Title
Spectral Clustering for Complex Settings

Permalink
https://escholarship.org/uc/item/722107r7

Author
Wang, Xiang

Publication Date
2013-02-01

Peer reviewed
Spectral Clustering for Complex Settings

By

XIANG WANG
BS (Tsinghua University) 2004
ME (Tsinghua University) 2008

Dissertation

Submitted in partial satisfaction of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Office of Graduate Studies

of the

University of California

Davis

Approved:

Ian Davidson, Chair

Zhaojun Bai

Owen Carmichael

Committee in Charge

2013
To my parents
# Contents

List of Figures ................................................................. vii  
List of Tables ................................................................. x  
Abstract .......................................................................... xi  
Acknowledgments ............................................................. xii

## 1 Introduction  
1.1 Spectral Clustering for a Single Graph .......................... 1  
1.2 Complex Graph Models ................................................. 2  
1.3 Summary of the Dissertation ......................................... 4

## 2 Constrained Spectral Clustering  
2.1 Introduction .............................................................. 7  
2.1.1 Background and Motivation ...................................... 7  
2.1.2 Our Contributions .................................................. 10  
2.2 Related Work ........................................................... 12  
2.3 Background and Preliminaries ...................................... 13  
2.4 A Flexible Framework for Constrained Spectral Clustering .... 15  
2.4.1 The Objective Function .......................................... 15  
2.4.2 Solving the Objective Function .................................. 17  
2.4.3 A Sufficient Condition for the Existence of Solutions ...... 20  
2.4.4 An Illustrative Example .......................................... 21  
2.5 Interpretations of Our Formulation ................................ 23  
2.5.1 A Graph Cut Interpretation .................................... 23  
2.5.2 A Geometric Interpretation ..................................... 23  
2.6 Implementation and Extensions ..................................... 24  
2.6.1 Constrained Spectral Clustering for 2-Way Partition ....... 24  
2.6.2 Extension to K-Way Partition .................................... 26  
2.6.3 Using Constrained Spectral Clustering for Transfer Learning ... 27
2.6.4 A Non-Parametric Version of Our Algorithm ........................................ 28
2.7 Testing and Innovative Uses of Our Work ............................................ 29
  2.7.1 Explicit Pairwise Constraints: Image Segmentation ......................... 30
  2.7.2 Explicit Pairwise Constraints: The Double Moon Dataset ............... 33
  2.7.3 Constraints from Partial Labeling: Clustering the UCI Benchmarks .... 35
  2.7.4 Constraints from Alternative Metrics: The Reuters Multilingual Dataset ............................................................ 37
  2.7.5 Transfer of Knowledge: Resting-State fMRI Analysis ................. 43
2.8 Summary ............................................................................................. 50

3 Active Extension to Constrained Spectral Clustering ............................... 51
  3.1 Introduction .......................................................................................... 51
  3.2 Related Work ....................................................................................... 54
  3.3 Our Active Spectral Clustering Framework ........................................ 55
    3.3.1 An Overview of Our Framework ............................................... 55
    3.3.2 The Constrained Spectral Clustering Algorithm ..................... 56
    3.3.3 The Query Strategy .................................................................... 57
  3.4 Empirical Results ................................................................................ 59
  3.5 Implementation Issues ........................................................................ 62
    3.5.1 Outliers ......................................................................................... 62
    3.5.2 Time Complexity ......................................................................... 64
    3.5.3 Stopping Criterion ....................................................................... 64
  3.6 Limitations and Extensions .................................................................. 65
  3.7 Summary ............................................................................................. 66

4 Constrained Spectral Clustering and Label Propagation ....................... 67
  4.1 Introduction .......................................................................................... 67
  4.2 Related Work and Preliminaries ....................................................... 69
    4.2.1 Label Propagation ........................................................................ 69
    4.2.2 Constrained Spectral Clustering (CSC) .................................... 70
List of Figures

1.1 Structure of the dissertation. ........................................ 4

2.1 A motivating example for constrained spectral clustering. ................ 8

2.2 An illustrative example: the affinity structure says \{1, 2, 3\} and \{4, 5, 6\} while the node labeling (coloring) says \{1, 2, 3, 4\} and \{5, 6\}. ............... 21

2.3 The solutions to the illustrative example in Fig. 2.2 with different $\beta$. The x-axis is the indices of the instances and the y-axis is the corresponding entry values in the optimal (relaxed) cluster indicator $u^*$. Notice that node 4 is biased toward nodes \{1, 2, 3\} as $\beta$ increases. ............... 22

2.4 The joint numerical range of the normalized graph Laplacian $\bar{L}$ and the normalized constraint matrix $\bar{Q}$, as well as the optimal solutions to unconstrained/constrained spectral clustering. ......................... 25

2.5 Segmentation of the elephant image. The images are reconstructed based on the relaxed cluster indicator $u^*$. Pixels that are closer to the red end of the spectrum belong to one segment and blue the other. The labeled pixels are as bounded by the black and white rectangles. ............... 31

2.6 Segmentation of the face image. The images are reconstructed based on the relaxed cluster indicator $u^*$. Pixels that are closer to the red end of the spectrum belong to one segment and blue the other. The labeled pixels are as bounded by the black and white rectangles. ............... 32

2.7 The convergence of our algorithm on 10 random samples of the Double Moon distribution. ........................................ 34

2.8 The partition of a noisy Double Moon sample. ........................ 35

2.9 The performance of our algorithm (CSP) on six UCI datasets, with comparison to unconstrained spectral clustering (Baseline) and the Spectral Learning algorithm (SL). Adjusted Rand index over 100 random trials is reported (mean, min, and max). ................................. 38

2.10 The ratio of constraints that are actually satisfied. ..................... 39
2.11 The box plot for 4 language pairs, 100 random trials each. Results are evaluated in terms of both ARI and NMI. 44

2.12 The box plot for another 4 language pairs, 100 random trials each. Results are evaluated in terms of both ARI and NMI. 45

2.13 The trial by trial breakdown of the performance gain of our technique (csp-p and csp-n) over the baseline (orig) on 4 language pairs. 46

2.14 The trial by trial breakdown of the performance gain of our technique (csp-p and csp-n) over the baseline (orig) on another 4 language pairs. 47

2.15 Transfer learning on fMRI scans. 48

2.16 The costs of transferring the idealized default mode network to the fMRI scans of two groups of elderly individuals. 49

3.1 Results on six UCI data sets, with comparison between our method active and the baseline method random. Y-axis is Rand index, X-axis is the number of constraints queried. The maximum number of queries is 2N, where N is the size of the corresponding data set. For the random method, the max/average/min performance over 10 runs (each with a randomly generated constraint set) are reported, respectively. 63

4.1 An illustration of the GFHF propagation model (N = 5, N_1 = 1). Node 4 is labeled. The propagation from 4 to 3 and 5 is governed by P_{ul} (directed edges); the propagation between 1, 2, and 3 is governed by P_{uu} (undirected edges). 70

4.2 An illustration of the LLGC propagation model (N = 5). Node 4 is labeled. The propagation between nodes is governed by Ā_{ij} (all edges are undirected). 71
4.3 An illustration of our stationary label propagation model \((N = 5)\). \(G\) is the unlabeled graph we want to propagate labels to. \(H\) is a latent node-bearing graph whose node set matches the node set of \(G\). The label propagation inside \(G\) is governed by the transition matrix \(P_{GG}\) (undirected edges). The propagation from \(H\) to \(G\) is governed by the transition matrix \(P_{GH}\) (directed edges).

