Y}})^\top(\mathbf{F}-\tilde{\mathbf{Y}})\right) \\ &= \frac{1}{2}\mathrm{tr}\left(\mathbf{F}^\top \mathbf{L}\mathbf{F} + \mu(\mathbf{F}-\mathbf{\Lambda Y})^\top(\mathbf{F}-\mathbf{\Lambda Y})\right). \end{aligned} \tag{3.9}$$

The diagonal matrix $\mathbf{\Lambda} = diag(\boldsymbol{\lambda}) = diag([\lambda_1, \cdots, \lambda_n])$ is introduced as a label weight to balance the influence of labels from different classes, and modulates the label importance based on the node degree. The value of λ_i $(i = 1, \cdots, n)$ is computed using the vertex degree d_i and label information

$$\lambda_i = \begin{cases} p_j \cdot \frac{d_i}{\sum_k y_{kj} d_k} & : \quad y_{ij} = 1 \\ 0 & : \quad \text{otherwise}, \end{cases} \tag{3.10}$$

where p_j is the prior of class j and is subject to the constraint $\sum_{j=1}^{c} p_j = 1$. The value of p_j can be either estimated from the labeled training set or simply set to be uniform $p_j = 1/c$ $(j = 1, \cdots, c)$ in agnostic situations (when no better prior is available or if the labeled data is plagued by biased sampling). Using the normalized label matrix $\tilde{\mathbf{Y}}$ in the bivariate formulation allows labeled nodes with high degrees to contribute more during the label propagation process. However, the total diffusion of each class is kept equal (for agnostic settings with no priors available) or proportional to the class prior (for the setting with prior information). Therefore, the influence of different classes is balanced even if the given class labels are imbalanced. If class proportion information is known, it can be integrated by scaling the diffusion with the appropriate prior. In other words, the label normalization attempts to enforce simple concentration inequalities which, in the *iid* case require the predicted label results to concentrate around the underlying class ratios [68]. This intuition is in line with prior work that uses class proportion information in transductive inference where class proportion is enforced as a hard constraint [29] or as a regularizer [103].

3.2.4 Graph Transduction via Alternating Minimization

We now present an alternating optimization procedure, named *graph transduction via alternating minimization* (*GTAM*), to efficiently minimize the above cost function. Starting with Eq. (3.9) and

repeating the similar derivation as in Section 3.2.2, we obtain the optimal solution $\mathbf{F}^*$ and the final cost function with respect to label variable $\mathbf{Y}$ as

$$\mathbf{F}^* = \mathbf{P}\tilde{\mathbf{Y}} = \mathbf{P}\mathbf{\Lambda}\mathbf{Y} \tag{3.11}$$

$$\mathcal{Q} = \frac{1}{2}\mathrm{tr}\left(\tilde{\mathbf{Y}}^\top \mathbf{A}\tilde{\mathbf{Y}}\right) = \frac{1}{2}\mathrm{tr}\left(\mathbf{Y}^\top \mathbf{\Lambda}\mathbf{A}\mathbf{\Lambda}\mathbf{Y}\right). \tag{3.12}$$

Instead of finding the global optimal $\mathbf{Y}^*$, we only take an incremental stop in each iteration to update one label in $\mathbf{Y}$. Namely, in each iteration, we find the optimal position (i^*, j^*) in the matrix $\mathbf{Y}$ and change the binary value of y_{i^*j*} from 0 to 1. To do this, we find the direction with the largest negative gradient guiding our choice of binary step on $\mathbf{Y}$. Specifically, we evaluate $\| \bigtriangledown \mathcal{Q}_{\mathbf{Y}} \|$ and find the largest negative value to determine (i^*, j^*).

Note that setting $y_{i^*j^*} = 1$ is equivalent to a similar operation on the normalized label matrix $\tilde{y}$ by setting $\tilde{y}_{i^*,j^*} = \epsilon_{i^*}, 0 < \epsilon_{i^*} < 1$, and $\mathbf{Y}, \tilde{\mathbf{Y}}$ have one-to-one correspondence. Thus, the greedy optimization of $\mathcal{Q}$ with respect to $\mathbf{Y}$ is equivalent to the greedy minimization of $\mathcal{Q}$ with respect to $\tilde{\mathbf{Y}}$. More formally, we derive the gradient of the above loss function $\nabla_{\tilde{\mathbf{Y}}}\mathcal{Q} = \frac{\partial \mathcal{Q}}{\partial \tilde{\mathbf{Y}}}$ and recover it with respect to $\mathbf{Y}$ as:

$$\frac{\partial \mathcal{Q}}{\partial \tilde{\mathbf{Y}}} = \mathbf{A}\tilde{\mathbf{Y}} = \mathbf{A}\mathbf{\Lambda}\mathbf{Y}. \tag{3.13}$$

As described earlier, we search the gradient matrix $\nabla_{\tilde{\mathbf{Y}}}\mathcal{Q}$ to find the minimal element

$$(i^*, j^*) = \arg\min_{\mathbf{x}_i \in \mathcal{X}_u, 1 \le j \le c} \nabla_{\tilde{y}_{ij}} \mathcal{Q}. \tag{3.14}$$

Because of the binary nature of $\mathbf{Y}$, we simply set $y_{i^*j^*} = 1$ instead of using a continuous update approach. Accordingly, the node weight matrix $\mathbf{\Lambda}^{t+1}$ can be recalculated with the updated $\mathbf{Y}^{t+1}$ in the $t+1$th iteration. The update of $\mathbf{Y}$ is greedy, and thus it could backtrack from predicted labels in previous iterations without convergence guarantees. We propose a straightforward way to guarantee convergence and avoid backtracking or unstable oscillation in the greedy propagation process. Once an unlabeled point has been labeled, its labeling can no longer be changed. Thus, we remove the most recently labeled point (i^*, j^*) from future consideration and conduct the search over the remaining unlabeled data only. In other words, to avoid changing nearly added labels, the labeled vertex $\mathbf{x}_{i^*}$ will be removed from $\mathbf{X}_u$ and added to $\mathbf{X}_l$.

Note that although the optimal $\mathbf{F}^*$ can be computed as in Eq. (3.11), we do not need to compute it explicitly. Instead, it is implicitly used in Eq. (3.14) to update $\mathbf{Y}$. In the following, we summarize the updating rules from step t to $t+1$ in the alternative minimization scheme.

. Compute gradient matrix:

$$(\nabla_{\tilde{\mathbf{Y}}}\mathcal{Q})^t = \mathbf{A}\tilde{\mathbf{Y}}^t = \mathbf{A}\mathbf{\Lambda}^t\mathbf{Y}^t, \mathbf{\Lambda}^t = diag(\boldsymbol{\lambda}^t). \tag{3.15}$$

. Update one label:

$$(i^*, j^*) = \arg\min_{\mathbf{x}_i \in \mathbf{X}_u, 1 \le j \le c} (\nabla_{\tilde{y}_{ij}}\mathcal{Q})^t \tag{3.16}$$

$$y_{i^*j^*}^{t+1} = 1. \tag{3.17}$$

. Update label normalization matrix:

$$\lambda^{t+1} = \begin{cases} \frac{\mathbf{D}_{ii}}{\sum_k \mathbf{Y}_{kj}^{t+1}\mathbf{D}_{kk}} & : \quad y_{ij}^{t+1} = 1 \\ 0 & : \quad \text{otherwise.} \end{cases} \tag{3.18}$$

. Update the list of labeled and unlabeled data:

$$\mathbf{X}_l^{t+1} \longleftarrow \mathbf{X}_l^t + \mathbf{x}_{i^*} \;\; ; \;\; \mathbf{X}_u^{t+1} \longleftarrow \mathbf{X}_u^t - \mathbf{x}_{i^*}. \tag{3.19}$$

From the above discussion, our method is unique in that it optimizes the loss function over both the continuous-valued $\mathbf{F}$ space and the binary-valued $\mathbf{Y}$ space. Starting with a few given labels, the method iteratively and greedily updates the label matrix $\mathbf{Y}$, label weight matrix $\mathbf{\Lambda}$, and gradient matrix $\nabla_{\tilde{\mathbf{Y}}}\mathcal{Q}$. In each iteration, new labeled samples are obtained to drive a better graph propagation in the next iteration. This procedure repeats until all points have been labeled. In our approach, we directly acquire new labels in each iteration without explicitly calculating $\mathbf{F}^*$, which is the regular procedure used in other graph transduction methods like *LGC* and *GFHF*. This unique feature makes the proposed algorithm very efficient since we only update the gradient matrix $\nabla_{\tilde{\mathbf{Y}}}\mathcal{Q}$ in each iteration.

Due to greedy assignment, the algorithm can repeat the alternative minimization (or the gradient computation equivalently) at most $n-l$ times. Each minimization step over $\mathbf{F}$ and $\mathbf{Y}$ thus requires $\mathcal{O}(n^2)$ complexity and thus the total complexity of the greedy *GTAM* algorithm is $\mathcal{O}(n^3)$. However,

Algorithm 3.1 Graph Transduction via Alternating Minimization (*GTAM*)

Input: data set $\mathbf{X} = \{\mathbf{X}_l, \mathbf{X}_u\}$ containing labeled subset $\mathbf{X}_l = \{\mathbf{x}_1, \cdots, \mathbf{x}_l\}$, unlabeled subset $\mathbf{X}_u = \{\mathbf{x}_{l+1}, \cdots, \mathbf{x}_n\}$, labels $\{z_1, \cdots, z_j, \cdots, z_l\}$, where $z_j \in \{1, \cdots, c\}$.

Weight matrix $\mathbf{W} = \{w_{ij}\}$, node degree matrix $\mathbf{D}$, initial label matrix $\mathbf{Y}^0$, graph Laplacian $\mathbf{L}$, propagation matrix $\mathbf{P}$, matrix $\mathbf{A}$;

Initialization:

iteration counter $t = 0$, initial label weight matrix $\mathbf{\Lambda}^t$;

repeat

Compute derivative: $(\nabla_{\tilde{\mathbf{Y}}} \mathcal{Q})^t = \mathbf{A}\tilde{\mathbf{Y}}^t$;

Find the optimal element in $\nabla_{\tilde{\mathbf{Y}}^t} \mathcal{Q}$: $(i^*, j^*) = \arg\min_{\mathbf{x}_i \in \mathbf{X}_u, 1 \le j \le c} \nabla_{\tilde{y}_{ij}} \mathcal{Q}$;

Update label matrix by setting: $y_{i^* j^*}^{t+1} = 1$; also $z_{i^*} = j^*$;

Add $\mathbf{x}_{i^*}$ to $\mathbf{X}_l$: $\mathbf{X}_l^{t+1} \longleftarrow \mathbf{X}_l^t + \mathbf{x}_{i^*}$;

Remove $\mathbf{x}_{i^*}$ from $\mathbf{X}_u$: $\mathbf{X}_u^{t+1} \longleftarrow \mathbf{X}_u^t - \mathbf{x}_{i^*}$;

Update label weight matrix $\mathbf{\Lambda}^{t+1}$ based on Eq. (3.10);

Update iteration counter: $t = t + 1$;

until $\mathbf{X}_u^t = \emptyset$

Output: The labels of unlabeled samples $\{z_{l+1}, \cdots, z_n\}$.

the update of the graph gradient can be done efficiently by modifying only a single entry in $\mathbf{Y}$ per iteration, which greatly reduces the computational cost to $\mathcal{O}(n^2)$. Empirically, the value of the loss function $\mathcal{Q}$ decreases rapidly in the first dozen iterations and steadily converges afterward. This phenomenon indicates that the alternating procedure could be stopped early. Once the first few iterations are completed, the new labels are added and the standard propagation step can be used to predict the optimal $\mathbf{F}^*$ as indicated in Eq. (3.11) over the whole graph in one step. The above algorithm chart (3.1) summarizes the proposed *GTAM* method.

3.3 Graph Cut View for the Bivariate Framework

In this section, we introduce a connection between the proposed bivariate graph transduction framework and the well-known maximum cut problem. Then, a greedy gradient based Max-Cut solution will be developed and related to the above *GTAM* algorithm.

3.3.1 Equivalence to a Constrained Max-Cut Problem

Recall the optimization problem defined in Eq. (3.8), which is exactly a linearly constrained binary integer programming (BIP) problem. Particularly in the case of a two-class problem, we can prove that it is equivalent to a weighted Max-Cut problem over a graph $\mathcal{G}_{\mathbf{A}} = \{\mathbf{X}, \mathbf{A}\}$ with linear constraints converted from the given labels, as described below. The cost function in Eq. (3.8) can be rewritten as below:

$$\mathcal{Q}(\mathbf{Y}) = \frac{1}{2}\mathrm{tr}\left(\mathbf{Y}^\top \mathbf{A}\mathbf{Y}\right) = \frac{1}{2}\mathrm{tr}\left(\mathbf{A}\mathbf{Y}\mathbf{Y}^\top\right) = \frac{1}{2}\mathrm{tr}\left(\mathbf{A}\mathbf{R}\right), \tag{3.20}$$

where $\mathbf{R} = \mathbf{Y}\mathbf{Y}^\top$. Considering the constraints $\sum_j y_{ij} = 1$ and $\mathbf{Y} \in \mathbb{B}^{n\times 2}$ for a two-class problem, we can let

$$\mathbf{Y} = [\mathbf{y} \quad \mathbf{e}-\mathbf{y}], \tag{3.21}$$

where $\mathbf{y} \in \mathbb{B}^n$ (i.e., $\mathbf{y} = \{y_i\}, y_i \in \{0,1\}, i = 1, \cdots, n$) and $\mathbf{e} = [1, 1, \cdots, 1]^\top$ are column vectors. Then rewrite $\mathbf{R}$ as

$$\begin{aligned}
\mathbf{R} &= \mathbf{Y}\mathbf{Y}^\top = [\mathbf{y} \quad \mathbf{e}-\mathbf{y}][\mathbf{y} \quad \mathbf{e}-\mathbf{y}]^\top \\
&= \mathbf{y}\mathbf{y}^\top + (\mathbf{e}-\mathbf{y})(\mathbf{e}^\top - \mathbf{y}^\top) \\
&= \mathbf{e}\mathbf{e}^\top + 2\mathbf{y}\mathbf{y}^\top - \mathbf{y}\mathbf{e}^\top - \mathbf{e}\mathbf{y}^\top \\
&= \mathbf{e}\mathbf{e}^\top - \mathbf{y}(\mathbf{e}^\top - \mathbf{y}^\top) - (\mathbf{e}-\mathbf{y})\mathbf{y}^\top.
\end{aligned} \tag{3.22}$$

Now rewrite the cost function in Eq. (3.20) by replacing $\mathbf{R}$ with Eq. (3.22):

$$\mathcal{Q}(\mathbf{y}) = \frac{1}{2}\mathrm{tr}\left(\mathbf{A}\left[\mathbf{e}\mathbf{e}^\top - \mathbf{y}(\mathbf{e}^\top - \mathbf{y}^\top) - (\mathbf{e}-\mathbf{y})\mathbf{y}^\top\right]\right). \tag{3.23}$$

Since $\mathbf{e}\mathbf{e}^\top$ is the all-ones matrix, we obtain

$$\frac{1}{2}\mathrm{tr}\left(\mathbf{A}\mathbf{e}\mathbf{e}^\top\right) = \frac{1}{2}\sum_i\sum_j \mathbf{A}_{ij}. \tag{3.24}$$

It is easy to show that $\mathbf{A}$ is symmetric and

$$\mathbf{y}(\mathbf{e}^\top - \mathbf{y}^\top) = [(\mathbf{e}-\mathbf{y})\mathbf{y}^\top]^\top. \tag{3.25}$$

Next, simplify the cost function $\mathcal{Q}$ as

$$\begin{aligned}
\mathcal{Q}(\mathbf{y}) &= \frac{1}{2}\mathrm{tr}\left(\mathbf{A}\mathbf{e}\mathbf{e}^\top\right) - \mathrm{tr}\left[(\mathbf{e}^\top - \mathbf{y}^\top)\mathbf{A}\mathbf{y}\right] \\
&= \frac{1}{2}\mathrm{tr}\left(\mathbf{A}\mathbf{e}\mathbf{e}^\top\right) - \mathbf{y}^\top\mathbf{A}(\mathbf{e}-\mathbf{y}).
\end{aligned} \tag{3.26}$$

Since the first part is a constant, the optimal value $\mathbf{y}^*$ of the above minimization problem is the argument of the maximization problem

$$\mathbf{y}^* = \arg\min_{\mathbf{y}} \mathcal{Q}(\mathbf{y}) = \arg\max_{\mathbf{y}} \mathbf{y}^\top \mathbf{A}(\mathbf{e} - \mathbf{y}). \tag{3.27}$$

Define a new function $f(\mathbf{y})$ as

$$f(\mathbf{y}) = \mathbf{y}^\top \mathbf{A}(\mathbf{e} - \mathbf{y}). \tag{3.28}$$

Again, the variable $\mathbf{y} \in \mathbb{B}^n$ is a binary vector and $\mathbf{e} = [1, 1, \cdots, 1]^\top$ is the unit column vector. Now we show that maximization of the above function $\max_{\mathbf{y}} f(\mathbf{y})$ is exactly a Max-Cut problem if we treat the symmetric matrix $\mathbf{A}$ as the weighted adjacency matrix of an undirected graph $\mathcal{G}_{\mathbf{A}} = \{V_{\mathbf{A}}, \mathbf{A}\}$. Note that the diagonal elements of $\mathbf{A}$ could be non-zero $\mathbf{A}_{ii} \neq 0, i = 1, 2, \cdots, n$, which indicates the undirected graph $\mathcal{G}_{\mathbf{A}}$ has self-connected nodes. Assume $\mathbf{A} = \mathbf{A}^0 + \mathbf{A}^\Lambda$, where $\mathbf{A}^0$ is the matrix by zeroing the diagonal elements of $\mathbf{A}$, and $\mathbf{A}^\Lambda$ is a diagonal matrix with $\mathbf{A}^\Lambda_{ii} = \mathbf{A}_{ii}, \mathbf{A}^\Lambda_{ij} = 0, i, j = 1, 2, \cdots, n, i \neq j$. It is straightforward to show that the function $f(\mathbf{y})$ can be written as

$$f(\mathbf{y}) = \mathbf{y}^\top (\mathbf{A}^0 + \mathbf{A}^\Lambda)(\mathbf{e} - \mathbf{y}) = \mathbf{y}^\top \mathbf{A}^0 (\mathbf{e} - \mathbf{y}) \tag{3.29}$$

In other words, the non-zero elements in $\mathbf{A}$ do not affect the value of $f(\mathbf{y})$. Therefore, in the remainder part of this thesis, we will assume that the matrix $\mathbf{A}$ has zero diagonal elements unless the text specifies otherwise.

Since $\mathbf{y} = \{y_1, y_2, \cdots, y_n\}$ is a binary vector, each setting of $\mathbf{y}$ partitions the vertex set $V_{\mathbf{A}}$ in the graph $\mathcal{G}_{\mathbf{A}}$ into two disjoint subsets (S_1, S_2). In other words, the two subsets $S_1 = \{v_i | y_i = 1\}$ and $S_2 = \{v_i | y_i = 0\}$ satisfy $S_1 \cup S_2 = V_{\mathbf{A}}$ and $S_1 \cap S_2 = \emptyset$. The maximum problem can be written as

$$\begin{aligned} \max f(\mathbf{y}) &= \max \sum_{i,j} \mathbf{A}_{ij} \cdot y_i (1 - y_j) \\ &= \max \frac{1}{2} \sum_{\substack{v_i \in S_1 \\ v_j \in S_2}} \mathbf{A}_{ij}. \end{aligned} \tag{3.30}$$

Because each binary vector $\mathbf{y}$ resulting in a partition (S_1, S_2) over the graph $\mathcal{G}_{\mathbf{A}}$ and $f(\mathbf{y})$ is a corresponding *cut*, the above maximization $\max f(\mathbf{y})$ is easily recognized as a Max-Cut problem [39].

However, in graph based semi-supervised learning, the variable $\mathbf{y}$ is partially specified by the initial label values. This given label information can be interpreted as a set of linear constraints on the Max-Cut problem. Thus, the optimal solution can achieved by solving a linearly constrained Max-Cut problem [82]. In addition, we also show that a multi-class problem equals a Max K-Cut problem ($K = c$) (refer to Appendix B).

However, since there is no guarantee that the weights of the graph $\mathcal{G}_{\mathbf{A}}$ are non-negative, such solutions could be problematic in practice, as discussed in [10]. Therefore, in the following subsection, we will propose a gradient greedy solution to efficiently solve the above Max-Cut problem, which can be treated as a different view of the previous alternating minimization solution.

3.3.2 Greedy Gradient Based Maximum Cut

For the standard Max-Cut problem, there have been many approximation techniques developed, including the most remarkable Goemans-Williamson algorithm using semidefinite programming [58] [59]. However, applying these guaranteed approximation schemes to solve the constrained Max-Cut problem for $\mathbf{Y}$ mentioned above is infeasible due to the constraints on initial labels. Furthermore, there is no guarantee that all edge weights a_{ij} of the graph $\mathcal{G}_{\mathbf{A}}$ are non-negative, a fundamental requirement in solving a standard Max-Cut problem [59]. Instead, here we use a greedy gradient based strategy to find the local optima by assigning each unlabeled vertex to the label set with minimum connectivity to maximize cross-set edge weights iteratively.

The greedy Max-Cut algorithm randomly selects the unlabeled vertices and places each of them into a proper subset depending on the edges between this unlabeled vertex and the vertices in the labeled subset. Given the label information, the initial label set for class j can be constructed as $\mathcal{S}_j = \{\mathbf{x}_i | y_{ij} = 1\}$ or $\mathcal{S}_j = \{\mathbf{x}_i | z_i = j\}$, $i = 1, 2, \cdots, n; j = 1, 2, \cdots, c$. Define the following connectivity measurement between unlabeled vertex $\mathbf{x}_i$ and label set $\mathcal{S}_j$ as:

$$c_{ij} = \sum_{m=1}^{n} a_{im} y_{mj} = \mathbf{A}_{i\cdot} \mathbf{Y}_{\cdot j}. \tag{3.31}$$

where $\mathbf{A}_{i\cdot}$ is the ith row vector of $\mathbf{A}$ and $\mathbf{Y}_{\cdot j}$ is the jth column vector of $\mathbf{Y}$. Intuitively, c_{ij} represents the sum of edge weights between vertex $\mathbf{x}_i$ to label set $\mathcal{S}_j$ given the graph $\mathcal{G}_{\mathbf{A}}$ with edge weights $\mathbf{A}$. Based on this definition, a straightforward local search for maximum cut is to place each unlabeled vertex $\mathbf{x}_i \in \mathbf{X}_u$ to the labeled set $\mathcal{S}_j$ with minimum connectivity c_{ij} to maximize the cross-set edge

Algorithm 3.2 Greedy Max-Cut for Label Propagation

Input: the graph $\mathcal{G}_{\mathbf{A}} = \{\mathbf{X}, \mathbf{A}\}$ and the given labeled vertex $\mathbf{X}_l$, and initial label $\mathbf{Y}$;

Initialization:

Obtain the initial cut $\{\mathcal{S}_j\}$ through assigning the labeled vertex $\mathbf{X}_l$ to each subset:

$$\mathcal{S}_j = \{x_i | y_{ij} = 1\}, j = 1, 2, \cdots, c$$

Unlabeled vertex set $\mathbf{X}_u = \mathbf{X} \setminus \mathbf{X}_l$;

repeat

Randomly select an unlabeled vertex $\mathbf{x}_i \in \mathbf{X}_u$

Compute connectivity $c_{ij}, j = 1, 2, \cdots, c$

Place the vertex to the labeled subject $\mathcal{S}_{j^*}$:

$$j^* = \arg\min_j c_{ij}$$

Add $\mathbf{x}_i$ to $\mathbf{X}_l$: $\mathbf{X}_l \longleftarrow \mathbf{X}_l + \mathbf{x}_i$;

Remove $\mathbf{x}_i$ from $\mathbf{X}_u$: $\mathbf{X}_u \longleftarrow \mathbf{X}_u - \mathbf{x}_i$;

until $\mathbf{X}_u = \emptyset$

Output: The final cut and the corresponding labeled subsets $\mathcal{S}_j, j = 1, 2, \cdots, c$

weights, as shown in Algorithm (3.2). In order to achieve a good solution, it is desirable to run Algorithm (3.2) multiple times with different random seeds and return the best cut [104].

Besides the significant computational cost, the above random solution may easily fall into undesired local optima and generate biased cuts. Following the definition in Eq. (3.31), the initial conditions of the given labels, such as the size of individual label sets and the density of each labeled vertex, determine the connectivity between unlabeled vertices and labeled subsets. In the case that the computed connectivity is negative, which occurs often in reality, the above random search will prefer assigning unlabeled vertices to the label set with the most labeled vertices, resulting in biased partitioning. Such biased partitioning also occurs in the case of minimum cut over an undirected graph with positive weights, as discussed in [134]. Other ill-label conditions may also make the random cut approach ineffective. For example, the numbers of labels among different classes may be disproportionate to the underlying class sizes, or the labeled vertices may reside in the regions of the graph with significantly different densities. Such labels often cause erroneous label prediction results, as reported in our empirical study [155]. Furthermore, the random selection of

an unlabeled vertex results in unstable predictions since unlabeled vertex $\mathbf{x}_i$ could have equally low connectivity to multiple label sets $\mathcal{S}_j$ under the condition of extremely sparse labels.

To address the aforementioned issues, we first modify the original definition of connectivity to alleviate the label imbalance among different classes. Motivated by the label weight term used in *GTAM*, a weighted connectivity is computed as

$$c_{ij} = p_j \cdot \sum_{m=1}^{n} \lambda_m a_{im} y_{mj} = p_j \cdot \mathbf{\Lambda} \mathbf{A}_{i.} \mathbf{Y}_{.j}. \tag{3.32}$$

The diagonal matrix $\mathbf{\Lambda} = diag([\lambda_1,\ \lambda_2,\ \cdots,\ \lambda_n]$ is called label weight matrix, same as the one used in *GTAM*:

$$\lambda_i = \begin{cases} d_i / d_{\mathcal{S}_j} & : \quad \text{if } \ \mathbf{x}_i \in \mathcal{S}_j, j = 1, \cdots, c \\ 0 & : \quad \text{otherwise,} \end{cases} \tag{3.33}$$

where $d_{\mathcal{S}_j} = \sum_{\mathbf{x}_m \in \mathcal{S}_j} d_m$ is the sum of the degrees of the vertices in the label set $\mathcal{S}_j$. This heuristic setting weighs the importance of each label based on the vertex degree, which alleviates the adverse impact from outliers or problematic labels. Without considering class priors, this definition of $\mathbf{\Lambda}$ is exactly the same as that in Eq. (3.10).

Finally, to handle unstable output from the random search algorithm, we propose a greedy gradient search approach, where the most beneficial vertex is assigned to the proper label set with minimum connectivity. In other words, we first compute the connectivity matrix $\mathbf{C} = \{c_{ij}\} \in \mathbb{R}^{n \times c}$ that gives the connectivity between all unlabeled vertices to existing label sets

$$\mathbf{C} = \mathbf{A} \mathbf{\Lambda} \mathbf{Y}. \tag{3.34}$$

Then we examine $\mathbf{C}$ to identify the element (i^*, j^*) with minimum value as

$$(i^*, j^*) = \arg \min_{i, j, \mathbf{x}_i \in \mathbf{X}_u} c_{ij}. \tag{3.35}$$

This means the unlabeled vertex $\mathbf{x}_{i^*}$ has the least connectivity with label set $\mathcal{S}_{j^*}$. Then we can update label set $\mathcal{S}_{j^*}$ by adding vertex $\mathbf{x}_{i^*}$ as one greedy step for maximizing the cross-set edge weights. This greedy search can be repeated until all the unlabeled vertices are assigned to labeled sets. In each iteration of the greedy cut process, the weighted connectivity of all unlabeled vertices to each labeled set is re-computed. Then the vertex with minimum connectivity is placed in the proper labeled set. The algorithm is described in the chart (3.3).

Algorithm 3.3 Greedy Gradient based Max-Cut for Label Propagation

Input: the graph $\mathcal{G}_{\mathbf{A}} = \{\mathbf{X}, \mathbf{A}\}$ and the given labeled vertex $\mathbf{X}_l$, and initial label $\mathbf{Y}$;

Initialization:

Obtain the initial cut $\{\mathcal{S}_j\}$ through assigning the labeled vertex $\mathbf{X}_l$ to each subset:

$$\mathcal{S}_j = \{x_i | y_{ij} = 1\}, j = 1, 2, \cdots, c$$

Unlabeled vertex set $\mathbf{X}_u = \mathbf{X} \setminus \mathbf{X}_l$;

repeat

for all $j = 0$ to $|\mathbf{X}_u|$ **do**

Compute weighted connectivity:

$$\mathbf{C}_{ij} = \sum_{k=1}^{n} \lambda_i a_{ik} y_{kj}, \mathbf{x}_i \in \mathbf{X}_u, j = 1, \cdots, c$$

end for

Update the cut $\{\mathcal{S}_i\}$ by placing the vertex $\mathbf{x}_{i^*}$ to the $\mathcal{S}_{j^*}$th subset:

$$(i^*, j^*) = \arg \min_{i,j,\mathbf{x}_i \in \mathbf{X}_u} c_{ij}$$

Add $\mathbf{x}_i$ to $\mathbf{X}_l$: $\mathbf{X}_l \longleftarrow \mathbf{X}_l + \mathbf{x}_i$;

Remove $\mathbf{x}_i$ from $\mathbf{X}_u$: $\mathbf{X}_u \longleftarrow \mathbf{X}_u - \mathbf{x}_i$;

until $\mathbf{X}_u = \emptyset$

Output: The final cut and the corresponding labeled subsets $\mathcal{S}_j, j = 1, 2, \cdots, c$

The connectivity matrix $\mathbf{C}$ can also be viewed as the gradient of the cost function $\mathcal{Q}$ in Eq. (3.12) with respect to $\tilde{Y}$, which is exactly the same as what was used in Eq. (3.13) of the *GTAM* algorithm

$$\mathbf{C} = \frac{\partial \mathcal{Q}}{\partial \tilde{\mathbf{Y}}} = \mathbf{A}\mathbf{\Lambda}\mathbf{Y}. \tag{3.36}$$

We named this the *greedy gradient Max-Cut* (*GGMC*) algorithm since in the greedy step, the unlabeled vertices are assigned labels in a manner that reduces the value of $\mathcal{Q}$ along the direction with the steepest gradient descent. Considering both the variables $\mathbf{Y}$ and $\mathbf{F}$ in the original bivariate formulation in Eq. (3.9), this greedy Max-Cut method is equivalent to the *alternating minimization* procedure between these two variables presented in our *GTAM* algorithm. Different from other graph cut based *SSL* methods, such as mincuts methods in [20][21], our *GGMC* algorithm tends to generate more nature graph cuts avoiding biased solutions due to the use of weighted connectivity. This allows us to effectively address the issues mentioned above and in practice achieve significant gains in accuracy while retaining the algorithm efficiency at the quadratic level (refer to the

following subsection for more details).

3.3.3 Complexity and Speed Up

Assume the graph has $n = |\mathbf{X}|$ vertices and a subset $\mathbf{X}_l$ with $l = |\mathbf{X}_l|$ labeled vertices (where $l \ll n$). The greedy gradient algorithm terminates after at most $n-l \simeq n$ iterations. In each iteration of the greedy gradient algorithm, the connectivity matrix $\mathbf{C}$ is updated by a matrix multiplication (an $n \times n$-matrix is multiplied by a $n \times c$-matrix). Hence, the complexity of the greedy algorithm is $\mathcal{O}(cn^3)$.

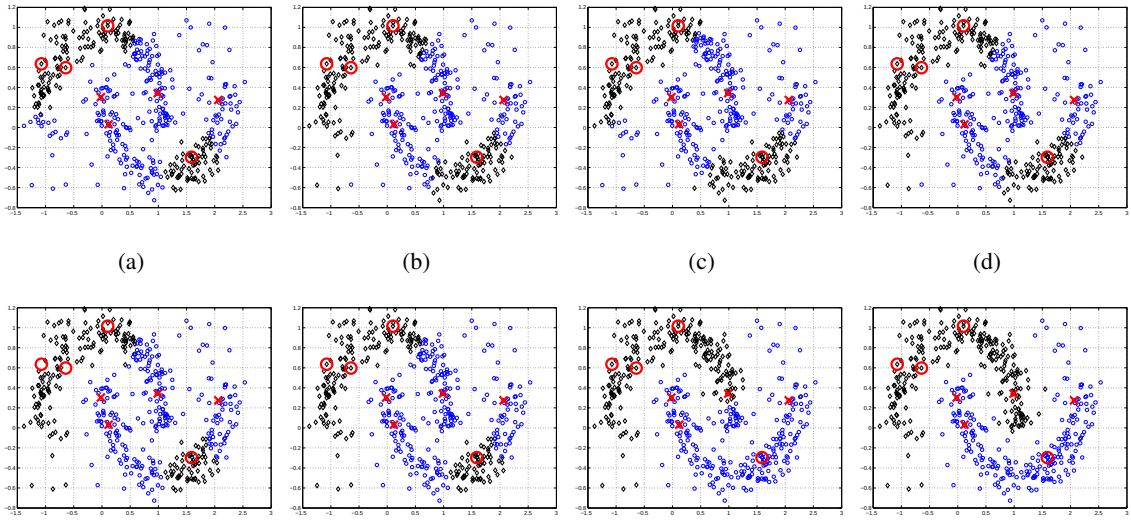
However, the greedy algorithm can be greatly accelerated in practice. For example, the computation of the connectivity in Eq. (3.32) can be done incrementally after assigning each new unlabeled vertex to a certain label set. This circumvents the re-calculation of all the entries in the $\mathbf{C}$ matrix. Assume in the tth iteration the connectivity is $\mathbf{C}^t = \{c^t_{jk}\}$ and an unlabeled vertex $\mathbf{x}_i$ with degree d_i is assigned to the labeled set $\mathcal{S}_j$. Clearly, for all remaining unlabeled vertices, the connectivity to the labeled sets remains unchanged except for the kth labeled set. In other words, only the jth column of $\mathbf{C}$ needs updating. This update is performed incrementally via

$$\mathbf{C}^{t+1}_{.k} = \frac{d_{\mathcal{S}^t_j}}{d_{\mathcal{S}^{t+1}_j}} \mathbf{C}^t_{.k} + \frac{d_i}{d_{\mathcal{S}^{t+1}_j}} \mathbf{A}_{.i}, \tag{3.37}$$

where $d_{\mathcal{S}^{t+1}_j} = d_{\mathcal{S}^t_j} + d_i$ is the sum of the degrees of the labeled vertices after assigning $\mathbf{x}_i$ to the labeled set $\mathcal{S}_j$. This incremental update reduces the complexity of the greedy gradient search algorithm to $\mathcal{O}(n^2)$.

3.4 Label Tuning over Graphs

As mentioned earlier, existing graph-based semi-supervised learning techniques convert the entire data to an undirected graph and then propagate the label information from labeled vertices to the entire graph. Obviously, all these approaches heavily rely on the quality of the training data. The fundamental assumption is that the initial labels are trustworthy. Moreover, the labeled data are expected to have adequate diversity and representation of the sample space. One challenging issue remains open - the labels may not always be reliable. Only few works have addressed this issue. The filter-based approaches were developed to eliminate the unreliable labels in [23][24][182]. Particularly, in [23], ensemble classifiers were developed as filters with a cross validation strategy to



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 3.1: Demonstration of the effect of incorrect labels on prediction results. Large red markers indicate known labels, including wrong labels, and the two-color small markers represent the classification results. a) *SVM*; b) *LapSVM* [15]; c) *RLS* [121]; d) *LapRLS* [15]; e) *GFHF* [182]; f) *LGC* [174]; g) *GTAM* [156]; h) our method *LDST*. Only *LDST* achieved fully correct results.

identify and eliminate mislabeled training instances. However, all these efforts assume sufficient labels and require training supervised classifiers.

The proposed *GTAM* method also suffers from the mislabeling problem. Although it achieved much better performance due to its robustness to labels and graphs, it still can not handle incorrect labels since the initial labels are treated as ground truth. To illustrate the problem, we show the mislabeling issue using the two-moon dataset in Figure 3.1. Among the eight labeled samples, two samples are falsely assigned. The results show that most of the existing techniques, either supervised or semi-supervised methods, generate erroneous prediction results (refer to Figure 3.1(a) - 3.1(g)).

In order to handle problematic labels, we further revise the unilateral greedy search strategy and adopt a bidirectional mechanism to perform label tuning over the graph. It leads to our new approach of *label diagnosis through self tuning* (*LDST*). While preserving the optimal $\mathbf{F}$, *LDST* explores greedy search among the most beneficial gradient directions of $\mathcal{Q}$ on both labeled and unlabeled samples. Following Eq. (3.13), we can compute the derivative $\nabla_{\tilde{\mathbf{Y}}}\mathcal{Q} = \frac{\partial \mathcal{Q}}{\partial \tilde{\mathbf{Y}}}$, which measures the rate of change of the cost function $\mathcal{Q}$ with respect to the normalized label $\tilde{\mathbf{Y}}$.

Note that $\mathbf{Y}$ is constrained in a binary space. Therein, the labeling operation changes the value from 0 to 1 for a certain element y_{ij} in the label matrix, and the unlabeling operation (i.e., removing the labels) does the reverse by setting $y_{ij} = 1 \rightarrow 0$. To reduce the value of the cost function $\mathcal{Q}$, we manipulate the label variable $\mathbf{Y}$ in both directions, labeling and unlabeling. Note that the labeling operation is carried out on the unlabeled nodes with the minimum value of the gradient $\min \nabla_{\tilde{\mathbf{Y}}} \mathcal{Q}$, while the unlabeling operation is executed on the labeled nodes with the maximum value of gradient $\max \nabla_{\tilde{\mathbf{Y}}} \mathcal{Q}$. To summarize, we have the following bidirectional gradient descent search, including both labeling and unlabeling operations, to achieve the steepest reduction on the cost function $\mathcal{Q}$:

$$\begin{aligned} (i^+, j^+) &= \arg\min_{\mathbf{x}_i \in \mathbf{X}_u, 1 \leq j \leq c} \nabla_{\tilde{y}_{ij}} \mathcal{Q}; \quad y_{i^- j^-} = 1 \\ (i^-, j^-) &= \arg\max_{\mathbf{x}_i \in \mathbf{X}_l, 1 \leq j \leq c} \nabla_{\tilde{y}_{ij}} \mathcal{Q}; \quad y_{i^- j^-} = 0, \end{aligned} \tag{3.38}$$

where (i^+, j^+) and (i^-, j^-) are the optimal elements in variable $\mathbf{Y}$ for labeling and unlabeling operations, respectively. Different from the labeling procedure, the optimal element for the unlabeling operation is only performed on the positions of variable $\mathbf{Y}_l$ where the element has nonzero values. In other words, through each bidirectional gradient descent iteration, we add one most reliable label, and remove one least confident label.

We summarize the *LDST* method in Algorithm (3.4). In the first s iterations, a number of unlabeling and labeling operations are executed in order to eliminate the problematic labels and add trustable new labels. We refer to this stage as *LDST-self-tuning*. In this self tuning stage, the same number of labels are added and removed to maintain a fixed size label pool. Moreover, each individual operation of labeling and unlabeling leads to the update of the label weight matrix $\mathbf{\Lambda}$. After executing s steps of label self tuning, the subsequent stage, called *LDST-propagation*, is conducted to propagate the updated labels to the rest of the graph. Note that the *LDST-propagation* procedure is essentially the same as *GTAM* since it continuously selects the most confident unlabeled vertices and assigns them with proper labels.