4.4 The average adjusted Rand index over 100 randomly chosen label sets of varying sizes.

4.5 The percentage of randomly chosen label sets that lead to positive performance gain with respect to the spectral clustering baseline.

5.1 The joint numerical range of the Wine dataset. The +’s correspond to points in \(\tilde{J}(G_1, G_2)\). The ◦’s are the Pareto optimal cuts found by our algorithm, which is \(\tilde{F}(G_1, G_2)\).

5.2 The Pareto embedding of the Wine dataset. (a)(b)(c) show the clusterings derived from the Pareto optimal cuts in Fig. 5.1. (d) shows the original labels of the dataset.

5.3 The mean difference (in terms of adjusted Rand index) of various techniques wrt the best-performing technique on each dataset, grouped by two cases (datasets with compatible views vs. datasets with incompatible views).

5.4 The results of applying our algorithm to resting-state fMRI scans. Illustrated is a horizontal slice of the scan (eyes are on the right-hand side). We use an exemplar scan (View 1) to induce the Default Mode Network (the red/yellow pixels in the figures) in a set of target scans (View 2). Our algorithm produced consistent partitions across different target scans.

5.5 The costs of induced DMN cuts on the target scans, grouped by 3 sub-populations. The costs increase as the cognitive symptom gets worse.
List of Tables

2.1 Table of notations ......................................................... 14
2.2 The UCI benchmarks ....................................................... 36
2.3 The Reuters multilingual dataset ...................................... 40
2.4 Average ARI of 7 algorithms on 8 language pairs .................. 42

3.1 Table of notations .......................................................... 55
3.2 The UCI benchmarks ....................................................... 60
3.3 The 20 Newsgroup data .................................................... 60

5.1 Table of notations .......................................................... 85
5.2 Statistics of datasets ....................................................... 95
5.3 The adjusted Rand index of various algorithms on six UCI datasets with incompatible views. Bold numbers are best results. The number in the parenthesis is the performance gain of our approach (Pareto) over the best competitor. Our method performs the best on the majority of datasets. ................................................................. 97
5.4 The adjusted Rand index of various algorithms on the 20 Newsgroups dataset with compatible views. Bold numbers are best results. The number in the parenthesis is the performance gain of our approach (Pareto) over the best competitor. Note our method is comparable to other methods. The best performing method MM, performs poorly in Table 5.3. .... 97
Many real-world datasets can be modeled as graphs, where each node corresponds to a data instance and an edge represents the relation/similarity between two nodes. To partition the nodes into different clusters, spectral clustering is used to find the normalized minimum cut of the graph (in the relaxed sense). As one of the most popular clustering schemes, spectral clustering is limited to a single graph. However, in practice, we often need to collectively consider rich information generated from multiple heterogeneous sources, e.g. scientific data (fMRI scans of different individuals), social data (different types of relationship among different people), and web data (multi-type contents). Such complex datasets demand complex graph models.

In this dissertation, we explore novel formulations to extend spectral clustering to a variety of complex graph models and study how to apply them to real-world problems. We start with incorporating pairwise constraints into spectral clustering, which extends spectral clustering from unsupervised setting to semi-supervised setting. Then we further extend our constrained spectral clustering formulation from passive learning to active learning. We justify the effectiveness of our approach by exploring its link to a classic graph-based semi-supervised learning technique, namely label propagation. Finally we study how to extend spectral clustering to the multi-view learning setting. Our proposed algorithms were not only tested on benchmark datasets but also successfully applied to real-world applications, such as machine translation aided document clustering and resting-state fMRI analysis.
ACKNOWLEDGMENTS

First and foremost, I thank my advisor Professor Ian Davidson for his guidance and support. He provided me with everything that a Ph.D. student can ever ask for, and some more.

I also thank my master program advisor Professor Xiaoming Jin for teaching me how to do good research and inspiring me to pursue a Ph.D. degree.

I am very grateful to my collaborators, Professor Owen Carmichael, Buyue Qian, Professor Jieping Ye, and my lab mates, Sean Gilpin and Kristin Lui, for their valuable input and help. I learn from them everyday.

I thank Professor Zhaojun Bai for his kind advice and encouragement.

I am truly grateful to Yiqi Fu at Tsinghua University, who gave me help when I was in need.

I owe my gratitude to Dr. Dou Shen, Dr. Bo Thiesson, and Dr. Jay Stokes for the wonderful opportunities they brought me.

Big shout-out to Stan, Buyue, Jishang, Yuan, Matt, Andrew, Kenes, Brian, Shubho, Jianqiang, Tom, Gabe, Angel, Anjul, Zijie, Yan, Grace, Sean, and Kristin for being awesome human beings and friends. I am proud to consider myself as your friends.

I would not have become who I am today without the unwavering support from my parents and my wife Xiaoli. You give meanings to everything I do and every goal I achieve. I love you.
Chapter 1

Introduction

1.1 Spectral Clustering for a Single Graph

Graph is a natural data model for many data mining applications because the node-edge structure of a graph matches the entity-relation structure underlying the datasets. For example, the World Wide Web can be modeled as a large graph where each node corresponds to a web page and the (directed) edges represent the hyper-link from one web page to another [11, 37]; the social network is a graph where each node is a person and an edge means the relationship or interaction between two people [23]; graph can also be used to model the interaction between proteins in biological data [28].

In general, given a set of data instances $D$, on which a pairwise relation $D \times D$ is defined, we can model $D$ as a single graph $G(V,E)$ which includes:

- A set of nodes $V$, where each node corresponds to a data instance in $D$.

- A set of edges $E \subseteq V \times V$, where each edge corresponds to the relation between the two nodes (instances).

In practice, commonly used pairwise relations include: 1) adjacency, where the edges are unweighted, and 2) a similarity function, where the edges are weighted.