Theoretically, the algorithm continues until all the unlabeled samples are labeled. However, this may result in a prohibitive level of computation if the data size is huge. There are two strategies we may use to speed up the algorithm. First, the iterative procedure of *LDST-propagation* can be terminated early after obtaining enough labels. The final prediction results are computed directly using Eq. (3.4). Second, the computation associated with matrix multiplication in calculating gra-

Algorithm 3.4 Label Diagnosis through Self Tuning (*LDST*)

Input: data set $\mathbf{X} = \{\mathbf{X}_l, \mathbf{X}_u\}$, where $\mathbf{X}_l = \{\mathbf{x}_1, \cdots, \mathbf{x}_l\}$ and $\mathbf{X}_u = \{\mathbf{x}_{l+1}, \cdots, \mathbf{x}_n\}$, labels $\mathcal{Z}_l = \{z_i\}, i = 1, \cdots, l$, $z_i \in \{1, \cdots, l\}$. Weight matrix $\mathbf{W} = \{w_{ij}\}$, node degree matrix $\mathbf{D}$, initial label matrix $\mathbf{Y}^0$, graph Laplacian $\mathbf{L}$, propagation matrix $\mathbf{P}$, matrix $\mathbf{A}$;

Initialization:

iteration counter $t = 0$, initial label weight matrix $\mathbf{\Lambda}^t$;

repeat

Compute partial derivatives: $(\nabla_{\tilde{\mathbf{Y}}} \mathcal{Q})^t = \mathbf{A}\tilde{\mathbf{Y}}^t$;

if $t \leq s$ **then**

Find the optimal element in $(\nabla_{\tilde{\mathbf{Y}}} \mathcal{Q})^t$: $(i^-, j^-) = \arg\max_{\mathbf{x}_i \in \mathbf{X}_l, 1 \leq j \leq c} (\nabla_{\tilde{y}_{ij}} \mathcal{Q})^t$;

Update label matrix by setting: $y_{i^- j^-}^{t+1} = 0$; also $z_{i^-} = 0$;

Update $\mathbf{X}_l, \mathbf{X}_u$: $\mathbf{X}_l^{t+1} \longleftarrow \mathbf{X}_l^t - \mathbf{x}_{i^-}$; $\mathbf{X}_u^{t+1} \longleftarrow \mathbf{X}_u^t + \mathbf{x}_{i^-}$;

end if

Find the optimal element in $(\nabla_{\tilde{\mathbf{Y}}} \mathcal{Q})^t$: $(i^+, j^+) = \arg\min_{\mathbf{x}_i \in \mathbf{X}_l, 1 \leq j \leq c} (\nabla_{\tilde{y}_{ij}} \mathcal{Q})^t$;

Update label matrix by setting: $y_{i^+ j^+}^{t+1} = 1$; also $z_{i^+} = j^+$;

Update $\mathbf{X}_l, \mathbf{X}_u$: $\mathbf{X}_l^{t+1} \longleftarrow \mathbf{X}_l^t + \mathbf{x}_{i^+}$; $\mathbf{X}_u^{t+1} \longleftarrow \mathbf{X}_u^t - \mathbf{x}_{i^+}$;

Update label weight matrix $\mathbf{\Lambda}^{t+1}$ based on Eq. (3.10);

Update iteration counter: $t = t + 1$;

until $\mathbf{X}_u^t = \emptyset$ or maximum iterations achieved.

Output: The labels of unlabeled samples $\{z_{l+1}, \cdots, z_n\}$.

dient $\nabla_{\tilde{\mathbf{Y}}} \mathcal{Q}^{t+1}$ can be converted to vector addition since each step only involves changing a single column entry in $\nabla_{\tilde{\mathbf{Y}}} \mathcal{Q}^t$, similar to the idea for speeding up *GTAM* in Section 3.3. Finally, in order to guarantee convergence and avoid backtracking in the bidirectional greedy search, we can constrain that after an unlabeled point has been labeled, its label can no longer be changed.

3.5 Multi-Graph GTAM

Note that all the above methods represent data in a single graph. However, in many applications, the data can have multiple representations naturally. For example, the web can be represented as different relationship maps, either by a directed graph with hyperlinks as edges or by an undirected

similarity graph in the feature space of the Bag-of-Word model [62]. For the applications of visual search, there are even more representations for images, such as SIFT features [100], GIST features [114], and sparse coding based features [95][168]. Even with the same feature space, graph construction also varies in many ways, including kernel selection, sparsification, and edge weighting, as discussed in Section 2.3. The choices of data representation and the graph construction process result in a myriad of graphs. In this section, we develop a new algorithm, called *Multi-Graph GTAM (MG-GTAM)*, which alternatively identifies the most confident unlabeled vertices for label assignment by considering multiple graphs, and combine the predictions from each individual graph to achieve more accurate labels over the entire label set.

Note that the learning problem using multiple representations simultaneously is termed as *multi-view learning* in previous literature [125]. For instance, in [137][139], a framework of joint complexity regularization, named *Co-Regularization*, was proposed for semi-supervised learning with multiple views. A closely related work using a mixture of Markov chains defined on different views was proposed in [175] for both spectral clustering and transductive learning. Another method of multi-view learning is to derive a convex combination of multiple kernels, each of which comes from one data representation [3][173]. However, most of the existing methods tend to learn a fixed combination of the regularization terms or graph weights and require sufficient labeled data.

Recall the cost function $\mathbf{Q}(\mathbf{F}, \mathbf{Y})$ defined over both the classification function $\mathbf{F}$ and the binary label matrix $\mathbf{Y}$ in the *GTAM* method, as shown in Eq. (3.9). Instead of using only one graph $\mathcal{G}$, now we have a set of graphs $\{\mathcal{G}_1, \cdots, \mathcal{G}_m\}$ constructed over the given data with either different features or graph construction processes. The natural extension of the formulation from a single graph to multiple graphs is to combine the smoothness terms from multiple graphs into a single cost function, as shown below:

$$\begin{aligned} \mathbf{F}^* = \quad & \arg\min_{\mathbf{F}\in\mathcal{R}^{|V|\times c}, \mathbf{Y}\in\mathcal{B}^{|V|\times c}, \boldsymbol{\alpha}\in\mathcal{R}^m} \mathcal{Q}(\mathbf{F}, \mathbf{Y}, \boldsymbol{\alpha}) = \arg\min \sum_{q=1}^{m} \alpha_q \|\mathbf{F}_{\mathcal{G}_q}\|^2 + \tfrac{\mu}{2}\|\mathbf{F} - \mathbf{V}\mathbf{Y}\|^2 \\ s.t. \quad & \sum_{q=1}^{m} \alpha_q = 1, \alpha_q \geq 0, \ \forall\, q \ \in \{1, \ldots, m\} \\ & \mathbf{Y}_{ij} \in \{0, 1\}, \ \textstyle\sum_j \mathbf{Y}_{ij} = 1, \ \forall\, i \ \in \{1, \ldots, n\}, j \ \in \{1, \ldots, c\}, \end{aligned} \tag{3.39}$$

where $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_m]$ are the weights corresponding to different graphs. The smoothness term

can be written in the following form with multiple graph Laplacians:

$$\begin{aligned}\sum_{q=1}^{m}\alpha_q\|\mathbf{F}_{\mathcal{G}_q}\|^2 &= \frac{1}{2}\mathrm{tr}\left(\sum_{q=1}^{m}\alpha_q\mathbf{F}^\top\mathbf{L}_q\mathbf{F}\right)\\ &= \frac{1}{2}\mathrm{tr}\mathbf{F}^\top\left(\sum_{q=1}^{m}\alpha_q\mathbf{L}_q\right)\mathbf{F} = \frac{1}{2}\mathrm{tr}\mathbf{F}^\top\mathcal{L}\mathbf{F},\end{aligned} \tag{3.40}$$

where we define the combined graph Laplacian as the weighted sum of Laplacians of all the graphs

$$\mathcal{L} = \sum_{q=1}^{m}\alpha_q\mathbf{L}_q. \tag{3.41}$$

The problem in Eq. (3.39) is intractable since it is a constrained mixed-integer programming problem. A similar idea was proposed in [3], called an optimal *combined kernel*, where an alternative optimization strategy was used [4]. A similar technique, called *co-regularization*, was proposed in [137]. However, both of these approaches are inefficient in practical applications due to the need for computing the combined graph Laplacian in each iteration of the optimization procedure. Moreover, the *co-regularization* method include additional $q(q-1)$ regularization terms, which significantly increases the computational cost. Another work using multiple graphs for clustering was proposed in [144], which uses a combined affinity matrix. In summary, this kind of early fusion strategy with multiple graphs suffers from the overhead of additional computational cost and thus is less desirable in practice.

Here we propose a more efficient way to extend the *GTAM* method from a single graph to multiple graphs by using a novel approach that aggregates the most confident labels captured from multiple graphs. First, consider the transductive inference over an individual graph by solving $\arg\min_{\mathbf{F}}\mathbf{Q}(\mathbf{F},\mathbf{Y})$ with the label variable $\mathbf{Y}$ fixed. Then the optimal prediction functions $\mathcal{F} = \{\mathbf{F}_1,\cdots,\mathbf{F}_m\}$ can be derived for all the given graphs $\{\mathcal{G}_1,\cdots,\mathcal{G}_m\}$ independently. Then the weighted combination over the prediction functions from individual graphs can be computed as:

$$\mathbf{F} = \sum_{q=1}^{m}\alpha_q\mathbf{F}_q, \tag{3.42}$$

where $\boldsymbol{\alpha} = [\alpha_1,\cdots,\alpha_m]$ are the weights, and large values of the components in $\boldsymbol{\alpha}$ indicate the most relevant graphs. Recall the definition of the label weight $\boldsymbol{\Lambda} = \{\lambda_i\}, i = 1,\cdots,n$ in Eq. (3.10)

and extend it to the multiple-graph case as:

$$\lambda_i = \begin{cases} \sum_{q=1}^{m} \alpha_q \frac{\mathbf{D}_{ii}^q}{\sum_k \mathbf{Y}_{kj}\mathbf{D}_{kk}} & : \quad \mathbf{Y}_{ij} = 1 \\ 0 & : \quad \text{otherwise.} \end{cases} \tag{3.43}$$

The above extension of label weight is very intuitive since the weighted sum of the normalized density, rather than the density from a single graph, is calculated as the importance measurement of each label. Given the above combined predictions and normalized density, we can define the following cost function over multiple graphs as:

$$\mathcal{Q}(\mathcal{F}, \tilde{\mathbf{Y}}, \boldsymbol{\alpha}) = \frac{1}{2}\sum_{q=1}^{m} \alpha_q \text{tr}\left(\mathbf{F}_q^\top \mathbf{L}_q \mathbf{F}_q\right) + \frac{\mu}{2}\sum_{q=1}^{m} \alpha_q \left(\mathbf{F}_q - \tilde{\mathbf{Y}}\right)^\top \left(\mathbf{F}_q - \tilde{\mathbf{Y}}\right). \tag{3.44}$$

Although the minimization problem of the above cost function is nontrivial, a similar optimizing strategy as that in *GTAM* can be applied to derive local optimal solutions. We first derive the optimal prediction function over each graph as:

$$\begin{aligned} \frac{\partial \mathcal{Q}}{\partial \mathbf{F}_q} = 0 \qquad & \Longrightarrow \alpha_q \mathbf{L}_q \mathbf{F}_q + \mu\alpha_q(\mathbf{F}_q - \tilde{\mathbf{Y}}) = 0 \\ & \Longrightarrow \mathbf{F}_q^* = (\mathbf{L}_q/\mu + \mathbf{I})^{-1}\tilde{\mathbf{Y}} = \mathbf{P}_q\tilde{\mathbf{Y}} \qquad (3.45) \\ & \mathbf{P}_q = (\mathbf{L}_q/\mu + \mathbf{I})^{-1}, \qquad (3.46) \end{aligned}$$

where $\mathbf{P}_q$ is the propagation matrix over graph $\mathcal{G}_q$. Replace the optimal $\mathbf{F}_q^*$ in Eq. (3.44) and we can derive the following:

$$\begin{aligned} \mathcal{Q}(\tilde{\mathbf{Y}}, \boldsymbol{\alpha}) &= \frac{1}{2}\text{tr}\left(\tilde{\mathbf{Y}}^\top \left[\sum_{q=1}^{m} \alpha_q \left(\mathbf{P}_q^\top \mathbf{L}_q \mathbf{P}_q + \mu(\mathbf{P}_q - \mathbf{I})^2\right)\right]\tilde{\mathbf{Y}}\right) \\ &= \frac{1}{2}\text{tr}\left[\tilde{\mathbf{Y}}^\top \left(\sum_{q=1}^{m} \alpha_q \mathbf{A}_q\right)\tilde{\mathbf{Y}}\right] = \frac{1}{2}\sum_{q=1}^{m} \alpha_q \text{tr}\left(\tilde{\mathbf{Y}}^\top \mathbf{A}_q \tilde{\mathbf{Y}}\right) \qquad (3.47) \\ & \mathbf{A}_q = \mathbf{P}_q^\top \mathbf{L}_q \mathbf{P}_q + \mu(\mathbf{P}_q - \mathbf{I})^2. \qquad (3.48) \end{aligned}$$

The constant matrix $\mathbf{A}_q$ is determined by the structure of graph $\mathcal{G}_q$ and can be computed off-line for each individual graph. Since the cost function $\mathcal{Q}$ is linear with respect to $\boldsymbol{\alpha}$, it can be treated as a convex combination over $\{\mathbf{A}_q\}, q = 1, \cdots, m$. We apply a gradient descent method, which alternatively optimizes over $\tilde{\mathbf{Y}}$ and $\boldsymbol{\alpha}$, similar as in [4].

Algorithm 3.5 Multi-Graph GTAM (*MG-GTAM*)

Input: data set $\mathbf{X} = \{\mathbf{X}_l, \mathbf{X}_u\}$, multiple graphs $\{\mathcal{G}_q\}$, weight matrices $\{\mathbf{W}_q\}$, vertex degree matrix $\{\mathbf{D}_q\}$, graph Laplacians $\{\mathbf{L}\}_q$, propagation matrix $\{\mathbf{P}_q\}$, matrix $\{\mathbf{A}_q\}$, $q = 1, \cdots, m$, initial label matrix $\mathbf{Y}^0$;

Initialization: iteration counter $t = 0$; iteration number T, $\alpha_q = 1/m, q = 1, \cdots, m$, label weights $\mathbf{\Lambda}^t = \{\lambda_i^t\}, i = 1, \cdots, n$;

for $i = 1$ to T **do**

Compute partial derivatives $(\frac{\partial \mathcal{Q}}{\partial \tilde{\mathbf{Y}}})^t$ as Eq. (3.49)

Find the optimal element $(i^*, j^*) = \arg\min_{\mathbf{x}_i \in \mathbf{X}_u, 1 \le j \le c} (\frac{\partial \mathcal{Q}}{\partial \tilde{\mathbf{Y}}})^t$

Update label matrix to obtain $\mathbf{Y}^{t+1}$ by setting: $y_{i^* j^*}^{t+1} = 1$; also $z_{i^*} = j^*$;

Update labeled and unlabeled sets: $\mathbf{X}_l^{t+1} \longleftarrow \mathbf{X}_l^t + \mathbf{x}_{i^*}$, $\mathbf{X}_u^{t+1} \longleftarrow \mathbf{X}_u^t - \mathbf{x}_{i^*}$;

Update label weights $\mathbf{\Lambda}^t$ as in Eq. (3.43)

Compute partial derivatives $(\frac{\partial \mathcal{Q}}{\partial \boldsymbol{\alpha}})^t = [(\frac{\partial \mathcal{Q}}{\partial \alpha_1})^t, \cdots, (\frac{\partial \mathcal{Q}}{\partial \alpha_q})^t]$ as Eq. (3.50)

Update $\boldsymbol{\alpha}$ as $\boldsymbol{\alpha} = \boldsymbol{\alpha} - \eta(\frac{\partial \mathcal{Q}}{\partial \boldsymbol{\alpha}})^t$ and truncate negative values to 0;

Normalize $\boldsymbol{\alpha}$ as $\alpha_q = \alpha_q / \sum_{k=1}^{m} \alpha_k, q = 1, \cdots, m$;

Update iteration counter: $t = t + 1$;

end for

Output: Compute final prediction function $\mathbf{F} = \sum_{q=1}^{m} \alpha_q \mathbf{F}_q$, where $\mathbf{F}_q$ is computed as Eq. (3.45).

The partial derivatives of $\mathcal{Q}$ over $\tilde{\mathbf{Y}}$ and $\boldsymbol{\alpha}$ can be computed as:

$$\frac{\partial \mathcal{Q}}{\partial \tilde{\mathbf{Y}}} = \sum_{q=1}^{m} \alpha_q \mathbf{A}_q \tilde{\mathbf{Y}} \tag{3.49}$$

$$\frac{\partial \mathcal{Q}}{\partial \alpha_q} = \frac{1}{2} \mathrm{tr} \left(\tilde{\mathbf{Y}}^\top \mathbf{A}_q \tilde{\mathbf{Y}} \right). \tag{3.50}$$

As clarified in Section 3.2, the update over the normalized label matrix $\tilde{\mathbf{Y}}$ is equivalent to updating the original label matrix $\mathbf{Y}$, where $\mathbf{Y}$ and $\tilde{\mathbf{Y}}$ have one-to-one correspondence. Therefore, we identify the minimal element of the unlabeled part in $\frac{\partial \mathcal{Q}}{\partial \tilde{\mathbf{Y}}}$ as:

$$(i^*, j^*) = \arg\min_{\mathbf{x}_i \in \mathbf{X}_u, 1 \le j \le c} \frac{\partial \mathcal{Q}}{\partial \tilde{\mathbf{Y}}}, \tag{3.51}$$

and update the label matrix by setting $y_{i^*, j^*} = 1$. The update of $\mathbf{Y}$ is indeed a labeling procedure that assigns the most confident unlabeled vertex with the proper label. With the updated $\mathbf{Y}$, we

recompute the node weight $\mathbf{\Lambda}$ using Eq. (3.43), and correspondingly update $\tilde{\mathbf{Y}}$. After finishing the update of the $\mathbf{Y}$ matrix, the coefficients $\boldsymbol{\alpha}$ can also be updated using the gradient descent approach

$$\alpha_q = \alpha_q - \eta \frac{\partial \mathcal{Q}}{\partial \alpha_q}, \tag{3.52}$$

where η is the step length. Since $\boldsymbol{\alpha} = \{\alpha_q\}, q = 1, \cdots, m$ is constrained as $\sum_q \alpha_q = 1$ and $\alpha_q \geq 0$, we need to normalize the α_q after each iteration. The updating procedure of the elements in $\boldsymbol{\alpha}$ can be interpreted as imposing higher weights to the most relevance graphs.

As summarized in the Algorithm (3.5), the alternative optimization procedure over both $\mathbf{\Lambda}$ and $\mathbf{Y}$ runs T iterations to find a solution as predictions over all data. Since the elements in $\boldsymbol{\alpha}$ are positively constrained, we check the signs of all α_q and truncate negative ones to be zero after each iteration. Note that the above optimization strategy is fairly efficient since we only need fusion prediction scores instead of having to compute new graph Laplacians or graph edge weights in each iteration. Specifically, the gradient descent for $\boldsymbol{\alpha}$ requires linear time computation and the complexity of the gradient descent for $\mathbf{Y}$ is $\mathcal{O}(n^2)$, which can further speed up to the $\mathcal{O}(n)$ complexity if we adopt an implementation trick as described in Section 3.3, which just incrementally updates single column vectors of the partial derivatives.

3.6 Experiments

In this section, we will present a set of experiments to evaluate the performance of the methods described in this chapter.

a. *Graph transduction via alternating minimization (GTAM):* We will demonstrate the superiority of the proposed *GTAM* method in comparison with the state-of-the-art semi-supervised learning methods over both synthetic and real data. Particularly, we compare with *TSVM* [79], *LapSVM* [138] [14], and two other closely related methods, namely *LGC* [174] and *GFHF* [182].

b. *Label diagnosis through self tuning (LDST):* By simulating noisy and incorrect labels on the two-moon artificial dataset, we show the unique capability of the *LDST* method in handling wrong labels.

c. *Multi-graph GTAM (MG-GTAM):* For the multiple-graph method, i.e. *MG-GTAM*, we show the comparison with the combined graph Laplacian (*CGL*) method [3], demonstrating the

significant performance gain when using very few labels. In addition, the *MG-GTAM* method requires much less computation than the *CGL* approach.

For fair comparison, we use the same graph construction process for different methods. Specifically, the widely used kNN approach is applied for sparsifying the original full adjacency matrix. For edge weighting, we use the following different strategies, as suggested in [74] [156]:

1. *Binary weighting:* All the linked edges in the graph are given the weight 1 and the edge weights of disconnected nodes are given weight 0. In other words, this setting simply uses $\mathbf{W} = \mathcal{P}$. Notice that these binary weights on graph edges can be sensitive, in particular if some of the graph nodes were improperly connected.

2. *Fixed Gaussian kernel weighting:* The edge weight between two connected samples $\mathbf{x}_i$ and $\mathbf{x}_j$ is computed as:

$$\mathbf{w}_{ij} = \mathcal{P}_{ij} \exp\left(-\frac{d(\mathbf{x}_i, \mathbf{x}_j)^2}{2\sigma^2}\right). \tag{3.53}$$

   The parameter σ is the kernel bandwidth parameter, which was uniformly set as the average distance between each selected sample and its kth nearest neighbor [27]. This fixed kernel size might be infeasible for some real data since the samples might not be sampled evenly and uniformly.

3. *Adaptive Gaussian kernel weighting:* As suggested in [64][156], here the kernel bandwidth parameter σ is locally scaled to the mean distance of k-nearest neighborhoods of the sample $\mathbf{x}_i, \mathbf{x}_j$. Assume the k nearest neighbor set of $\mathbf{x}_i$ is $\mathcal{N}_{\mathbf{x}_i}$ with cardinality $N_1 = |\mathcal{N}_{\mathbf{x}_i}|$ and the neighbor set of $\mathbf{x}_j$ is $\mathcal{N}_{\mathbf{x}_j}$ with cardinality $N_2 = |\mathcal{N}_{\mathbf{x}_j}|$, the adaptive kernel bandwidth is computed as:

$$\sigma = \frac{1}{N_1 + N_2}\left(\sum_{\mathbf{x}_p \in \mathcal{N}_{\mathbf{x}_i}} d(\mathbf{x}_i, \mathbf{x}_p) + \sum_{\mathbf{x}_q \in \mathcal{N}_{\mathbf{x}_j}} d(\mathbf{x}_j, \mathbf{x}_q)\right). \tag{3.54}$$

In the experiments, we tested the above three different edge weighting strategies and showed the robustness of the proposed methods. For all the experiments, the ℓ_2 distance $d_{\ell_2}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\|\mathbf{x}_i - \mathbf{x}_j\|^2}$ was used.

3.6.1 Evaluation of *GTAM*

Noisy Two-Moon Dataset:

We first compared our *GTAM* method with state-of-the-art *SSL* algorithms over the noisy two-moon dataset, as shown in Figure 2.4. Despite the near-perfect classification results reported on the clean two-moon dataset in the literature [138][174], we showed how a small level of noise added to the dataset could affect the results of the previous algorithms. In Figure 2.4, two separable manifolds containing 600 two-dimensional points are mixed with 100 noisy outlier samples. This modification makes the prior methods fail to achieve reasonable prediction results because they are overly sensitive to locations of the initial labels, ratios of the two-class labels, and the level of ambient noise or outliers. In order to obtain reliable evaluation of the performance, 100 independent trials with random selection are repeated under each setting and the average error rates are recorded.

The first group of experiments were conducted by varying the number of labels. We uniformly set $k = 6$ to construct the neighborhood graph and applied the above three types of edge weighting schemes. The average error rates of the predictions over the unlabeled points are shown in Figures 3.2(a), 3.2(b), 3.2(c), corresponding to binary edge weighting, fixed Gaussian kernel edge weighting, and adaptive Gaussian kernel edge weighting, respectively. The results clearly show that the *GTAM* method is robust to a range of label set sizes and generates perfect prediction results for all three versions of edge weights.

The second group of experiments demonstrate the influence of the number of linkages, i.e., the value of k, for neighborhood graph constructions. Specifically, we varied the value of k from 4 to 20 and Figures 3.2(d), 3.2(e), 3.2(f) show the results for different versions of edge weights. Apparently, *GTAM* generates significantly better performance than other algorithms for most of the test cases.

Finally, a thorough study is also conducted to assess the effect of imbalanced labels on *SSL* algorithms. We first fix one class to have only one label and select r labels fore the other class. Here, r indicates the imbalance ratio and we study the range $1 \leq r \leq 20$. Figures 3.2(g), 3.2(h), 3.2(i) show the results with different edge weighting methods. Clearly, *GTAM* is insensitive to the imbalance condition since the per-class label weight normalization is shown to be effective in compensating the differences of label size.

In summary, Figure 3.2 demonstrates the performance advantage of the proposed *GTAM* approach versus the *LGC*, *GFHF*, *LapRLS*, and *LapSVM* methods. From the figure, we can conclude

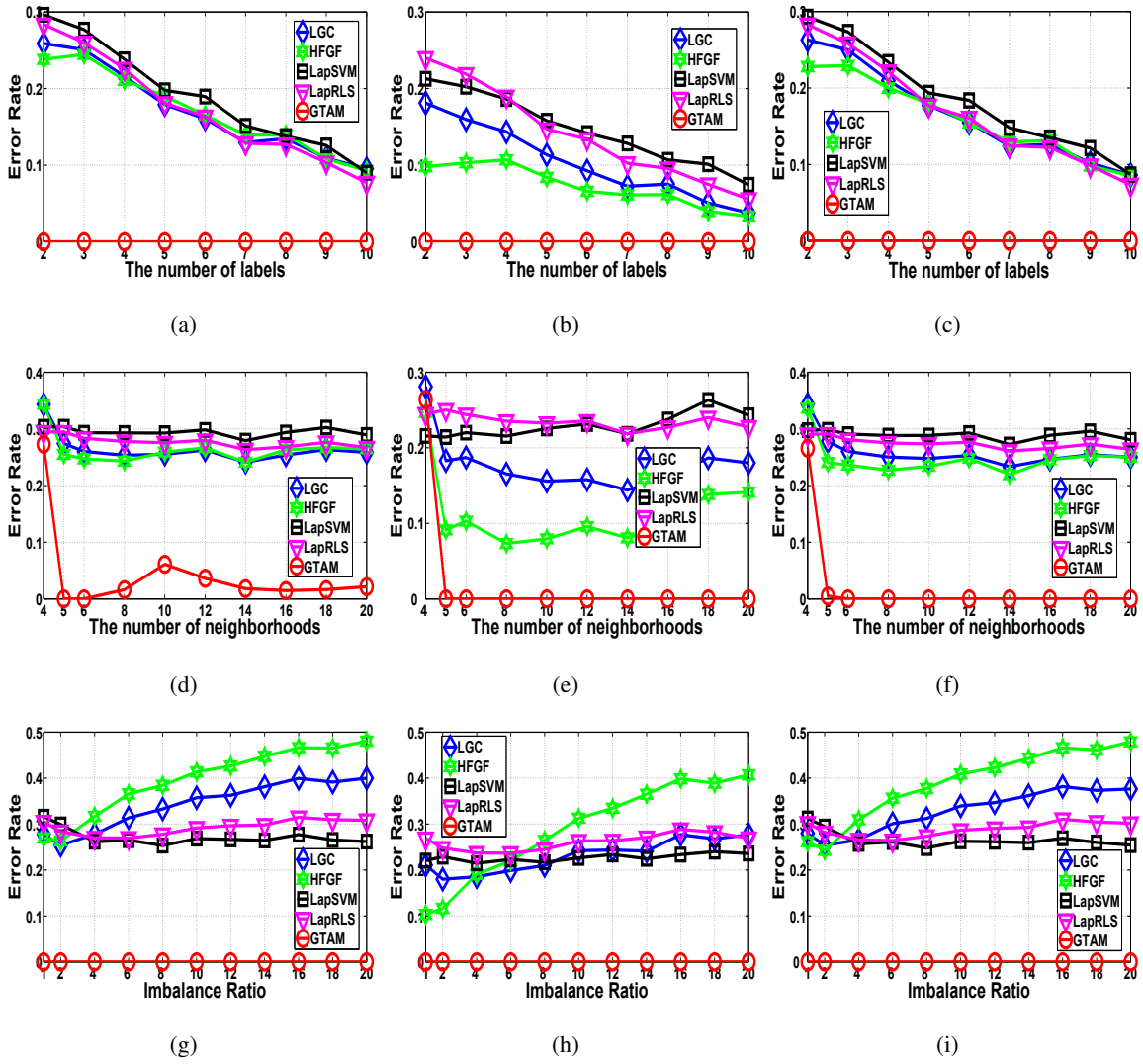
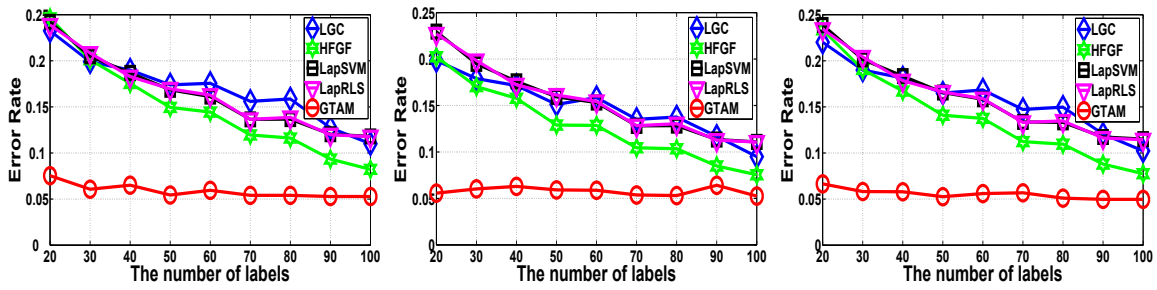


Figure 3.2: Experimental results on the noisy two-moon dataset simulating different graph construction approaches and label conditions. Figures a) d) g) use binary weighting. Figures b) e) h) use fixed Gaussian kernel weighting. Figures c) f) i) use adaptive Gaussian kernel weighting. Figures a) b) c) vary the number of labels. Figures d) e) f) vary the value of k in the graph construction. Figures g) h) i) vary the label imbalance ratios.



(a) (b) (c)

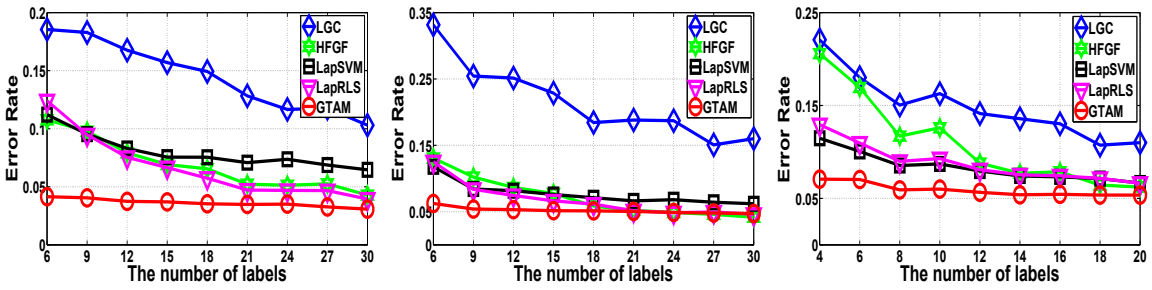
Figure 3.3: Experimental results on the USPS digits dataset under varying levels of labeling using: a) binary weighting; b) fixed Gaussian kernel weighting and c) adaptive Gaussian kernel weighting.

that all the four prior approaches are very sensitive to the initial labels since none of them can produce perfect predictions, and the error rates of *LGC* and *GFHF* are significantly increased when the label class becomes more imbalanced, even when more labels become available. However, *GTAM* is clearly superior, achieving the best accuracy regardless of the imbalance ratio and the size of the label set. Furthermore, the *GTAM* approach is also more robust to graph construction in terms of different weighting strategies and number of edge linkages.

USPS Handwritten Digits:

We also evaluated the proposed method in an image recognition task, i.e. classifying handwritten digits $0 \sim 9$ in the USPS database. The dataset contains gray scale handwritten digit images with the resolution 16×16 scanned from envelopes by the U.S. Postal Service. We used a subset of the data by randomly selecting 4000 samples. For all the constructed graphs, the value of k was uniformly set as 6 and three different edge weighting approaches were tested. We varied the total number of labels from 20 to 100 and guaranteed that each digit class had at least one label. For each setting, the average error rate was computed over 20 random trials.

The experimental results are shown in Figures 3.3(a), 3.3(b), 3.3(c), corresponding to three different edge weighting approaches. We can conclude that *GTAM* significantly improves the classification accuracy, compared to all the other approaches, especially when very few labeled samples are available. The mean error rates of *GTAM* are consistently low with a very small standard deviation (10^{-4} level). This again demonstrates that the *GTAM* method is insensitive to the numbers and specified locations of the initial labels.



(a) (b) (c)

Figure 3.4: Performance of *LGC*, *GFHF*, *LapRLS*, *LapSVM*, and *GTAM* algorithms using the UCI datasets. The horizontal axis is the number of training labels provided while the vertical axis is the average error rate achieved over 100 random folds. Results are shown in a) for the Iris dataset, in b) for the Wine dataset and in c) for the Breast Cancer dataset.

UCI Datasets:

Finally, we tested *GTAM* and other methods over several of the most popular benchmark datasets from the UCI Machine Learning Repository [52] in real application domains. Specifically, we use datasets of Iris, Wine, and Breast Cancer. The datasets are collected from several different disciplines and are related to different specific applications. For instance, the Iris dataset has been used to predict the type of the iris plant. In the Wine dataset, some measured chemical properties are used to determine the origin of wines. The Breast Cancer dataset is related to the characteristics of the cell nuclei extracted from digitized images of a fine needle aspirate (FNA) of the breast mass for discrimination between a benign and malignant diagnosis. Since each attribute of these datasets could have significantly different scales, we normalized each feature to the range $[0, 1]$. For all three datasets, we applied the same graph construction process, where $k = 6$ was used for constructing the neighborhood graph and the edge weight is computed using the Gaussian kernel with fixed bandwidth.

Figure 3.4 shows the results of *GTAM* and several other *SSL* methods. The vertical axis indicates the average error rate computed over 100 random trials and the horizontal axis shows the number of labeled samples. Again, our *GTAM* method outperformed others in most of the test cases with significant performance gain when very few labeled data were available.

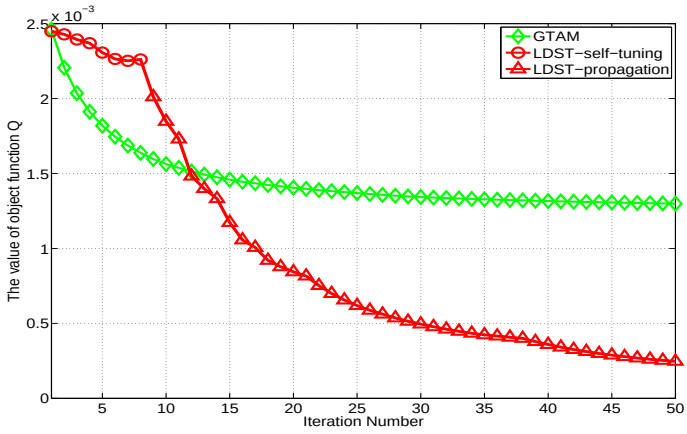


Figure 3.5: The cost function $\mathcal{Q}$ during optimization procedures of the *LDST* and *GTAM* methods. The *LDST* method reaches a much lower value of the cost function after the initial steps of label tuning.

3.6.2 Evaluation of *LDST*

In this test, we evaluate the performance of the proposed *LDST* method by using a manipulated version of the two-moon dataset in which a certain number of label errors are added, as shown in Figure 3.1. Large red markers are the labeled samples and the shape represents different classes, i.e., positive or negative. Each class is assigned four labeled samples, among which one is mislabeled.

We compared a few different approaches, including supervised methods, like the standard *SVM*, existing *GSSL* algorithms, and our *LDST* method. Again for all the graph-based approaches, we use the common setting to build a KNN graph with the same number of neighborhoods ($k = 6$). For parameters of other methods, we adopt the best setting reported in previous literature [15][156][174][182].

Figure 3.1 shows the classification results using different methods. From this test, several observations can be obtained. First, the performance of supervised methods like *SVM* is heavily degraded due to the blind trust on the false labels and the exclusive reliance on the labeled data. Second, most *SSL* methods generate erroneous results even though both the labeled and unlabeled data are used. It is because the labeled samples outweigh the unlabeled data in driving the inference process. In particular, some algorithms, like *GFHF*, require the prediction results to exactly match the initial labels. *LGC* incorporates an elastic fitness term in the cost function but can not rectify the wrong

Method	SVM	LapSVM	RLS	LapRLS	GFHF	LGC	GTAM	LDST
Error (%)	34.64	30.16	34.01	30.26	38.76	23.77	5.99	**0.91**
Std (%)	7.03	10.63	11.23	10.67	3.69	6.82	11.24	**2.43**

Table 3.1: The mean and standard deviation of the error rates on 20 random tests.

labels. *GTAM* achieves the best accuracy among the methods without tuning initial labels due to its bivariate formulation and iterative label propagation procedure. However, it still fails to completely separate the manifolds due to the significant level of noise. It is clear that these *SSL* methods lack the capability to identify and eliminate false labels, and thus produce inaccurate predictions by propagating erroneous label information. On the contrary, the proposed *LDST* method uses a self tuning process to eliminate unreliable labels leading to accurate prediction results without breaking the structure of the two manifolds, as shown in Figure 3.1(g). Since *GTAM* achieved a high accuracy close to that achieved by *LDST*, and both share the common bivariate optimization framework, we analyze the cost function value $\mathcal{Q}$ of these two methods by showing the first 50 iterations during the optimization process in Figure 3.5. This figure clearly shows that after pruning the wrong labels by self tuning (the first 8 iterations marked as red circles), the cost descends rapidly in the propagation stage thereafter. This demonstrates that most of the performance improvement of *LDST* over *GTAM* can be attributed to the label self tuning procedure.

In addition, a comprehensive comparison study was conducted by 20 rounds of random tests. For each round, three correct labels and one wrong label were randomly assigned to each class. The graph construction options and several parameters, like the number of self tuning iterations $s = 8$, were fixed for all the runs. The mean and standard deviation of the classification errors are recorded in Table 3.1. From this evaluation, *LDST* achieves much higher and more stable performance, which further confirms the superiority of the *LDST* method for predicting labels using an initial set of imperfectly labeled samples.