Once the dataset is turned into a graph, we can perform a variety of data mining techniques on the graph. In this dissertation, we focus on spectral clustering [48, 51, 55], which finds the normalized min-cut of the graph and partitions the nodes of graph into
different clusters. As compared to other popular clustering schemes, such as $K$-means, hierarchical clustering, density based clustering, etc., spectral clustering has some unique advantages:

1. Some clustering schemes make implicit assumptions about the dataset, e.g. the geometric shape of the clusters, the density function of the underlying distribution, the conditional independence between features, and so on. These assumptions are critical to the effectiveness of the clustering algorithm; yet their validity is difficult, if not impossible, to test in practice. In contrast, spectral clustering makes no assumptions other than we have a graph where higher edge weight means we are less inclined to separate that pair of nodes.

2. Some clustering techniques are based on heuristics, whereas spectral clustering has a global objective to optimize, namely the normalized min-cut of the graph. This gives the users a clear understanding of what properties are being sought after and how the clustering result should be interpreted.

3. Some clustering algorithms such as $K$-means can only find a local optimum and their results are sensitive to random initialization. In contrast, spectral clustering has closed form solution. The objective of spectral clustering can be solved by eigenvalue decomposition in polynomial time with a deterministic solution.

1.2 Complex Graph Models

Spectral clustering have been extensively studied for a single graph, which is sufficient to encode a set of data instances with pairwise similarity. However, in practice, there are heterogeneous datasets that a single graph can no longer fully characterize. For example, in additional to the hyperlink structure, web data have heterogeneous attributes, such as texts, images, videos, etc; social networks include not only friendship relations between people but also heterogeneous personal information; scientific data, such as fMRI scans, often have multiple observations for the same object.

To encode the rich information associate with a heterogeneous dataset, we need to
introduce more complex graph models. In particular, there are several complex settings that have been proven to be useful for real-world data:

**Graphs with pairwise constraints** The pairwise constraints associated with a graph do not replace but are in addition to the original affinity structure of the graph. Typical constraints are Must-Link, meaning that the two nodes belong to the same cluster, and Cannot-Link, meaning that the two nodes belong to different clusters. In practice, the constraints often come from prior knowledge, domain expert supervision, or auxiliary data sources. Our goal is to incorporate these constraints into the spectral clustering objective such that the cut cost is minimized while the constraint satisfaction is maximized. The main challenges include inconsistent constraints, degree-of-belief constraints rather than binary constraints, and massive amount of constraints.

**Graph with node labels** In some graphs the nodes have labels. The labeling could either be partial or complete. Each node could have one or more labels. In practice, the labels can come from the tag keywords on web pages (for web data), affiliation information of people (for social networks), domain knowledge (for scientific data), and so on. Our goal is to use the (partial) labeling to improve the result of spectral clustering, assuming that the original graph is a noisy sample of the ground truth distribution. The challenge is to find a unified and meaningful encoding of heterogeneous label sets, which are often noisy, inconsistent, and massive in amount.

**Graph with multiple views** A multi-view graph has a unique node set, but multiple edge sets. It can be viewed as multiple observations of the same graph (e.g. multiple fMRI scans of the same patient), an edge-evolving graph, or a graph with affinity structures constructed from heterogeneous data sources (e.g. multilingual documents). Our goal is to combine the views to find an optimal partition of the nodes. The challenge is that different views are often incompatible with each other, i.e. combining them in a wrong way could hurt the clustering result.

To extend spectral clustering to these complex settings is non-trivial. Besides of the aforementioned challenges, the core computation of spectral clustering, namely the eigenvalue decomposition, is confined to a single matrix (the graph Laplacian in our
Therefore we need to devise novel formulations to encode the auxiliary information from the complex graphs.

### 1.3 Summary of the Dissertation

In this dissertation we extend spectral clustering to various complex settings. We present a unified and principled framework, under which the objective functions can be solved efficiently using generalized eigendecomposition. We demonstrate the effectiveness of our algorithms on not only benchmark datasets but also two real-world applications: resting state fMRI analysis and machine translation aided document clustering. We provide intuitive interpretations, both geometrically and probabilistic, to our approach and relate it to a classic graph-based semi-supervised learning technique: label propagation. Furthermore we extend our formulation to the active learning setting and the multi-view learning setting. The structure of the dissertation is illustrated in Fig. 1.1.
In Chapter 2, we proposed a constrained spectral clustering framework that can encode a variety of side information types (pairwise constraints, partial labeling, multiple metrics, etc.) as pairwise constraints. In contrast to some previous efforts that implicitly encode Must-Link and Cannot-Link constraints by modifying the graph Laplacian or constraining the underlying eigenspace, we present a more natural and principled formulation, which explicitly encodes the constraints as part of a constrained optimization problem. Our method offers several practical advantages: it can encode the degree of belief in Must-Link and Cannot-Link constraints; it guarantees to lower-bound how well the given constraints are satisfied using a user-specified threshold; it can be solved deterministically in polynomial time through generalized eigendecomposition. Furthermore, by inheriting the objective function from spectral clustering and encoding the constraints explicitly, much of the existing analysis of unconstrained spectral clustering techniques remains valid for our formulation. We validate the effectiveness of our approach by empirical results on both artificial and real datasets. We also demonstrate an innovative use of encoding large number of constraints: transfer learning via constraints. Some materials in this chapter appeared in [60, 61, 63].

In Chapter 3, we extend our constrained spectral clustering framework to the active learning setting. In this setting, the constraints are actively acquired by querying a black-box oracle. Since in practice each query comes with a cost, our goal is to maximally improve the result by using as few queries as possible. The advantages of our approach include: 1) it is principled by querying the constraints which maximally reduce the expected error; 2) it can incorporate both hard and soft constraints which are prevalent in practice. We empirically show that our method significantly outperforms the baseline approach, namely constrained spectral clustering with randomly selected constraints. A version of this chapter appeared in [59].

In Chapter 4, we present a label propagation interpretation to our constrained spectral clustering framework. In label propagation, the known labels are populated to the unlabeled nodes based on the affinity structure of the graph. In constrained clustering, known labels are first converted to pairwise constraints (Must-Link and Cannot-Link),
then a constrained cut is computed as a tradeoff between minimizing the cut cost and maximizing the constraint satisfaction. Both techniques are evaluated by their ability to recover the ground truth labeling, i.e. by 0/1 loss function either directly on the labels or on the pairwise relations derived from the labels. These two fields have developed separately, but in this chapter, we show that they are indeed related. This insight allows us to propose a novel way to generate constraints from the propagated labels, which outperforms and is more stable than the state-of-the-art label propagation and constrained spectral clustering algorithms. A version of this chapter appeared in [62].

In Chapter 5, we present a Pareto optimization framework for multi-view spectral clustering. Previous work on multi-view clustering formulates the problem as a single objective function to optimize, typically by combining the views under a compatibility assumption and requiring the users to decide the importance of each view a priori. In this chapter, we propose a multi-objective formulation and show how to solve it using Pareto optimization. The Pareto frontier captures all possible good cuts without requiring the users to set the “correct” parameter. The effectiveness of our approach is justified by both theoretical analysis and empirical results. We also demonstrate a novel application of our approach: resting-state fMRI analysis. A version of this chapter appeared in [64].

Chapter 6 concludes the dissertation.
Chapter 2
Constrained Spectral Clustering

2.1 Introduction

2.1.1 Background and Motivation

Spectral clustering is an important clustering technique that has been extensively studied in the image processing, data mining, and machine learning communities [48, 51, 55]. It is considered superior to traditional clustering algorithms like K-means in terms of having deterministic polynomial-time solution, the ability to model arbitrary shaped clusters, and its equivalence to certain graph cut problems. For example, spectral clustering is able to capture the underlying moon-shaped clusters as shown in Fig. 2.1(b), whereas K-means would fail (Fig. 2.1(a)). The advantage of spectral clustering has also been validated by many real-world applications, such as image segmentation [51] and mining social networks [65].

Spectral clustering was originally proposed to address an unsupervised learning problem: the data instances are unlabeled, and all available information is encoded in the graph Laplacian. However, there are cases where unsupervised spectral clustering becomes insufficient. Using the same toy data, as shown in (Fig. 2.1(c)), when the two moons are under-sampled, the clusters become so sparse that the separation of them becomes difficult. To help spectral clustering recover from an undesirable partition, we can introduce side information in various forms, in either small or large amounts. For example:
1. **Pairwise constraints**: Domain experts may explicitly assign constraints that state a pair of instances must be in the same cluster (Must-Link, ML for short) or that a pair of instances cannot be in the same cluster (Cannot-Link, CL for short). For instance, as shown in Fig. 2.1(d), we assigned several ML (solid lines) and CL (dashed lines) constraints, then applied our constrained spectral clustering algorithm, which we will describe later. As a result, the two moons were successfully recovered.

2. **Partial labeling**: There can be labels on some of the instances, which are neither complete nor exhaustive. We demonstrate in Fig. 2.9 that even small amounts of labeled information can greatly improve clustering results when compared against
the ground truth partition, as inferred by the labels.

3. **Alternative weak distance metrics**: In some situations there may be more than one distance metrics available. For example, in Section 2.7.4 and accompanying paragraphs we describe clustering documents using distance functions based on different languages (features).

4. **Transfer of knowledge**: In the context of transfer learning [49], if we treat the graph Laplacian as the target domain, we could transfer knowledge from a different but related graph, which can be viewed as the source domain. We discuss this direction in Section 2.6.3 and 2.7.5.

All the aforementioned side information can be transformed into pairwise ML and CL constraints, which could either be hard (binary) or soft (degree of belief). For example, if the side information comes from a source graph, we can construct pairwise constraints by assuming that the more similar two instance are in the source graph, the more likely they belong to the same cluster in the target graph. Consequently the constraints should naturally be represented by a degree of belief, rather than a binary assertion.

How to make use of these side information to improve clustering falls into the area of constrained clustering [7]. In general, constrained clustering is a category of techniques that try to incorporate ML and CL constraints into existing clustering schemes. It has been well studied on algorithms such as $K$-means clustering, mixture model, hierarchical clustering, and density-based clustering. Previous studies showed that satisfying all constraints at once [16], incrementally [18], or even pruning constraints [17] is intractable. Furthermore, it was shown that algorithms that build set partitions incrementally (such as $K$-means and EM) are prone to being over-constrained [15]. In contrast, incorporating constraints into spectral clustering is a promising direction since, unlike existing algorithms, all data instances are assigned simultaneously to clusters, even if the given constraints are inconsistent.

Constrained spectral clustering is still a developing area. Previous work on this topic can be divided into two categories, based on how they enforce the constraints. The first
category [32, 34, 45, 57, 67] directly manipulate the graph Laplacian (or equivalently, the affinity matrix) according to the given constraints; then unconstrained spectral clustering is applied on the modified graph Laplacian. The second category use constraints to restrict the feasible solution space [13, 20, 43, 68, 69]. Existing methods in both categories share several limitations:

- They are designed to handle only binary constraints. However, as we have stated above, in many real-world applications, constraints are made available in the form of real-valued degree of belief, rather than a yes or no assertion.

- They aim to satisfy as many constraints as possible, which could lead to inflexibility in practice. For example, the given set of constraints could be noisy, and satisfying some of the constraints could actually hurt the overall performance. Also, it is reasonable to ignore a small portion of constraints in exchange for a clustering with much lower cost.

- They do not offer any natural interpretation of either the way that constraints are encoded or the implication of enforcing them.

2.1.2 Our Contributions

In this chapter, we study how to incorporate large amounts of pairwise constraints into spectral clustering, in a flexible manner that addresses the limitations of previous work. Then we show the practical benefits of our approach, including new applications previously not possible.

We extend beyond binary ML/CL constraints and propose a more flexible framework to accommodate general-type side information. We allow the binary constraints to be relaxed to real-valued degree of belief that two data instances belong to the same cluster or two different clusters. Moreover, instead of trying to satisfy each and every constraint that has been given, we use a user-specified threshold to lower bound how well the given constraints must be satisfied. Therefore, our method provides maximum flexibility in terms of both representing constraints and satisfying them. This, in addition to handling large amounts of constraints, allows the encoding of new styles
of information such as entire graphs and alternative distance metrics in their raw form without considering issues such as constraint inconsistencies and over-constraining.

Our contributions are:

- We propose a principled framework for constrained spectral clustering that can incorporate large amounts of both hard and soft constraints.

- We show how to enforce constraints in a flexible way: a user-specified threshold is introduced so that a limited amount of constraints can be ignored in exchange for lower clustering cost. This allows incorporating side information in its raw form without considering issues such as inconsistency and over-constraining.

- We extend the objective function of unconstrained spectral clustering by encoding constraints explicitly and creating a novel constrained optimization problem. Thus our formulation naturally covers unconstrained spectral clustering as a special case.

- We show that our objective function can be turned into a generalized eigenvalue problem, which can be solved deterministically in polynomial time. This is a major advantage over constrained $K$-means clustering, which produces non-deterministic solutions while being intractable even for $K = 2$ [17, 22].

- We interpret our formulation from both the graph cut perspective and the Laplacian embedding perspective.

- We validate the effectiveness of our approach and its advantage over existing methods using standard benchmarks and new innovative applications such as transfer learning.