3.6.3 Evaluation of *MG-GTAM*

We evaluated the performance of the proposed *MG-GTAM* method using the USPS dataset. Similar to the experimental setting in [3], four different pairwise classification tasks were conducted to

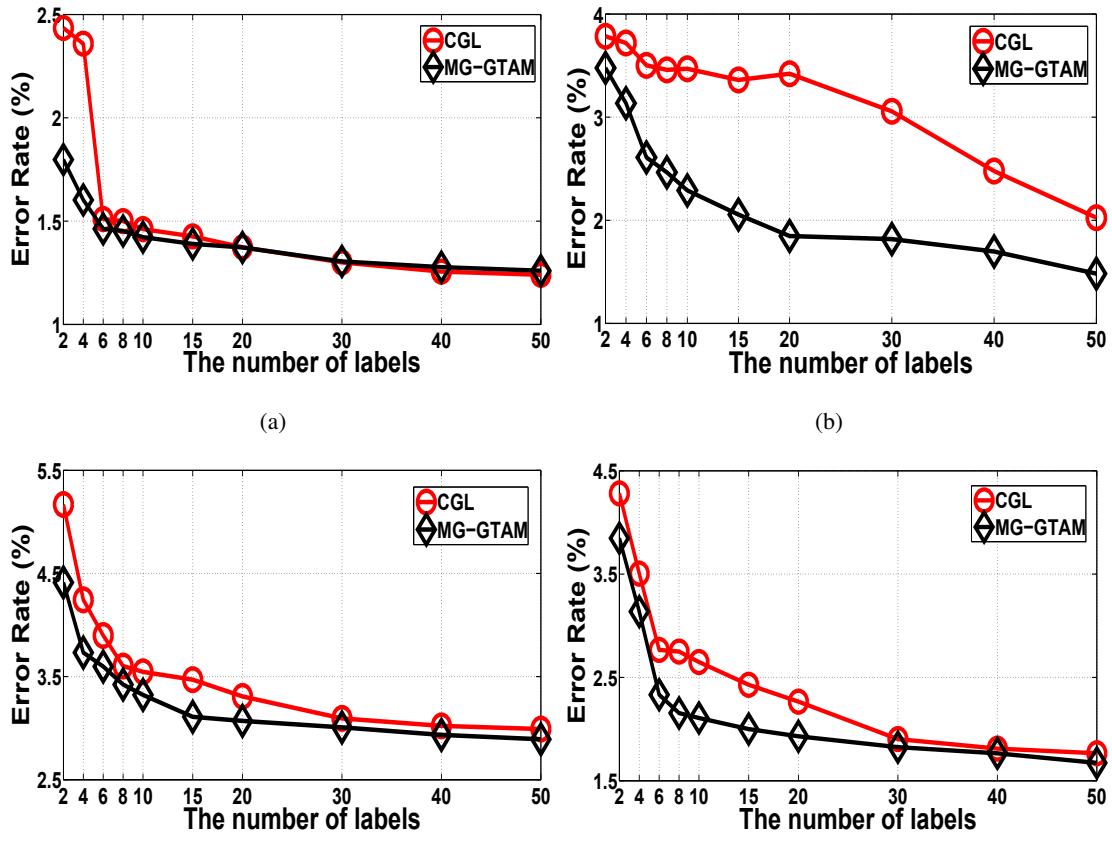


Figure 3.6: Performance comparison of the *CGL* [3] method and our *MG-GTAM* method on the experiments on USPS digit recognition. The figures show the error rates when using different numbers of labeled samples for classifying digits: a) 1 vs 7; b) 2 vs. 3; c) 2 vs. 7 d) 4 vs. 7.

discriminate ambiguous digit pairs, such as 1 vs. 7, 2 vs. 3, 3 vs. 7, and 4 vs. 7,. For each test, the dataset consisted of 200 images for each digit and the number of labeled points randomly varied between 2 and 50. To generate different graphs, we first varied the value of k from 1 to 5 when constructing kNN graphs. Then all three types of edge weighting schemes (binary weighting, fixed Gaussian kernel weighting, and adaptive Gaussian kernel weighting) were applied to compute the weight matrix, resulting in a total of 15 graphs. We evaluated the proposed *MG-GTAM* method as well as the prior work of combining graph Laplacians (*CGL*) [3]. Figure 3.6 shows the average error rates over 100 random trials for each classification task. The results demonstrate that the proposed *MG-GTAM* method clearly outperforms the *CGL* method. Particularly, when the number of labels

is small, such as 2 and 4, the performance gain provided by *multi-graph GTAM* is fairly significant. This property indicates that our multi-graph approach is suitable for image search applications, in which usually only very few queries are provided. In addition, our multi-graph transduction method is more efficient than *CGL* - it takes only about 25% of the time it takes to run *CGL* to complete the label prediction process.

3.7 Summary and Discussion

The effectiveness of existing graph-based *SSL* methods heavily depends on the availability of accurate labels and appropriately constructed graphs. However, the performance often significantly degrades if the labels are not distributed evenly across classes, if the initial label locations are biased, or if excessive noise samples or outliers corrupt the underlying manifold structure. These degenerate settings seem to plague real world problems as well, compromising the performance of state-of-the-art *SSL* algorithms. In addition, there remain many heuristic choices in the graph design process which usually requires empirical trials when applying these methods to specific domains. These shortcomings have been confirmed in our experiments over synthesis data sets and real data sets, as described in this chapter.

We propose a novel graph-based semi-supervised learning method, bivariate graph transduction method, to address these problems. Our main contributions include:

1. We extended the existing quadratic regularization framework based on a single variable to a new one including both the label matrix and the classification function as variables. Such extension allows us to treat input labels as part of the optimization targets and thereby address the label sensitivity problem.

2. We demonstrated that the bivariate formulation is actually a mixed integer programming problem, which can be reduced to a pure binary integer programming (BIP) problem. An alternative optimizer, named *Graph Transduction via Alternating Minimization* (*GTAM*), was designed to achieve a local optimal solution with high efficiency for real applications.

3. We further proved that the proposed bivariate formulation is equivalent to a Max-Cut problem for two-class cases, and a Maximum K-cut problem for multi-class cases. In addition,

we proposed an efficient solution with $\mathcal{O}(n^2)$ complexity by using a greedy gradient search to achieve approximation of the above Max-Cut problem. This greedy gradient Max-Cut solution can be interpreted as the optimization procedure of *GTAM* in the content of the equivalent graph-cut formulation.

4. Based on the bivariate framework, we further developed a graph-based label tuning strategy, called *Label Diagnosis through Self Tuning*, to handle errors in the initial labeled set.

5. Finally, we developed a multi-graph based transductive learning approach, called *multi-graph GTAM*, and showed its superiority in both prediction accuracy and computational efficiency, compared with prior work.

Future work includes extensions of the proposed methods to handle out-of-sample cases such that new data points can be added without requiring a full retraining procedure. Another interesting extension of the bivariate framework is to incorporate active learning to achieve efficient manual labeling and accurate prediction.

Chapter 4

Semi-Supervised Hashing for Large Scale Visual Search

4.1 Introduction

In previous chapters, we have shown the promising results using graph-based semi-supervised learning techniques. However, graph-based approaches suffer from the critical bottleneck in the scalability of graph construction and label propagation procedure. For instance, the widely used neighborhood graph relies on nearest neighbor search, which leads to quadratic complexity for the graph construction process. In addition, most large-scale visual search problems involve handling high-dimensional visual descriptors, thereby causing another challenge in excessive storage requirement. Thus, existing graph-based approaches are prohibitive for large-scale applications like image and video retrieval due to both computational and storage cost.

In reality, visual data, both image and video, is growing rapidly over the Internet. For example, the photo sharing website, *Flickr*, has over 5 billion images. Another sharing website, *YouTube*, receives more than 24 hours of uploaded videos per minute. There is an emerging need of effective tools for retrieving relevant content from such massive databases. Besides the widely used text-based commercial search engines, like *Google* and *Bing*, content based image retrieval (CBIR) has attracted substantial attention over the past decade [38]. Instead of taking textual key words as input, CBIR techniques directly take a visual query q and try to return its nearest neighbors from a given database of images or videos $\mathbf{X}$ using certain features and distance measure.

Searching nearest neighbors is a fundamental step in many machine learning algorithms [132]. Exhaustively comparing the query q with each sample in the database $\mathbf{X}$ is infeasible because linear complexity $\mathcal{O}(|\mathbf{X}|)$ is not scalable in practical settings. Besides the scalability issue, most large-scale CBIR applications also suffer from the *curse of dimensionality* [16] since visual descriptors usually have hundreds, or even thousands, of dimensions. Therefore, besides the infeasibility of exhaustive search, storage of the original data also becomes a critical bottleneck.

Fortunately, in many applications, it is sufficient to return approximate nearest neighbors. Instead of an exact nearest neighbor search through linear scan, a fast and accurate indexing method with sublinear ($o(|\mathcal{X}|)$), logarithmic ($\mathcal{O}(\log|\mathcal{X}|)$) or even constant ($\mathcal{O}(1)$) query time is desired for approximate nearest neighbors (ANN) search. For example, ϵ-ANN aims to find $p \in \mathcal{X}$ satisfying $d(p,q) \leq (1+\epsilon)d(p',q)$, where ϵ satisfies $\epsilon > 0$ and p' is the nearest neighbor of query q [71]. Over the past several decades, many techniques have been developed for fast and efficient ANN search. Especially, tree-based approaches store data with efficient data structures, which makes the search operation extremely fast, typically with the complexity of $\mathcal{O}(\log(|\mathcal{X}|))$. The representative tree-based algorithms include KD tree [17] [54] [135], ball tree [115], metric tree [150], and vantage point tree [169]. A detailed survey of the tree-based ANN search algorithms can be found in [110]. However, the performance drops significantly for high-dimensional data, and the efficiency is degraded to the worst level, identical to an exhaustive search [70].

Recently, hashing based ANN techniques have attracted much attention. They have constant query time and need substantially reduced storage as only compact binary codes are stored. In this work, we focus on compact binary hash codes $\mathbf{Y} \in \mathbb{B}^{K\times n}$ of original feature data $\mathbf{X} \in \mathbb{R}^{D\times n}$ ($K \ll D$). To generate a K-bit hash code $\mathbf{Y}$, K binary hash functions are used. The linear projection-based hash function family has been widely used since it is very efficient. Also, it has achieved state-of-the-art performance for various tasks [56] [165] [120]. In linear projection-based hashing, the k^{th} hash function is of the following form:

$$h_k(\mathbf{x}) = \operatorname{sgn}\left(f(\mathbf{w}_k^\top \mathbf{x} + b_k)\right), \tag{4.1}$$

where $\mathbf{x}$ is a data point, $\mathbf{w}_k$ is a projection vector, b_k is a threshold and $f(\cdot)$ is a function to be determined. Since $h(\mathbf{x}) \in \{-1, 1\}$, the corresponding binary hash bit can be simply expressed as: $y_k(\mathbf{x}) = (1 + h_k(\mathbf{x}))/2$.

Different choices of $\mathbf{w}$ and $f(\cdot)$ lead to different hashing approaches. Roughly, hashing methods can be divided into two main categories: unsupervised methods and supervised methods. Unsupervised methods design the hash function using random samples or the unlabeled data $\mathbf{X}$. Locality Sensitive Hashing (*LSH*) [56] is arguably one of the most popular unsupervised hashing methods and has been applied to many applications, including information retrieval and computer vision. Its kernelized version has recently been developed in [90]. Another representative unsupervised method called Spectral Hashing (*SH*) was proposed recently in [165]. Since unsupervised methods do not require any labeled data, they can be easily applied to different data domains given a pre-specified distance metric.

Note that hashing-based *ANN* methods aim to return an approximate set of nearest neighbors. However, in typical CBIR, even returning the exact nearest neighbors does not guarantee search quality. This is due to the well-known problem called *semantic gap*, where the high level semantic description of visual content often differs from the low level visual descriptors [141]. Furthermore, most hashing techniques provide theoretic guarantees only when certain distance metrics are used. For instance, the *LSH* function family works for the $\ell_p(p \in [0, 2])$, Mahalanobis, and Jaccard distances. But there exist other popular distance measures, like χ^2 distance, which have been shown to be effective for histogram style descriptors, such as the popular Bag-of-Visual-Words (BoW) representation [96][172]. In summary, due to the lack of appropriate semantic visual descriptors and associated similarity measure, existing unsupervised hashing-based ANN search does not yield satisfactory results.

For the specific application of image search, the similarity (or distance) between image pairs sometimes is not defined with a simple metric. Ideally, one would like to provide pairs of images that one believes contain 'similar' or 'dissimilar' images. From such pairwise labeled data, the hashing mechanism should be able to generate codes that preserve the semantic consistency, i.e. semantically similar images should have similar hash codes. Supervised learning techniques have been explored to handle this issue. For example, in [91], authors have suggested merging standard *LSH* with a learned Mahalanobis metric to achieve semantic-level indexing. Since this approach uses labeled sample pairs for training distance metric, it was categorized as a semi-supervised learning paradigm. However, the hash functions still use a random hyperplane, same as in the standard *LSH*.

Method	Hash Function	Projection	Hamming Distance	Learning Paradigm
LSH [56]	$\mathrm{sgn}(\mathbf{w}^\top \mathbf{x} + b)$	data-independent	non-weighted	unsupervised
SIKH [120]	$\mathrm{sgn}(cos(\mathbf{w}^\top \mathbf{x} + b) + t)$	data-independent	non-weighted	unsupervised
SH [165]	$\mathrm{sgn}(cos(k\mathbf{w}^\top \mathbf{x}))$	data-dependent	non-weighted	unsupervised
BSSC [131]	–	data-dependent	weighted	supervised
RBMs [65]	–	–	non-weighted	supervised
BRE [89]	$\mathrm{sgn}(\mathbf{w}^\top \mathbf{k}_\mathbf{x})$	data-dependent	non-weighted	supervised
LAMP [109]	$\mathrm{sgn}(\mathbf{w}^\top \mathbf{x} + b)$	data-dependent	non-weighted	supervised
SSH [158]	$\mathrm{sgn}(\mathbf{w}^\top \mathbf{x} + b)$	data-dependent	non-weighted	semi-supervised

Table 4.1: The conceptual comparison of the proposed SSH method with other binary encoding methods.

In addition, another supervised hashing technique, called Boosted Similarity Sensitive Coding (*BSSC*), was proposed in [131] which tries to learn a series of weighted hash functions from labeled data. Kulis and Darrell recently proposed learning hash functions by explicitly minimizing the reconstruction error between the metric space and hamming space, termed as Binary Reconstructive Embedding (*BRE*) [89]. Other binary encoding methods, like deep neural network stacked with Restricted Boltzmann Machines (*RBMs*), were recently applied to learn binary codes [65], and have shown superior performance over *BoostSSC* given sufficient training labels [148]. Although *RBMs* use both labeled and unlabeled data, the latter is only used in a pre-training phase to provide a good initialization for the supervised back-propagation phase. So *RBMs* are still categorized as a supervised method. One of the problems with all of these supervised methods is that they are much slower in comparison with the unsupervised methods. Another problem for supervised methods stems from limited or noisy training data, which can easily lead to overfitting.

In this chapter, we propose a *Semi-Supervised Hashing* (*SSH*) framework that can leverage semantic similarity using labeled data while preventing overfitting by incorporating unlabeled data. The objective function of *SSH* consists of two main components: supervised empirical fitness and unsupervised information theoretic regularization. Specifically, we provide a rigorous formula-

tion in which a supervised term aims to minimize the empirical error on the labeled data while an unsupervised term provides effective regularization by ensuring desirable properties like variance and independence of individual bits. After relaxation and imposing orthogonality constraints, one can derive uncorrelated hash codes. Finally, we present a sequential learning technique to learn bit-dependent hash codes with the capability of correcting errors made by previous hash bits. In Table 4.1, we compare the proposed methods with other popular ones by using a taxonomy based on important properties of the methods.

The remainder of this chapter is organized as follows. In Section 4.2, we briefly describe several popular hashing methods. Section 4.3 presents the formulation of our proposed approach, i.e. semi-supervised hashing (*SSH*). In Section 4.4, we present three different solutions for designing semi-supervised hash functions, followed by an extension of designing unsupervised sequential hash functions in Section 4.5. Section 4.6 provides extensive experimental validation over real datasets. The conclusions and future work are given in Section 4.7.

4.2 Related Work on Hashing

In this section, we review a few important hashing methods, such as *spectral hashing*, *boosted similarity sensitive coding*, *binary reconstructive embedding* based hashing, and deep belief network along with pros and cons when using these methods in practical image retrieval applications.

4.2.1 Locality Sensitive Hashing

A key ingredient of *Locality Sensitive Hashing* (*LSH*) is mapping "similar" samples to the same bucket with a high probability [56]. In other words, the locality in the original space is preserved in the hamming space. More precisely, the hash functions $h(\cdot)$ from *LSH* family satisfy the following elegant locality preserving property:

$$P\left\{h(\mathbf{x}) = h(\mathbf{y})\right\} = sim(\mathbf{x}, \mathbf{y}), \tag{4.2}$$

where the similarity is usually associated with a the distance measure. For example, RBF kernel is often used to compute similarity from ℓ_2 distance as

$$sim(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{\sigma^2}\right). \tag{4.3}$$

A popular class of *LSH* functions involve random projections and thresholds as:

$$h(x) = sign(\mathbf{w}^\top \mathbf{x} + b), \tag{4.4}$$

where $\mathbf{w}$ is a random hyperplane and b is a random intercept. Here, the random vector $\mathbf{w}$ is data-independent, and is usually constructed by sampling each component of $\mathbf{w}$ randomly from a p-stable distribution corresponding to a general ℓ_p metric, where $p \in (0, 2]$, e.g., standard Gaussian for ℓ_2 distance [37].

Due to the asymptotic theoretical guarantee, many *LSH*-based large-scale search systems have been developed. For instance, a self-tuning indexing technique, called *LSH* forest was proposed in [12] to improve the performance without additional storage and query overhead. However, the practical efficiency of *LSH* is still very limited since it usually requires long codes to achieve high precisions at the cost of greatly reduced recalls. A practical solution for this problem is to use multiple tables to increase recall [56]. For instance, constructing a total of l K-bit length hash tables can map the original data point $\mathbf{x}$ to a Hamming embedding representation $H(x) = [h_1(x), \cdots, h_K(x)]$. The corresponding Hamming distance between points $\mathbf{x}_i$ and $\mathbf{x}_j$ can be efficiently computed as

$$d_{\mathcal{H}} = \sum_{k=1}^{K} \left| h_k(\mathbf{x}_i) - h_k(\mathbf{x}_j) \right|. \tag{4.5}$$

Accordingly, *LSH* using l tables with K-bit codes provides the following collision probability:

$$P\left\{H(\mathbf{x}) = H(\mathbf{y})\right\} \propto l \cdot \left(1 - \frac{cos^{-1}\mathbf{x}^\top \mathbf{y}}{\pi}\right)^K. \tag{4.6}$$

For a large-scale application, the value of K should be considerably large to reduce false collisions (i.e., the number of non-neighbor sample pairs falling into the same hash bucket). However a large value of K decreases the collision probability between similar samples as well. In order to overcome this drawback, a compromise strategy using multiple hash tables is usally adopted in practice. Obviously, this is inefficient due to the extra storage cost and query time. In [101], a technique called MultiProbe *LSH* was developed to reduce the number of required hash tables through intelligently probing multiple buckets within each hash table. Although such ideas are shown to be useful, data-independent random projection based methods suffer from the lack of effective discrimination over data. Therefore, recent research of hash functions tends to leverage data-driven machine learning techniques to improve the performance of hashing [26].

In addition, incorporating kernel learning further generalizes *LSH* by extending the standard metric space to a broad class of kernel-based similarity functions [90][109]. Metric learning has been incorporated into randomized *LSH* functions using pairwise similarity and dissimilarity constraints between some examples [91]. However, all these methods still use random projections to design hash functions, which usually has limited performance with short codes.

4.2.2 Boosted Similarity Sensitive Hashing

To improve discrimination among hash codes, boosted similarity sensitive coding (*BSSC*) was developed in [131] to learn a weighted hamming embedding for task specific similarity search, as shown below

$$H : \mathcal{X} \rightarrow \{\alpha_1 h_1(\mathbf{x}), \cdots, \alpha_K h_K(\mathbf{x})\}. \tag{4.7}$$

Therein, the conventional hamming distance $d_{\mathcal{H}}$ is replaced by the weighted version as

$$d_{\mathcal{WH}} = \sum_{k=1}^{K} \alpha_k |h_k(\mathbf{x}_i) - h_k(\mathbf{x}_j)| n \tag{4.8}$$

Through learning the optimal weights $\{\alpha_1, \cdots, \alpha_k\}$, the objective is to lower the collision probability of non-neighbor pairs, while improving the collision probability of neighborhood pairs. If we treat each hash function as a decision stump, a straightforward approach to learn the weights is to directly use the adaptive boosting algorithm [53], as described in [131].

4.2.3 Spectral Hashing

Spectral Hashing (*SH*) was recently proposed to design compact binary codes for ANN search by exploring data-driven machine learning techniques. Besides the desired property of locality preservation, *SH* also aims at designing balanced and uncorrelated binary codes. Balanced codes ensure an even size of hash buckets, while uncorrelated codes help reduce redundancy. The hash

functions of *SH* satisfy the following criteria [165]:

$$\begin{aligned} \min \sum_{ij} sim(\mathbf{x}_i, \mathbf{x}_j) \|H(\mathbf{x}_i) - H(\mathbf{x}_j)\|^2 \\ \text{subject to:} \quad h_k(\mathbf{x}_i) \in \{-1, 1\} \\ \sum_i h_k(\mathbf{x}_i) = 0, k = 1, \cdots, K \\ \sum_i h_k(\mathbf{x}_i) h_l(\mathbf{x}_i) = 0, \text{for } \ k \neq l. \end{aligned} \tag{4.9}$$

The direct solution for the above optimization is non-trivial even for a single bit since it is a balanced graph partition problem, which is NP hard. The combination of K-bit balanced partitioning is even harder because of the pairwise independence constraints. After relaxing the constraints, the above optimization was solved using spectral graph analysis [51]. Especially, with the assumption of uniform data distribution, the spectral solution can be efficiently computed using 1D-Laplacian eigenfunctions [165].

The final *SH* algorithm consists of three key steps: 1) extraction of maximum variance directions through Principal Component Analysis (PCA) over the data; 2) direction selection, which prefers projections with large spread and small spatial frequency; 3) partition of projected data by a sinusoidal function. *SH* has been shown to be very effective in encoding large-scale, low-dimensional data since the dominant PCA directions are selected multiple times to create binary bits. However, its performance degrades severely for high-dimensional cases. For high-dimensional problems ($D >> K$) where many directions contain enough variance, usually each PCA direction is picked only once. This is because the top few projections have similar range and thus, a low spatial frequency is preferred. In this case, *SH* can be simply considered as a PCA projection followed by a mean partition. In *SH*, the projection directions are data dependent but learned in an unsupervised manner; thus, it does not really take into account the information available in the labeled subset. Moreover, the assumption of uniform data distribution is usually not true for real-world data.

4.2.4 Deep Belief Networks

Learning deep belief networks (*DBN*) via *Restricted Boltzmann Machines* (*RBMs*) was recently proposed in [65][128] to obtain binary codes. Such learned networks are capable of capturing higher order correlations between different layers of the network. Through the multi-layer network

structure, the number of units in each layer is reduced and high-dimensional input can be projected to a much more compact binary vector space.

In practice, *RBMs* have two stages: unsupervised pre-training and supervised fine-tuning. The greedy pre-training phase is progressively executed layer by layer from input to output. After achieving convergence of the parameters of a layer via contrastive divergence, the derived activation probabilities are fixed and treated as input to drive the training of the next layer. During the fine-tuning stage, the labeled data is used to help refine the trained network through back-propagation. Specifically, a cost function is first defined to estimate the number of correctly classified points in the training set [60]. Then, the network weights are refined to maximize this objective function through gradient descent. *RBM*-based binary encoding involves estimating a large number of weights. For example, the *RBMs* structure used in [148] has four layers of size $512 - 512 - 256 - 32$ nodes requiring learning of a total of 663552 weights. This not only incurs an extremely costly training procedure, but also demands a large pool of training data for fine-tuning. For instance, in [148], 20 batches of 1000×1000 neighborhood matrices are used during the back-propagation stage, which is fairly large compared to the actual size of the dataset, LabelMe + Peekaboom (70000 images).

4.2.5 Binary Reconstruction Embedding

Instead of using data-independent random projections in *LSH* or principal components in *SH*, Kulis and Darrell proposed data-dependent and bit-correlated hash functions as [89]:

$$h_k(\mathbf{x}) = \operatorname{sgn}\left[\sum_{q=1}^{s} \mathbf{W}_{kq}\kappa(\mathbf{x}_{kq}, \mathbf{x})\right]. \tag{4.10}$$

The sample set $\{\mathbf{x}_{kq}\}, q = 1, \cdots, s$ is the training data for learning hash function h_k, and $\kappa(\cdot)$ is a kernel function. The weight matrix $\mathbf{W}$ is used to combine multiple kernel functions.

Based on the above formulation, a method called binary reconstruction embedding (*BRE*) was designed to minimize the difference between the metric space and reconstructed distance in hamming space. The Euclidean metric $d_{\mathcal{M}}$ and the binary reconstruction distance $d_{\mathcal{R}}$ are defined as:

$$d_{\mathcal{M}}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2 \tag{4.11}$$

$$d_{\mathcal{R}}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{K}\sum_{k}^{K}\left(h_k(\mathbf{x}_i) - h_k(\mathbf{x}_j)\right)^2. \tag{4.12}$$

The objective is to minimize the following reconstruction error to derive the optimal $\mathbf{W}$:

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{N}} \left[d_{\mathcal{M}}(\mathbf{x}_i,\mathbf{x}_j) - d_{\mathcal{R}}(\mathbf{x}_i,\mathbf{x}_j)\right]^2, \tag{4.13}$$

where $\mathcal{N}$ is the data pool consisting of the training sample pairs. Optimizing the above objective function is difficult due to the non-differentiability of the $\mathrm{sgn}(\cdot)$ function. Instead, a coordinate-descent algorithm was applied to iteratively update the hash functions to obtain a local optimum. This hashing method can be easily extended to supervised scenarios by setting neighbor pairs with zero distance and non-neighbor pairs with a large distance. However, since the binary reconstruction distance $d_{\mathcal{R}}$ is bounded in $[0, 1]$ while the metric distance $d_{\mathcal{M}}$ has no upper bound, the minimization problem in Eq. (4.13) is only meaningful when input data is appropriately normalized. In practice, the original data point $\mathbf{x}$ is often mapped to a hypersphere with unit length to produce bounded $d_{\mathcal{M}}$ as $0 \leq d_{\mathcal{M}} \leq 1$. This normalization removes the scale of data points, which is often not negligible for practical applications of nearest neighbor search.

4.3 Semi-Supervised Paradigm for Hashing

In this section, we present the formulation of our hashing method, *Semi-Supervised Hashing* (*SSH*). In this setting, one is given a set of n points, $\mathcal{X} = \{\mathbf{x}_i\}$, $i = 1 \ldots n$, $\mathbf{x}_i \in \mathbb{R}^D$, in which a subset of pairs is associated with two categories of label information, $\mathcal{M}$ and $\mathcal{C}$. Specifically, a pair $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}$ is denoted as a neighbor-pair when $(\mathbf{x}_i, \mathbf{x}_j)$ are neighbors in a metric space or share common class labels. Similarly, $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}$ is called a non-neighbor-pair if $\mathbf{x}_i$ and $\mathbf{x}_j$ are far away in the metric space or have different class labels. Let us denote the data matrix by $\mathbf{X} \in \mathbb{R}^{D\times n}$ where each column is a data point. Also, suppose there are l points $\mathbf{X}_l \in \mathbb{R}^{D\times l}$, $l \ll n$, which are associated with at least one of the categories $\mathcal{M}$ or $\mathcal{C}$. The goal of *SSH* is to learn hash functions that minimize the error on the labeled training data $\mathbf{X}_l$, while maximally satisfying the desirable properties, e.g., independence of bits and balanced partitioning. We discuss the formulation and solution of our method in the following subsections.

4.3.1 Empirical Fitness

SSH aims to map the data $\mathbf{X} \in \mathbb{R}^{D\times n}$ to a Hamming space to obtain its compact representation. Suppose we want to learn K hash functions leading to a K-bit Hamming embedding of $\mathbf{X}$. Without

loss of generality, let $\mathbf{X}$ be normalized to have zero mean. In this work, we use linear projection coupled with mean thresholding as a hash function. In other words, given a vector $\mathbf{w}_k \in \mathbb{R}^D$, the k^{th} hash function is defined as

$$h_k(\mathbf{x}_i) = \text{sgn}(\mathbf{w}_k^\top \mathbf{x}_i + b_k), \tag{4.14}$$

where b_k is the mean of the projected data, i.e.,

$$b_k = -\frac{1}{n}\sum_{j=1}^{n} \mathbf{w}_k^\top \mathbf{x}_j = 0, \tag{4.15}$$

since $\mathbf{X}$ is zero-mean. One can get the corresponding binary bit as

$$y_{ki} = \frac{1}{2}\left[1 + h_k(\mathbf{x}_i)\right] = \frac{1}{2}\left[1 + \text{sgn}(\mathbf{w}_k^\top \mathbf{x}_i)\right], \tag{4.16}$$

and $\mathbf{Y} = \{y_{ki}\} \in \mathbb{B}^{K\times n}$ represents the hash codes of all samples. Let $\mathbf{H} = [h_1, \ldots, h_K]$ be a sequence of K hash functions and $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_K] \in \mathbb{R}^{D\times K}$. We want to learn a $\mathbf{W}$ that gives the same bits for $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}$ and different bits for $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}$. We define the following objective function measuring the empirical accuracy on the labeled data for a family of hashing functions $\mathbf{H}$:

$$J(\mathbf{H}) = \sum_k \left[\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}} h_k(\mathbf{x}_i)h_k(\mathbf{x}_j) - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}} h_k(\mathbf{x}_i)h_k(\mathbf{x}_j) \right]. \tag{4.17}$$

One can express the above objective function in a compact matrix form by first defining a matrix $\mathbf{S} \in \mathbb{R}^{l\times l}$ incorporating the pairwise labeled information from $\mathbf{X}_l$ as:

$$S_{ij} = \begin{cases} 1 & : \quad (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M} \\ -1 & : \quad (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C} \\ 0 & : \quad otherwise. \end{cases} \tag{4.18}$$

Also, suppose $\mathbf{H}(\mathbf{X}_l) \in \mathbb{R}^{K\times l}$ maps the points in $\mathbf{X}_l$ to their K-bit hash codes. Then, the objective function $J(\mathbf{H})$ can be represented as

$$\begin{aligned} J(\mathbf{H}) &= \frac{1}{2}\mathbf{tr}\left(\mathbf{H}(\mathbf{X}_l)\ \mathbf{S}\ \mathbf{H}(\mathbf{X}_l)^\top\right) \\ &= \frac{1}{2}\mathbf{tr}\left(\text{sgn}(\mathbf{W}^\top\mathbf{X}_l)\ \mathbf{S}\ \text{sgn}(\mathbf{W}^\top\mathbf{X}_l)^\top\right), \end{aligned} \tag{4.19}$$

where $\text{sgn}(\mathbf{W}^\top\mathbf{X}_l)$ is the matrix of signs of individual elements. Since the above function measures only the empirical accuracy, it is prone to overfitting, especially when the size of the labeled set is

small compared to the entire dataset (i.e. $l \ll n$). To get better generalization ability, one needs to add regularization by incorporating some desirable conditions. These additional regularization terms use all the data $\mathbf{X}$, including both unlabeled and labeled data and thus belong to the semi-supervised learning category.

Motivated by spectral hashing [165], we would like to generate hash codes in which each bit maximizes the information by generating a balanced partition of the data. The balancing property specifies that each hash function $h_k(\cdot)$ should partition the dataset $\mathbf{X}$ into two sets with equal size. In summary, we intend to learn optimal hashing functions $\mathbf{H}$ by maximizing the objective function with the following constraints:

$$\begin{aligned} \mathbf{H}^* &= \arg\max_{\mathbf{H}} J(\mathbf{H}) \\ \text{subject to} \quad & \sum_{i=1}^{n} h_k(\mathbf{x}_i) = 0, k = 1, \ldots, K. \end{aligned} \tag{4.20}$$

Since the objective function $J(\mathbf{H})$ itself is non-differentiable, the above problem is difficult to solve even without considering any constraints. As an alternative, we first present a simple relaxation of the empirical fitness.

In the relaxed version of the objective function, we replace the sign of projection with its *signed magnitude* in (4.17). This relaxation is intuitive in the sense that it not only desires similar points to have the same sign but also large projection magnitudes, and it projects dissimilar points not only with different signs but also as far as possible. With this relaxation, the new objective can be directly written as a function of $\mathbf{W}$ as

$$J(\mathbf{W}) = \sum_k \left(\sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}} \mathbf{w}_k^\top \mathbf{x}_i \mathbf{x}_j^\top \mathbf{w}_k - \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}} \mathbf{w}_k^\top \mathbf{x}_i \mathbf{x}_j^\top \mathbf{w}_k \right). \tag{4.21}$$

Without loss of generality, we also assume $\|\mathbf{w}_k\| = 1, \ \forall k$. The above function can be expressed in a matrix form as

$$J(\mathbf{W}) = \frac{1}{2} \mathbf{tr} \left(\mathbf{W}^\top \mathbf{X}_l \mathbf{S} \mathbf{X}_l^\top \mathbf{W} \right). \tag{4.22}$$

4.3.2 Information Theoretic Regularization

Maximizing empirical accuracy for just a few pairs can lead to severe overfitting, as illustrated in Figure 4.1. Hence, we use a regularizer which utilizes both labeled and unlabeled data. From the

information-theoretic point of view, one would like to maximize the information provided by each bit [9]. Using the maximum entropy principle, a binary bit that gives balanced partitioning of $\mathbf{X}$ provides maximum information. Thus, it is desired to have $\sum_{i=1}^{n} h_k(\mathbf{x}_i) = 0$. However, finding mean-thresholded hash functions that meet the balancing requirement is hard. Instead, we use this property to construct a regularizer for the empirical accuracy given in (4.22). We now show that maximum entropy partitioning is equivalent to maximizing the variance of a bit.

Proposition 4.3.1. *[maximum variance condition] A hash function with maximum entropy $H(h_k(\mathbf{x}))$ must maximize the variance of the hash values, and vice-versa, i.e.,*

$$\max H(h_k(\mathbf{x})) \Longleftrightarrow \max \operatorname{var}[h(\mathbf{x})]. \tag{4.23}$$

Proof. Assume h_k has a probability p of assigning the hash value $h_k(\mathbf{x}) = 1$ to a data point and $1-p$ for $h_k(\mathbf{x}) = -1$. The entropy of $h_k(\mathbf{x})$ can be computed as

$$H(h_k(\mathbf{x})) = -p \log_2 p - (1-p) \log_2 (1-p). \tag{4.24}$$

It is easy to show that the maximum entropy is $\max H(h_k(\mathbf{x})) = \mathbf{1}$ when the partition is balanced, i.e., $p = 1/2$. Now we show that balanced partitioning implies maximum bit variance. The mean of hash value is $E[h(\mathbf{x})] = \mu = 2p - 1$ and the variance is:

$$\begin{aligned} &\operatorname{var}[h_k(\mathbf{x})] = E[(h_k(\mathbf{x}) - \mu)^2] \\ &= 4(1-p)^2 p + 4p^2(1-p) = 4p(1-p). \end{aligned} \tag{4.25}$$

Clearly, $\operatorname{var}[h(\mathbf{x})]$ is concave with respect to p and its maximum is reached at $p = 1/2$, i.e. balanced partitioning. Also, since $\operatorname{var}[h(\mathbf{x})]$ has a unique maximum, it is easy to see that the maximum variance partitioning also maximizes the entropy of the hash function. □

Using the above proposition, the regularizer term is defined as

$$R(\mathbf{W}) = \sum_k \operatorname{var}[h_k(\mathbf{x})] = \sum_k \operatorname{var}[\operatorname{sgn}(\mathbf{w}_k^\top \mathbf{x})]. \tag{4.26}$$

Maximizing the above function with respect to $\mathbf{W}$ is still hard due to its non-differentiability. To overcome this problem, we first show that the maximum variance of a hash function is lower-bounded by the scaled variance of the projected data.

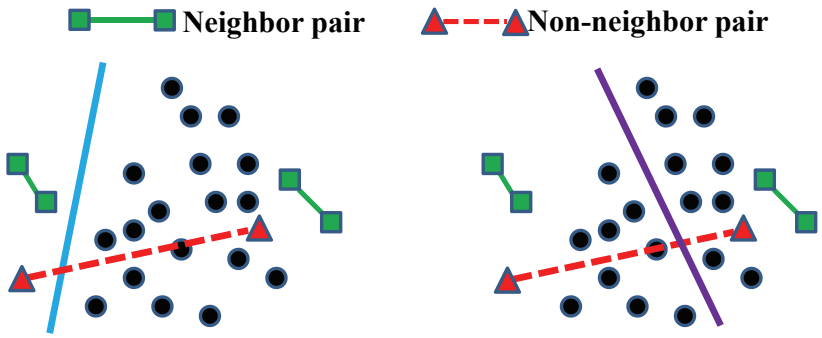


Figure 4.1: An illustration of partitioning with maximum empirical fitness and entropy. Both of the partitions satisfy the given pairwise labels, while the one on the right is more informative due to higher entropy.

Proposition 4.3.2. *[lower bound on maximum variance of a hash function] The maximum variance of a hash function is lower-bounded by the scaled variance of the projected data, i.e.,*

$$\max \operatorname{var}[h_k(\mathbf{x})] \geq \alpha \cdot \operatorname{var}[\mathbf{w}_k^\top \mathbf{x}]. \tag{4.27}$$

where α is a positive constant.

Proof. Suppose, $\|\mathbf{x}_i\|^2 \leq \beta \; \forall i, \beta > 0$. Since $\|\mathbf{w}_k\|^2 = 1 \; \forall k$, from Cauchy-Schwarz inequality,

$$\begin{aligned} &\|\mathbf{w}_k^\top \mathbf{x}\|^2 \leq \|\mathbf{w}_k\|^2 \cdot \|\mathbf{x}\|^2 \leq \beta = \beta \cdot \|\operatorname{sgn}(\mathbf{w}_k^\top \mathbf{x})\|^2 \\ &\Rightarrow E\left[\|\operatorname{sgn}(\mathbf{w}_k^\top \mathbf{x})\|^2\right] \geq \frac{1}{\beta} E\left[\|\mathbf{w}_k^\top \mathbf{x}\|^2\right] \\ &\Rightarrow \max \operatorname{var}[h_k(\mathbf{x})] \geq \frac{1}{\beta} \operatorname{var}[\mathbf{w}_k^\top \mathbf{x}]. \end{aligned} \tag{4.28}$$

Here, we have used the properties that the data is zero-centered, i.e., $E[\mathbf{w}_k^\top \mathbf{x}] = 0$, and for maximum bit variance $E[\operatorname{sgn}(\mathbf{w}_k^\top \mathbf{x})] = 0$. □

Given the above proposition, we use the lower bound on the maximum variance of a hash function as a regularizer, which is easy to optimize, i.e.,

$$\begin{aligned} R(\mathbf{W}) &= \frac{1}{\beta} \sum_k E[\|\mathbf{w}_k^\top \mathbf{x}\|^2] = \frac{1}{n\beta} \sum_k \mathbf{w}_k^\top \mathbf{X}\mathbf{X}^\top \mathbf{w}_k \\ &= \frac{1}{n\beta} \mathbf{tr}\left(\mathbf{W}^\top \mathbf{X}\mathbf{X}^\top \mathbf{W}\right). \end{aligned} \tag{4.29}$$

4.3.3 Final Objective Function

Combining the relaxed empirical fitness term from Eq. (4.22) and the relaxed regularization term from Eq. (4.29), the overall semi-supervised objective function is given as

$$\begin{aligned} J(\mathbf{W}) &= \frac{1}{2}\mathbf{tr}\left(\mathbf{W}^\top \mathbf{X}_l \mathbf{S} \mathbf{X}_l^\top \mathbf{W}\right) + \frac{\eta}{2}\mathbf{tr}\left(\mathbf{W}^\top \mathbf{X}\mathbf{X}^\top \mathbf{W}\right) \\ &= \frac{1}{2}\mathbf{tr}\left[\mathbf{W}^\top \left(\mathbf{X}_l \mathbf{S} \mathbf{X}_l^\top + \eta \mathbf{X}\mathbf{X}^\top\right)\mathbf{W}\right] \\ &= \frac{1}{2}\mathbf{tr}\left(\mathbf{W}^\top \mathbf{M} \mathbf{W}\right), \end{aligned} \tag{4.30}$$

where the constants n and β are absorbed in the coefficient η. Here the *adjusted* covariance matrix is represented as

$$\mathbf{M} = \mathbf{X}_l \mathbf{S} \mathbf{X}_l^\top + \eta \mathbf{X}\mathbf{X}^\top. \tag{4.31}$$

It is very interesting to note the form of $\mathbf{M}$, the unsupervised data variance part $\mathbf{X}\mathbf{X}^T$ adjusted by $\mathbf{X}_l \mathbf{S} \mathbf{X}_l^\top \mathbf{W}$ arising from the given partial pair-wise labeled data.

4.4 Semi-Supervise Projection Learning for Hashing

4.4.1 Orthogonal Projection Learning

While learning compact codes, one would like to avoid redundancy in bits as much as possible, in addition to having each bit be highly informative. One way to achieve this is by making the projection directions orthogonal. Combining this, the optimal projections are:

$$\begin{aligned} \mathbf{W}^* &= \arg\max_{\mathbf{W}} J(\mathbf{W}) \\ \text{subject to} \quad & \mathbf{W}^\top \mathbf{W} = \mathbf{I}. \end{aligned} \tag{4.32}$$

Now, the learning of optimal projections $\mathbf{W}$ becomes a typical eigenproblem, which can be easily solved by doing an eigenvalue decomposition on matrix $\mathbf{M}$:

$$\begin{aligned} \max_{\mathbf{W}} J(\mathbf{W}) &= \sum_{k=1}^{K} \lambda_k \\ \mathbf{W}^* &= [\mathbf{e}_1 \cdots \mathbf{e}_K], \end{aligned} \tag{4.33}$$

where $\lambda_1 > \lambda_2 > \cdots > \lambda_K$ are the top eigenvalues of $\mathbf{M}$ and $\mathbf{e}_k, k = 1, \cdots, K$ are the corresponding eigenvectors.

To summarize, the objective function in Eq. (4.30) consists of two components. The first (supervised) term is the empirical accuracy of the learned hash functions over the pairwise labeled data, while the second (unsupervised) term is a regularizer that prefers those directions that maximize the variance of the projections subject to orthogonality constraints. Mathematically, it is very similar to finding the maximum variance direction using PCA except that the original covariance matrix gets "adjusted" by another matrix arising from the labeled data. Hence, our framework provides an intuitive and easy way to learn hash functions in a semi-supervised paradigm.

4.4.2 Non-Orthogonal Projection Learning

In the previous section, we imposed orthogonality constraints on the projection directions in order to approximately decorrelate the hash bits. However, these orthogonality constraints sometimes lead to a practical problem. It is well known that for most real-world datasets, most of the variance is contained in the top few projections. The orthogonality constraints force one to progressively pick those directions that have very low variance, substantially reducing the quality of lower bits, and hence the whole embedding. We empirically verify this behavior in Section 4.6. Depending on the application, it may make sense to pick a direction that is not necessarily orthogonal to the previous directions but has higher variance as well as low empirical error on the labeled set. On the other hand, one doesn't want to pick a previous direction again since the fixed thresholds will generate the same hash codes in our case. Hence, instead of imposing hard orthogonality constraints, we convert them into a penalty term added to the objective function. This allows the learning algorithm to pick suitable directions by balancing various terms. With this, one can write the new objective function as

$$\begin{aligned} J(\mathbf{W}) &= \frac{1}{2}\mathbf{tr}(\mathbf{W}^\top \mathbf{M}\mathbf{W}) - \frac{\rho}{2}\|\mathbf{W}^\top\mathbf{W} - \mathbf{I}\|_{\mathcal{F}}^2 \\ &= \frac{1}{2}\mathbf{tr}(\mathbf{W}^\top \mathbf{M}\mathbf{W}) - \frac{\rho}{2}\mathbf{tr}\left[(\mathbf{W}^\top\mathbf{W} - \mathbf{I})^\top(\mathbf{W}^\top\mathbf{W} - \mathbf{I})\right]. \end{aligned} \tag{4.34}$$

The new formulation has a certain tolerance to non-orthogonality, which is modulated by a positive coefficient ρ. However, the above objective function is non-convex and there is no easy way to find the global solution unlike the previous case. To maximize with respect to $\mathbf{W}$, we set the derivative to zero and absorb all constants into ρ as

$$\frac{\partial J(\mathbf{W})}{\partial \mathbf{W}} = 0 \Rightarrow (\mathbf{W}\mathbf{W}^\top - \mathbf{I} - \frac{1}{\rho}\mathbf{M})\mathbf{W} = 0. \tag{4.35}$$

Though the above equation admits an unlimited number of solutions since $\mathbf{W}$ has a non-empty nullspace, we can obtain a solution by ensuring

$$\mathbf{W}\mathbf{W}^\top\mathbf{W} = \left(\mathbf{I} + \frac{1}{\rho}\mathbf{M}\right)\mathbf{W}. \tag{4.36}$$

One can get a simple solution for the above condition if $\mathbf{I} + \frac{1}{\rho}\mathbf{M}$ is positive definite. From (4.30), $\mathbf{M}$ is symmetric but not necessarily positive definite. Let $\mathbf{Q} = \mathbf{I} + \frac{1}{\rho}\mathbf{M}$. Clearly, $\mathbf{Q}$ is also symmetric. In the following proposition we show that $\mathbf{Q}$ is positive definite if the coefficient ρ is chosen appropriately.

Proposition 4.4.1. *The matrix* $\mathbf{Q}$ *is positive definite if* $\rho > \max(0, -\bar{\lambda}_{min})$*, where* $\bar{\lambda}_{min}$ *is the smallest eigenvalue of* $\mathbf{M}$*.*

Proof. By definition in (4.34), $\rho > 0$. Since $\mathbf{M}$ is symmetric, it can be represented as $\mathbf{M} = \mathbf{U}diag(\lambda_1, \cdots, \lambda_D)\mathbf{U}^\top$ where all λ_i's are real. Let $\bar{\lambda}_{min} = \min(\lambda_1, \cdots, \lambda_D)$. Then $\mathbf{Q}$ can be written as

$$\begin{aligned}\mathbf{Q} &= \mathbf{I} + \mathbf{U}diag\left(\frac{\lambda_1}{\rho}, \cdots, \frac{\lambda_D}{\rho}\right)\mathbf{U}^\top \\ &= \mathbf{U}diag\left(\frac{\lambda_1}{\rho} + 1, \cdots, \frac{\lambda_D}{\rho} + 1\right)\mathbf{U}^\top. \end{aligned} \tag{4.37}$$

Clearly, $\mathbf{Q}$ will have all eigenvalues positive if $\frac{\lambda_{min}}{\rho} + 1 > 0 \Rightarrow \rho > -\lambda_{min}$. □

If $\mathbf{Q}$ is positive definite, it can be decomposed as $\mathbf{Q} = \mathbf{L}\mathbf{L}^\top$ using Cholesky decomposition. Then, one can easily verify that $\mathbf{W} = \mathbf{L}\mathbf{U}$ satisfies Eq. (4.36). To achieve a meaningful *approximate* solution to our problem, we truncate the computed matrix $\mathbf{W}$ by selecting its first k columns. The final non-orthogonal projections are derived as

$$\mathbf{W}_{\text{nonorth}} = \mathbf{L}\mathbf{U}_k, \tag{4.38}$$

where $\mathbf{U}_k$ are the top k eigenvectors of $\mathbf{M}$.

4.4.3 Sequential Projection Learning

The above non-orthogonal solution is achieved via single-shot adjustment over the orthogonal ones. However, the objective function in Eq. (4.34) is ill posed and the approximate solution is somewhat

ad-hoc since it is sensitive to the choice of the penalty coefficient ρ. Moreover, such single-shot solutions do not have the sequential error correcting property where each hash function tries to correct the errors made by the previous one. Hence, we propose an alternative solution to learn a sequence of projections, which implicitly incorporates bit correlation through iteratively updating the pair-wise label matrix.

The idea of sequential projection learning is quite intuitive. The hash functions are learned iteratively such that at each iteration, the pairwise label matrix $\mathbf{S}$ in (4.18) is updated by imposing higher weights on point pairs violated by the previous hash function. This sequential process implicitly creates dependency between bits and progressively minimizes empirical error. The sign of $\mathbf{S}_{ij}$, representing the logical relationship in a point pair $(\mathbf{x}_i, \mathbf{x}_j)$, remains unchanged in the entire process and only its magnitude $|\mathbf{S}_{ij}|$ is updated. Chart 4.1 describes the algorithm of the proposed semi-supervised sequential projection learning method.

Here, $\tilde{\mathbf{S}}^k \in \mathbb{R}^{l\times l}$ measures the *signed magnitude* of pairwise relationships of the k^{th} projections of $\mathbf{X}_l$:

$$\tilde{\mathbf{S}}^k = \mathbf{X}_l^\top \mathbf{w}_k \mathbf{w}_k^\top \mathbf{X}_l. \tag{4.39}$$

Mathematically, $\tilde{\mathbf{S}}^k$ is simply the derivative of empirical accuracy of the k^{th} hash function, i.e., $\tilde{\mathbf{S}}^k = \nabla_{\mathbf{S}} J_k$, where $J_k = \mathbf{w}_k^\top \mathbf{X}_l \mathbf{S} \mathbf{X}_l^\top \mathbf{w}_k$. The function $\mathrm{T}(\cdot)$ implies the truncated gradient of J_k:

$$\mathrm{T}(\tilde{\mathbf{S}}_{ij}^k, \mathbf{S}_{ij}) = \begin{cases} \tilde{\mathbf{S}}_{ij}^k & : \quad \mathrm{sgn}(\mathbf{S}_{ij} \cdot \tilde{\mathbf{S}}_{ij}^k) < 0 \\ 0 & : \quad \mathrm{sgn}(\mathbf{S}_{ij} \cdot \tilde{\mathbf{S}}_{ij}^k) \geq 0. \end{cases} \tag{4.40}$$

The condition $\mathrm{sgn}(\mathbf{S}_{ij} \cdot \tilde{\mathbf{S}}_{ij}^k) < 0$ for a labeled pair $(\mathbf{x}_i, \mathbf{x}_j)$ indicates that hash bits $h_k(\mathbf{x}_i)$ and $h_k(\mathbf{x}_j)$ contradict the given pairwise label. In other words, points in a neighbor pair $(\mathbf{x}_i, \mathbf{x}_i) \in \mathcal{M}$ are assigned different bits or those in $(\mathbf{x}_i, \mathbf{x}_i) \in \mathcal{C}$ are assigned the same bit. For each such violation, $\mathbf{S}_{ij}$ is updated as $\mathbf{S}_{ij} = \mathbf{S}_{ij} - \alpha \tilde{\mathbf{S}}_{ij}^k$. The step size α is chosen such that $\alpha \leq \frac{1}{\beta}$ where $\beta = \max_i \|\mathbf{x}_i\|^2$, ensuring $|\alpha \tilde{\mathbf{S}}_{ij}^k| \leq 1$. This leads to numerically stable updates without changing the sign of $\mathbf{S}_{ij}$. Those pairs for which the current hash function produces the correct bits, i.e., $\mathrm{sgn}(\mathbf{S}_{ij} \cdot \tilde{\mathbf{S}}_{ij}^k) > 0$, $\mathbf{S}_{ij}$ are kept unchanged by setting $\mathrm{T}(\tilde{\mathbf{S}}_{ij}^k, \mathbf{S}_{ij}) = 0$. Thus, those labeled pairs for which the current hash function does not predict the bits correctly exert more influence on the learning of the next function, biasing the new projection to produce correct bits for such pairs. Intuitively, it has a flavor of boosting-based methods commonly used for classification.

Algorithm 4.1 Sequential projection learning for hashing (*SPLH*)

Input: data $\mathbf{X}$, pairwise labeled data $\mathbf{X}_l$, initial pairwise labels $\mathbf{S}_1$, length of hash codes K, constant α

for $k = 1$ **to** K **do**

Compute adjusted covariance matrix:

$$\mathbf{M}_k = \mathbf{X}_l \mathbf{S}_k \mathbf{X}_l^\top + \eta \mathbf{X}\mathbf{X}^\top$$

Extract the first eigenvector $\mathbf{e}$ of $\mathbf{M}_k$ and set:

$$\mathbf{w}_k = \mathbf{e}$$

Update the labels from vector $\mathbf{w}_k$:

$$\mathbf{S}_{k+1} = \mathbf{S}_k - \alpha \mathrm{T}\left(\tilde{\mathbf{S}}^k, \mathbf{S}_k\right)$$

Compute the residual:

$$\mathbf{X} = \mathbf{X} - \mathbf{w}_k \mathbf{w}_k^\top \mathbf{X}$$

end for

After extracting a projection direction using $\mathbf{M}_k$, the contribution of the subspace spanned by that direction is removed from $\mathbf{X}$ to minimize the redundancy in bits. Note that this is not the same as imposing the orthogonality constraints on $\mathbf{W}$ discussed earlier. Since the supervised term $\mathbf{X}_l \mathbf{S}_k \mathbf{X}_l^\top$ still contains information potentially from the whole space spanned by the original $\mathbf{X}$, the new direction may still have a component in the subspace spanned by the previous directions. Thus, the proposed formulation automatically decides the level of desired correlations between successive hash functions. If empirical accuracy is not affected, it prefers to pick uncorrelated projections. Thus, unlike the single-shot solution discussed earlier, the proposed sequential method aggregates various desirable properties in a single formulation leading to superior performance on real-world tasks as shown in Section 4.6. In fact, one can extend this sequential learning method in unsupervised cases as well, as shown in the next subsection.

4.5 Unsupervised Sequential Learning

Unlike the semi-supervised case, pairwise labels are not available in the unsupervised case. To apply the general framework of sequential projection learning to an unsupervised setting, we propose the idea of generating pseudo labels at each iteration of learning. In fact, while generating a bit via a

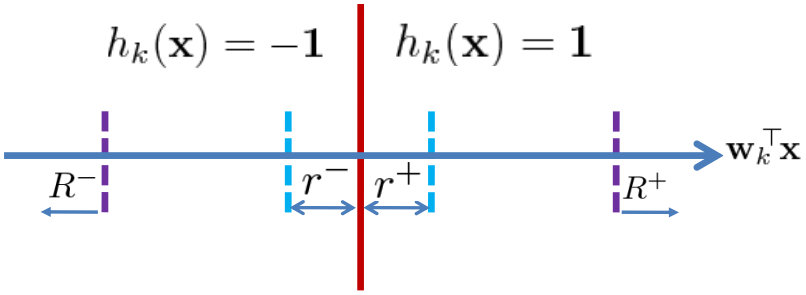


Figure 4.2: Potential errors due to thresholding (red line) of the projected data to generate a bit. Points in r^- and r^+, are assigned different bits even though they are quite close. Also, points in R^- (R^+) and r^- (r^+) are assigned the same bit even though they are quite far.

binary hash function, there are two types of boundary errors one encounters due to thresholding of the projected data. Suppose all the data points are projected on a one-dimensional axis as shown in Figure 4.2, and the red vertical line is the partition boundary, i.e. $\mathbf{w}_k^\top \mathbf{x} = 0$. The points left of the boundary are assigned a hash value $h_k(\mathbf{x}) = -1$ and those of the right are assigned a value $h_k(\mathbf{x}) = 1$. The regions marked as r^-, r^+ are located very close to the boundary and regions R^-, R^+ are located far from it. Due to thresholding, points in the pair $(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{x}_i \in r^-$ and $\mathbf{x}_j \in r^+$, are assigned different hash bits even though their projections are quite close. On the other hand, points in pair $(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{x}_i \in r^-$ and $\mathbf{x}_j \in R^-$ or $\mathbf{x}_i \in r^+$ and $\mathbf{x}_i \in R^+$, are assigned the same hash bit even though their projected values are quite far apart. To correct these two types of boundary "errors", we first introduce a neighbor-pair set $\mathcal{M}$ and a non-neighbor-pair set $\mathcal{C}$:

$$\mathcal{M} = \{(x_i, x_j)\} : h(x_i) \cdot h(x_j) = -1, |\mathbf{w}^\top (\mathbf{x}_i - \mathbf{x}_j)| \leq \epsilon \tag{4.41}$$

$$\mathcal{C} = \{(x_i, x_j)\} : h(x_i) \cdot h(x_j) = 1, |\mathbf{w}^\top (\mathbf{x}_i - \mathbf{x}_j)| \geq \zeta. \tag{4.42}$$

Then, given the current hash function, a desired number of point pairs are sampled from both $\mathcal{M}$ and $\mathcal{C}$. Suppose $\mathbf{X}_{\mathcal{MC}}$ contains all the points that are part of at least one sampled pair. Using the labeled pairs and $\mathbf{X}_{\mathcal{MC}}$, a pairwise label matrix $\mathbf{S}^k_{\mathcal{MC}}$ is constructed similar to Equation (4.18). In other words, for a pair of samples $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}$, a pseudo label $\mathbf{S}^k_{\mathcal{MC}} = 1$ is assigned while for those $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}$, $\mathbf{S}^k_{\mathcal{MC}} = -1$ is assigned. In the next iteration, these pseudo labels force the point pair in $\mathcal{M}$ to be assigned the same hash values and those in $\mathcal{C}$ different ones. Thus, it

Algorithm 4.2 Unsupervised sequential projection learning for hashing (*USPLH*)

Input: data $\mathbf{X}$, length of hashing codes K

Initialize $\mathbf{X}^0_{\mathcal{MC}} = \emptyset, \mathbf{S}^0_{\mathcal{MC}} = \mathbf{0}$.

for $k = 1$ **to** K **do**

Compute adjusted covariance matrix:

$$\mathbf{M}_k = \sum_{i=0}^{k-1} \lambda^{k-i} \mathbf{X}^i_{\mathcal{MC}} \mathbf{S}^i_{\mathcal{MC}} {\mathbf{X}^i_{\mathcal{MC}}}^\top + \eta \mathbf{X}\mathbf{X}^\top$$

Extract the first eigenvector $\mathbf{e}$ of $\mathbf{M}_k$ and set:

$$\mathbf{w}_k = \mathbf{e}$$

Generate pseudo labels from projection $\mathbf{w}_k$:

Sample $\mathbf{X}^k_{\mathcal{MC}}$ and construct $\mathbf{S}^k_{\mathcal{MC}}$

Compute the residual:

$$\mathbf{X} = \mathbf{X} - \mathbf{w}_k \mathbf{w}_k^\top \mathbf{X}$$

end for

sequentially tries to correct the potential errors made by the previous hash functions.

Note that each hash function $h_k(\cdot)$ produces a pseudo label set $\mathbf{X}^k_{\mathcal{MC}}$ and the corresponding label matrix $\mathbf{S}^k_{\mathcal{MC}}$. The new label information is used to adjust the data covariance matrix in each iteration of sequential learning, similar to that for the semi-supervised case. However, the unsupervised setting does not have a boosting-like update of the label matrix unlike the semi-supervised case. Each iteration results in its own pseudo label matrix depending on the hash function. Hence, to learn a new projection, all the pairwise label matrices since the beginning are used but their contribution is decayed exponentially by a factor λ at each iteration. Note that one does not need to store these matrices explicitly since incremental updates can be done at each iteration, resulting in the same memory and time complexity as for the semi-supervised case. The detailed learning procedure is described in algorithm chart (4.2). Since there exist no pseudo labels at the beginning, the first vector $\mathbf{w}_1$ is just the first principal direction of the data. Then, each hash function is learned to satisfy the pseudo labels iteratively by adjusting the data covariance matrix, similar to *SPLH* approach.

In summary, besides the three different versions of semi-supervised hashing methods, i.e., the orthogonal solution *SSH*$_{orth}$, the non-orthogonal solution *SSH*$_{nonorth}$, and the sequential solution *SPLH*, we further provided an unsupervised extension of the sequential learning method, named as *USPLH*. Table 4.2 conceptually summarizes the four proposed hashing methods.

Method	Projection Correlation	Learning Paradigm	Learning Style
SSH_{orth}	orthogonal	semi-supervised	single-short
$SSH_{nonorth}$	non-orthogonal	semi-supervised	single-short
SPLH	non-orthogonal	semi-supervised	sequential
USPL	non-orthogonal	unsupervised	sequential

Table 4.2: The conceptual summary of the proposed hashing methods.

4.6 Experiments

We evaluated all the three versions of the proposed semi-supervised hashing methods, including orthogonal solution SSH_{orth}, non-orthogonal solution $SSH_{nonorth}$ (with orthogonality constraint relaxed), and sequential solution *SPLH*, as well as the unsupervised extension (*USPLH*) on several benchmark datasets, and compared with other popular binary coding methods, including locality sensitive hashing (*LSH*), spectral hashing (*SH*), binary reconstructive embedding (*BRE*) method, boosted similarity sensitive coding (BSSC), and shift invariant kernel based hashing (*SIKH*). These methods cover both unsupervised and supervised categories. Especially, previous work has shown that *SH* performs better than other binary encoding methods [165] [158], such as Restricted Boltzmann Machines (*RBMs*) [128]. For both *SH* and *BRE*, we used the best setting reported in previous literature. For *LSH*, we randomly select projections from a Gaussian distribution with zero-mean and identity covariance and apply random partitioning to construct hashing functions. In addition, for all the supervised and semi-supervised methods, a very small portion of labeled samples were used during training. For example, only 1000 labeled images are used in the experiments on the CIFAR10 dataset and 2000 on the *Flickr* image data.

In the following subsections, we will first discuss our evaluation protocols, followed by brief descriptions of the benchmark datasets. Finally, the extensive experimental results and comparison study is presented.

4.6.1 Evaluation Protocols

To perform a reasonable and practical evaluation, we adopt two criteria commonly used in the literature:

1. **Hamming ranking**: All the points in the database are ranked according to their Hamming distance from the query and the desired neighbors are returned from the top of the ranked list. The complexity of Hamming ranking is linear even though it is very fast in practice.

2. **Hash lookup**: A lookup table is constructed using the database codes, and all the points in the buckets that fall within a small Hamming radius r of the query are returned. The complexity of the hash lookups is constant time.

Note that evaluations based on *Hamming ranking* and *hash lookup* focus on different characteristics of hashing techniques. For instance, *hash lookup* emphasizes more on the practical search speed. However, when using many hash bits and a single hash table, the Hamming space becomes increasingly sparse, and very few samples fall within the Hamming radius r ($r = 2$ in our experiments), resulting in many failed queries. In this situation, *Hamming ranking* provides better quality measurement of the Hamming embedding, while neglecting the issue of search speed.

All the experiments were conducted using a *single* hash table with relatively compact codes (up to 64 bits for the largest image collection dataset with around 80 million points). The search results are evaluated based on whether the returned images and the query sample share the same semantic labels for supervised and semi-supervised tests. We use several metrics to measure the quantitative performance of different methods. For Hamming ranking based evaluation, we compute the precision of the top ranked M neighbors, where M is uniformly set as 500 in the experiments. Finally, similar to [165], a Hamming radius of 2 is used to retrieve the neighbors in the case of hash lookup. The precision of the returned samples falling within Hamming radius 2 is reported. If a query returns no neighbors inside Hamming ball with radius 2, it is treated as a failed query with zero precision.

4.6.2 Datasets

We used three image datasets in our experiments, i.e., CIFAR-10, a *Flickr* image collection, and the 80 million tiny images [147], with the number of samples ranging from tens of thousands to hundred

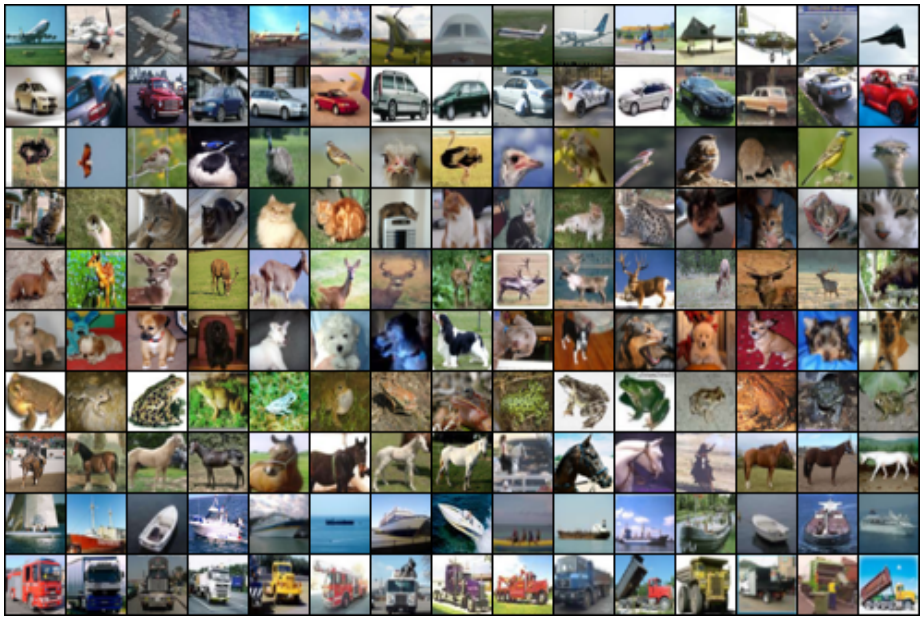


Figure 4.3: Some example images from the CIFAR10 dataset. From top row to bottom row, the image classes are *airplane*, *automobile*, *bird*, *cat*, *deer*, *dog*, *frog*, *horse*, *ship*, and *truck*.

of thousands to millions. In addition, we use 1 million image SIFT features for the experiments of unsupervised hashing methods. For the first two datasets, since they are fully annotated, we focus on the quantitative evaluation of the search accuracy. For the 80 million image data, we demonstrate the scalability of the proposed methods and provide qualitative evaluation by providing the search results of some exemplar queries.

CIFAR-10 dataset

The CIFAR-10 dataset contains a labeled subset of the 80-million tiny images collection [147]. It consists of a total of 60000 32×32 color images in 10 classes, each of which has 6000 samples [7]. Each sample in this dataset is associated with a mutually exclusive class label. Examples of the images in the CIFAR10 dataset are shown in Figure 4.3. Since this dataset is fully annotated, the ground truth semantic neighbors can be easily retrieved based on the class labels. The entire dataset is partitioned into two parts: a training set with 59000 samples and a test set with 1000

[7] http://www.cs.toronto.edu/ kriz/cifar.html

samples. The training set is used for learning hash functions and constructing the hash look-up tables. For *BSSC*, *BRE* and the proposed semi-supervised hashing methods, we additionally sample 1000 random points from the training set and assign semantic nearest neighbor information (i.e. construct the pairwise label matrix $\mathbf{S}$) based on the image labels. The binary encoding is performed in the 384-dim GIST feature space [114].

Flickr Images

Here we use a set of *Flickr* consumer images collected by NUS lab, i.e. NUS-WIDE dataset [33]. It was created as a benchmark for evaluating multimedia search techniques. This dataset contains around 270000 images associated with 81 ground truth concept tags. Unlike the CIFAR10 dataset, each sample in NUS-WIDE could be assigned with multiple labels, since this kind of multiple tagging occurs very often in real annotation scenarios. The Bag-of-Visual-Word model with local SIFT features is used for extracting the image descriptor [100]. Particularly, a visual vocabulary with 500-length code book and a soft assignment strategy was used for deriving the image features, as described in [77].

Similar to the CIFAR10 dataset, this dataset is partitioned into two parts, 1K for query test and around 269K for training and constructing hash tables. In addition, $2K$ images are randomly sampled with labels for *BSSC*, *BRE* and our proposed semi-supervised hashing methods. The precision is evaluated based on whether the returned images and the query share *at least one* common semantic label. The performance was evaluated with different code lengths varying from 8-bit to 64-bit.

SIFT-1M Dataset

We also test the performance of our unsupervised sequential learning method (*USPLH*) using SIFT-1M dataset. It contains 1 million local SIFT descriptors [100] extracted from random images. Each point in the dataset is a 128-dim vector representing histograms of gradient orientations. We use 1 million samples for training and an additional $10K$ for testing. Euclidean distance is used to determine the nearest neighbors. Following the criterion used in [165] [158], a returned point is considered a true neighbor if it lies in the top 2 percentile points closest to a query. Since no labels are available in this experiment, both SSH_{orth} and $SSH_{nonorth}$ have no adjustment term. Because it results in the same hash functions by using just principal projections, we named it as PCA-based hashing (*PCAH*). We also compare it with a few unsupervised hashing techniques, including *LSH*,

SH, *SIKH* on this dataset. For *USPLH*, to learn each hash function sequentially, we select 2000 samples from each of the four boundary and margin regions r^-, r^+, R^-, R^+. A label matrix $\mathbf{S}$ is constructed by assigning pseudo-labels to pairs generated from the samples.

80 **Million Tiny Images**

Besides the quantitative evaluation on the above two datasets, we also apply our techniques on a large collection of images with gist feature, i.e., 80 million tiny images dataset [147], which has been used as a benchmark dataset for designing binary encoding approaches [89] [49] [165] [148]. However, only a small portion of the dataset is manually labeled and the associated meta information is fairly noisy. Since CIFAR10 is a fully annotated subset of this gigantic image collection, we combine these two datasets in our experiments. The experimental protocol for quantitative evaluation is described as below. A subset of two million data points is sampled to construct the training set, especially for computing the data covariance matrix for all the eigen-decomposition based approaches, and a separate set of 2K samples from the CIFAR10 dataset is used as labeled samples. For *SH*, the hash functions were designed using this two-million dataset. For *BRE* and the proposed semi-supervised hashing methods, both the two-million dataset and 2K labeled data were used to learn the hash functions. After obtaining the hash functions, the Hamming embedding of the *entire* dataset with a total of $79,302,017$ samples is computed with 64-bit hash codes. Finally, some examples are randomly selected for query test, and qualitative comparison is made with other methods.

4.6.3 Results

For quantitative evaluation on the CIFAR10 and *Flickr* datasets, the number of bits is varied from 8 to 48 for the CIFAR10 dataset, and from 8 to 64 for the *Flickr* image dataset. The performance curves of both Hamming ranking and hash lookup are shown in Figure 4.4 and 4.5, respectively.

From these figures, it is not very surprising to see that the single table based *LSH* provides the worst performance since the random hash functions lack discrimination for small bits. The orthogonal solution for *SSH* described in Section 4.4, i.e. SSH_{orth}, has comparable performance to the non-orthogonal solution, $SSH_{nonorth}$, for small numbers of bits (i.e. 8, 12, and 16 bits) since there is enough variance in the top few orthogonal directions computed in our semi-supervised formulation. But when using large numbers of bits, SSH_{orth} performs much worse since the orthogonal solution

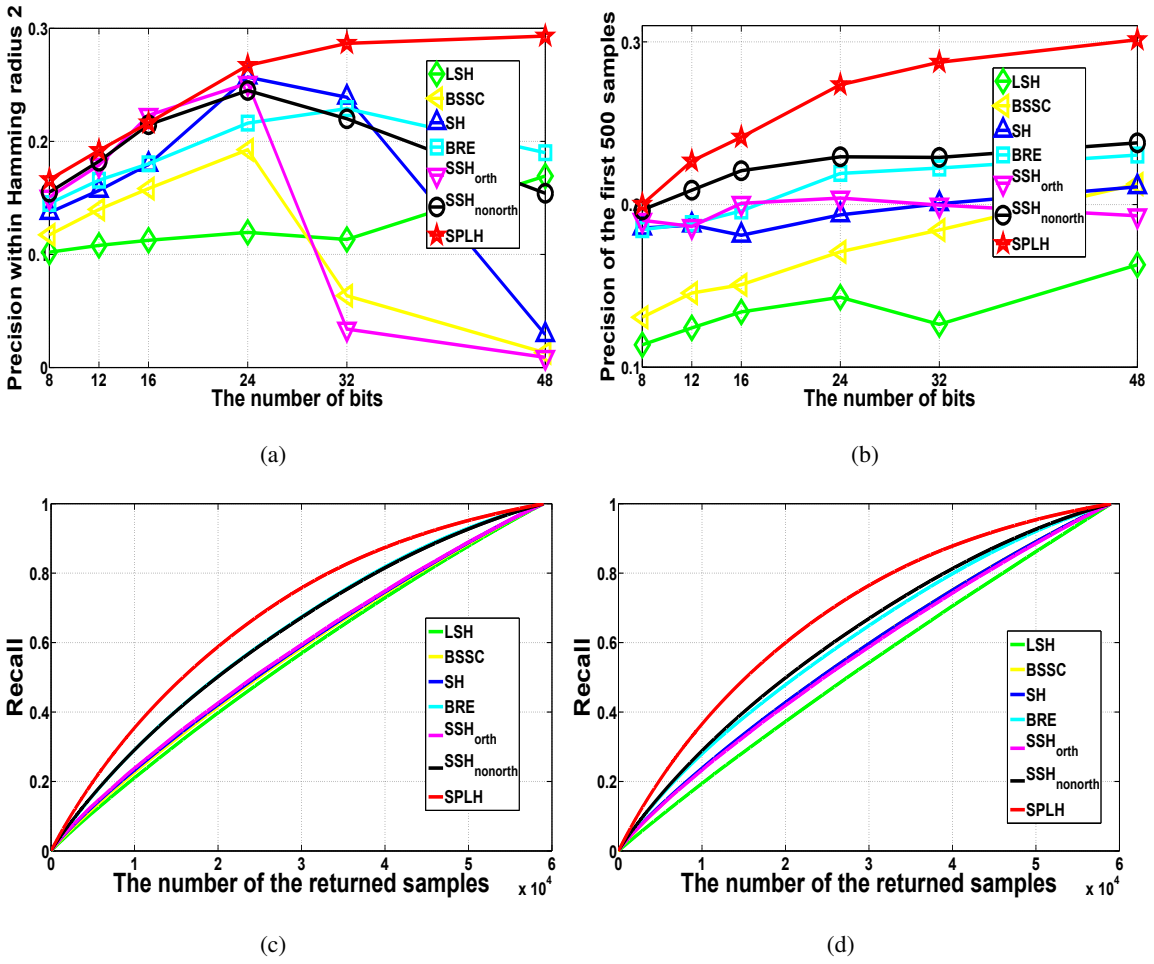


Figure 4.4: Results on the CIFAR10 dataset. a) Precision within Hamming radius 2 using hash lookup; b) Precision of the top 500 returned samples using Hamming ranking; c) Recall curves with 24 bits; d) Recall curves with 32 bits;

forces one to progressively pick the low variance projections, substantially reducing the quality of the whole embedding. Both Figures 4.4 and 4.5 clearly show that $SSH_{nonorth}$ is significantly better than SSH_{orth} when using long hash codes. Note that although *SH* also uses principal projection directions, it can somewhat alleviate the negative impact of low variance projections by reusing the large variance projections with higher frequency sinusoidal binarization. The sequential method of *SSH*, i.e., *SPLH*, provides the best performance for all bits. Particularly, in the evaluation of hash lookup within Hamming radius 2 (Figures 4.4(a) and 4.5(a)), the precision for most of the compared methods drops significantly when longer codes are used. This is because, for longer codes, the num-

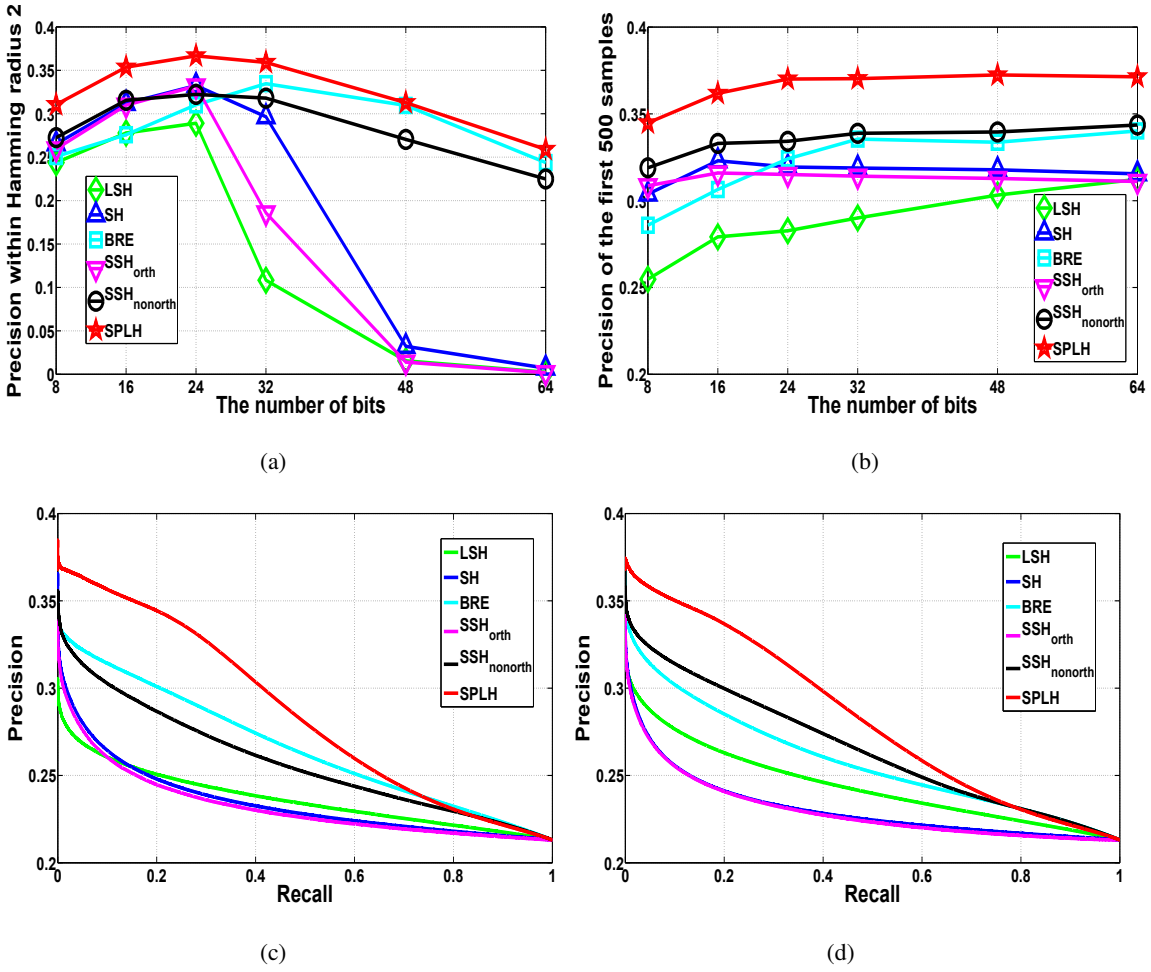


Figure 4.5: Results on the *Flickr* image dataset. a) Precision within Hamming radius 2 using hash lookup; b) Precision of the top 500 returned samples using Hamming ranking; c) Precision-Recall curve with 32 bits; d) Precision-Recall curve with 64 bits.

ber of points falling in a bucket decrease exponentially. Thus, many queries fail by not returning any neighbor even in a Hamming ball of radius 2. This shows a practical problem with hash lookup tables even though they have a faster query response than Hamming ranking. Even in this case, *SPLH* provides the best performance for most of the cases. Also, the drop in precision for longer codes is much less compared to others, indicating less failed queries for *SPLH*. Besides, the recall curves of the CIFAR10 tests are given in Figures 4.4(c) 4.4(d), and the precision-recall curves of the *Flickr* tests are given in Figures 4.5(c) 4.5(d). The results demonstrate a significant improvement of the proposed semi-supervised hashing approaches, especially *SPLH*, compared with other methods.

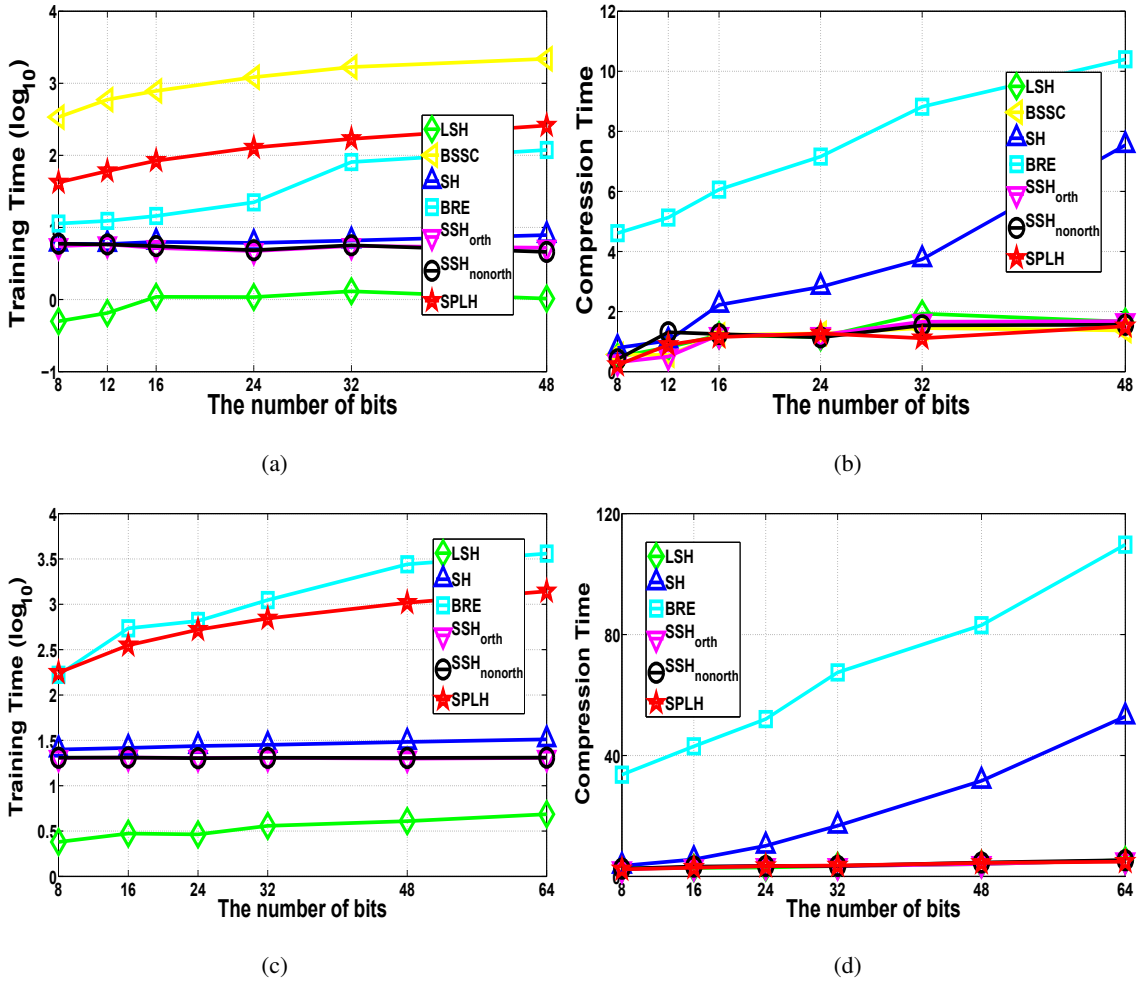


Figure 4.6: Computational cost for different binary encoding methods. a) Training cost on the CIFAR10 dataset; b) Compression cost on the CIFAR10 dataset; a) Training cost on the *Flickr* dataset; b) Compression cost on the *Flickr* dataset.