The rest of the chapter is organized as follows: In Section 2.2 we briefly survey previous work on constrained spectral clustering; Section 2.3 provides preliminaries for spectral clustering; in Section 2.4 we formally introduce our formulation for constrained spectral clustering and show how to solve it efficiently; in Section 2.5 we interpret our
objective from two different perspectives; in Section 2.6 we discuss the implementation of our algorithm and possible extensions; we empirically evaluate our approach in Section 2.7; Section 2.8 concludes the chapter.

2.2 Related Work

Constrained clustering is a category of methods that extend clustering from unsupervised setting to semi-supervised setting, where side information is available in the form of, or can be converted into, pairwise constraints. A number of algorithms have been proposed on how to incorporate constraints into spectral clustering, which can be grouped into two categories.

The first category manipulates the graph Laplacian directly. Kamvar et al. [34] proposed the spectral learning algorithm that sets the \( (i, j) \)-th entry of the affinity matrix to 1 if there is a ML between node \( i \) and \( j \); 0 for CL. A new graph Laplacian is then computed based on the modified affinity matrix. In [67], the constraints are encoded in the same way, but a random walk matrix is used instead of the normalized Laplacian. Kulis et al. [39] proposed to add both positive (for ML) and negative (for CL) penalties to the affinity matrix (they then used kernel \( K \)-means, instead of spectral clustering, to find the partition based on the new kernel). Lu and Carreira-Perpiñán [45] proposed to propagate the constraints in the affinity matrix. In [32, 57], the graph Laplacian is modified by combining the constraint matrix as a regularizer. The limitation of these approaches is that there is no principled way to decide the weights of the constraints, and there is no guarantee that how well the give constraints will be satisfied.

The second category manipulates the eigenspace directly. For example, the subspace trick introduced by [20] alters the eigenspace which the cluster indicator vector is projected onto, based on the given constraints. This technique was later extended in [13] to accommodate inconsistent constraints. Yu and Shi [68, 69] encoded partial grouping information as a subspace projection. Li et al. [43] enforced constraints by regularizing the spectral embedding. This type of approaches usually strictly enforce given constraints. As a result, the results are often over-constrained, which makes the algorithms sensitive
to noise and inconsistencies in the constraint set. Moreover, it is non-trivial to extend these approaches to incorporate soft constraints.

In addition, Gu et al. [26] proposed a spectral kernel design that combines multiple clustering tasks. The learned kernel is constrained in such a way that the data distributions of any two tasks are as close as possible. Their problem setting differs from ours because we aim to perform single-task clustering by using two (disagreeing) data sources. Wang et al. [58] showed how to incorporate pairwise constraints into a penalized matrix factorization framework. Their matrix approximation objective function, which is different from our normalized min-cut objective, is solved by an EM-like algorithm.

We would like to stress that the pros and cons of spectral clustering as compared to other clustering schemes, such as K-means clustering, hierarchical clustering, etc., have been thoroughly studied and well established. We do not claim that constrained spectral clustering is universally superior to other constrained clustering schemes. The goal of this work is to provide a way to incorporate constraints into spectral clustering that is more flexible and principled as compared with existing constrained spectral clustering techniques.

2.3 Background and Preliminaries

We first introduce the standard graph model that is commonly used in the spectral clustering literature. Important notations used throughout the rest of the chapter are listed in Table 2.1.

A collection of $N$ data instances is modeled by an undirected, weighted graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, A)$, where each data instance corresponds to a vertex (node) in $\mathcal{V}$; $\mathcal{E}$ is the edge set and $A$ is the associated affinity matrix. $A$ is symmetric and non-negative. The diagonal matrix $D = \text{diag}(D_{11}, \ldots, D_{NN})$ is called the degree matrix of graph $\mathcal{G}$, where

$$D_{ii} = \sum_{j=1}^{N} A_{ij}.$$  

Then

$$L = D - A$$
Table 2.1. Table of notations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{G}$</td>
<td>An undirected (weighted) graph</td>
</tr>
<tr>
<td>$A$</td>
<td>The affinity matrix</td>
</tr>
<tr>
<td>$D$</td>
<td>The degree matrix</td>
</tr>
<tr>
<td>$I$</td>
<td>The identity matrix</td>
</tr>
<tr>
<td>$L/\bar{L}$</td>
<td>The unnormalized/normalized graph Laplacian</td>
</tr>
<tr>
<td>$Q/Q$</td>
<td>The unnormalized/normalized constraint matrix</td>
</tr>
<tr>
<td>$\text{vol}$</td>
<td>The volume of graph $\mathcal{G}$</td>
</tr>
</tbody>
</table>

is called the unnormalized graph Laplacian of $\mathcal{G}$. Assuming $\mathcal{G}$ is connected (i.e. any node is reachable from any other node), $L$ has the following properties:

Property 2.3.1 (Properties of graph Laplacian [55]). Let $L$ be the graph Laplacian of a connected graph, then:

1. $L$ is symmetric and positive semi-definite.
2. $L$ has one and only one eigenvalue equal to 0, and $N - 1$ positive eigenvalues: 
   \[ 0 = \lambda_0 < \lambda_1 \leq \ldots \leq \lambda_{N-1}. \]
3. $\mathbf{1}$ is an eigenvector of $L$ with eigenvalue 0 ($\mathbf{1}$ is a constant vector whose entries are all 1).

Shi and Malik [51] showed that the eigenvectors of the graph Laplacian can be related to the normalized min-cut (Ncut) of $\mathcal{G}$. The objective function can be written as:

\[
\arg\min_{\mathbf{v} \in \mathbb{R}^N} \mathbf{v}^T \bar{L} \mathbf{v}, \text{ s.t. } \mathbf{v}^T \mathbf{v} = \text{vol}, \; \mathbf{v} \perp D^{1/2} \mathbf{1}. \tag{2.1}
\]

Here

\[ \bar{L} = D^{-1/2} LD^{-1/2} \]

is called the normalized graph Laplacian [55]; $\text{vol} = \sum_{i=1}^{N} D_{ii}$ is the volume of $\mathcal{G}$; the first constraint $\mathbf{v}^T \mathbf{v} = \text{vol}$ normalizes $\mathbf{v}$; the second constraint $\mathbf{v} \perp D^{1/2} \mathbf{1}$ rules out the
principal eigenvector of $\bar{L}$ as a trivial solution, because it does not define a meaningful cut on the graph. The relaxed cluster indicator $u$ can be recovered from $v$ as:

$$u = D^{-1/2}v.$$ 

Note that the result of spectral clustering is solely decided by the affinity structure of graph $\mathcal{G}$ as encoded in the matrix $A$ (and thus the graph Laplacian $L$). We will then describe our extensions on how to incorporate side information so that the result of clustering will reflect both the affinity structure of the graph and the structure of the side information.