Figure 4.6 provides the comparison of the computational cost, including training time and compression time, for different techniques. The training time indicates the cost of learning the hash functions from training data and the compression time measures the encoding time from the original test data to binary codes. It is not surprising that *LSH* needs negligible training time since the projections are randomly generated, instead of being learned. The three eigenvalue decomposition based techniques, i.e. *SH*, SSH_{orth}, and $SSH_{nonorth}$, requires similar training cost. Since *SPLH* needs to update its pair-wise label matrix and performs eigenvalue decomposition at each iteration, its training time is longer but comparable with *BRE*. Compared with off line training cost, the compression time is usually more important for realistic tasks since it is done in real time. As shown in Figures 4.6(b) and 4.6(d), *BRE* is the most expensive method in terms of computing the binary codes. *SH* method requires a little more time than the remaining methods due to the calculation of the sinusoidal function. The code generation time can be ranked as: $BRE \gg SH > LSH \simeq SSH_{orth} \simeq SSH_{nonorth} \simeq SPLH$.

Figure 4.7 shows the experimental results of the unsupervised tests on the SIFT-1M dataset. Methods that learn data-dependent projections, i.e., *USPLH*, *SH* and *PCAH*, perform generally much better than *LSH* and *SIKH*. *SH* performs better than *PCAH* for longer codes since, for this dataset, *SH* tends to pick the high-variance directions again. *USPLH* yields the best precision for all bits. Particularly, Figure 4.7(a) shows precision curves for different methods using a hash lookup table, and Figure 4.7(b) shows the precision curves using hamming ranking. *USPLH* gives the best performance for most cases. Also, the performance of *USPLH* does not drop as rapidly as SH and *PCAH* with an increase in bits. Thus, *USPLH* leads to less query failures in comparison to other methods. Figures 4.7(c) and 4.7(d) show the recall curves for different methods using 24-bit and 48-bit codes. Higher precision and recall for *USPLH* indicate the advantage of learning hash functions sequentially even with noisy pseudo labels.

Finally, we present the experimental results on the large 80-million image data set to show the scalability of the proposed semi-supervised hashing methods. We design the hash tables with 64-bit length to index the 384-dimension gist descriptor, which dramatically reduces the storage of the entire dataset from hundreds of gigabytes to a few hundred megabytes. A random set of queries was sampled from the database and used for tests. Here we compared the visual search results of the three best performing methods, i.e., *BRE*, $SSH_{nonorth}$, and *SPLH*. After obtaining the search results in hash lookup (within Hamming radius $r = 2$), we computed the Euclidian distance of the

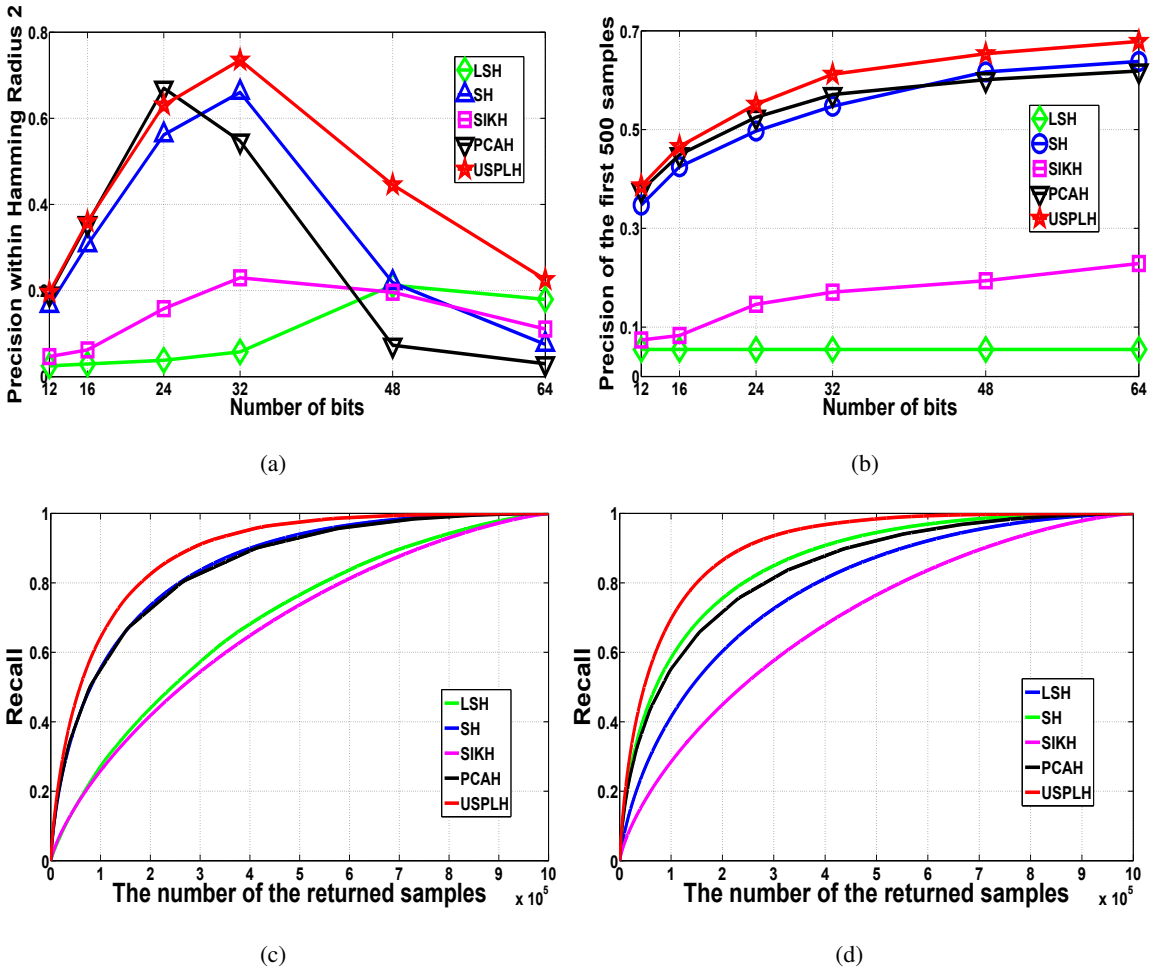
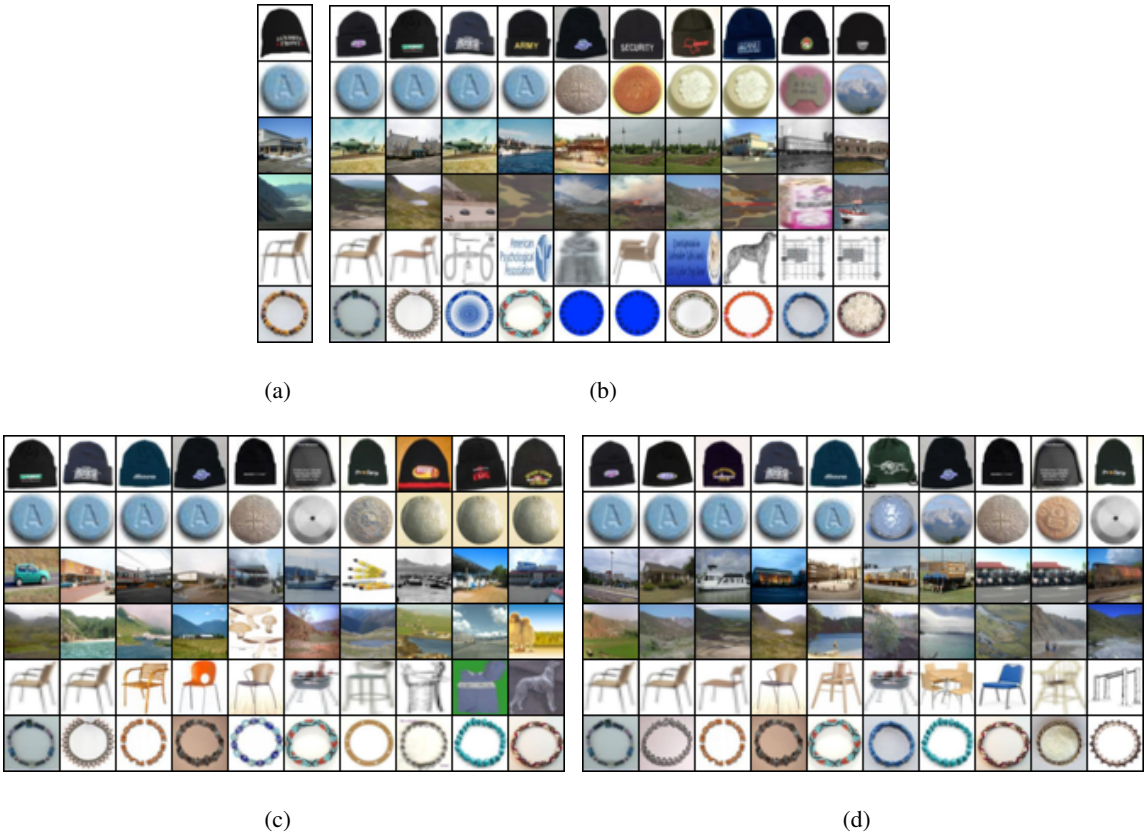


Figure 4.7: Results on the SIFT-1M dataset. The horizontal axis indicates the number of bits, and the vertical axis represents: a) precision of the top 500 returned samples; b) precision within Hamming radius 2. Recall curves (c) with 24 bits, and (d) with 48 bits.

collected nearest neighbors and query images in Gist feature space and then sorted the results. The top ten returned images for some exemplar queries are shown in Figure 4.8. *SPLH* presents more visually consistent search results than *BRE* and SSH_{orth}. This exhaustive search usually involves only around $0.005\% \sim 0.01\%$ of the entire 80 million samples, which indicates that the search is scalable for real time applications.



(a) (b)

(c) (d)

Figure 4.8: Qualitative evaluation over the 80 million image set though visualizing the search results with different hashing methods. a) query images; top 10 returned images using b) *BRE* method; c) $SSH_{nonorth}$ method, and d) *SPLH* method. All these hashing approaches use 64-bit binary coding.

4.7 Summary and Discussion

In this chapter, we have proposed a semi-supervised paradigm to learn efficient hash codes by simple linear mapping which can handle semantic similarity/dissimilarity among the data points. It tries to maximize empirical accuracy over the labeled data while regularizing the solution using an information-theoretic term. Specifically, the proposed method combines empirical loss over the pair-wise labeled data with other desirable properties, e.g., maximum entropy of the hash bits over both labeled and unlabeled data points. We proved that the maximum entropy condition equals the maximum variance of hash bits, which can be furthered relaxed. Therefore, the proposed method leads to a very simple eigen-decomposition based solution which is extremely efficient. Based on this framework, we proposed the following ways to obtain:

1. **Orthogonal hash functions** (*SSH*$_{orth}$): By adding orthogonality as hard constraints, the hash codes can be directly obtained by conducting eigen-decomposition over an adjusted covariance matrix.

2. **Non-orthogonal hash functions** (*SSH*$_{nonorth}$): The orthogonality constraints can be removed and imposed as a penalty term in the objective function. Then, an *approximate* non-orthogonal projections can be derived through a single-shot adjustment over the previous orthogonal solution.

3. **Sequential hash functions** (*SPLH*): More specifically, the sequential hash function is learned iteratively such that in each iteration the new bit tends to minimize the errors made by the previous bit. For this, the pairwise label is updated by imposing higher weights on point pairs violated by the previous hash function.

4. **Unsupervised sequential hash functions** (*USPLH*): Further, the sequential learning method was extended for the unsupervised setting, where a set of pseudo-labels are generated sequentially using the probable mistakes made by the previous bit. Each new hash function tries to minimize these errors subject to the same regularizer as in the semi-supervised case.

We conducted extensive experiments on four large datasets containing up to 80 million points and provided both quantitative and qualitative comparison with the state-of-the-art hashing techniques. The experimental results show superior performance of the proposed semi-supervised hashing methods. Particularly, *SPLH* achieved the best performance for semi-supervised and supervised cases and *USPLH* performs the best for unsupervised cases. The sequential techniques, i.e. *SPLH* and *USPLH*, need more time for training than *LSH* or the eigen-decomposition based methods, *SH*, *SSH*$_{orth}$, and *SSH*$_{nonorth}$, but comparable or even faster than the *BRE* method, and must faster than *BSSC*. In terms of binary code generation time, all the four proposed hashing methods are as fast as *LSH*, which is significantly faster than *SH* and *BRE* methods. In the future, we would like to investigate if the semi-supervised hashing methods can provide theoretical guarantees on the performance of approximate nearest neighbor search.

Chapter 5

Application: Visual Reranking and Pattern Discovery

In this chapter, we present a few visual search systems for real world applications to validate and demonstrate the power of the semi-supervised learning methods described in previous chapters. In Section 5.1, we will present a general image annotation and search framework using the developed graph transduction method. Section 5.2 describes a specific application of the proposed method for microscopic image analysis. Specifically, we show how to keep the expert in the loop and minimize the user labeling cost for efficient cellular phenotype identification. In Section 5.3, we apply the *LDST* algorithm to improve web image search by re-ranking initial search results based on keywords.

5.1 Columbia TAG System - Transductive Annotation by Graph

The Columbia TAG (Transductive Annotation by Graph) system is deigned to facilitate rapid retrieval and exploration of large image and video collections. It incorporates novel graph-based label propagation methods and intuitive graphic user interfaces (GUI) that allow users quickly browse and annotate a small number of images/videos, and then in real or near real time receive refined labels for all remaining unlabeled data in the collection. Using such refined labels, additional positive images/videos matching user's interest can be quickly discovered. It can be used as a fast search system alone, or a bootstrapping system for developing additional target recognition tools needed

in critical image application domains such as intelligence, surveillance, consumer, biomedical, and Web.

TAG system differs from the traditional approaches that are based on automatic image classification. These methods usually require a sufficiently large number of labeled samples to train classifiers - a method referred to as supervised learning. Instead, TAG minimizes the burden of manual labeling on users. The objective is to leverage the best use of whatever user input available (as few as one or two samples per class) and propagate such information in the most effective way to all the remaining data in the database. Specifically, we use our bivariate graph transductive learning methods developed in Chapter 3 to address several challenging issues, such as imbalanced, noisy, and biased labels, and achieve promising performance in several application domains, including bimolecular images, web documents, and satellite images.

TAG system is different from prior works using semi-supervised learning, which utilize both labeled and unlabeled data in learning the classifier or inferring labels of new data. Most semi-supervised techniques focus on separation of labeled samples of different classes while taking into account distribution of the unlabeled data. The performance of such methods often suffers from the scarcity of labeled data, invalid assumptions about classification models or distributions, and sensitivity to non-ideal label conditions. To overcome these issues, we adopt the bivariate graph transduction paradigm that makes least assumptions about the data and labels. One central principle of this paradigm is that data share and propagate their labels with other data in their proximity, defined in the context of a graph. Data are represented as vertices in a graph and the structure and edges in the graph define the relation among data. Propagation of labels among data in a graph is intuitive, flexible, and effective, without requiring complex models for the classifier and data distributions. Moreover, our graph inference method improved the existing graph learning approach in terms of the sensitivity to weak labels, graph construction, and noisy situations, as described in Chapter 3.

Starting with the small number of labels given by users, our graph-based transductive learning method propagates the initial labels to the remaining data and predicts the most likely labels (or scores) for each vertex in the graph. The propagation process is optimized with respect to several criteria. How well do the predictions fit the already known labels given by the user? What's the regularity of the predictions over data in the graph? Is the propagation process amicable to addition

of new labels? Are the results sensitive to quality of the initial labels and specific ways the labeled data are selected?

The output of TAG system consists of refined or predicted labels of images in the collection. They can be used to identify additional positive samples matching targets of interest, which in turn can be used to train more robust classifiers, arrange the best presentation order for image browsing, or rearrange image presentations for EEG-based image visualization.

The TAG system can be used in different modes - interactive and automatic. The interactive mode is designed for applications in which a user uses the GUI to interact with the system in browsing, labeling, and providing feedback. The automatic mode takes the initial labels or scores produced by other processes and then output refined scores or labels for all the data in the collection. The processes providing the initial labels may come from various sources, such as other classifiers using different modalities, models, or features, rank information of the data from other search engines, or even other manual annotation tools. When dealing with labels/scores from imperfect sources (e.g., search engines), special care is needed to filter the initial labels and assess their reliability before using them as input for propagation.

5.1.1 Comparison with Prior Work

There have been prior works exploring use of user feedback in improving the image retrieval experience. In [124], relevance feedback provided by the user is used to indicate which images in the returned results are relevant or irrelevant to the search target user has in mind. Such feedback can be indicated explicitly (by marking labels of relevance or irrelevance) or implicitly (by tracking specific images viewed by the user). Given such feedback information, the initial query (either in the form of keywords or example images) can be modified. Alternatively, the underlying features and distance metrics used in representing and matching images can be refined using relevance feedback information [117]. Though such ideas are intuitive and easy to implement, applications in practical domains have not shown effective results. There is no guarantee that the refined query, feature, or metric will improve the capability of retrieving additional targets that have been missed in the initial results.

In another thread of research, researchers attempt to answer the question that given a small set of labels, what will be the best data sample in the next iteration of user inspection or observation?

The objective is to actively select the most beneficial sample for observation so that the uncertainty about the classification model can be reduced to the largest extent. In contrast with the conventional machine learning methods that passively sample data for labeling, such approaches select sample data in an active way, therefore referred to as active learning in the literatures [35][69][146]. Active learning methods have shown very promising results in interactive multimedia retrieval. However, in most cases supervised learning techniques are used and a non-trivial number of labeled data are needed in order to learn a classifier with reasonable quality. Such requirements make them non-competitive when there are only very few labeled samples available. In addition, most active learning methods select data that are difficult to classify, aiming at resolving the uncertainty near the local point. However, such methods ignore the impact of additional labels to a larger extent, including other data in the unlabeled collection.

Semi-supervised learning techniques have attracted a lot of attention from researchers due to its major advantage that the manual labeling cost can be greatly reduced. As discussed in Chapter 2, the existing graph-based transductive learning techniques, though promising, are still inadequate under several challenging conditions in practice. For example, the interactive retrieval process considered in this chapter often lead to imbalanced situations in which labeled samples from one class often significantly outnumber those from different classes. Such conditions often cause inaccurate results from label propagation. In addition, the data samples and their features may be subject to a large level of noise, causing confusing and ambiguous cases for classification. Furthermore, data labeled by users may be sampled in a biased way, leading to biased coverage of the data set and thereby incorrect classification results.

In the TAG system, we implement several novel ideas described in Chapter 3 to address the problems mentioned above. Specifically, we use an iterative optimization method to improve the label propagation accuracy. During each iteration of the process, the most informative label is automatically selected and its class label is predicted. The added label sample is then added to the labeled pool and the optimal predicted labels for the rest of the unlabeled data are then computed. Such techniques improve the quality of the label propagation results by avoiding an aggressive step of predicting a large number of labels from a small number of labels. Instead, it implements a judicious procedure to predict new labels incrementally, starting from the most informative ones.

In addition, we apply a novel label normalization method to address the class imbalance issue.

Each class is assigned an equal amount of weights and each member of a class is assigned a weight proportionally to its connection density and inversely proportional to the number of samples sharing the same class.

Finally, the TAG system includes a novel incremental learning method that allows addition of new labeled samples efficiently. Each time when user labels more data, the prediction results of the rest of the images can be quickly updated using a superposition process without repeating the entire propagation process. Influence by the new labels can be easily added to the original predicted labels. Such incremental learning capabilities are important for achieving real-time responses interaction between the users and the system.

5.1.2 TAG System Overview:

We present the system diagram of TAG system in Figure 5.1. Given a collection of images or video clips, TAG system builds an affinity graph to capture the relationship among individual images or videos. The graph is also used to propagate information from labeled data to a large number of data in the same collection. In the following, we will walk through the main processed involved in building the graph and using the graph for label propagation.

Feature Extraction and Graph Construction: Each vertex in the graph represents a basic entity (data sample) of retrieval and annotation. It can be an image, a video clip, a multimedia document, or an object contained in an image or video. In the ingestion process, each data sample is first pre-processed (e.g., scaling, partitioning, noise reduction, smoothing, quality enhancement etc). After pre-processing, features are extracted from each sample. TAG does not dictate usage of specific features. Any feature set preferred by practical applications may be used, such as global features (color, texture, edge), local features (such as local interest points), and spatial information (such as layout). Multiple types and modalities of features may also be aggregated or combined. Given the extracted features, affinity (or similarity) between each pair of samples is computed. The pair-wise affinity values are then assigned to be weights of the corresponding edges in the graph. Usually, weak edges with small weights are pruned to reduce the complexity of the affinity graph. Alternatively, a fixed number of edges may be set for each vertex to construct neighborhood or b-matching graph, as described in Chapter 2.

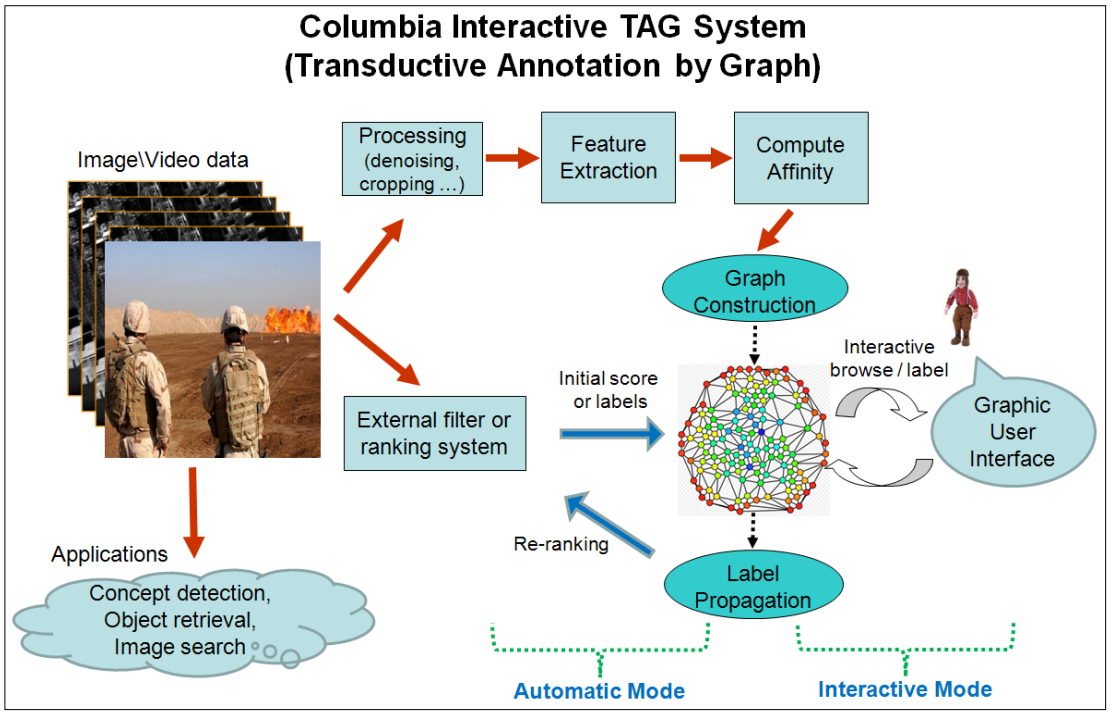


Figure 5.1: The system diagram and usage modes of Columbia TAG (Transductive Annotation by Graph) System.

Annotation and Browsing With the affinity graph in place, TAG system is ready to be used for retrieval and annotation. TAG currently provides two different modes for such processes. In the *Interactive Mode*, users browse, view, inspect, and label images or videos through a graphic user interface (GUI) described later in this document. Initially before any label is assigned, a subset of data may be shown in the browsing window by using certain metadata (time, ID, etc) or simply random sampling of the collection. Using the GUI, user may view any image of interest and then provide feedback about relevance of the result (e.g., marking the image as relevant or irrelevant). Such labels can then be encoded as labels and assigned to the corresponding vertices in the graph.

In the *Automatic Mode*, the initial labels of a subset of vertices in the graph may be provided by some external filters, classifiers, or ranking systems. For example, if the target of interest is ”helipad” for military intelligence analysis in satellite imagery, an external classifier using image features and computer vision classification models may be used to predict whether the target is present in an image and assign the image to the most likely class (positive vs. negative). If the target is product image search for Web images (say ”automobile”), external Web image search engines

may be used to retrieve most likely images using keyword search. The rank information of each returned image can then be used to estimate the likelihood of detecting the target in the image and approximate the class scores which can be assigned to the corresponding vertex in the graph. As mentioned above, each vertex in the graph is associated with either a binary label (positive vs. negative) or a continuous-valued score approximating the likelihood of detecting the target.

Graph-Based Label Propagation and Results Visualization

Given the assigned labels or scores for some subset of the vertices in the graph (usually a very small portion of the entire graph), a key function of the TAG system is to propagate the labels to other vertices on the graph in an accurate and efficient way. Such propagation process needs to be fast, completed in the real time or near-real time in order to keep users engaged. After the propagation process is completed, the predicted labels of all the vertices of the graph are used to determine the best order of presenting the results to the user. A standard option is to rank the images in the database in the descending order of likelihood so that user can quickly find additional relevant images. Figure 5.2 shows the GUI used in TAG following this standard method. An alternative is to determine the most informative data to show to the user so that human inspection and labels may be collected for such critical samples. The objective is to maximize the utility of the user interaction so that the best prediction model and classification results can be obtained with the least amount of manual user input. The graph propagation process may also be applied to predict labels for new data that are not yet included in the graph. Such processes may be based on nearest neighbor voting or some forms of extrapolation from existing graph to external vertices.

5.1.3 Sample Applications

Here, we present a case study in searching images downloaded from Internet photo sharing site *Flickr*. In this application, users are given a collection of images that have been filtered using keywords, and would like to quickly retrieve images of a specific class (for example *Statue of Liberty*) through interactive browsing and relevance feedback. We assume that no prior defined recognition models have been trained to simulate the scenarios in which users may change their targets of interest dynamically depending on the contexts and tasks. Using the TAG system, users are able to quickly zero in on the images matching their specific interest by browsing and annotating returned results as positive (relevant to target) or negative (irrelevant to target). The TAG system

Figure 5.2: The graphic user interface (GUI) of Columbia TAG System. Images shown in this example are from the photo sharing website *Flickr* using a text search "*statue of liberty*".

uses the label propagation method described earlier to infer likelihood scores for each image in the collection indicating whether the image contains the desired target. User can repeat the procedure of labeling and propagation to refine the results until the output results satisfy the requirements.

Data

For this proof-of-concept experiment, we acquired an image collection from *Flickr* first by a a simple text search "Statue of Liberty". Images on such web sharing sites usually are already associated with certain textual tags, assigned by users who upload the images. However, it has been well recognized that such manually assigned tags are inaccurate - the error rate could be as high as 50% [85]. Such discrepancy may be due to the ambiguity of labels or lack of control of the labeling process. Here, in this experiment, we show how TAG system can be used to quickly refine the accuracy of the labels. In the specific experiment, we use "*statue of liberty*" as the keyword and download the top 3000 returned images, which are fed as input to the TAG label propagation system. As shown in Figure 5.2, many of the initial returned images from *Flickr* are not correct - the visual content in the images does not actually show the scene or object of the statue. Using the initial 3000 images

from *Flickr*, we extract features and construct a TAG graph. Users then interact with the TAG system to browse images and provide relevance labels for a few images, from which additional images showing the object/scene of the statue are predicted and retrieved.

Features and Graph Each candidate image is processed to extract features. We applied the soft weight version of bag of visual word (BoW) feature, which has been shown effective in improving robustness of object and scene recognition [77]. Note TAG system is scalable in terms of feature representation. Therefore, other application specified features can also be utilized to improve the graph propagation.

With the extracted image features, we can compute the pair wise affinity between samples to construct the undirected and weighted graph. The procedure of building a efficient and robust graph contains several key steps, such as graph sparsification and edge weighting, as described in Chapter 2. In our experiments, we build kNN graph with a fixed number of nearest neighbors ($k = 25$). Then Gaussian kernel weighting with adaptive kernel size is used to to compute the edge weights between image pairs.

Results

In this experiment, we use a data set that includes 3000 *Flickr* images in response to a search keyword "*statue of liberty*". We formulated the problem as a two-class problem - distinguishing images containing Statue of Liberty from those not containing Statue of Liberty. We use TAG to label a small number of samples and then conduct graph-based label propagation to predict the likelihoods and re-rank the images in the database. We evaluate the performance of the TAG system by measuring the accuracy of the first page of results (precision of the top 20 returned images). Specifically, we compute the top-20 precision (average over multiple runs) and evaluate the influence of the number of labeled images on the accuracy. We will demonstrate that a very high precision can be achieved by using the TAG system even only a very small number of labels are given by users.

Figure 5.3 shows the performance curve of the TAG system for the experiment on the statue of liberty dataset. The horizontal axis is the number of manually labeled samples and the vertical axis is the error rate among the top 20 ranked images. As shown in this figure, TAG system demonstrates very good performance (error rate as low as 1%) with only 4 labels on the average. In Figure 5.4, we show samples of the TAG retrieval results (top 20 images) with only one label given by the user. Two different scenarios are shown - one targeting at the far view of the location and the

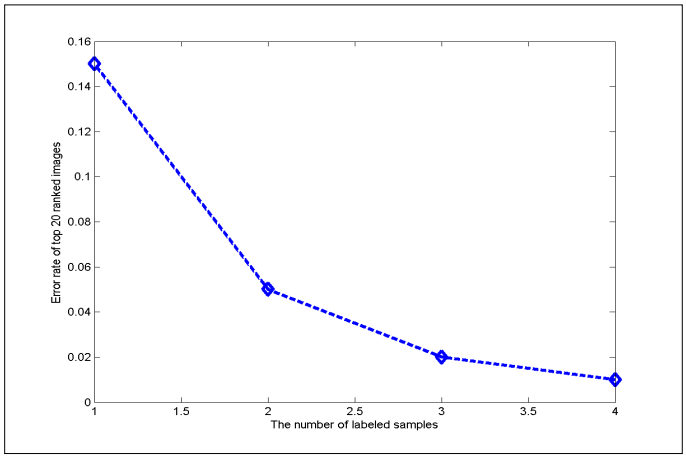


Figure 5.3: The top-20 accuracy of TAG label propagation results with different number of manual labels.

other focusing on the near view. Excellent accuracies are achieved, 95% for the far-view statue of liberty and 100% for the near view. These confirm the effectiveness of using the TAG system to rapidly refine the search results and retrieve relevant images from large collections for any arbitrary targets of interest, without depending on pre-defined target classes and time consuming labeling and learning processes.

5.1.4 Discussion and Summary

Columbia TAG system implements novel graph-based label propagation methods for rapid searching of images and videos that match user interest related to either predefined categories or arbitrary targets without prior definition. Its unique features include

. a new framework for real-time interactive image search and label propagation;

. novel graph-based transductive learning methods for solving the challenging issues, such as small labeled data set, imbalanced labels, noisy locations of labeled data, and unreliable labels;

. a superposable graph transduction method for decomposing the label propagation process into subprocesses, each of which involves only a single or subset of label inputs;

Figure 5.4: Examples of image search results by TAG system. The left and the right figure show the TAG propagation results of far-view and close-view of *statue of liberty*, respectively. With only one manual label by user, TAG successfully propagates the labels to correct samples with 95% and 100% accuracy.

. implementation of the above label propagation algorithms in an interactive image retrieval system, in which user labels are used as input for propagation in each iteration;

. implementation of the above label propagation algorithms in a fully automatic label refinement system without user interaction.

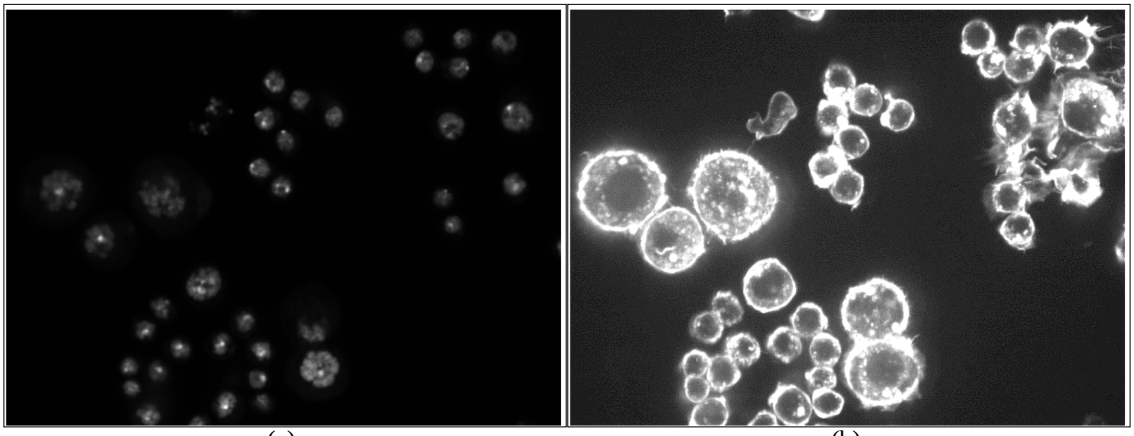
In the next section, we will present a specific application of the the TAG system for biomedical microcopy image analysis.

5.2 Interactive Visual Annotation for Microscopic Images

5.2.1 Introduction and Motivation

Cellular Microscopic Screening: Gene function can be assessed by analyzing disruptive effects on a biological process caused by the absence or disruption of genes. With recent advances in fluorescence microscopy imaging and gene interference techniques like RNA interference (RNAi), genome-wide high-content screening (HCS) has emerged as a powerful approach to systematically study the functions of each individual gene. These microscopic screenings generate a large number of biological readouts, including cell size, cell viability, cell cycle, and cell morphology. A typical

HCS cellular image usually contains a population of cells shown in multi-channel signals, such as DNA channel (indicating locations of nuclei) and F-actin channel (indicating information of cytoplasm) (Figure 5.5).



(a) (b)

Figure 5.5: Typical microscopic images of Drosophila K_{c167} embryonic cells. (a) image of the DNA channel; (b) image of the F-actin channel after homomorphic enhancement.

Recently through manual analysis of fluorescence microscopy images, cellular phenotypes visible in RNAi cell images (e.g., cytoskeletal organization and cell shape) have been found important for HCS study [87]. Specifically, when an individual gene is "turned off" by the RNAi technology, the resulting changes of the morphological structures of the cells in the images can be used to infer the function of the gene on the biological process under investigation (e.g., drug design, disease mechanism). However, a critical barrier preventing successful deployment of large-scale genome-wide HCS is the lack of efficient and robust methods for automating phenotype classification and quantitative evaluation of the rapidly increasing collection of HCS images.

Interactive Microscopy Annotation: One important task in HCS is to rapidly retrieve the most relevant cellular images from the database given a certain cell phenotype of interest specified by biologists. Currently this is handled in a manual way - biologists first examine a few example images showing the phenotype of interest, and then manually browse through individual microscopic images, and assess the relevance of each image to the cellular phenotypes. Apparently, this manual procedure is very expensive and relies on well trained domain experts. In [161][162], we have developed a supervised learning manner based cellular phenotype identification system. However, it

still requires a significant amount of input from experts.

In this section, we present an efficient interactive annotation framework for RNAi microscopic cellular images using the TAG system described in the previous section. Starting with the expert labeling of a few cells according to some predefined phenotypes, the system infers the phenotype classes of all unlabeled cells on the microscopic images. As mentioned earlier, the learning process is done in a semi-supervised manner that both the labeled and unlabeled data are utilized. Given the predicted phenotype label for the cells, image-level relevance scores are also computed. Then the system recommends the most relevant cell images to the biologist who will review the results and make further cell-level inspection.

The objective of the proposed interactive system is to drastically improve the throughput of finding relevant images from a large RANi cellular image collection. The underlying technical goal is to develop a novel graph transduction approach that can infer accurate cell phenotype prediction, and also work in an incremental manner to handle new cell labels obtained from the interactive annotation procedure. Note the proposed system is different from the regular relevance feedback or active learning systems for image retrieval. Here the annotation is done at the cell level, while relevance scoring and recommendation are computed at the image level.

Motivation: There are two major problems in directly applying such the semi-supervised learning techniques to the cellular image annotation task. First, the manual cell labeling on microscopic images easily generates imbalanced labels since the priors of different phenotypes vary a lot and the most images shown to the users often belong to a certain phenotype. In such situations, existing methods usually fail, as illustrated in the experiments using a toy example in Figure 2.4. Second, the interactive annotation system needs to respond fast to the input from experts in the practical setting. To solve these problems, we developed an interactive annotation and search system for microscopic image analysis that utilizes the power of graph transduction. Specifically, we apply the label weight normalization process (described in Chapter 3) to handle the imbalanced label issue, and a new superposable graph propagation approach to achieve real-time response to user interaction. Through extensive experiments over realistic RANi cellular images, we demonstrate the proposed techniques can significantly improve annotation accuracy while improving the speed at the same time.

5.2.2 Graph Transduction with Superposition Law

First, we will show the graph transduction procedure can be decomposed into superposable components, each of which can be computed incrementally. First the label matrix $\mathbf{Y}$ can be decomposed to the sum of a series individual sample label mask. For each individual labeled sample $\mathbf{x}_i$, the corresponding label mask is defined as $\hat{\mathbf{Y}}_i = \{\hat{y}_{ij}\} \in \mathcal{R}^{n \times c}$, where only one nonzero element $\hat{y}_{ij} = 1$ if the class label of $\mathbf{x}_i$ is j. So we can write $\mathbf{Y} = \sum_{i=1}^{l} \hat{\mathbf{Y}}_i$, where l is the cumulative number of labels given by the user so far. Replace $\mathbf{Y}$ in Eq. (3.11) by the sum of individual label mask, we can get:

$$\mathbf{F} = \mathbf{P}\mathbf{\Lambda} \sum_{i=1}^{l} \hat{\mathbf{Y}}_i = \sum_{i=1}^{l} \mathbf{P}\mathbf{\Lambda}\hat{\mathbf{Y}}_i = \sum_{i=1}^{l} \hat{\mathbf{F}}_i, \tag{5.1}$$

where $\hat{\mathbf{F}}_i = \mathbf{P}\mathbf{\Lambda}\hat{\mathbf{Y}}_i$ is the classification function predicted only by using labeled sample $\mathbf{x}_i$. In other words, the classification function $\mathbf{F}$ obtained by graph propagating using the labeled sample set $\mathbf{X}_l = \{\mathbf{x}_1, \cdots, \mathbf{x}_l\}$ equals the sum of a functional set $\hat{\mathcal{F}} = \{\hat{\mathbf{F}}_1, \cdots, \hat{\mathbf{F}}_l\}$, where each element of $\hat{\mathbf{F}}$ is the classification function propagated from a individual sample in $\mathbf{X}_l$. We call this as the superposition law in graph propagation procedure [155]. This allows us to update the classification function $\mathbf{F}$ incrementally when each new labeled is received without recalculating the whole propagation from the entire label set. The superposition solution in Eq. (5.1) can be further extended to class-based superposition cases as:

$$\mathbf{F} = \sum_{j=1}^{c} \sum_{z_i=j} \mathbf{P}\mathbf{\Lambda}\hat{\mathbf{Y}}_i = \sum_{j=1}^{c} \sum_{z_i=j} \hat{\mathbf{F}}_i, \tag{5.2}$$

where $\sum_{z_i=j} \hat{\mathbf{F}}_i$ denote the contribution to the final prediction results from labels of class j. Apparently, it only has nonzero values in the jth column vector.

It is easy to see how the imbalanced labeling mentioned above affects prediction results. For example, consider a two-class case and assume there are l_1 positive and l_2 negative labels. From the class superposition Eq. 5.2, it is easy to derive the following:

$$\mathbf{F} = \sum_{i=1}^{l_1+l_2} \hat{\mathbf{F}}_i = \sum_{i=1}^{l_1} \hat{\mathbf{F}}_i + \sum_{i=l_1+1}^{l_1+l_2} \hat{\mathbf{F}}_i \simeq \sum_{i=l_1+1}^{l_1+l_2} \hat{\mathbf{F}}_i \tag{5.3}$$

If $l_1 << l_2$ for the imbalanced labeling condition, the predicted classification results will have strong bias towards the positive class since most vertices will be dominated by the scores propagated from labels of the majority class. We have discussed this imbalanced label issue using the two-moon

toy data earlier in Figure 2.4(g). Most of the existing *SSL* methods generate biased classification results, as shown in Figure 2.4(h)-2.4(k). Compared to other methods, our proposed bivariate graph transduction algorithm, i.e., *GTAM*, produced more reasonable predictions (Figure 2.4(l)). The robustness of our approach to imbalanced labels is mainly attributed to the label weight term $\mathbf{\Lambda}$ and the alternating minimization strategy, as discussed in Chapter 3.

To handle the label imbalance issue, we apply the label weight normalization procedure discussed in Chapter 3. Consider the classification function $\mathbf{F}$ and label matrix $\mathbf{Y}$ as the concatenation of column vectors as:

$$\mathbf{F} = [\mathbf{F}_{\cdot 1} \cdots \mathbf{F}_{\cdot j} \cdots \mathbf{F}_{\cdot c}] \tag{5.4}$$

$$\mathbf{Y} = [\mathbf{Y}_{\cdot 1} \cdots \mathbf{Y}_{\cdot j} \cdots \mathbf{Y}_{\cdot c}], \tag{5.5}$$

where $\mathbf{F}_{\cdot j}$ and $\mathbf{Y}_{\cdot j} (j = 1, \cdots, c)$ correspond to predictions and labels from class j. Applying the superposition method, the column vector $\mathbf{F}_{\cdot j}$ can be computed as:

$$\mathbf{F}_{\cdot j} = \mathbf{P}\mathbf{\Lambda}\mathbf{Y}_{\cdot j}. \tag{5.6}$$

The above equation can be seen as the vector version of superposition law of Eq. (5.2). Given a new labeled sample $\mathbf{x}_s$ with degree d_s belong to class $z_s = j$. In this case, only the jth column of the label matrix needs to be updated, which is vector $\mathbf{Y}_{\cdot j}$. From Eq. (5.6), only the vector $\mathbf{F}_{\cdot j}$ need to be recomputed. Let D_j denotes the total degree of the existing labeled samples in class j excluding the new labeled sample $\mathbf{x}_s$. We can calculate two weighting coefficients σ, γ as:

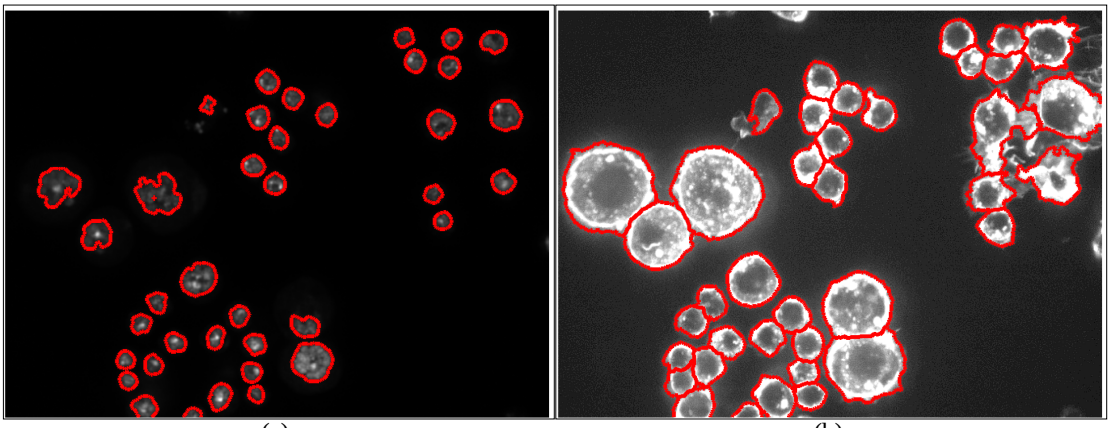
$$\sigma = \frac{D_j}{D_j + d_s} \tag{5.7}$$

$$\gamma = \frac{d_s}{D_j + d_s} = 1 - \sigma. \tag{5.8}$$

Then the new vector $\mathbf{F}_j^{new}$ can be updated as:

$$\mathbf{F}_{\cdot j}^{new} = \sigma \mathbf{F}_{\cdot j} + \gamma \hat{\mathbf{F}}_s = \sigma \mathbf{F}_{\cdot j} + \gamma \mathbf{P}_{\cdot s}, \tag{5.9}$$

where $\hat{\mathbf{F}}_s$ propagation results based on $\mathbf{x}_s$ only. Note that $\hat{\mathbf{F}}_{\cdot s}$ is exactly the sth column vector of $\mathbf{P}$, i.e. $\hat{\mathbf{F}}_s = \mathbf{P}_{\cdot s}$. Based on the superposition law discussed in the previous section, the above updating step by replacing the jth column of $\mathbf{F}$ with $\mathbf{F}_{\cdot j}^{new}$ is equivalent to the the optimal prediction directly



(a) (b)

Figure 5.6: The automatic segmentation result of the microscopy image of Figure 5.5. (a) nuclei segmentation; (b) extracted cell bodies.

derived from Eq. (5.1). However, the superposition approach is much efficient since we reduce the computation cost from matrix multiplication to scalar multiplication and vector addition.

During the active annotation process, the cells in the microscopic images are assigned with soft labels indicating the class memberships of various phenotypes. We use a simple method to aggregate the class labels for cells and measure the class labels of the whole image. The image relevance score vector $\mathbf{r} = \{r_j\}, (j = 1, \cdots, c)$ representing the relevance score of an image is computed as the average of class scores of all cells $\mathbf{x}_t$ in the image $\hat{\mathbf{X}}$:

$$r_j = \sum_{\mathbf{x}_t \in \hat{\mathbf{X}}} \mathbf{F}_{tj} / n_{cell}, \tag{5.10}$$

where n_{cell} is the number of cells in the image $\hat{\mathbf{X}}$. Finally, the image-level scores are used to rank images in the databases for user annotation and browsing.

5.2.3 Experiments and Evaluation

5.2.3.1 Material and Preprocessing

We collaborated with the bioinformatics group and Genetics Laboratory in Harvard University for the experiments of cellular microscope image annotation. The microscopic images of *Drosophila* K_{c167} embryonic cells were used to validate the active annotation approach. The images were ac-

Cell Phenotype	*Appearance Description*
Actin Accumulation (*AA*)	actin accumulation in the cell body, bright intensity, may have non-round nuclei;
Cell Cycle Arrest (*CycA-sti*) [8]	large size, round cells with multi-nuclei;
Longthin-LPA (*LL*)	long punctuate actin, with a cell shape like prolonged water drops or long thin poles;
LS-Fla (*LF*)	cells with a large spiky and filamentous structure;
Rho	large and flat shape, with multi-nuclei, non-round.

Table 5.1: Biologically pre-defined cellular phenotypes and the description about their appearances.

quired by automated microscopy with a Universal Imaging AutoScope Nikon TE300 [161]. The previous study on this dataset shows that the image appearance at the cell level reflected the underlying gene function expression [7]. We used 70 HCS microscopy screening sets, resulting a set of 210 cell images with three channels (only DNA and F-actin images are used for analysis). First we applied homomorphic filtering on the raw data to perform image enhancement and denoising. Since the DNA signal is fairly strong, clearly noticeable with respect to a relatively uniform dark background, we first segmented nuclei by a simple histogram thresholding technique. However, cytoplasmic boundary segmentation was a challenging task due to intensity variation and cellular phenotype diversity. Starting with the segmented nuclei region, we further applied a seeded watershed algorithm incorporating deformable models to separate both isolated and attached cell bodies [177]. Figure 5.6 shows examples of the segmented cells in cellular microscopic images. After segmentation, we obtained a total of 3162 valid cell segments, among of which 191 (6%) cells were manually labeled as ground truth.

For these cell segments, biologists pre-defined five distinct cellular phenotypes (refer to Table 5.1). All these cellular phenotypes exhibit unique texture and geometric characteristics, as the

[8] abbreviated as CycA-sti since this cellular phenotype is frequently found when the genes *CycA* and *sti* are knocked down by RNAi.

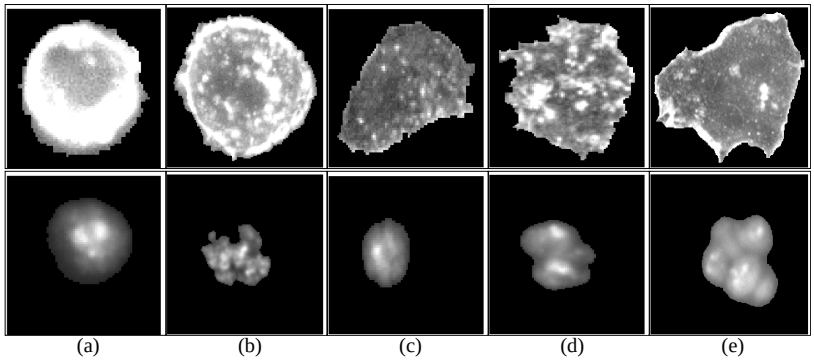


Figure 5.7: The cell segments examples of predefined cellular phenotype prototypes. The top row is the cytoplasm and the bottom row is the corresponding nuclei. (a) Actin Accumulation (*AA*); (b) Cell Cycle Arrest (*CycA-sti*); (c) Longthin-LPA (*LL*); (d) LS-Fla (*LF*); and (e) Rho.

examples shown in Figure 5.7. In order to capture the morphological and appearance properties of different cellular phenotypes, a total of 214 dimensional attributes, including wavelet features, Zernike moments features, Haralick features, region properties, were computed from the cell segments [161]. The ℓ_2 distance and Gaussian kernel were applied to compute pair-wise similarity among cells and a neighborhood graph is constructed from these cellular features.

5.2.3.2 Results

Since each microscopic image contains a population of cells, some of which belongs to different phenotypes. However, the most dominant cellular phenotype in an image reveals the underlying gene "turn down" function expression. Hence, the microscopic images are categorized into five types, corresponding to the five phenotype in cell level. The task of annotating the image class is to rank all the images in the collection based on the relevance to a certain cell phenotype query. It helps the scientists rapidly target the most relevant genes related to a biological hypothesis. It also can be used to collect the positive samples for further mining task.

Staring from 10 initial cellular labels, at least one for each phenotypes, we simulated the interactive annotation procedure by subsequentially adding 10 more cell labels in the next round. In the experiments, we combine the above active graph transduction approaches on top of *LGC* and *GFHF* to design so called label regularized *LGC* (*LR-LGC*) and label regularized *GFHF* (*LR-GFHF*) meth-

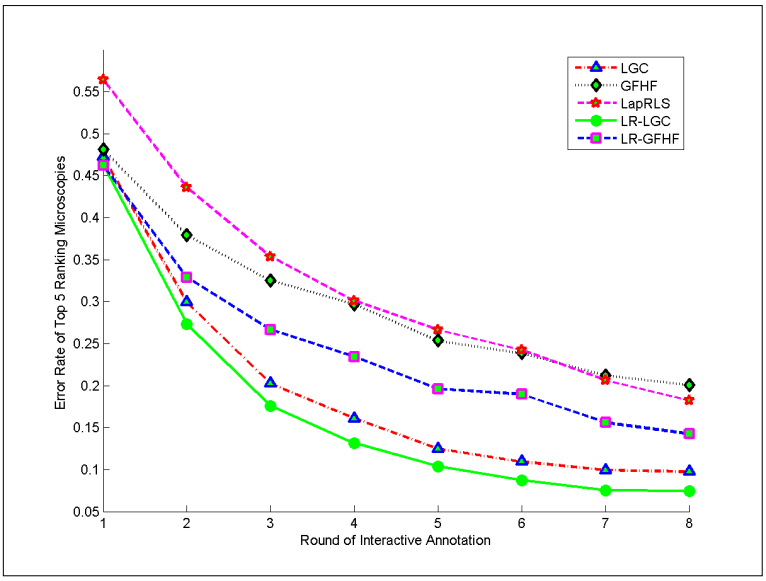
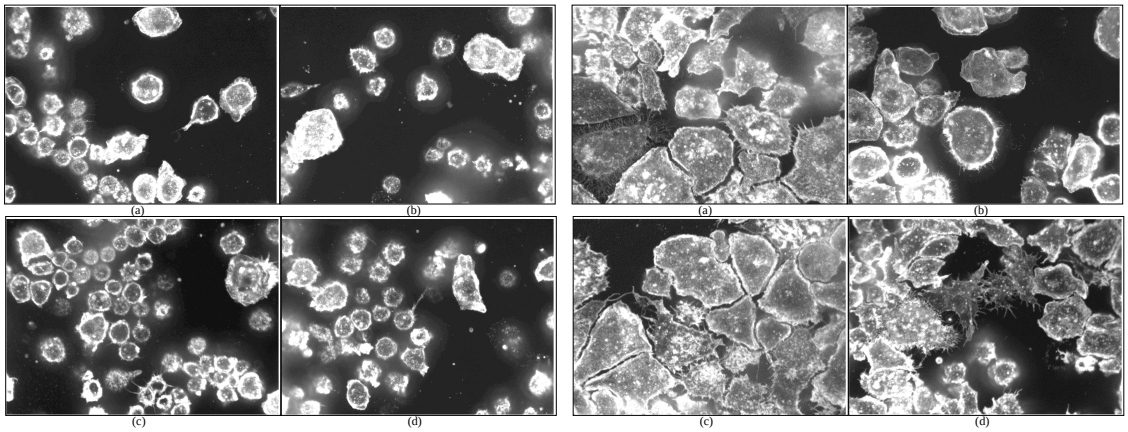


Figure 5.8: The performance of active cellular image annotation using the graph transductive learning approach. X coordinate denotes the number of labeling rounds and Y coordinate denotes the accuracy of top 5 ranked microscopy images.

ods. We compared these two new methods with the standard *LGC*, *GFHF*, and *LapRLS*. Figure 5.8 gives the performance comparison of the five approaches. We can see that the annotation accuracy on the microscopies increases as more cell labels are available. Our proposed idea using label weight normalization improved both *LGC* and *GFHF* significantly. With 8 rounds of annotation, only 80 cell labels (around 2.5% of the total cell segments) are given, but an annotation accuracy (by *LR-LGC*) as high as 92.6% can be achieved. Figure 5.9(a) and 5.9(b) shows two examples of the top four images in response to the cellular phenotype query of *AA* and *Rho*. Additionally, the computational cost of the active annotation is analyzed in Table 5.2. Since the graph construction can be implemented off line, the table only includes the computation cost associated with the online active annotation procedure. The superposable solution *LR-LGC* discussed in this section reduced the computation cost and matches the real-time requirement in practical applications. However, *LG-GFHF* method does not benefit from the superposition procedure since the prediction results have to be recomputed after receiving new labels, as discussed in our previous work [155].

Method	LGC	GFHF	LapRLS	LR-LGC	LR-GFHF
Computation Cost (sec.)	0.81	70.05	218.9	0.14	70.28

Table 5.2: Computation cost of graph-based transductive annotation after 8 rounds of user interaction.



(a) (b)

Figure 5.9: Examples of cellular image annotation results by graph-based transductive learning: a) query results of *AA* cellular phenotype; b) query results of *Rho* cellular phenotype.

5.3 Label Diagnosis through Self Tuning for Web Image Search

5.3.1 Motivation and Introduction

Mislabeling issue occurs frequently in web image annotation due to uncontrolled labeling procedure and semantic ambiguity. For example, *Flickr*, one of the most popular photo sharing website, allows users to assign textual tags when they upload images. However, it has been well recognized that there exists high inaccuracy among such manually assigned textual tags, whose accuracy is only around 50% accuracy [85]. Most of the current image search techniques directly utilize the textual tag associated with the images. Apparently, this significantly degrades the accuracy of the search results. For instance, when the user types in the keyword "*tiger*", visually inconsistent results, as shown in Figure 5.10, could be returned though all contain the keyword "*tiger*". The results may include images of images of apex predator, butterfly, flower, tank, or golf professional. As discussed earlier, such falsely tagged images, if used as labels in machine learning methods, will

cause significant performance drop.

In this section, we apply our bivariate graph transduction approach, i.e. label diagnosis through self tuning (*LDST*), to address this critical problem with mislabeled samples. The objective is to diagnose the quality of the given labels and remove unreliable labels while preserving visual consistency. As described in Chapter 3, a floating greedy approach is executed to simultaneously carry out correction over labeled samples and prediction over unlabeled samples. In the case of *Flickr* image search, *LDST* is used to refine the text based image search results by preserving only "high-quality" labels, and then use *GSSL* methods to propagate initial labels to the rest of the databases, and retrieve additional true positive samples. Conceptually, such refinement and propagation techniques belong to the broad category of works in search re-ranking [66]. Though most of the prior work did not address the issue of false labels and thus suffers from performance degradation when it occurs.

5.3.2 Related Work on Web Image Search Reranking

Most works on visual search reranking start with the text-based query and then apply various re-rank methods to refine initial search results. For example, the approaches based on pseudo-relevance feedback (PRF) use an initial search result set as pseudo labels, among which a small number of the top-ranked samples are assumed as positive, and bottom ranked images as negative [111]. Such labeled samples are then used to train classifiers modeling the semantic target of the query. Then the learned classifiers are used to re-rank the initial search results. PRF methods highly rely on the quality of the pseudo-positive or negative samples since they are considered as ground truths in the subsequent learning procedure. Another category of approaches apply probabilistic models, such as constellation model [48], probabilistic latent semantic analysis and latent Dirichlet allocation [47], to train region based bag-of-words classifiers. The principle of information bottleneck was applied to find the optimal clusters of images that preserve the maximal mutual information in the top result set [66]. However, the calculation of mutual information, which involves probability density estimation with a small number of samples, remains challenging in high dimensional feature space. VisualRank is recently developed to exploit visual content to re-rank image search results from *Google*. It uses random walk over an affinity graph to rank images based on the visual hyperlinks (similarity) among the images. Finally, the re-ranked image list is sorted based on the

Figure 5.10: Example images using text search "*tiger*" from the photo sharing website *Flickr*.

"importance" of graph vertices. This method suffers from several potential shortcomings. First, the top images from re-ranked results are often nearly duplicated with each other because near duplicate images tend to share similar vertex importance, thus produce equal high re-ranking scores. Second, the re-ranked images may be inconsistent if the initial text-search results have multiple dense sub-graphs corresponding to different patterns. For example, the text search by "*tiger*" returns multiple disparate clusters, such as apex predator and the professional golfer, both of which have high vertex importance.

In summary, the current visual search re-rank methods are failure in either blindly trusting the quality of top pseudo labels from the initial search results, or completely ignoring the initial ranking order and just resorting to data-driven unsupervised methods. In this section, we will apply the *LDST* method described in Chapter 2 to handle the semi-supervised scenario with unreliable labels. Specifically, we treat the top ranked images in the initial search results as possibly noisy positive samples, and use the *LDST* method to diagnose and tune such labels before propagating to the a large set of candidate images. The graph-based propagation results are then used to estimate the relevance scores and re-rank the images.

5.3.3 Refine Keyword Based Web Image Search

To evaluate our approach on the web image search task, a total of nine categories of images are acquired from the photo sharing website *Flickr* using text search. The selected categories cover a diverse range of targets, including animals, plants, man-made objects and scenes. For each set

Figure 5.11: Example images of text search results from the photo sharing website *Flickr*. A total of nine text queries are used: *dog*, *tiger*, *panda*, *bird*, *flower*, *airplane*, *forbidden city*, *statue of liberty*, *golden bridge*.

of text search results, about 1500 returned images are collected for re-ranking. Example images corresponding to these text queries are shown in Figure 5.11.

For image feature representation, we adopt the widely used Bag-of-Visual-Words (*BoW*) derived from local key points, which has been shown effective in many applications of object and scene classification. We use difference of Gaussian as key point detector and SIFT as descriptor [100]. To quantize the local features to visual words, we adopt the soft-assignment strategy which has been shown effective in [77].

The pair-wise affinity value is computed using cosine similarity between *BoW* vectors. The number of nearest neighbors is uniformly set as 200 (a typical setting for cosine similarity graph [14] [156]) to construct kNN graphs for the returned images from each individual query. Since there is no clear cue for selecting negative samples for each individual query, the classification task is degenerated to a ranking problem [176]. Here the top ranked 60 samples are treated as pseudo positive labels. Label self tuning is used to remove visually inconsistent samples and afterward propagation is done to rank the remaining images (the number of self tuning iteration is uniformly set as 30). We compare *LDST* with other automatic re-ranking methods, like Pseudo-relevance feedback (*PRF*) framework [111]. Specifically, we implemented *PRF-SVM*, *PRF-LGC* and *PRF-*

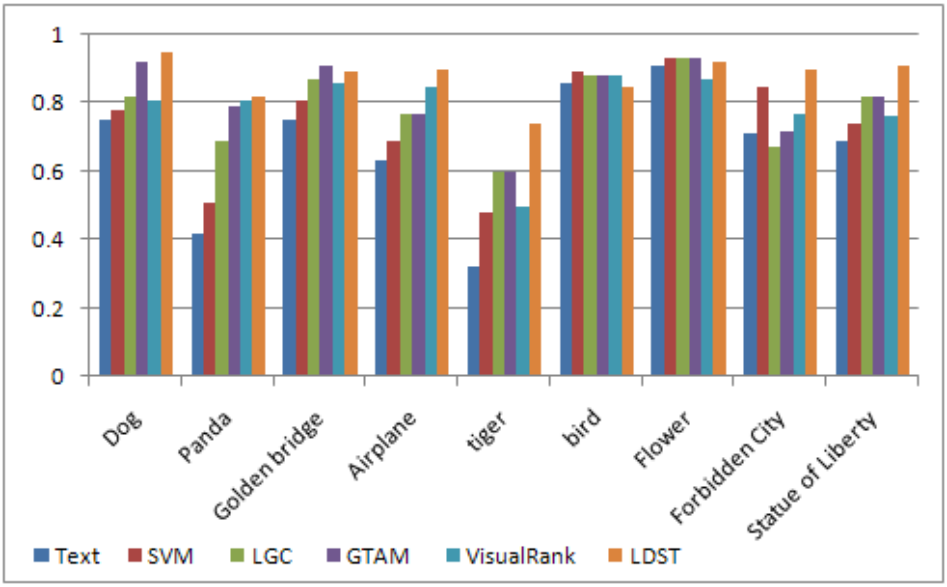


Figure 5.12: Comparison of the precision of the top 100 reranked images over different categories of images.

GTAM for comparison. In addition, we included the recent VisualRank technique [78], which has shown empirical success on product image search on *Google*. The same parameter setting is applied as suggested by [78]. Moreover, all the graph-based approaches use the same graph, as described earlier. The precision of the top 100 reranked images are calculated to evaluate the performance, as shown in Figure 5.12.

From Figure 5.12, *LDST* achieves significant performance improvement over most semantic categories, like *tiger*, *panda*, and *dog*. For these cases, the visual content of targets exhibits consistent pattern though there is strong ambiguity associated with the keywords. However, *LDST* does not show much gain for the categories of *bird* and *flower*. Because the pseudo positive samples associated with these two tags are quite diverse, which affects the stability of the label self tuning procedure. Overall, *LDST* improves the average accuracy of text search results from 67.11% to 87.56% on the nine categories (Table 5.3). Compared to VisualRank method, *LDST* also enjoys a clear performance gain of 10.8%.

There are two parameters in *LDST* for web image search, the initial number of positive l and the number of tuning iteration s. In the above experiments, we empirically set $l = 60$ images from

Method	Text	SVM	LGC	GTAM	VisualRank	LDST
Accuracy (%)	67.11	74.22	79.89	81.44	79.00	87.56

Table 5.3: The accuracy of the top ranked *Flickr* images by different approaches.

top ranked list as positive samples and fix $s = l/2$. We have carried out an extensive study by varying the value of l from 20 to 100. The result shows that *LDST* achieved fairly consistent and stable performance under different choices of l with the precision of the top 100 ranked images as $87.23 \pm 0.6\%$. In addition, the proposed method can be made very efficient by applying the superposable update model mentioned in the previous subsections. The current implementation takes only a few seconds to re-rank more than 1500 images for each query on a regular PC.

5.4 Summary and Discussion

Based on the bivariate graph transduction algorithms developed in Chapter 3, we designed several systems for image annotation and search in this chapter. Our main contributions include:

1. We proposed a general image annotation and search platform, called Transductive Annotation by Graph (TAG). It combines intuitive graphic user interfaces with the graph-based semi-supervised method that allow users quickly browse and annotate a small number of images/videos, and then in real or near real time predict the labels for all remaining unlabeled data. The TAG system can be used as a search system alone, or a bootstrapping system for developing additional target recognition tools.

2. We extended the general TAG framework to a specific application of real-time analysis of cellular images. Specifically, the cellular image annotation task is formulated as a joint procedure of cell label propagation and image relevance ranking. The developed system can be used to annotate and retrieve cellular images showing a large variety of cell phenotypes. It is critical for various biomedical applications such as large scale gene function study and drug designs. To the best of our knowledge, this is the first multi-level graph transduction learning system successfully validated over real microscopic cellular images.

3. Finally, we explored the power of label correction of the *LDST* algorithm and applied it to re-

rank keyword-based web image search results. By combining label diagnosis, manipulation, and propagation in a uniform optimization framework, the system can identify and eliminate unreliable labels, and return more consistent visual ranking results.

Chapter 6

Application: Image Retrieval via Brain Machine Interface

Different from the traditional image annotation and retrieval systems developed in the previous chapter, we here propose an integrated paradigm that marries the strengths of brain computer interface and machine learning systems in a unique and synergistic way. The brain state decoding subsystem utilizes advanced spatio-temporal analysis of neural response signals measured by Electroencephalography (EEG) in a single trial setting. The content analytics component applies our bivariate graph transduction and label diagnosis methods described in Chapter 3 to handle rare targets, small label sizes, and noisy conditions. The neural signal analysis component has a unique power for recognizing generic target classes, while the visual content analysis component is suitable for high throughput processing. The proposed system explores the synergy of these two and has shown promising performance in detecting generic target classes in a high throughput fashion.

6.1 Introduction

The human brain is widely considered to be the most powerful visual information processing system. The human visual system is able to get the "gist" of a scene in a few hundred milliseconds [86][113] [145]. As a result, many efforts have been made to understand the human vision mechanism through decoding brain state from neural signals. By monitoring the neural response signals, e.g., those recorded non-invasively via electroencephalography (EEG) [41], promising re-

sults have been shown in detecting objects of interests (OOI) contained in the visual stimuli presented to subjects [19][55][81][116][118].

One of the ultimate goals for automated computer vision or media content analysis is to detect and recognize objects, scenes, people, and events in images or videos. Such capabilities, if realized, will greatly enhance the performance and utility of many applications, such as human computer interaction and visual information search. Recently, impressive progress has been reported in the literature, including advances in image feature extraction, visual matching, and object categorization. Several widely participated benchmarking efforts, such as Caltech 101 [46], PASCAL [45], ImageClef [130], and TRECVID [140], have been organized to demonstrate and evaluate the state of the art in this field. A common framework used in such efforts is to learn object models from a pool of training data, which may have been wholly or partly annotated over pre-defined object classes. Such a learning framework has been shown to be powerful. However, it is limited in its scalability to large-scale applications. One of main barriers is the dependence on the manual annotation process, which is laborious and time consuming. To overcome this, efforts have been reported using interactive annotation with relevance feedback and active learning in order to reduce the amount of the required manual input [69] [124], including our work reported in Chapter 5. Recent works have also started to explore the freely available (but imperfect) metadata associated with images on the Web [78] [157].

In this chapter, we propose a novel framework that combines the power of brain state decoding and visual content analysis to maximize the efficiency of the image annotation and retrieval task in a completely hand free streamlined fashion. The proposed *brain computer interface and visual pattern mining* (BCI-VPM) based image annotation system, as shown in Figure 6.1, consists of two critical components, EEG-based generic interest detector and graph-based salient visual pattern discovery. The EEG-based interest detector mentioned above is generic - a subject adaptive EEG-based detector trained over a generic class can be applied to detect any new objects of interest. Likewise, the subsequent graph-based visual saliency discovery is general as it does not assume any prior knowledge about image classes or data sets. Additionally, by completely freeing users from any manual operation (e.g., button pressing) in the viewing stage, we achieve the maximal throughput of annotation or retrieval from a fast stream of image sequence via a process called *rapid serial visual presentation* (RSVP) paradigm [119]. RSVP involves images being flashed to the viewer

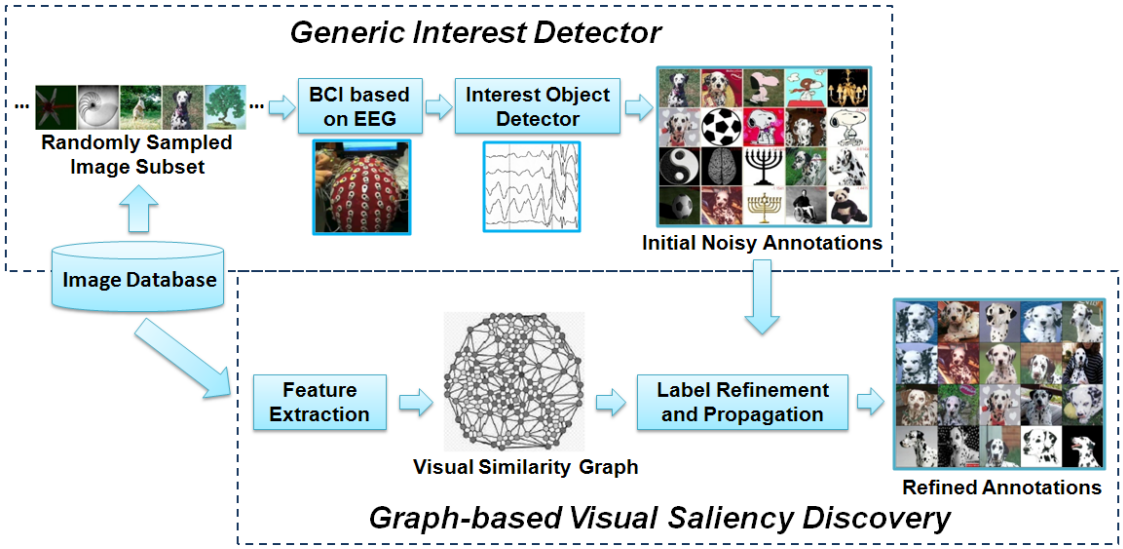


Figure 6.1: System diagram of the proposed BCI-VPM image annotation system. A small subset of images is shown to users, whose EEG-based neural response signals are used to detect objects of interest that catch users' attention. The EEG scores of the subset are then refined and propagated to the entire image collection through recovering the visual consistency and discovering the salient visual pattern over a visual similarity graph.

at high rates, such as 10 frames per second. EEG signal recorded from an array of transducers placed on the scalp of the subject are recorded continuously and analyzed to extract discriminative spatial-temporal features. Techniques like Hierarchical and Bilinear Discriminant Component Analysis [42] can be applied to analyze EEG signal to generate a probabilistic interest score for each image. The image sequence ordered by the EEG scores is then given further analysis to determine the salient visual patterns that catch users' attention. It utilizes our bivariate graph transduction approach described in Chapter 3 to recover visual consistency on a graph constructed from visual similarity. Label diagnosis and tuning techniques are also used to refine the interest score decoded from EEG neural signals. The unreliable EEG labels are eliminated, while the EEG scores associated with the small image subset that have undergone the EEG-based brain state decoding step are propagated to the entire database to retrieve further relevant images, including those not yet shown to users. The regular graph and the diffusion process with label correction have unique power in handling imperfect labels and sparse targets with extreme low prior.

In summary, we propose exploration of an integrated paradigm that marries the strengths of human vision and machine learning systems in a unique and synergistic way - human vision for its superb capability in detecting general objects in diverse and complex conditions and content analytics for automatic processing of large volumes of data.

We will show effectiveness of the proposed approach over images from multiple domains. The first set includes a total of 3798 Internet images with 62 object categories from the well-known Caltech 101 dataset. The second is for target detection in high-resolution satellite imagery (DigiGlobe images), containing 1051 images with "*Helipad*" targets. Experimental results from multiple subjects indicate a very promising performance. In the Internet image experiments, the annotation accuracy, measured in terms of average precision (AP), of one of the object classes *Dalmatian* was improved from 1.76% (random baseline) to 36.7% by EEG interest detector, and further boosted to 69.1% by the visual pattern mining process.

6.2 System Overview

The proposed BCI-VPM image annotation framework consists of two subsystems (as shown in Figure 6.1) - one using the neural signals measured by EEG to detect generic OOI present in images, while the other using graph-based visual pattern mining methods to refine the detection results from the first subsystem and propagate results to an expanded data set. We provide a brief overview here and give further details on each components in later sections.

For EEG-based OOI detection, a small subset of images (on the order of few hundred) is first randomly sampled from an image collection and presented as visual stimuli to the subject. The selection process avoids long EEG recording sessions which may cause subject fatigue. One of our design goals is to require minimal subject participation, yielding just sufficient information for the neural state decoder and the pattern mining module to effectively infer objects that have attracted a user's attention and generate labels for all the images in the collection.

The sampled images are then presented to the subject in a sequential fashion, following a paradigm called Rapid Serial Visual Presentation (RSVP) [55], as shown in in the left part of Figure 6.2. The subject is instructed to focus on a fixed marker in the screen center in the first 2 seconds, then each image is shown to the subject for a fixed period of time, ranging from 100ms to 200ms

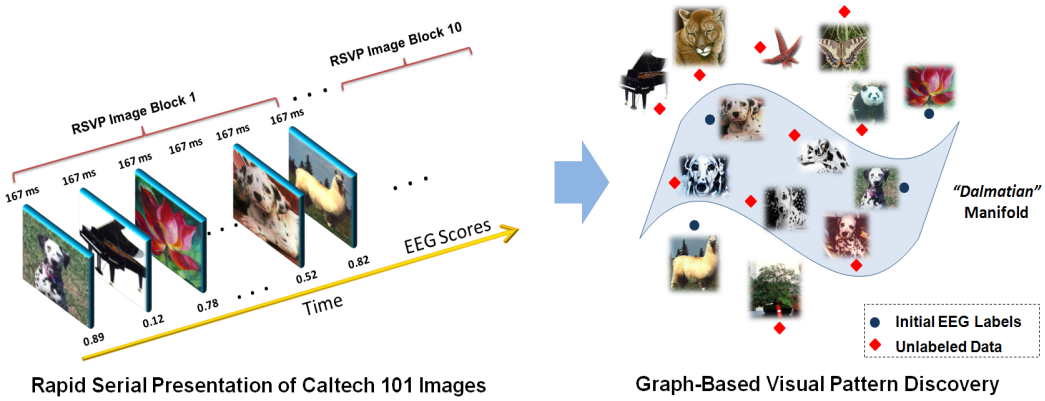


Figure 6.2: Overview of the processing pipeline of the proposed BCI-VPM image annotation system. The left demonstrates the RSVP paradigm used for presenting visual stimuli to subjects. The RSVP sequence contains 1000 randomly selected images, which are further partitioned into 10 blocks and 100 images each. An image block typically is shown at $5-10$Hz and each image lasts around $100-200$ milliseconds. The right shows the process of graph-based visual pattern mining. After ingesting the estimated EEG "interest" scores as initial labels, the underlying data manifold structure is explored to discover the salient visual pattern among the top EEG score ranked images to refine the initial labels, retrieve additional relevant images, and propagate labels to a much large image pool.

each (equivalent to $5-10$ Hz). The subject may be given instructions ahead of time to look for a specific class of object or simply allowed to choose any object class to their interest on the spot. An array of EEG electrodes placed on the subject scalp are used to continuously record multiple channels of neural response signals (e.g., at 1K sampling rate). Spatio-temporal processing and discriminative analysis are performed on the post-stimulus signals to compute a score, which predicts the confidence in detecting the OOI in each image viewed by the subject. Based on the EEG scores, images are ranked to form initial results, from which top ranked results are used as pseudo positive labels and fed to the pattern discovery module for further refinement and prediction. Note that due to low signal quality and subject variations, the EEG-based OOI detector is pre-trained for each subject. However, such one-time offline training can be done very efficiently without restricting the generality of the detector. As the detector is trained to detect shifts in the user's attention as opposed to detect the recognition of a specific object, an object class uncorrelated with the test objects can

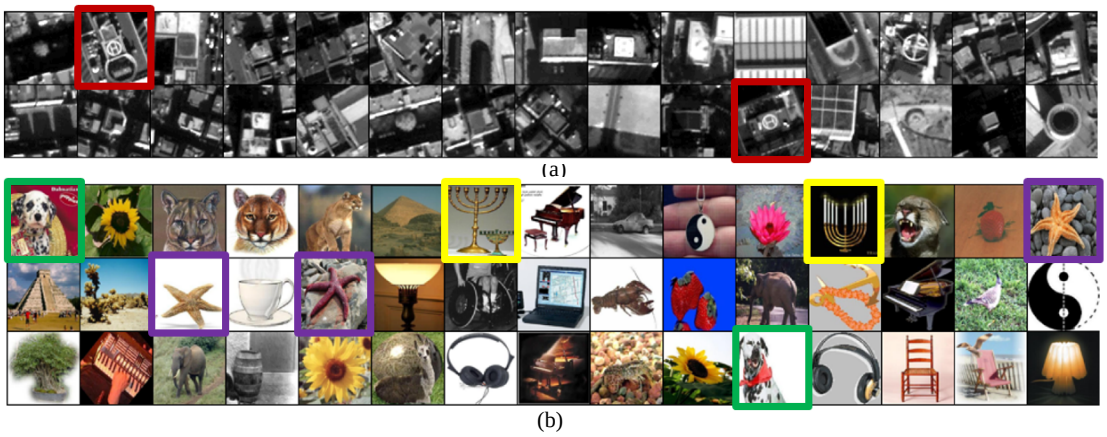


Figure 6.3: Example images shown to subjects with target objects highlighted with a color bounding box. a) Satellite imagery with target "*helipad*"; b) Caltech101 images with targets "*dalmatian*", "*starfish*" and "*menorah*".

be used to train the detector.

The pattern discovery subsystem starts with construction of an affinity graph, which captures the pairwise visual content similarity among vertices (corresponding to images) and the underlying subspace structures in the high dimensional space (as shown in the right part of Figure 6.2). Such a construction process is done offline before user interaction. The small set of pseudo positive labels generated by the EEG-based interest detector is fed to the initially unlabeled graph as assigned labels for a small number of vertices, which are used to drive the subsequent processes of label identification, refinement, and propagation. Bivariate graph transduction technique plays a critical role here since we will rely on both the initial noisy labels and the large pool of unlabeled data points throughout the diffusion process. Finally, the propagated label predictions over the entire graph can be used to generate annotations for every single image in the collection, or to re-rank the images based on the detection scores. The top ranked results, as shown in Figure 6.6, 6.7, 6.8, and 6.12 (b), are expected to be more accurate (in terms of both precision and recall) than the baseline of using EEG-based detection alone.