## 2.4 A Flexible Framework for Constrained Spectral Clustering

In this section, we show how to incorporate side information into spectral clustering as pairwise constraints. Our formulation allows both hard and soft constraints. We propose a new constrained optimization formulation for constrained spectral clustering. Then we show how to solve the objective function by converting it into a generalized eigenvalue system.

### 2.4.1 The Objective Function

We encode side information with an $N \times N$ constraint matrix $Q$. Traditionally, constrained clustering only accommodates binary constraints, namely Must-Link and Cannot-Link:

$$Q_{ij} = Q_{ji} = \begin{cases} 
+1 & \text{if } ML(i, j) \\
-1 & \text{if } CL(i, j) \\
0 & \text{no side information available}
\end{cases}.$$ 

Let $u \in \{-1, +1\}^N$ be a cluster indicator vector, where $u_i = +1$ if node $i$ belongs to cluster $+$ and $u_i = -1$ if node $i$ belongs to cluster $-$, then

$$u^T Q u = \sum_{i=1}^{N} \sum_{j=1}^{N} u_i u_j Q_{ij}$$
is a measure of how well the constraints in $Q$ are satisfied by the assignment $u$: the measure will increase by 1 if $Q_{ij} = 1$ and node $i$ and $j$ have the same sign in $u$; it will decrease by 1 if $Q_{ij} = 1$ but node $i$ and $j$ have different signs in $u$ or $Q_{ij} = -1$ but node $i$ and $j$ have the same sign in $u$.

We extend the above encoding scheme to accommodate soft constraints by relaxing the cluster indicator vector $u$ as well as the constraint matrix $Q$ such that:

$$u \in \mathbb{R}^N, Q \in \mathbb{R}^{N \times N}.$$  

$Q_{ij}$ is positive if we believe nodes $i$ and $j$ belong to the same cluster; $Q_{ij}$ is negative if we believe nodes $i$ and $j$ belong to different clusters; the magnitude of $Q_{ij}$ indicates how strong the belief is.

Consequently, $u^T Q u$ becomes a real-valued measure of how well the constraints in $Q$ are satisfied in the relaxed sense. For example, $Q_{ij} < 0$ means we believe nodes $i$ and $j$ belong to different clusters; in order to improve $u^T Q u$, we should assign $u_i$ and $u_j$ with values of different signs; similarly, $Q_{ij} > 0$ means nodes $i$ and $j$ are believed to belong to the same cluster; we should assign $u_i$ and $u_j$ with values of the same sign. The larger $u^T Q u$ is, the better the cluster assignment $u$ conforms to the given constraints in $Q$.

Now given this real-valued measure, rather than trying to satisfy all the constraints in $Q$ individually, we can lower-bound this measure with a constant $\alpha \in \mathbb{R}$:

$$u^T Q u \geq \alpha.$$  

Following the notations in Eq.(2.1), we substitute $u$ with $D^{-1/2}v$, above inequality becomes

$$v^T \bar{Q} v \geq \alpha,$$

where

$$\bar{Q} = D^{-1/2} Q D^{-1/2}$$

is the normalized constraint matrix.

We append this lower-bound constraint to the objective function of unconstrained spectral clustering in Eq.(2.1) and we have:
Problem 2.4.1 (Constrained Spectral Clustering). Given a normalized graph Laplacian $\bar{L}$, a normalized constraint matrix $\bar{Q}$ and a threshold $\alpha$, we want to optimize the following objective function:

$$\arg\min_{v \in \mathbb{R}^N} v^T \bar{L} v, \text{ s.t. } v^T \bar{Q} v \geq \alpha, \ v^T v = \text{vol}, \ v \neq D^{1/2} 1.$$  \hfill (2.2)

Here $v^T \bar{L} v$ is the cost of the cut we want to minimize; the first constraint $v^T \bar{Q} v \geq \alpha$ is to lower bound how well the constraints in $Q$ are satisfied; the second constraint $v^T v = \text{vol}$ normalizes $v$; the third constraint $v \neq D^{1/2} 1$ rules out the trivial solution $D^{1/2} 1$. Suppose $v^*$ is the optimal solution to Eq.(2.2), then $u^* = D^{-1/2} v^*$ is the optimal cluster indicator vector.

It is easy to see that the unconstrained spectral clustering in Eq.(2.1) is covered as a special case of Eq.(2.2) where $\bar{Q} = I$ (no constraints are encoded) and $\alpha = \text{vol}$ ($v^T \bar{Q} v \geq \text{vol}$ is trivially satisfied given $\bar{Q} = I$ and $v^T v = \text{vol}$).

2.4.2 Solving the Objective Function

To solve a constrained optimization problem, we follow the Karush-Kuhn-Tucker Theorem [38] to derive the necessary conditions for the optimal solution to the problem. We can find a set of candidates, or feasible solutions, that satisfy all the necessary conditions. Then we choose the optimal solution among the feasible solutions using brute-force method, given the size of the feasible set is finite and small.

For our objective function in Eq.(2.2), we introduce the Lagrange multipliers as follows:

$$\Lambda(v, \lambda, \mu) = v^T \bar{L} v - \lambda (v^T \bar{Q} v - \alpha) - \mu (v^T v - \text{vol}).$$  \hfill (2.3)

Then according to the KKT Theorem, any feasible solution to Eq.(2.2) must satisfy the following conditions:

$$\text{(Stationarity) } \bar{L} v - \lambda \bar{Q} v - \mu v = 0,$$  \hfill (2.4)

$$\text{(Primal feasibility) } v^T \bar{Q} v \geq \alpha, v^T v = \text{vol},$$  \hfill (2.5)

$$\text{(Dual feasibility) } \lambda \geq 0,$$  \hfill (2.6)

$$\text{(Complementary slackness) } \lambda (v^T \bar{Q} v - \alpha) = 0.$$  \hfill (2.7)
Note that Eq.(2.4) comes from taking the derivative of Eq.(2.3) with respect to \( v \). Also note that we dismiss the constraint \( v \neq D^{1/2}1 \) at this moment, because it can be checked independently after we find the feasible solutions.