6.3 Generic Interest Detector via Single Trial EEG Decoding

There has been increasing interest in investigating the application of BCI for image annotation and search [55][116][81][19]. Motivated by humans' ability to make very rapid and accurate visual decisions in "the blink of an eye" [57], we extend the usage of the BCI based image search system in [55] to design a generic interest detector, where users are instructed to look for specific object classes in sequences of images presented using the RSVP paradigm. Examples of segments of the RSVP image sequences used in our experiments are shown in Figure 6.3 [9]. A crucial aspect of this particular annotation system is that we can measure brain signals, in real-time, that can be used to annotate or rank an image given a desired object class. From neuroimaging studies, the neural signals can be measured non-invasively to detect and recognize objects in rapidly shown image sequences [86][145]. A very robust signal measurable from the EEG is the P300. It reflects a perceptual "orienting response" or shift of attention which can be driven by the content of the sensory input stream. Below we briefly describe the method we use to map the EEG into an "interest score" to be used for annotating the imagery. Given the RSVP paradigm for presenting a rapid sequence images to the subject, we simultaneously record EEG, using 64 scalp electrodes (Figure 6.4), and map the activity to an "interest score" for each image. The interest score is meant to reflect how much of a user's attention was directed toward an image. From a neuroscience perspective it can be seen as the single trial correlate of the P300-related orienting response. The algorithm we use to decode the EEG and ultimately map it to an interest score has been described previously [55]. Briefly, our approach begins with the linear model

$$z_t = \sum\nolimits_i \alpha_i s_{it}, \tag{6.1}$$

where s_{it} represents the electrical potential measured at time t for electrode i on the scalp surface, while α_i represents the spatial weights which will be learned based on a set of training data. The goal is to combine voltages in the electrodes linearly such that the sum z_t is maximally different between two conditions. The two conditions are "target of interest" vs "distracter". We also assume that this maximally discriminant activity is not constant but changes its spatial distribution within the second that follows the presentation of an image, thus we assume a stationarity time T of approximately

[9] The satellite images are provided by DigiGlobe.

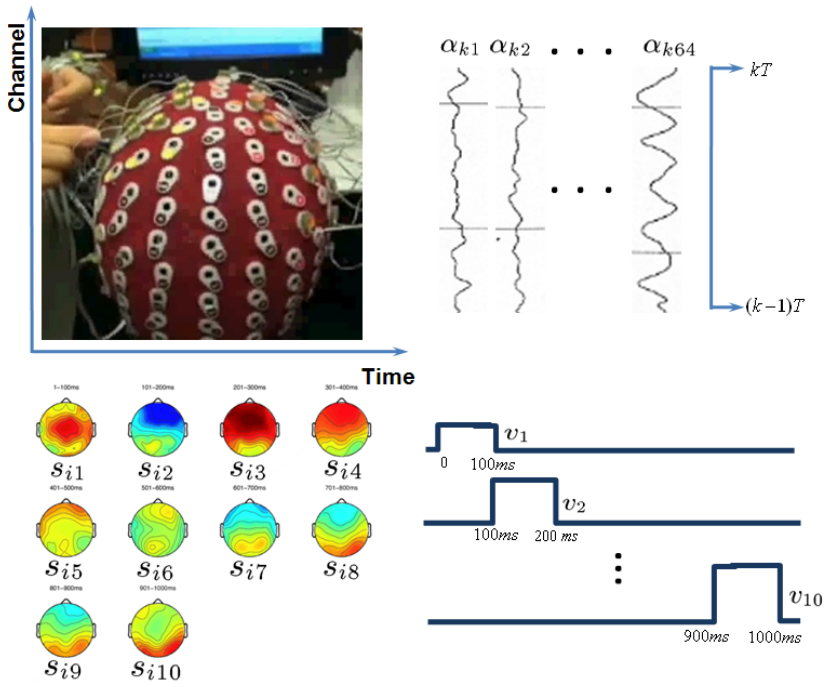


Figure 6.4: Demonstration of the EEG-based generic interest detector. The scalp surface shown is monitored with 64 electrodes. The bottom left color scalp maps show the spatial distribution of the recorded cortical signal at different time intervals. The right part shows the signal decoding procedure by Hierarchical Discriminant Component Analysis.

$100ms$. As a result, we find distinct optimal weight vectors, α_{ki} for each $100ms$ window following the presentation of the image (index k labels the time window):

$$z_{kt} = \sum_i \alpha_{ki} s_{it} \,, \quad t = (k-1)T \cdots kT. \tag{6.2}$$

These different z_{kt} are then combined in an average over time to provide the optimal discriminant activity over the entire second of data, with the result being our "interest score" e for the image as:

$$e = \sum_t \sum_k v_k z_{tk} \,. \tag{6.3}$$

For on-line implementation purposes we use the method of Fisher Linear Discriminants to train coefficients α_{ik} within each 100ms time window. The coefficients v_k are learned using penalized logistic regression after all exemplars have been observed. Because of the two step process of first

combining activity in space, and then again in time, the above EEG decoding method is termed Hierarchical Discriminant Component Analysis [116]. One advantage of such two-stage hierarchical modeling is the significant reduction of the number of model parameters that need to be learned - from about 10^5 to 10^4 (a 10 fold reduction). Colored scalp maps indicating spatial distribution of recorded cortical signal with different time interval are shown in Figure 6.4. It is important to confirm a strong correlate with the P300 attention orienting neural response, suggested by the neurological studies. Detectors built based on such single trial spatio-temporal EEG signal analysis have shown very promising results in various tasks such as people detection and image triage [55]. We will discuss the effectiveness of such a detector in detecting diverse objects such as those in Caltech 101 database in Section 6.5.1.

6.4 Visual Pattern Mining with Noisy EEG Labels

Assume that the generic interest detector outputs the EEG score $\mathbf{e} = \{e_1, e_2, \cdots, e_n\}$ from a RSVP sequence $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ shown to the subject [10]. Usually, the top ranked images based on scores $\mathbf{e}$ do not match the desired OOI due to the noisy nature of EEG signals in practice, as shown in Figure 6.6, 6.7, 6.8, and 6.12 (a). In Chapter 3, we showed that the existing semi-supervised methods cannot handle cases with extremely noisy labels. In order to refine the noisy EEG scores, we apply the bivariate graph transduction method to extract salient image patterns and eliminate mislabeled samples (by EEG interest detector) among the top ranked images. In other words, an improved interest measurement $\mathbf{f}$ is estimated using an image based representation and initial EEG scores as $\{\mathbf{X}, \mathbf{e}\} \rightarrow \mathbf{f}$. We formulate the following process of EEG label refinement and visual pattern mining.

1. Convert the image representation to a visual similarity graph $\mathbf{X} \rightarrow \mathcal{G} = \{\mathbf{V}, \mathbf{E}, \mathbf{W}\}$, where vertices $\mathbf{V}$ are the image samples $\mathbf{X}$ and the edges $\mathbf{E}$ with weights $\mathbf{W}$ measure the pairwise similarity of images.

2. Transfer the interest scores to pseudo EEG labels $\mathbf{e} = \{e_1, e_2, \cdots, e_n\} \rightarrow \mathbf{y} = \{y_1, y_2, \cdots, y_n\}$. In other words, a binarization function $g(\cdot)$ is applied to convert EEG scores to EEG labels as

[10] For an RSVP image sequence, the decoded EEG score vector $\mathbf{e} = \{e_1, e_2, \cdots, e_n\}$ is usually normalized as $e_i \in [0, 1], i = 1, \cdots, n$.

$\mathbf{y} = g(\mathbf{e})$, where $y_i \in \{1, 0\}$ and $y_i = 1$ for $e_i > \epsilon$, otherwise $y_i = 0$. The value ϵ is called interest level for discretizing the EEG scores.[11]

3. Apply the bivariate graph transduction framework to define the following risk function

$$\mathcal{Q}(\mathbf{f}, \mathbf{y}) = \mathcal{Q}_{fit}(\mathbf{f}, \mathbf{y}) + \gamma \mathcal{Q}_{\mathcal{G}}(\mathbf{f}), \tag{6.4}$$

which imposes the tradeoff between the smoothness measurement $\mathcal{Q}_{\mathcal{G}}(\mathbf{f})$ of function $\mathbf{f}$ over the undirected graph $\mathcal{G}$ and empirical error $\mathcal{Q}_{fit}(\mathbf{f}, \mathbf{y})$.

4. Alternatively minimize the above risk function with respect to $\mathbf{f}$ and $\mathbf{y}$ to finally achieve the optimal interest score $\mathbf{f}^*$:

$$\mathbf{f}^* = \arg\min_{\mathbf{f},\mathbf{y}} \mathcal{Q}(\mathbf{f}, \mathbf{y}). \tag{6.5}$$

In the following discussion, we follow the above procedure to detail our method for salient visual pattern mining with noisy EEG labels.

6.4.1 Image Features and Graph Construction

For the image feature extraction, we applied the widely used Bag-of-Visual-Words (*BoW*) derived from local key points, which has been shown to be effective in many applications of object and scene classification. In particular, we use the difference of Gaussian (DOG) and Harris-Affine as key point detector and SIFT as descriptor [100][105]. To weigh the importance of a visual word to an image, a soft-assignment strategy for computing the frequency of visual words is adopted [77]. With a constructed visual vocabulary (size is 5000), the sparse representation of *BoW* features for each image is extracted. The χ^2 distance is often used for the calculation of dissimilarity $\mathcal{D}_{ij}$ between histograms of *BoW* as in Eq (2.10). We have also used global features (texture and shape feature) and Euclidean distance for part of the experiments, such as satellite images. Such features have been shown to be efficacious in previous work [155]. Starting from the distance matrix $\mathcal{D}$, the graph construction $\mathcal{X} \rightarrow \mathcal{G} = \{\mathbf{V}, \mathbf{E}, \mathbf{W}\}$ is addressed in two steps, graph sparsification and edge weighting. As discussed in Chapter 2, sparsity is important to ensure that graph-based algorithms remain efficient and robust to noise. The most common algorithm for recovering a sparse

[11] In practice, the value of ϵ is set dynamically to achieve a fixed-number l of EEG positive labels, i.e. $\sum_i y_i = l$.

subgraph is the k nearest neighbors algorithm (kNN), where each vertex greedily connects with its k neighbors with the minimal distance. However, the kNN method typically produces asymmetric and irregular graphs, where the connectivity is uneven over different parts of the graph. This situation generates unreliable learning results if $\mathbf{X}$ contains very imbalanced class ratios, which occur very often in realistic image annotation settings, such as the 1.76% prior of target images in our Caltech 101 experiments. We have shown that b-matched graph is superior to kNN graph in terms of stability and accuracy for semi-supervised learning approaches [74]. Though the comparison between kNN and b-matched graph is not provided in this chapter, kNN graphs poorly performed in our experiments because the OOI is very infrequent in the tested image database.

With the vertex sparsified subgraph, the edge weights w_{ij} are estimated by applying heat kernel function on the χ^2 distance $\mathcal{D}_{ij}$ as: $w_{ij} = \exp(-\frac{\mathcal{D}_{ij}}{2\sigma^2})$. Realize the samples $\mathbf{X}$ might be draw unevenly, here we re-weight the similarity measure using an adaptive kernel size σ as suggested in Chapter 2.

6.4.2 Graph-based Visual Pattern Mining

Given the constructed b-matched graph with edge weight $\mathbf{W}$ and the vertex degree matrix $\mathbf{D}$, the normalized graph Laplacian is computed as $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. Starting from the regularization framework in Eq. (6.4), we formulate the following cost function:

$$E(\mathbf{f}, \mathbf{y}) = \|\mathbf{f} - \mathbf{y}\|^2 + \gamma \|\mathbf{f}\|_{\mathcal{G}}^2, \tag{6.6}$$

where $\|\mathbf{f} - \mathbf{y}\|^{\mathbf{2}}$ is the empirical square loss with pseudo EEG label $\mathbf{y} = g(\mathbf{e})$. The semi-inner product $\|\mathbf{f}\|_{\mathcal{G}}$ measures the function smoothness over the graph $\mathcal{G}$, which reflects the visual consistency:

$$\|\mathbf{f}\|_{\mathcal{G}}^{\mathbf{2}} = \langle \mathbf{f}, \mathbf{f} \rangle = \mathbf{f}^\top \, \mathbf{L} \, \mathbf{f}. \tag{6.7}$$

Note that the above problem is formed in a bivariate cost function and the empirical risk is estimated using unreliable pseudo EEG labels. Similar to the the alternating optimization procedure developed in Chapter 3, we derive partial differentials with respect to $\mathbf{y}$ and $\mathbf{f}$, respectively, and iteratively update the function values to refine EEG labels. Since $\mathbf{f} \in \mathbb{R}^n$ is a continuous valued function, the optimal one can be derived by zeroing the partial differential $\nabla_{\mathbf{f}} E$:

$$\begin{aligned} \frac{\partial E}{\partial \mathbf{f}} &= \frac{\partial \mathcal{Q}(\mathbf{f}, \mathbf{y})}{\partial \mathbf{f}} + \gamma \frac{\partial \mathcal{V}(\mathbf{f})}{\partial \mathbf{f}} = 2(\mathbf{f} - \mathbf{y}) + 2\gamma \mathbf{L}\mathbf{f} = 0 \\ &\Rightarrow \quad \mathbf{f}^* = (\mathbf{I} + \gamma \mathbf{L})^{-1} \mathbf{y} = \mathbf{p}\mathbf{y}, \end{aligned} \tag{6.8}$$

Algorithm 6.1 EEG label refinement and visual pattern mining

Input: initial EEG scores $\mathbf{e}$;

Graph $\mathcal{G}$ and normalized graph Laplacian $\mathbf{L}$;

Constant matrix: $\mathbf{p} = (\mathbf{I} + \gamma\mathbf{L})^{-1}$.

Initialization:

The number of pseudo EEG labels l;

Iteration number $M = l/2$;

Initialize function $\mathbf{f}$ with EEG scores $\mathbf{f}^0 = \mathbf{e}$.

Loop: $\tau = 0, \cdots, M$

Convert EEG scores to labels:

$\mathbf{y}^\tau = g(\mathbf{f}^\tau)$, satisfying $\sum_i y_i^\tau = l$;

Compute gradient:

$\nabla_{\mathbf{y}^\tau} E = \left(\|\mathbf{p} - \mathbf{I}\|^2 + \gamma \mathbf{p}^\top \mathbf{L}\mathbf{p}\right) \mathbf{y}^\tau$;

Update EEG label with truncated gradient:

$\mathbf{y}^{\tau+1} = \mathbf{y}^\tau - T(\nabla_{\mathbf{y}^\tau} E)$;

Recalculate interest level function:

$\mathbf{f}^{\tau+1} = \mathbf{p}\mathbf{y}^{\tau+1}$;

Output: the refined EEG scores $\mathbf{f}^*$.

where $\mathbf{p} = (\mathbf{I} + \gamma\mathbf{L})^{-1}$ is a constant matrix in $\mathbb{R}^{n\times n}$ and $\mathbf{I}$ is identity matrix. Replace function $\mathbf{f}$ in Equation 6.6 by the optimal one $\mathbf{f}^*$ and rewrite the risk function as:

$$E(\mathbf{y}) = \|\mathbf{p} - \mathbf{I}\|^2 \|\mathbf{y}\|^2 + \gamma \mathbf{y}^\top \mathbf{p}^\top \mathbf{L}\mathbf{p}\mathbf{y}. \tag{6.9}$$

Derive the partial differential $\nabla_{\mathbf{y}} E$ and ignore the constant coefficient:

$$\frac{\partial E}{\partial \mathbf{y}} \propto \left(\|\mathbf{p} - \mathbf{I}\|^2 + \gamma \mathbf{p}^\top \mathbf{L}\mathbf{p}\right) \mathbf{y}. \tag{6.10}$$

Note that $\mathbf{y} \in \mathbb{B}^n$ is a binary vector representing class labels. The conventional approaches, such as zeroing $\nabla_{\mathbf{y}} E$ or standard stochastic gradient is not appropriate for minimizing E with respect to

the binary-valued variable $\mathbf{y}$. Here, we truncate the gradient $\nabla_{\mathbf{y}} E$ and discretize it to $T(\nabla_{\mathbf{y}} E)$:

$$T(\nabla_{y_i} E) = \begin{cases} 1 & : \quad \nabla_{y_i} E = \max(\nabla_{\mathbf{y}_l} E) \\ -1 & : \quad \nabla_{y_i} E = \min(\nabla_{\mathbf{y}_u} E) \\ 0 & : \quad otherwise, \end{cases} \tag{6.11}$$

where $\mathbf{y}_l, \mathbf{y}_u$ are the labeled and unlabeled parts of the label variable $\mathbf{y}$. Then the variable $\mathbf{y}$ can be updated with this truncated stochastic gradient $T(\nabla_{\mathbf{y}} E)$:

$$\mathbf{y} \leftarrow \mathbf{y} - T(\nabla_{\mathbf{y}} E). \tag{6.12}$$

Intuitively, the above updating by truncated gradient descent will remove one unreliable EEG label, and meanwhile choose the most suitable one from the remaining data as a new EEG label. Through iteratively repeating this truncated gradient descent updating, the EEG label set will be gradually refined to derive visually consistent visual pattern from the top ranked image list. Note that the gradient truncation approach is different with the method developed in [94], where the truncated gradient is applied to induce sparsity in the continuous-valued weights for online learning. Algorithm chart 6.1 presents the summary of the proposed method for EEG label refinement and visual pattern mining. Compared with the bivariate framework developed in Chapter 3, here we apply the following modifications. First, the BCI based image retrieval application is formulated as a ranking problem, instead of the standard classification problem in our previous bivariate framework. Second, the initial noisy labels are given in the form of continuous scores acquired from the EEG interested detector, instead of the binary-valued label information. Both these two modifications are critical in adapting the the approach to the hybrid neuro-computer vision system.

6.5 Experiments

We tested the developed BCI-VPM annotation system on the image data from various domains, including Internet image collections (i.e. Caltech 101) and satellite imagery (DigiGlobe images), to show the scalability and generalization. The detailed experimental setting and performance evaluation are reported below. The experimental scenario is that a user is instructed to look for a certain OOI in each presentation of an RSVP image sequence. The BCI interest detector generates probability based EEG confidence scores, which measure the relevance of the presented image to the

Figure 6.5: The summary of the experimental results on Caltech 101 dataset. The performance is evaluated in terms of average precision (AP). A total of four subjects and three OOI are tested (12 trials of RSVP presentations). The APs of random sequence, EEG detector and BCI-VPM refinement are recorded. The yellow color table cells highlight the significant improved trials (8 out of 12).

instructed OOI. The estimated EEG score are then fed into the subsystem of visual pattern mining for refinement and propagation. During these tests, EEG data was recorded at 2048 Hz using a 64-electrode EEG recording system (Biosemi, BrainProduct, Germany) in a standard (10-20) setting.

6.5.1 Caltech 101 Object Annotation

Data filtering: Caltech 101 is a very challenging dataset for EEG decoding because it contains fairly common and diverse object categories with large intra class variations. Moreover, images greatly vary in both resolutions and scales. This can significantly affect the user's detection performance during the EEG signal decoding. To design a practical set of intial EEG experiments, we first filter the object categories by selecting 62 categories that provide 3798 images that have similar scales and resolutions. When displayed during the RSVP, these images are re-scaled to a size of 240×240 to achieve the desired uniformity in view.

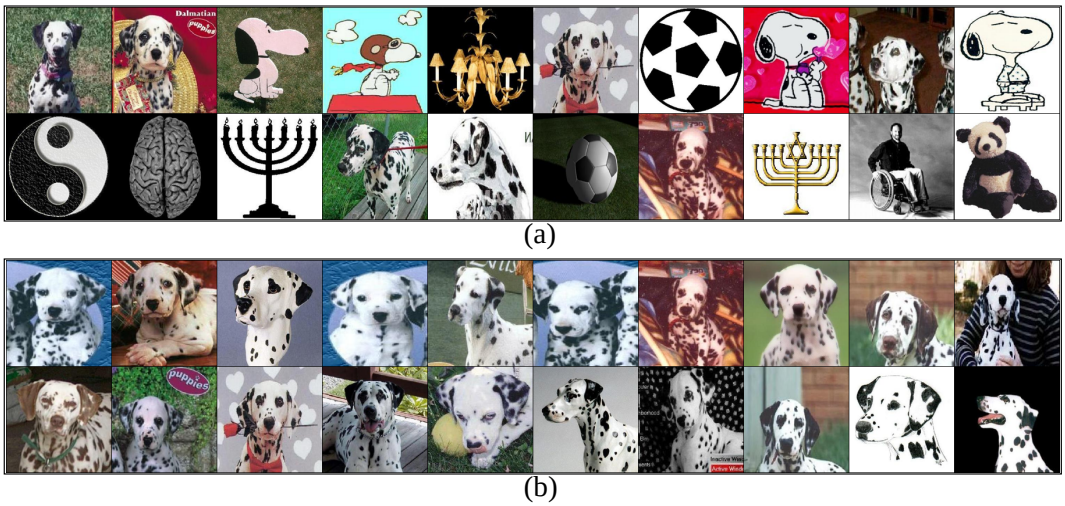


Figure 6.6: The experimental results (top 20 images) of the trial from Subject A on Caltech 101 RSVP with OOI as *Dalmatian*. a) ranking by interest scores from EEG detector; b) ranking by scores after label refinement.

Experimental scenario: As it is impractical to require a user to perceive 62 OOI simultaneously, we also narrowed th choice of target categories that a user has to detect to one specific target each time. Specifically, users were instructed perceive *Starfish*, *Dalmatian*, and *Chandelier/Menorah* [12] for each pass of RSVP (with the target order being varied between subjects). Notice that this simplification still did not reduce much of the challenge due to the diverse object categories and sparse targets. For example, "*Starfish*" and "*Dalmatian*" only account for 2.26% and 1.76% of the data set, respectively. For the BCI interest detector training, we use two popular objects, *Soccer Ball* or *Baseball Gloves* as OOI to train the EEG-based detector. Most subjects are familiar with such objects without the need of special instruction and thus they serve as adequate common patterns that can catch user's attention. The Caltech 256 database was used to select training images to differentiate between the training and testing images.

Image database down sampling: Furthermore, in order to show the scalability of the proposed method, we randomly partition the 62-object database into two parts $\mathcal{X} = \{\mathcal{X}_s, \mathcal{X}_u\}$. The subset

[12]The experiments were initially designed to annotate the object class *Chandelier*. Realizing the ambiguity between *Chandelier* and *Menorah* due to visual similarity, we decided to treat the samples from these two object categories as the same class in these experiments.

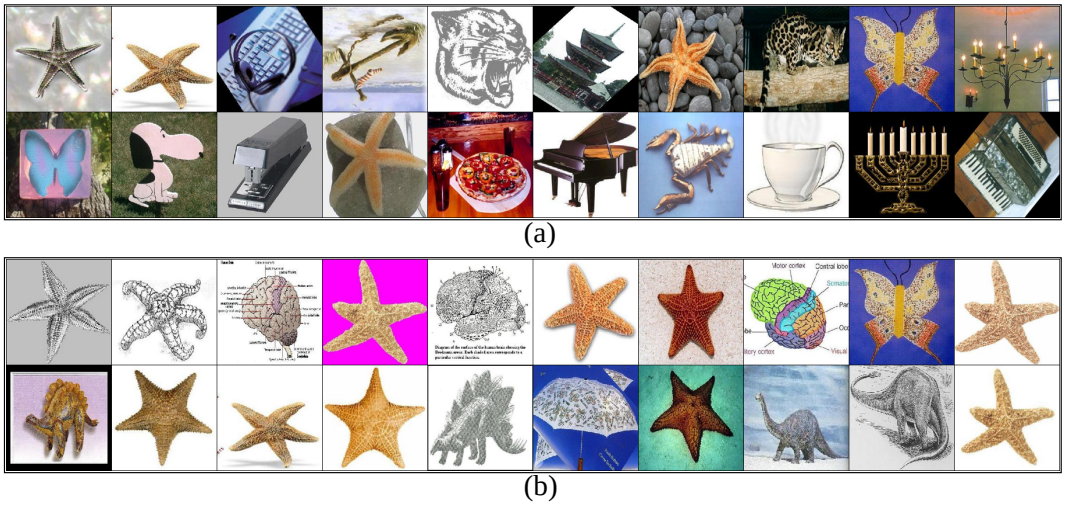


Figure 6.7: The experimental results (top 20 images) of the trial from Subject B on Caltech 101 sequence with OOI as *Starfish*. a) ranking by interest scores from EEG detector; b) ranking by scores after label refinement.

$\mathcal{X}_s$ containing 1000 images is randomly ordered to form a RSVP image sequence shown to all test subjects. The RSVP image sequences were shown in 10 blocks of 100 images each, with images shown at 6Hz within each block. The other subset $\mathcal{X}_u$ containing a total of 2798 images is treated as EEG-unlabeled samples in later label refinement stage by setting their EEG scores as $\mathbf{e}_u = \vec{\mathbf{0}}$. This strategy shows that with only partial images processed by the BCI detector, the proposed BCI-VPM system can extend the usage of the BCI annotation to the images that are not processed by BCI interest detector. This merit is extremely critical because it provides the capability and scalability to annotating a large collection of images.

Results: Four subjects are drawn from undergraduate and graduate students, staff and faculty that were not digital media analysts, but were familiar with EEG work. These subjects participated in the experiments and were instructed to identify three object classes from RSVP presentations. A total of 12 trials of RSVP presentations are evaluated in terms of average precision (AP), as shown in Figure 6.5. AP is a performance metric commonly used in information retrieval [140]. It approximates the area under the precision-recall curve. The experiments show promising results. For example, the BCI detector achieved 33.73% and BCI-VPM label refinement further improved to 69.1% for subject A annotating *"Dalmatian"*. Among the 12 trials of tests, 8 trials achieved

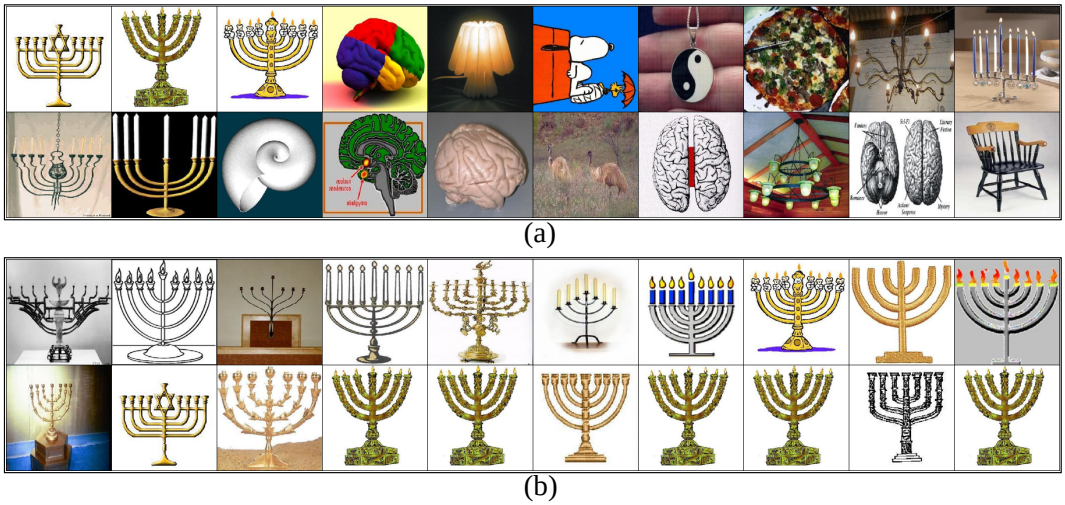


Figure 6.8: The experimental results (top 20 images) of the trial from Subject C on Caltech 101 sequence with OOI as *Chandelier/Menorah*. a) ranking by interest scores from EEG detector; b) ranking by scores after label refinement.

significant performance improvement. Even for some tough cases, where the BCI detector only obtained less than 10% AP, the label refinement process still significantly bootstraps the annotation accuracy. Figure 6.6, 6.7, and 6.8 showed the top 20 images from the BCI detector and BCI-VPM label refinement from three different test subjects, where significant precision gain can be observed. In Figure 6.9, we analyze the precision-recall(PR) curves of one of the successful case (Subject A annotating "*Dalmatian*"), which further confirms the efficiency of the proposed BCI-VPM annotation system.

However, there are some cases, where possibly due to users' misunderstanding of the OOI or some uncontrolled distractions, the BCI detector generated very poor performance, typically less than 7% AP. In those cases, the top ranked images do not contain a majority consistent pattern. Therefore, the subsequent label refinement process is unable to extract the salient visual patterns.

To analyze the sensitivity of the combined BCI-VPM accuracy to the quality of the front end BCI detection precision, we further evaluate the effectiveness of the BCI-VPM system with a varying number of true positive samples contained in the top images (e.g., 20) of the initial EEG-based ranking. The positive images are randomly drawn from the target category and the negative images are randomly drawn from the database. Average performances of 200 random runs per category

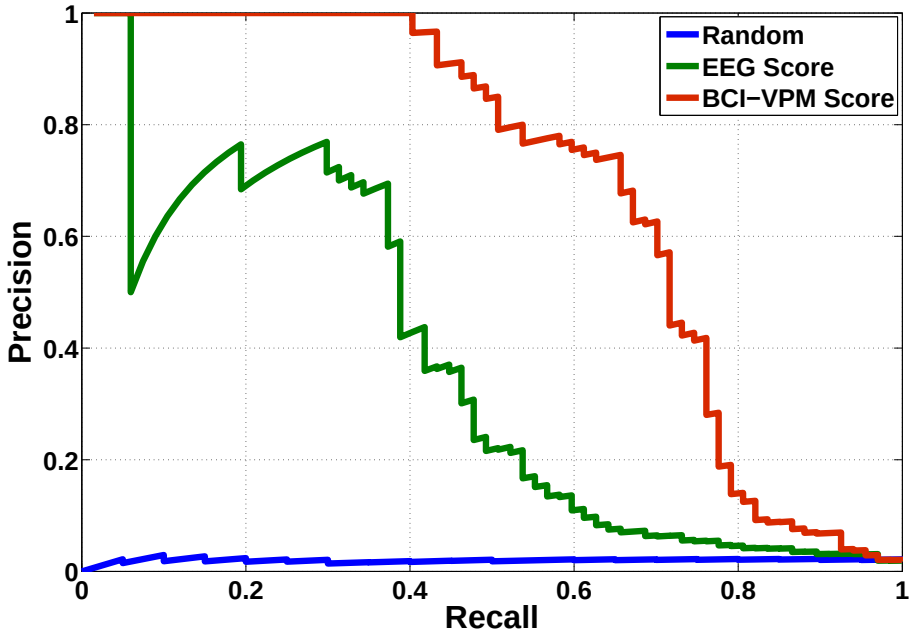


Figure 6.9: The performance evaluation on Caltech101 image sequence by Precision-Recall (PR) curve (the trial of Subject A annotating "*Dalmatian*").

and the average results among the three targets are shown in Figure 6.10. The results confirm that the combined BCI-VPM approach can effectively improve the precision by using the EEG detector alone. They also confirm the monotonic relationship between the final accuracy and the initial EEG detection accuracy. For example, the average top-20 precision is improved from 20% to 35% and from 30% to about 55%. Such performance curves can be used to determine the required accuracy for the initial EEG component given a target detection performance for the final combined system. The results can also be used to roughly measure the visual pattern complexity and the difficulty in re-ranking the three targets - *Starfish* is more difficult than *Chandelier/Menorah*, which is in turn more difficult than *Dalmatian*. Such ordering matches the intuitive expectation of the relative complexities of the object classes.

Subject	Method	AP-30	AP-60	AP-100	AP-ALL
A	EEG score	19.76	15.83	14.9	30.97
	BCI-VPM score	**50.19**	**32.65**	**25.46**	**37.89**
B	EEG score	8.76	9.79	9.56	23.71
	BCI-VPM score	**87.31**	**63.29**	**46.07**	**57.41**
C	EEG score	12.70	19.58	16.54	29.62
	BCI-VPM score	**90.82**	**63.30**	**41.21**	**53.66**
D	EEG score	10.68	11.87	11.75	24.45
	BCI-VPM score	**91.87**	**60.62**	**40.24**	**52.70**

Table 6.1: The performance comparison of annotation performance of EEG interest detector and the BCI-VPM refined EEG score in terms of average precision of top 30, 60, 100 ranked images and entire satellite image dataset (the number of pseudo EEG labels $l = 30$).

6.5.2 Target Annotation in Satellite Imagery

The other type of RSVP image sequence shown to test subjects consists of blocks of chipped images taken from satellite imagery with each chip potentially containing a target, as shown in Figure 6.3 (a). Among the 1051 samples in the RSVP sequence of satellite imagery, 105 images have a target "*helipad*" centered. This sequence is displayed to the subjects with a speed of 10 Hz, i.e. 10 images per second. A total of 4 subjects were tested with single trial of presentation of RSVP. The performance is evaluated in terms of average precision of the top 30, 60, and 100 and entire dataset of the BCI interest detector and the final refined results, as shown in Table 6.1. Compared with the Caltech 101 experiments, Table 6.1 shows much more consistent performance gain for all trials. The reason lies in that the targets of "*helipad*" have salient visual clue of "H" symbol, which easily attracts users' attention. In addition, the non-target image blocks are very noisy and mostly unmeaningful, which reduces level of distraction.

Since the value of l is applied for truncating the EEG scores to create initial binary labels, it is necessary to evaluate the performance using different l. We vary the value of l from 20 to 50

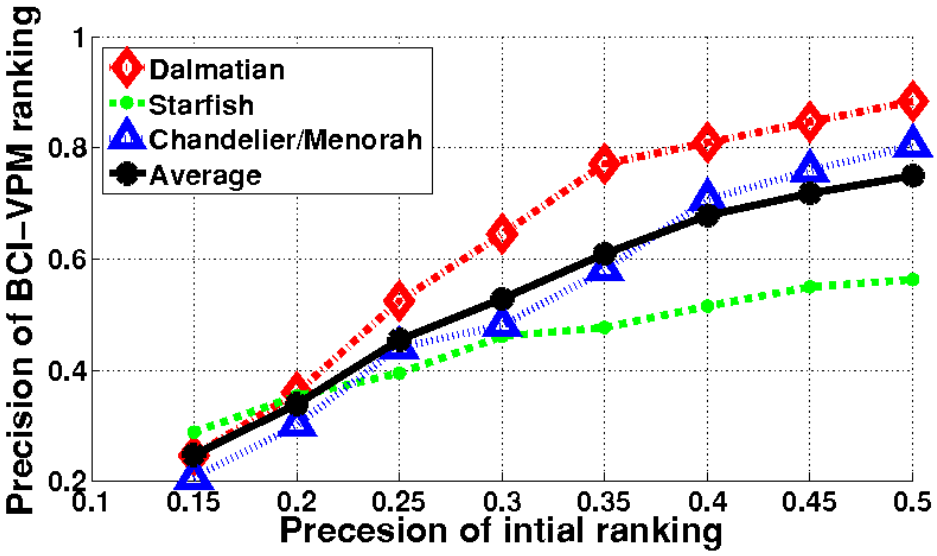


Figure 6.10: Simulated evaluation of the dependency of the BCI-VPM re-ranking performance (in terms of top-20 precision) on the performance of the initial EEG detection. Individual curves for different classes (*Dalmatian*, *Chandelier/Menorah*, *Starfish*) and the average results across three categories are shown.

and evaluate the performance in terms of mean average precision (MAP) (averaging among four subjects). As shown in Figure 6.11, a fairly stable performance range is with the value of l in $[30, 50]$. As an illustration, Figure 6.12 shows the test results from Subject C. The sub-figure (a) gives the top 20 image blocks by ranking the original EEG interest scores $\mathbf{e}$ and the sub-figure (b) shows the top 20 images from the ranking of the refined interest score $\mathbf{f}$.

6.6 Comparison with Prior Works and Unique Contribution

Despite the growing interest in BCI, few works can be found in using BCI for image annotation and search. We summarize the ideas of the prior works and point out the unique contributions of the work presented in this chapter.

The pioneer system (called Cortically-Couple Computer Vision, C3Vision) using EEG-based neural measurement in image target detection was first reported in [55], and further elaborated in [126]. Its RSVP visual presentation paradigm and spatio-temporal discriminant analysis ap-

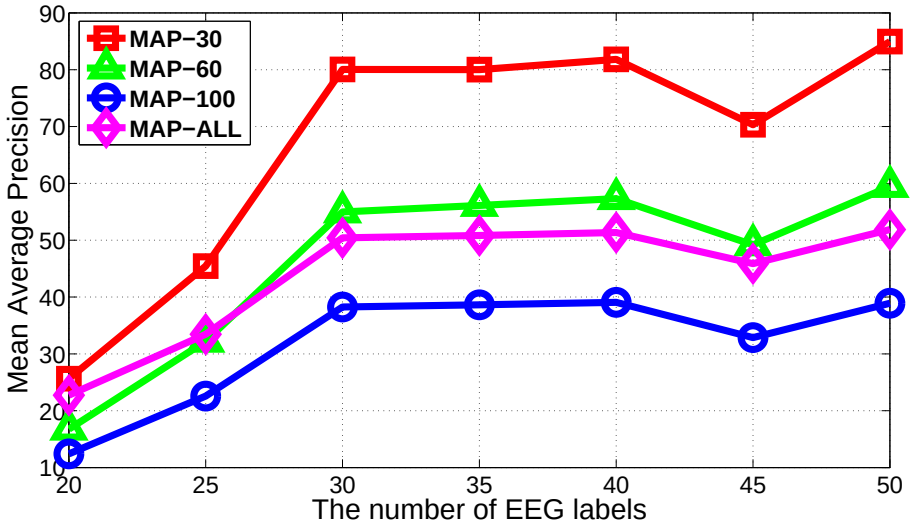


Figure 6.11: The Mean Average Precision of top 30, 60, 100, and entire satellite image set using different numbers of EEG scores as initial labels.

proaches are extended to develop the interest detector in this chapter. Compared to the current work, it focused on less diverse object classes (e.g., people vs. background) and had not been combined with the pattern discovery subsystem for label refinement and propagation.

Recently, a supervised learning approach (called Human Aided Computing, HAC) was proposed in [133] to develop an EEG-based classifier for recognition of distinct objects in images, such as face vs. no-face, or animals vs. inanimate objects. The proposed method showed performance improvement when neural response measurements from multiple trials (i.e., same images presented multiple times) and/or multiple subjects were combined. However, the technique is limited due to the requirement of predefined object classes and ground truth labels for training the object detectors. In contrast, our work focuses on robust detection using only single trial EEG signals and scalability to detection of arbitrary object classes. The only training session is done offline only once per subject using an object class that is independent of the test targets.

Computer vision component was added to the HAC method (HAC-CV) in [81] to fuse the EEG signals and the image features in the same target classifier. Multiple trials were used and improved performance was reported compared to detection using EEG signals alone. Again, the method is restricted as target classes are predefined, labeled, and trained in a supervised fashion.

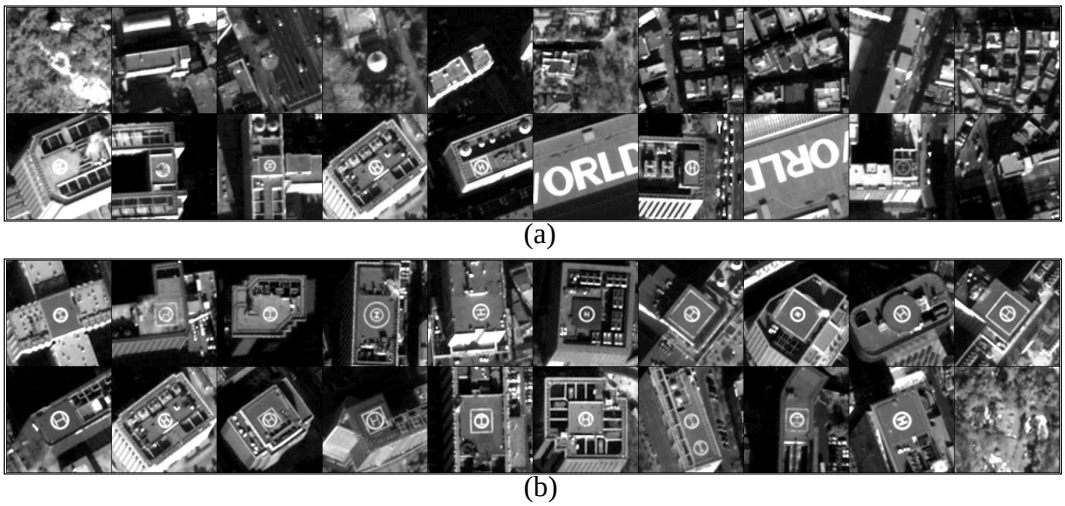


Figure 6.12: The experimental results of Subject C on "*helipad*" target RSVP, showing the top 20 ranked images . a) ranking by original EEG scores; b) ranking by the BCI-VPM refined interest score.

Other works have also explored the use of fMRI imaging to decode brain state [107][83] [108]. Such approaches enjoy a higher spatial resolution at the cost of lower temporal throughput. In our work, we focus on the system utilizing a non-invasive continuous recording BCI framework based on EEG. Table 6.2 lists the comparison between our proposed method and the prior works discussed above. The key features and unique contributions of the proposed system include:

. **Generality**: No predefined target classes and annotations are needed.

. **Robustness**: The combination of BCI and pattern discovery results in greatly improved accuracy in generic object detection.

. **High-throughput**: Only a small subset of images need to be viewed by the subject via a single trial setting. The results are automatically propagated over to the rest of images in a large collection.

In addition, our proposed system can handle rare target classes with a prior as low as 1.8% (as demonstrated in Section 6.5.1) at a high speed (5-10 images per second).

System	**C3Vision**	**HAC**	**HAC-CV**	**BCI-VPM**
System Structure	pure BCI	pure BCI	hybrid BCI+CV	hybrid BCI+CV
EEG Trials	single trial single subject	single/multiple trials single/multiple subjects	multiple trials single/multiple subjects	single trial single subject
Object Class	people vs. background	face vs. animal animal vs. inanimate	face, animal, and inanimate	general object
Target Frequency	2%	25%	50%	$\sim 2\%$
Manual Labels	No	Yes	Yes	No
RSVP Speed	$5-10$ HZ	$1-2$HZ	$1-2$HZ	$5-10$HZ
Learning Method	unsupervised	supervised	supervised	unsupervised

Table 6.2: Comparison of the existing BCI-based image analysis system, including **C3Vision** [55], **HAC** [133], **HAC-CV** [81], and our proposed **BCI-VPM** image annotation system.

6.7 Summary and Discussion

In this chapter, we propose an integrated system that combines neural signal processing for brain state decoding and content analytics for improving the performance of image retrieval systems. The brain state decoding subsystem utilizes advanced spatio-temporal analysis of neural response signals measured by EEG in a single trial setting. The content analytics component implements the graph-based diffusion method (as described in earlier chapters) that is capable of handling rare targets, small label size, and noisy conditions. The former has unique power in recognizing generic target classes, while the latter is suitable for high-throughput processing. The proposed system explores the synergy of these two subsystems and has shown promising performance in detecting generic target classes in a high throughput fashion.

Several aspects of the system merit further investigation. First, in the current setting, the subjects are instructed to look for certain object classes during the image viewing session. It will be interesting to relax this and allow subjects to "lock in" on any object class fitting his/her interest on the fly without any instructions. This may result in greater ambiguity in users' expectation and

perception of targets and less consistent target-specific neural responses. Second, the output of the graph-based pattern discovery can be used to further assess the quality of the EEG interest detector and provide a closed-loop feedback mechanism to incrementally update the detector. The re-ranked images could also be used to adjust the order of image presentation in the RSVP stream in order to improve sensitivity/specificity of the neural response signals. Prediction results from the updated neural signal decoder can be used as input to start the graph-based visual pattern mining process again, forming an iterative loop for continuously enhancing the image target retrieval performance. Such iterative closed-loop system also offers an excellent framework allowing the study of the interesting issues involved in the co-learning processing between the human vision, the EEG neural decoder, and the machine learning for image target detection.

Chapter 7

Conclusions

7.1 Summary and Contributions

This thesis is dedicated to developing advanced semi-supervised learning techniques for the applications of visual search and retrieval. We focus on two major techniques: bivariate graph transduction and semi-supervised hashing learning. Both of these methods can be either used for general semi-supervised learning tasks or incorporated in practical visual search systems. In particular, bivariate graph transduction aims at improving search accuracy and providing a suitable framework for incorporating user feedback, while semi-supervised hashing generates efficient and compact hash codes for searching large-scale image and video databases.

The main contributions of this thesis include both algorithm design and practical system development. Specifically, we developed the following machine learning algorithms:

a. **Bivariate Graph Transduction**: We developed a novel bivariate regularization framework for graph-based semi-supervised learning. Such bivariate formulation allows us to treat input labels as part of the optimization variables and thereby alleviate the issue of label sensitivity. Theoretically, we proved that this new formulation is equivalent to a constrained Max-Cut problem. In addition, we designed an efficient solution by alternatively minimizing the cost function with respect to two variables, i.e. labels and prediction functions, which can be regarded as a greedy gradient-based Max-Cut solution from the viewpoint of graph cut. Finally, two advanced extensions, label tuning algorithm and multi-graph transduction, were developed to achieve even higher performance in real applications.

b. **Semi-Supervised Hashing**: Targeting general approximate nearest neighbor search problems, we proposed a semi-supervised paradigm to learn efficient hash codes which exploit information of semantic similarity/dissimilarity among the data points. We showed this hashing technique is particularly useful for searching large-scale visual databases since it can help overcome the well-known *semantic gap* by leveraging the given information about semantic consistency among a small set of images. Starting from this semi-supervised hashing paradigm, we developed three different methods for generating compact and robust hash codes, and an extension for producing hashing codes in an unsupervised manner.

To handle real applications, we applied the bivariate graph transduction algorithm and developed the following image search systems:

1. **Interactive Image Search**: A general system for interactive image search, named Columbia TAG (Transductive Annotation by Graph) system, was developed to provide a suitable framework for keeping user annotation in loop and incorporates relevance feedback. The empirical study demonstrated that this interactive search mode can significantly reduce the workload of annotation and achieve satisfactory performance with only a few user labels. It can be used as a fast search system alone, or as a bootstrapping system for developing additional target recognition tools needed in critical image application domains. For instance, we have extended the TAG system to microscopic image analysis, which shows promising performance in rapidly discovering different cellular phenotypes.

2. **Automatic Image Search**: We apply the bivariate framework to design an automatic mode for improving web image search. The top returned search results are treated as labeled positive samples with potential labeling errors. Then the label tuning algorithm, i.e., *LDST*, is applied to refine labeled set and propagate label information over the entire image collection. The experimental results over a collection of *Flickr* images confirm the significant performance gain over both text search baselines and other search re-ranking methods.

3. **Hybrid Image Search**: A unique image search platform was developed through incorporating the proposed graph transduction approach with an Electroencephalogram (EEG) based brain computer interface (BCI). This hybrid paradigm marries the strengths of human vision and machine learning systems in a unique and synergistic way. Extensive experiments over

satellite imagery and Web images demonstrate the promising results for detecting generic targets in a high throughput fashion. To the best of our knowledge, this hybrid system is the first solution for combining human vision and computer vision systems in retrieving image targets of arbitrary classes.

7.2 Future Work

Despite the exciting results shown in this thesis, there are many open issues associated with the two developed methods, i.e. bivariate graph transduction and semi-supervised hashing. In the following, we discuss a few topics for future work.

1. **Semi-supervised graph construction**: Most work on *GSSL* has focused on label prediction, and efforts on graph construction have been relatively limited [74][98], as mentioned in Chapter 2. Most of the current graph construction methods rely on certain heuristic criteria under an unsupervised setting. It will be desirable to combine the available label information associated with the given data to design proper graphs in a semi-supervised fashion, similar to the way they are used to optimize the label predictions.

2. **Kernelized semi-supervised hashing**: Kernel methods haven been proved critical for general learning problems, such as support vector machines. Previous work demonstrates that investigating the power of kernel learning can help improve the performance of locality sensitive hashing [63][72][90]. It is no doubt that introducing kernel methods into the proposed semi-supervised hashing paradigm can increase the capabilities of ANN search beyond the Euclidian space.

3. **Combining hashing and graph-based learning**: Hashing and graph-based learning are complementary in terms of scalability and accuracy. Applying hashing techniques can help rapid construction of large-scale sparse graphs. Efforts for large-scale semi-supervised learning have started to emerge recently [49][99]. The joint problem of hashing and graph-based learning remains an exciting, yet unexploited, direction for scaling up semi-supervised classification methods.

Part I

Bibliography

Bibliography

[1] D. Aldous and J. Fill. Reversible Markov chains and random walks on graphs, 2002.

[2] R. K. Ando and T. Zhang. Two-view feature generation model for semi-supervised learning. In Z. Ghahramani, editor, *Proceedings of the 24th international conference on Machine learning*, pages 25–32, Corvalis, Oregon, June 2007. ACM.

[3] A. Argyriou, M. Herbster, and M. Pontil. Combining graph laplacians for semi–supervised learning. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 67–74. MIT Press, Cambridge, MA, 2006.

[4] A. Argyriou, C. Micchelli, and M. Pontil. Learning convex combinations of continuously parameterized basic kernels. In *Proceedings of the 18th conference on learning theory*, pages 338–352. Springer, 2005.

[5] A. Azran. The rendezvous algorithm: Multiclass semi-supervised learning with markov random walks. In Z. Ghahramani, editor, *Proceedings of the International Conference on Machine learning*, pages 49–56, Corvalis, Oregon, USA, June 2007.

[6] F. R. Bach and M. I. Jordan. Learning spectral clustering, with application to speech separation. *The Journal of Machine Learning Research*, 7:1963–2001, 2006.

[7] C. Bakal, J. Aach, G. Church, and N. Perrimon. Quantitative Morphological Signatures Define Local Signaling Networks Regulating Cell Morphology. *Science*, 316(5832):1753, 2007.

[8] M.-F. Balcan, A. Blum, and K. Yang. Co-training and expansion: Towards bridging theory and practice. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 89–96. MIT Press, Cambridge, MA, 2005.

[9] S. Baluja and M. Covell. Learning to hash: forgiving hash functions and applications. *Data Mining and Knowledge Discovery*, 17(3):402–430, 2008.

[10] F. Barahona, M. Grötschel, M. Jünger, and G. Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36(3):493–513, 1988.

[11] S. Basu, I. Davidson, and K. Wagstaff. *Constrained clustering: Advances in algorithms, theory, and applications*. Chapman & Hall/CRC, 2008.

[12] M. Bawa, T. Condie, and P. Ganesan. LSH forest: self-tuning indexes for similarity search. In *Proceedings of the 14th international conference on World Wide Web*, pages 651–660, Chiba, Japan, May 2005.

[13] M. Bayati, D. Shah, and M. Sharma. Maximum weight matching via max-product belief propagation. In *Int. Symp. on Information Theory*, pages 1763–1767, 2005.

[14] M. Belkin, P. Niyogi, and V. Sindhwani. On manifold regularization. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, Barbados, January 2005.

[15] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[16] R. Bellman. *Dynamic programming*. Rand Corporation research study. Princeton University Press, 1957.

[17] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[18] T. D. Bie and N. Cristianini. Convex methods for transduction. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[19] N. Bigdely-Shamlo, A. Vankov, R. Ramirez, and S. Makeig. Brain Activity-Based Image Classification From Rapid Serial Visual Presentation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 16(5):432–441, Oct. 2008.

[20] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of International Conference on Machine Learning*, pages 19–26, San Francisco, CA, USA, 2001.

[21] A. Blum, J. Lafferty, M. R. Rwebangira, and R. Reddy. Semi-supervised learning using randomized mincuts. In *Proceedings of the twenty-first international conference on Machine learning*, pages 13–20, New York, NY, USA, 2004.

[22] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100, Madison, Wisconsin, United States, 1998. ACM.

[23] C. E. Brodley and M. A. Friedl. Identifying and eliminating mislabeled training instances. In *Proc. the Thirteenth National Conference on Artificial Intelligence*, pages 799–805, 1996.

[24] C. E. Brodley and M. A. Friedl. Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, 11:131–167, 1999.

[25] M. A. Carreira-Perpiñán and R. S. Zemel. Proximity graphs for clustering and manifold learning. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 225–232. MIT Press, Cambridge, MA, 2005.

[26] L. Cayton and S. Dasgupta. A learning framework for nearest neighbor search. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 233–240. MIT Press, Cambridge, MA, 2008.

[27] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

[28] O. Chapelle, V. Sindhwani, and S. Keerthi. Optimization techniques for semi-supervised support vector machines. *The Journal of Machine Learning Research*, 9:203–233, 2008.

[29] O. Chapelle, V. Sindhwani, and S. S. Keerthi. Branch and bound for semi-supervised support vector machines. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 217–224. MIT Press, Cambridge, MA, 2007.

[30] O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, Barbados, January 2005.

[31] N. V. Chawla and G. Karakoulas. Learning from labeled and unlabeled data: An empirical study across techniques and domains. *Journal of Artificial Intelligence Research*, 23(1):331–366, 2005.

[32] Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng. On evolutionary spectral clustering. *ACM Transactions on Knowledge Discovery from Data*, 3(4), 2009.

[33] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng. Nus-wide: A real-world web image database from national university of singapore. In *Proc. of ACM Conf. on Image and Video Retrieval*, Santorini, Greece., July 2009.

[34] F. Chung and N. Biggs. *Spectral graph theory*. American Mathematical Society Providence, RI, 1997.

[35] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[36] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158, Shaker Heights, Ohio, United States, 1971. ACM.

[37] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual Symposium on Computational Geometry*, pages 253–262, 2004.

[38] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):1–60, 2008.

[39] M. M. Deza and M. Laurent. *Geometry of cuts and metrics*. Springer Verlag, 2009.

[40] W. Donath and A. Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5):420–425, 1973.

[41] J. Donoghue. Bridging the brain to the world: a perspective on neural interface systems. *Neuron*, 60(3):511–521, 2008.

[42] M. Dyrholm, C. Christoforou, and L. Parra. Bilinear discriminant component analysis. *Journal of Machine Learning Research*, 8:1097–1111, 2007.

[43] J. Edmonds. Paths, trees and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[44] J. Edmonds and E. Johnson. Matching: A well-solved class of integer linear programs. *Combinatorial OptimizationEureka, You Shrink!*, pages 27–30, 2003.

[45] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html.

[46] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[47] R. Fergus, F. F. Li, P. Perona, and A. Zisserman. Learning object categories from google's image search. In *Proc. of Tenth IEEE International Conference on Computer Vision*, pages 1816–1823, Beijing, China, 2005.

[48] R. Fergus, P. Perona, and A. Zisserman. A visual category filter for google images. *Lecture notes in computer science*, pages 242–256, 2004.

[49] R. Fergus, Y. Weiss, and A. Torralba. Semi-supervised learning in gigantic image collections. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 522–530. MIT Press, Cambridge, MA, 2009.

[50] M. Filippone, F. Camastra, F. Masulli, and S. Rovetta. A survey of kernel and spectral methods for clustering. *Pattern Recogn.*, 41(1):176–190, 2008.

[51] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2):214–225, 2004.

[52] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[53] Y. Freund and R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory*, pages 23–37, Santa Cruz, California, USA, 1995.

[54] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions Math. Softw.*, 3(3):209–226, 1977.

[55] A. Gerson, L. Parra, and P. Sajda. Cortically coupled computer vision for rapid image search. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):174–179, 2006.

[56] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. of 25th International Conference on Very Large Data Bases*, pages 518–529, 1999.

[57] M. Gladwell. *Blink: The power of thinking without thinking*. Little, brown and company: Time warner book group, New York, 2005.

[58] M. X. Goemans and D. P. Williamson. .879-approximation algorithms for MAX CUT and MAX 2SAT. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 422–431, Montreal, Quebec, Canada, 1994.

[59] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[60] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, Cambridge, MA, 2005.

[61] S. Goldman and Y. Zhou. Enhancing supervised learning with unlabeled data. In *Proceedings of the 17th international conference on Machine learning*, pages 327–334, San Francisco, CA, 2000. Morgan Kaufmann.

[62] Z. Harris. Distributional structure. *Word*, 10(23):146–162, 1954.

[63] J. He, W. Liu, and S.-F. Chang. Scalable similarity search with optimized kernel hashing. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1129–1138, Washington, DC, USA, 2010. ACM.

[64] M. Hein and M. Maier. Manifold denoising. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 561–568. MIT Press, Cambridge, MA, 2007.

[65] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[66] W. Hsu, L. Kennedy, and S. Chang. Video search reranking via information bottleneck principle. In *Proceedings of the 14th annual ACM international conference on Multimedia*, pages 44–51, 2006.

[67] B. Huang and T. Jebara. Loopy belief propagation for bipartite maximum weight b-matching. In *Int. Workshop on Artificial Intelligence and Statistics*, 2007.

[68] B. Huang and T. Jebara. Collaborative filtering via rating concentration. In Y. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume Volume 9 of JMLR: W&CP, pages 334–341, May 13-15 2010.

[69] T. Huang, C. Dagli, S. Rajaram, E. Chang, M. Mandel, G. Poliner, and D. Ellis. Active Learning for Interactive Multimedia Retrieval. *Proceedings of the IEEE*, 96(4):648–667, 2008.

[70] P. Indyk. Nearest-neighbor searching in high dimensions. In J. E. Goodman and J. O'Rourke, editors, *Handbook of discrete and computational geometry*. CRC Press LLC, Boca Raton, FL, 2004.

[71] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. of 30th ACM Symposium on Theory of Computing*, pages 604–613, 1998.

[72] P. Jain, B. Kulis, and K. Grauman. Fast image search for learned metrics. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, Alaska, USA, June 2008.

[73] T. Jebara and V. Shchogolev. B-Matching for Spectral Clustering. In *The European Conf. on Mach. Learn.*, pages 679–686. Springer, 2006.

[74] T. Jebara, J. Wang, and S.-F. Chang. Graph construction and b-matching for semi-supervised learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 441–448, Montreal, Quebec, Canada, June 2009. ACM.

[75] W. Jiang. *Advanced Techniques for Semantic Concept Detection in General Videos*. PhD thesis, Graduate School of Arts and Sciences, Columbia University, 2010.

[76] Y.-G. Jiang. *Large Scale Semantic Concept Detection, Fusion, and Selection for Domain Adaptive Video Search*. PhD thesis, City University of Hong Kong, 2009.

[77] Y.-G. Jiang, C.-W. Ngo, and J. Yang. Towards optimal bag-of-features for object categorization and semantic video retrieval. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 494–501, Amsterdam, The Netherlands, 2007.

[78] Y. Jing and S. Baluja. VisualRank: Applying PageRank to large-scale image search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1877–1890, 2008.

[79] T. Joachims. Transductive inference for text classification using support vector machines. *Proceedings of International Conference on Machine Learning*, pages 200–209, 1999.

[80] T. Joachims. Transductive learning via spectral graph partitioning. In *Proceedings of International Conference on Machine Learning*, volume 20, pages 290–297, 2003.

[81] A. Kapoor, P. Shenoy, and D. Tan. Combining brain computer interfaces with vision for object categorization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

[82] R. M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, 43:85–103, 1972.

[83] K. Kay, T. Naselaris, R. Prenger, and J. Gallant. Identifying natural images from human brain activity. *Nature*, 452(7185):352–355, 2008.

[84] L. Kennedy. *Advanced Techniques for Multimedia Search: Leveraging Cues from Content and Structure*. PhD thesis, Graduate School of Arts and Sciences, Columbia University, 2009.

[85] L. Kennedy, S. Chang, and I. Kozintsev. To search or to label?: predicting the performance of search-based automatic image classifiers. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, page 258, 2006.

[86] C. Keysers, D. Xiao, P. Foldiak, and D. Perrett. The speed of sight. *Journal of Cognitive Neuroscience*, 13(1):90–101, 2001.

[87] A. Kiger, B. Baum, S. Jones, M. Jones, A. Coulson, C. Echeverri, and N. Perrimon. A functional genomic analysis of cell morphology using RNA interference. *Journal of biology*, 2(4):27, 2003.

[88] M.-A. Krogel and T. Scheffer. Multi-relational learning, text mining, and semi-supervised learning for functional genomics. *Machine Learning*, 57(1):61–81, 2004.

[89] B. Kulis and T. Darrell. Learning to Hash with Binary Reconstructive Embeddings. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Proc. of Advances in Neural Information Processing Systems*, volume 20, pages 1042–1050. 2009.

[90] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *Proc. of the IEEE International Conference on Computer Vision*, pages 2130–2137, Kyoto, Japan, September 2009.

[91] B. Kulis, P. Jain, and K. Grauman. Fast similarity search for learned metrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2143–2157, 2009.

[92] B. Kveton, M. Valko, A. Rahimi, and L. Huang. Semi-Supervised Learning with Max-Margin Graph Cuts. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 421–428, 2010.

[93] P. Lang, K. Yeow, A. Nichols, and A. Scheer. Cellular imaging in drug discovery. *Nature Reviews Drug Discovery*, 5(4):343–356, 2006.

[94] J. Langford, L. Li, and T. Zhang. Sparse online learning via truncated gradient. *Journal of Machine Learning Research*, 10:777–801, 2009.

[95] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 801–808. MIT Press, Cambridge, MA, 2007.

[96] D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. *Lecture Notes in Computer Science*, 1398:4–18, 1998.

[97] Z. Li, J. Liu, and X. Tang. Pairwise constraint propagation by semidefinite programming for semi-supervised classification. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, pages 576–583, New York, NY, USA, 2008. ACM.

[98] W. Liu and S.-F. Chang. Robust multi-class transductive learning with graphs. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 381–388, Miami Beach, Florida, USA.

[99] W. Liu, J. He, and S.-F. Chang. Large graph construction for scalable semi-supervised learning. In J. Fürnkranz and T. Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning*, pages 679–686, Haifa, Israel, June 2010. Omnipress.

[100] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[101] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li. Multi-probe LSH: efficient indexing for high-dimensional similarity search. In *Proceedings of the 33rd international conference on Very large data bases*, pages 950–961, University of Vienna, Austria, September 2007.

[102] M. Maier, U. von Luxburg, and M. Hein. Influence of graph construction on graph-based clustering measures. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in neural information processing systems*, volume 22, pages 1025–1032. 2009.

[103] G. S. Mann and A. McCallum. Simple, robust, scalable semi-supervised learning via expectation regularization. In *Proceedings of International Conference on Machine Learning*, pages 593–600, Corvalis, Oregon, 2007. ACM Press New York, NY, USA.

[104] C. Mathieu and W. Schudy. Yet another algorithm for dense max cut: Go greedy. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 176–182. Society for Industrial and Applied Mathematics, 2008.

[105] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[106] T. Mitchell. The role of unlabeled data in supervised learning. In *In Proceedings of the Sixth International Colloquium on Cognitive Science*. Citeseer, 1999.

[107] T. Mitchell, R. Hutchinson, R. Niculescu, F. Pereira, X. Wang, M. Just, and S. Newman. Learning to decode cognitive states from brain images. *Machine Learning*, 57(1):145–175, 2004.

[108] Y. Miyawaki, H. Uchida, O. Yamashita, M. Sato, Y. Morito, H. Tanabe, N. Sadato, and Y. Kamitani. Visual Image Reconstruction from Human Brain Activity using a Combination of Multiscale Local Image Decoders. *Neuron*, 60(5):915–929, 2008.

[109] S. J.-Y. S. Mu, Y. Weakly-Supervised Hashing in Kernel Space. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3344 – 3351, San Francisco, USA, June 2010.

[110] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications*, pages 331–340, Algarve, Portugal, 2009.

[111] A. P. Natsev, A. Haubold, J. Tešić, L. Xie, and R. Yan. Semantic concept-based query expansion and re-ranking for multimedia retrieval. In *Proceedings of the 15th international conference on Multimedia*, pages 991–1000, Augsburg, Germany, 2007. ACM.

[112] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems 14*, pages 849–856, 2001.

[113] A. Oliva. Gist of the scene. In *Encyclopedia of Neurobiology of Attention*, pages 251–256, San Diego, CA, 2005. Elsevier.

[114] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[115] S. Omohundro. Efficient algorithms with neural network behavior. *Complex Systems*, 1(2):273–347, 1987.

[116] L. Parra, C. Christoforou, A. Gerson, M. Dyrholm, A. Luo, M. Wagner, M. Philiastides, and P. Sajda. Spatiotemporal linear decoding of brain state: Application to performance augmentation in high-throughput tasks. *IEEE Signal Processing Magazine*, 25(1):95–115, January 2008.

[117] J. Peng, B. Bhanu, and S. Qing. Probabilistic feature relevance learning for content-based image retrieval. *Computer Vision and Image Understanding*, 75(1):150–164, 1999.

[118] P. Poolman, R. Frank, P. Luu, S. Pederson, and D. Tucker. A single-trial analytic framework for EEG analysis and its application to target detection and classification. *NeuroImage*, 42(2):787–798, 2008.

[119] M. Potter and E. Levy. Recognition memory for a rapid sequence of pictures. *Journal of Experimental Psychology*, 81(1):10, 1969.

[120] M. Raginsky and S. Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1509–1517. 2009.

[121] R. Rifkin and R. Lippert. Notes on regularized least squares. Technical Report MIT-CSAIL-TR-2007-025, Computer Sciences and Artificial Intelligence Laboratory, MIT, 2007.

[122] S. T. Roweis and L. K. Saul. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science*, 290(5500):2323–2326, 2000.

[123] Y. Rui, T. Huang, and S. Chang. Image Retrieval: Current Techniques, Promising Directions, and Open Issues* 1. *Journal of visual communication and image representation*, 10(1):39–62.

[124] Y. Rui, T. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: a power tool for interactive content-based image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5):644–655, 1998.

[125] S. Rüping and T. Scheffer. Learning with multiple views. In *Proc. ICML Workshop on Learning with Multiple Views*, 2005.

[126] P. Sajda, E. Pohlmeyer, J. Wang, B. Hanna, L. C. Parra, and S.-F. Chang. Cortically-coupled computer vision. In D. S. Tan and A. Nijholt, editors, *Brain-Computer Interfaces: Applying our Minds to Human-Computer Interaction*. Elsevier, 2010.

[127] P. Sajda, E. Pohlmeyer, J. Wang, L. C. Parra, C. Christoforou, J. Dmochowski, B. Hanna, C. Bahlmann, M. K. Singh, and S.-F. Chang. In a Blink of an Eye and a Switch of a transistor: Cortically Coupled Computer Vision. *Proceedings of the IEEE*, 98(3):462–478, 2010.

[128] R. Salakhutdinov and G. Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009.

[129] J. Salez and D. Shah. Optimality of Belief Propagation for Random Assignment Problem. In *ACM-SIAM Symp. on Discrete Algorithms*, 2009.

[130] M. Sanderson and P. Clough. cross-language image retrieval track. http://imageclef.org/.

[131] G. Shakhnarovich. *Learning task-specific similarity*. PhD thesis, Massachusetts Institute of Technology, 2005.

[132] G. Shakhnarovich, T. Darrell, and P. Indyk. *Nearest-neighbor methods in learning and vision: theory and practice*. MIT Press, 2005.

[133] P. Shenoy and D. Tan. Human-aided computing: Utilizing implicit human processing to classify images. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 845–854, 2008.

[134] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[135] C. Silpa-Anan and R. Hartley. Optimised kd-trees for fast image descriptor matching. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[136] V. Sindhwani, J. Hu, and A. Mojsilovic. Regularized co-clustering with dual supervision. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, pages 976–983. 2008.

[137] V. Sindhwani, P. Niyogi, and M. Belkin. A co-regularization approach to semi-supervised learning with multiple views. In *Proc. ICML Workshop on Learning with Multiple Views*, pages 74–79, 2005.

[138] V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 824–831, Bonn, Germany, 2005. ACM.

[139] V. Sindhwani and D. S. Rosenberg. An rkhs for multi-view learning and manifold co-regularization. In *Proceedings of the 25th international conference on Machine learning*, pages 976–983, 2008.

[140] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, 2006.

[141] A. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

[142] J. Smith and S. Chang. VisualSEEk: a fully automated content-based image query system. In *Proceedings of the fourth ACM international conference on Multimedia*, pages 87–98. ACM, 1997.

[143] M. Szummer and T. Jaakkola. Partially labeled classification with markov random walks. In *Advances in Neural Information Processing Systems 17*, volume 14, pages 945–952. MIT Press, Cambridge, MA, 2002.

[144] W. Tang, Z. Lu, and I. Dhillon. Clustering with multiple graphs. In *2009 Ninth IEEE International Conference on Data Mining*, pages 1016–1021, Miami, FL, USA,, 2009.

[145] S. Thorpe, D. Fize, and C. Marlot. Speed of processing in the human visual system. *Nature*, 381(6582):520–522, 1996.

[146] S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 107–118. ACM New York, NY, USA, 2001.

[147] A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: A large data set for non-parametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008.

[148] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, Alaska, USA, June 2008.

[149] A. Turpin and F. Scholer. User performance versus precision measures for simple search tasks. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval*, pages 11–18, Seattle, Washington, USA, 2006. ACM.

[150] J. Uhlmann. Satisfying general proximity/similarity queries with metric trees. *Information Processing Letters*, 40(4):175–179, 1991.

[151] V. Vapnik. *Statistical learning theory*. Wiley, New York, 1998.

[152] U. Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[153] F. Wang and C. Zhang. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, 2008.

[154] J. Wang and S.-F. Chang. Columbia tag system - transductive annotation by graph version 1.0. Technical report, Columbia University, October 2008.

[155] J. Wang, S.-F. Chang, X. Zhou, and T. C. S. Wong. Active Microscopic Cellular Image Annotation by Superposable Graph Transduction with Imbalanced Labels. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Alaska, USA, June 2008.

[156] J. Wang, T. Jebara, and S.-F. Chang. Graph transduction via alternating minimization. In *Proceedings of International Conference on Machine Learning*, pages 1144–1151, Helsinki, Finland, 2008.

[157] J. Wang, Y.-G. Jiang, and S.-F. Chang. Label diagnosis through self tuning for web image search. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1390–1397, Miami Beach, Florida, USA.

[158] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3424 – 3431, San Francisco, USA, June 2010.

[159] J. Wang, S. Kumar, and S.-F. Chang. Sequential projection learning for hashing with compact codes. In J. Fürnkranz and T. Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning*, pages 1127–1134, Haifa, Israel, June 2010. Omnipress.

[160] J. Wang, E. Pohlmeyer, B. Hanna, Y.-G. Jiang, P. Sajda, and S.-F. Chang. Brain state decoding for rapid image retrieval. In *Proceedings of the seventeen ACM international conference on Multimedia*, pages 945–954. ACM, 2009.

[161] J. Wang, X. Zhou, P. L. Bradley, S.-F. Chang, N. Perrimon, and S. T.C. Wong. Cellular Phenotype Recognition for High-Content RNAi Genome-Wide Screening. *Journal of Biomolecular Screening*, 13(1):29–39, February 2008.

[162] J. Wang, X. Zhou, F. Li, and N. P. S. T. C. W. Pamela L. Bradley, Shih-Fu Chang. An image score inference system for RNAi genome-wide screening based on fuzzy mixture regression modeling. *Journal of Biomedical Informatics*, 42(1):32–40, 2009.

[163] W. Wang and Z.-H. Zhou. A new analysis of co-training. In J. Fürnkranz and T. Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning*, pages 1135–1142, Haifa, Israel, June 2010. Omnipress.

[164] X. Wang and I. Davidson. Flexible constrained spectral clustering. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 563–572, New York, NY, USA, 2010. ACM.

[165] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Proc. of Advances in Neural Information Processing Systems*, volume 21, pages 1753–1760. 2008.

[166] Z. Xu, R. Jin, J. Zhu, I. King, and M. Lyu. Efficient convex relaxation for transductive support vector machine. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1641–1648. MIT Press, Cambridge, MA, 2008.

[167] D. Yan, L. Huang, and M. I. Jordan. Fast approximate spectral clustering. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 907–916, New York, NY, USA, 2009. ACM.

[168] J. Yang, K. Yu, and T. Huang. Supervised translation-invariant sparse coding. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3517–3524, San Francisco, USA, June 2010.

[169] P. N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *Proc. of the fourth annual ACM-SIAM Symposium on Discrete algorithms*, pages 311–321, Austin, Texas, United States, 1993.

[170] E. Zavesky. *A Guided, Low-Latency, and Relevance Propagation Framework for Interactive Multimedia Search*. PhD thesis, Graduate School of Arts and Sciences, Columbia University, 2010.

[171] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. *Advances in Neural Information Processing Systems*, 17:1601–1608, 2004.

[172] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study. *International Journal of Computer Vision*, 73(2):213–238, 2007.

[173] T. Zhang, A. Popescul, and B. Dom. Linear prediction models with graph regularization for web-page categorization. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 821–826, 2006.

[174] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 321–328. MIT Press, Cambridge, MA, 2004.

[175] D. Zhou and C. J. C. Burges. Spectral clustering and transductive learning with multiple views. In *Proceedings of the 24th international conference on Machine learning*, pages 1159–1166, 2007.

[176] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Scholkopf. Ranking on Data Manifolds. In *Proc. NIPS*, volume 16, pages 169–172, 2004.

[177] X. Zhou, K. Liu, P. Bradley, N. Perrimon, and S. Wong. Towards automated cellular image segmentation for RNAi genome-wide screening. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2005*, pages 885–892, 2005.

[178] X. Zhou and S. Wong. Informatics challenges of high-throughput microscopy. *Signal Processing Magazine, IEEE*, 23(3):63–72, 2006.

[179] Z.-H. Zhou and M. Li. Tri-training: exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1529–1541, 2005.

[180] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[181] X. Zhu and A. B. Goldberg. *Introduction to semi-supervised learning*. Morgan & Claypool Publishers, 2009.

[182] X. Zhu, X. Wu, and Q. Chen. Eliminating class noise in large datasets. In *International Conference on Machine Learning*, volume 20, pages 920–927, 2003.

Part II

Appendices

Appendix A

b-Matching via Belief Propagation

Given an adjacency matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ of a graph $\mathcal{G} = (\mathbf{X}, \mathbf{E})$ with n nodes $\mathbf{X}$ and $\mathcal{O}(n^2)$ edges $\mathbf{E}$, the maximum weight b-matching problem finds a subgraph of $\mathcal{G}_b$ with maximum weight while constraining the number of edges for each vertex to exactly b. It is a direct generalization of the maximum weight unipartite matching problem ($b = 1$) which can be solved by Edmonds' algorithm in polynomial time $\mathcal{O}(n^3)$ [43]. In [13], the 1-matching problem was formulated with a discrete probability distribution and solved by max-product loopy belief propagation (BP) to recover the maximum a posteriori (MAP) assignment in $\mathcal{O}(n^3)$. In [67], the extension of loopy belief propagation from 1-matching to b-matching was provided and a proof of convergence in $\mathcal{O}(bn^3)$ was included. Furthermore, various implementation issues and changes to the canonical message passing rules of belief propagation were provided to improve computational efficiency (standard message passing involves messages of an exponential size) leading to an efficient solution for the b-matching problem. Here we present an algorithm in the bipartite case where the nodes in the graph are split into two sets a priori before being b-matched. The extension to the unipartite case is straightforward simply by modifying the algorithm such that messages are passed between all pairs of nodes instead of across the bipartition only.

Assume u_i and v_j are two vertices on the graph and b-matching returns the neighbor vertex sets $\mathcal{N}(u_i)$ and $\mathcal{N}(v_j)$ for u_i and v_j, respectively. The combinatorial problem in Eq. (2.8) can be written as:

$$\max_{\mathcal{N}} \mathcal{W}(\mathcal{N}) = \max_{\mathcal{N}} \sum_{i=1}^{n} \sum_{v_k \in \mathcal{N}(u_i)} \mathbf{K}_{ik} + \sum_{j=1}^{n} \sum_{u_l \in \mathcal{N}(v_j)} \mathbf{K}_{lj} \tag{A.1}$$

For each vertex, two random variables are defined as $z_i \in Z$ and $s_j \in S$ and $z_i = \mathcal{N}(u_i)$ and $s_j = \mathcal{N}(v_j)$. Hence we can have the following potential functions:

$$\phi(z_i) = \exp\left(\sum_{v_j \in z_i} \mathbf{K}_{ij}\right) \tag{A.2}$$

$$\phi(s_j) = \exp\left(\sum_{u_i \in s_j} \mathbf{K}_{ij}\right) \tag{A.3}$$

$$\psi(z_i, s_j) = \neg(v_j \in z_i \oplus u_i \in s_j). \tag{A.4}$$

By multiplying the above potential functions and pairwise clique functions, the objective function for weighted b-matching problem can be formulated as a probability distribution via $p(Z, S) \propto \exp(\mathcal{W}(\mathcal{N}))$ [34], where the joint distribution can be expressed as:

$$p(Z, S) = \frac{1}{Z} \prod_{i=1}^{n} \prod_{j=1}^{n} \psi(z_i, s_j) \prod_{k=1}^{n} \phi(z_i)\phi(s_j) \tag{A.5}$$

Max-product message passing on the above distribution is guaranteed to converge to the true maximum in $\mathcal{O}(n^3)$ time on bipartite graphs, the proof provided in [67] is omitted here for brevity. In practice, convergence is much faster than this worst case. Furthermore, [129] prove that, under mild assumptions on the matrix $\mathbf{K}$, belief propagation for matching problems can converge in $\mathcal{O}(n^2)$ for dense graphs. If the graph is sparse, convergence is typically $\mathcal{O}(|\mathbf{E}|)$ or proportional to the number of edges in the graph.

Appendix B

Multi-Class Graph Transduction as a Max K-Cut Problem

Here, we briefly described that K-class bivariate graph transduction is equivalent to a Max K-Cut problem. If the number of classes is K, the label variable $\mathbf{Y}$ is a $n \times K$ matrix denoting classification results, where $\mathbf{Y}_{ij} = 1$ indicates that vertex $\mathbf{x}_i$ is associated with label j. Then, we can rewrite the objective function in Eq. (3.8) as

$$\mathcal{Q}(\mathbf{Y}) = \frac{1}{2}\mathrm{tr}\left(\mathbf{Y}^\top \mathbf{A}\mathbf{Y}\right) = \frac{1}{2}\sum_k \mathbf{Y}_{.k}^\top \mathbf{A}\mathbf{Y}_{.k} \tag{B.1}$$

Let $\mathbf{y}_k = \mathbf{Y}_{.k}$ is the column vector from $\mathbf{Y}$. Let non-zero elements in $\mathbf{y}_k$ denote the vertices in subset S_k, where $k = 1, 2, \cdots, K$, $S_1 \cup S_2 \cup \cdots \cup S_K = V_{\mathbf{A}}$, and $S_m \cap S_n = \emptyset$ if $m \neq n$. Then the above objective function is equivalent to

$$\begin{aligned}\mathcal{Q}(\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_K) = \tfrac{1}{2}\textstyle\sum_k \mathbf{y}_k^\top \mathbf{A}\mathbf{y}_k &= \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in S_k \\ i<j}} \mathbf{A}_{ij} \\ &= \sum_{i<j} \mathbf{A}_{ij} - \sum_{m=1}^{K-1}\sum_{n=m+1}^{K}\sum_{\substack{\mathbf{x}_i \in S_m \\ \mathbf{x}_j \in S_n}} \mathbf{A}_{ij}\end{aligned} \tag{B.2}$$

Therefore the original minimization problem equals to maximize the sum of the weight of the edges between the disjoint sets S_k, i.e., maximum K-cut problem.

$$\max \sum_{m=1}^{K-1}\sum_{n=m+1}^{K}\sum_{\substack{\mathbf{x}_i \in S_m \\ \mathbf{x}_j \in S_n}} \mathbf{A}_{ij} \tag{B.3}$$