To solve Eq.(2.4)-(2.7), we start with looking at the complementary slackness requirement in Eq.(2.7), which implies either \( \lambda = 0 \) or \( v^T \bar{Q}v - \alpha = 0 \). If \( \lambda = 0 \), we will have a trivial problem because the second term from Eq.(2.4) will be eliminated and the problem will be reduced to unconstrained spectral clustering. Therefore in the following we focus on the case where \( \lambda \neq 0 \). In this case, for Eq.(2.7) to hold \( v^T \bar{Q}v - \alpha \) must be 0. Consequently the KKT conditions become:

\[
\begin{align*}
\bar{L}v - \lambda \bar{Q}v - \mu v &= 0, \\
v^T v &= \text{vol}, \\
v^T \bar{Q}v &= \alpha, \\
\lambda &> 0.
\end{align*}
\] (2.8-2.11)

Under the assumption that \( \alpha \) is arbitrarily assigned by user and \( \lambda \) and \( \mu \) are independent variables, Eq.(2.8-2.11) cannot be solved explicitly, and it may produce infinite number of feasible solutions, if one exists. As a workaround, we temporarily drop the assumption that \( \alpha \) is an arbitrary value assigned by the user. Instead, we assume \( \alpha \triangleq v^T \bar{Q}v \), i.e. \( \alpha \) is defined as such that Eq.(2.10) holds. Then we introduce an auxiliary variable, \( \beta \), which is defined as the ratio between \( \mu \) and \( \lambda \):

\[
\beta \triangleq -\frac{\mu}{\lambda \text{vol}}.
\] (2.12)

Now we substitute Eq.(2.12) into Eq.(2.8) we obtain:

\[
\bar{L}v - \lambda \bar{Q}v + \frac{\lambda \beta}{\text{vol}} v = 0,
\]

or equivalently:

\[
\bar{L}v = \lambda (\bar{Q} - \frac{\beta}{\text{vol}} I)\nu
\] (2.13)

We immediately notice that Eq.(2.13) is a generalized eigenvalue problem for a given \( \beta \).

Next we show that the substitution of \( \alpha \) with \( \beta \) does not compromise our original intention of lower bounding \( v^T \bar{Q}v \) in Eq.(2.2).
**Lemma 2.4.2.** $\beta < v^T\bar{Q}v$.

**Proof** Let $\gamma \triangleq v^T\bar{L}v$, by left-hand multiplying $v^T$ to both sides of Eq.(2.13) we have

$$v^T\bar{L}v = \lambda v^T(\bar{Q} - \frac{\beta}{vol}I)v.$$  

Then incorporating Eq.(2.9) and $\alpha \triangleq v^T\bar{Q}v$ we have

$$\gamma = \lambda(\alpha - \beta).$$

Now recall that $L$ is positive semi-definite (Property 2.3.1), and so is $\bar{L}$, which means

$$\gamma = v^T\bar{L}v > 0, \forall v \neq D^{1/2}1.$$  

Consequently, we have

$$\alpha - \beta = \frac{\gamma}{\lambda} > 0 \implies v^T\bar{Q}v = \alpha > \beta.$$

\[ \square \]

In summary, our algorithm works as follows (the exact implementation is shown in Algorithm 1):

1. **Generating candidates**: The user specifies a value for $\beta$, and we solve the generalized eigenvalue system given in Eq.(2.13). Note that both $\bar{L}$ and $\bar{Q} - \beta/volI$ are Hermitian matrices, thus the generalized eigenvalues are guaranteed to be real numbers.

2. **Finding the feasible set**: Removing generalized eigenvectors associated with non-positive eigenvalues, and normalize the rest such that $v^Tv = vol$. Note that the trivial solution $D^{1/2}1$ is automatically removed in this step because if it is a generalized eigenvector in Eq.(2.13), the associated eigenvalue would be 0. Since we have at most $N$ generalized eigenvectors, the number of feasible eigenvectors is at most $N$.

3. **Choosing the optimal solution**: We choose from the feasible solutions the one that minimizes $v^T\bar{L}v$, say $v^*$. According to Lemma 2.4.2, $v^*$ is the optimal solution to the objective function in Eq.(2.2) for any given $\beta$ and $\beta < \alpha = v^*T\bar{Q}v^*$. 

-19-
Algorithm 1: Constrained Spectral Clustering

**Input**: Affinity matrix $A$, constraint matrix $Q$, $\beta$;

**Output**: The optimal (relaxed) cluster indicator $u^*$;

1. $vol \leftarrow \sum_{i=1}^N \sum_{j=1}^N A_{ij}$, $D \leftarrow \text{diag}(\sum_{j=1}^N A_{ij})$;
2. $\bar{L} \leftarrow I - D^{-1/2}AD^{-1/2}$, $\bar{Q} \leftarrow D^{-1/2}QD^{-1/2}$;
3. $\lambda_{\text{max}}(\bar{Q}) \leftarrow$ the largest eigenvalue of $\bar{Q}$;
4. if $\beta \geq \lambda_{\text{max}}(\bar{Q}) \cdot vol$ then
5. return $u^* = \emptyset$;
6. end
7. else
8. Solve the generalized eigenvalue system in Eq.(2.13);
9. Remove eigenvectors associated with non-positive eigenvalues and normalize the rest by $v \leftarrow \frac{v}{\|v\|} \sqrt{vol}$;
10. $v^* \leftarrow \text{argmin}_v v^T \bar{L} v$, where $v$ is among the feasible eigenvectors generated in the previous step;
11. return $u^* \leftarrow D^{-1/2}v^*$;
12. end

2.4.3 A Sufficient Condition for the Existence of Solutions

On one hand, our method described above is guaranteed to generate a finite number of feasible solutions. On the other hand, we need to set $\beta$ appropriately so that the generalized eigenvalue system in Eq.(2.13) combined with the KKT conditions in Eq.(2.8-2.11) will give rise to at least one feasible solution. In this section, we discuss such a sufficient condition:

$$\beta < \lambda_{\text{max}}(\bar{Q}) \cdot vol,$$

where $\lambda_{\text{max}}(\bar{Q})$ is the largest eigenvalue of $\bar{Q}$. In this case, the matrix on the right hand side of Eq.(2.13), namely $\bar{Q} - \beta/vol \cdot I$, will have at least one positive eigenvalue. Consequently, the generalized eigenvalue system in Eq.(2.13) will have at least one positive eigenvalue. Moreover, the number of feasible eigenvectors will increase if we make $\beta$
smaller. For example, if we set $\beta < \lambda_{\min}(\bar{Q})\cdot\text{vol}$, $\lambda_{\min}(\bar{Q})$ to be the smallest eigenvalue of $\bar{Q}$, then $\bar{Q} - \beta/\text{vol} \cdot I$ becomes positive definite. Then the generalized eigenvalue system in Eq.(2.13) will generate $N - 1$ feasible eigenvectors (the trivial solution $D^{1/2}1$ with eigenvalue 0 is dropped).

In practice, we normally choose the value of $\beta$ within the range

$$(\lambda_{\min}(\bar{Q}) \cdot \text{vol}, \lambda_{\max}(\bar{Q}) \cdot \text{vol}).$$

In that range, the greater $\beta$ is, the more the solution will be biased towards satisfying $\bar{Q}$. Again, note that whenever we have $\beta < \lambda_{\max}(\bar{Q}) \cdot \text{vol}$, the value of $\alpha$ will always be bounded by

$$\beta < \alpha \leq \lambda_{\max}\text{vol}.$$ 

Therefore we do not need to take care of $\alpha$ explicitly.

### 2.4.4 An Illustrative Example

To illustrate how our algorithm works, we present a toy example as follows. In Fig. 2.2, we have a graph associated with the following affinity matrix:

$$A = \begin{bmatrix}
0 & 1 & 1 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 0 \\
\end{bmatrix}$$

Unconstrained spectral clustering will cut the graph at edge (3, 4) and split it into two symmetric parts \{1, 2, 3\} and \{4, 5, 6\} (Fig. 2.3(a)).
Figure 2.3. The solutions to the illustrative example in Fig. 2.2 with different $\beta$. The x-axis is the indices of the instances and the y-axis is the corresponding entry values in the optimal (relaxed) cluster indicator $u^*$. Notice that node 4 is biased toward nodes $\{1, 2, 3\}$ as $\beta$ increases.

Then we introduce constraints as encoded in the following constraint matrix:

$$Q = \begin{bmatrix}
+1 & +1 & +1 & +1 & -1 & -1 \\
+1 & +1 & +1 & +1 & -1 & -1 \\
+1 & +1 & +1 & +1 & -1 & -1 \\
+1 & +1 & +1 & +1 & -1 & -1 \\
-1 & -1 & -1 & -1 & +1 & +1 \\
-1 & -1 & -1 & -1 & +1 & +1
\end{bmatrix}. $$

$Q$ is essentially saying that we want to group nodes $\{1, 2, 3, 4\}$ into one cluster and $\{5, 6\}$ the other. Although this kind of “complete information” constraint matrix does not happen in practice, we use it here only to make the result more explicit and intuitive.

$Q$ has two distinct eigenvalues: 0 and 2.6667. As analyzed above, $\beta$ must be smaller than 2.6667$\text{vol}$ to guarantee the existence of a feasible solution, and larger $\beta$ means we want more constraints in $Q$ to be satisfied (in a relaxed sense). Thus we set $\beta$ to $\text{vol}$ and $2\text{vol}$ respectively, and see how it will affect the resultant constrained cuts. We solve the generalized eigenvalue system in Eq.(2.13), and plot the cluster indicator vector $u^*$ in Fig. 2.3(b) and 2.3(c), respectively. We can see that as $\beta$ increases, node 4 is dragged from the group of nodes $\{5, 6\}$ to the group of nodes $\{1, 2, 3\}$, which conforms to our expectation that greater $\beta$ value implies better constraint satisfaction.
2.5 Interpretations of Our Formulation

2.5.1 A Graph Cut Interpretation

Unconstrained spectral clustering can be interpreted as finding the Ncut of an unlabeled graph. Similarly, our formulation of constrained spectral clustering in Eq. (2.2) can be interpreted as finding the Ncut of a labeled/colored graph.

Specifically, suppose we have an undirected weighted graph. The nodes of the graph are colored in such a way that nodes of the same color are advised to be assigned into the same cluster while nodes of different colors are advised to be assigned into different clusters (e.g. Fig. 2.2). Let \( \mathbf{v}^* \) be the solution to the constrained optimization problem in Eq. (2.2). We cut the graph into two parts based on the values of the entries of \( \mathbf{u}^* = \mathbf{D}^{-1/2} \mathbf{v}^* \). Then \( \mathbf{v}^T \tilde{\mathbf{L}} \mathbf{v}^* \) can be interpreted as the cost of the cut (in a relaxed sense), which we minimize. On the other hand, \( \alpha = \mathbf{v}^T \tilde{\mathbf{Q}} \mathbf{v}^* = \mathbf{u}^T \mathbf{Q} \mathbf{u}^* \) can be interpreted as the purity of the cut (also in a relaxed sense), according to the color of the nodes in respective sides. For example, if \( \mathbf{Q} \in \{-1, 0, 1\}^{N \times N} \) and \( \mathbf{u}^* \in \{-1, 1\}^N \), then \( \alpha \) equals to the number of constraints in \( \mathbf{Q} \) that are satisfied by \( \mathbf{u}^* \) minus the number of constraints violated. More generally, if \( Q_{ij} \) is a positive number, then \( u_i^* \) and \( u_j^* \) having the same sign will contribute positively to the purity of the cut, whereas different signs will contribute negatively to the purity of the cut. It is not difficult to see that the purity can be maximized when there is no pair of nodes with different colors that are assigned to the same side of the cut (0 violations), which is the case where all constraints in \( \mathbf{Q} \) are satisfied.

2.5.2 A Geometric Interpretation

We can also interpret our formulation as constraining the joint numerical range [27] of the graph Laplacian and the constraint matrix. Specifically, we consider the joint numerical range:

\[
J(\mathbf{L}, \mathbf{Q}) \triangleq \{ (\mathbf{v}^T \tilde{\mathbf{L}} \mathbf{v}, \mathbf{v}^T \tilde{\mathbf{Q}} \mathbf{v}) : \mathbf{v}^T \mathbf{v} = 1 \}.
\] (2.14)
$J(\bar{L}, \bar{Q})$ essentially maps all possible cuts $\mathbf{v}$ to a 2-D plane, where the $x$-coordinate corresponds to the cost of the cut, and the $y$-axis corresponds to the constraint satisfaction of the cut. According to our objective in Eq. (2.2), we want to minimize the first term while lower-bounding the second term. Therefore, we are looking for the leftmost point among those that are above the horizontal line $y = \alpha$.

In Fig. 2.4(c), we visualize $J(\bar{L}, \bar{Q})$ by plotting all the unconstrained cuts given by spectral clustering and all the constrained cuts given by our algorithm in the joint numerical range, based on the graph Laplacian of a Two-Moon dataset with a randomly generated constraint matrix. The horizontal line and the arrow indicate the constrained area from which we can select feasible solutions. We can see that most of the unconstrained cuts proposed by spectral clustering are far below the threshold, which suggests spectral clustering cannot lead to the ground truth partition (as shown in Fig. 2.4(b)) without the help of constraints.

2.6 Implementation and Extensions

In this section, we discuss some implementation issues of our method. Then we extend our algorithm to several practical settings, namely $K$-way partition, transfer learning, and non-parametric solution.

2.6.1 Constrained Spectral Clustering for 2-Way Partition

The routine of our method is similar to that of unconstrained spectral clustering. The input of the algorithm is an affinity matrix $A$, the constraint matrix $Q$, and a threshold $\beta$. Then we solve the generalized eigenvalue problem in Eq. (2.13) and find all the feasible generalized eigenvectors. The output is the optimal (relaxed) cluster assignment indicator $\mathbf{u}^*$. In practice, a partition is often derived from $\mathbf{u}^*$ by assigning nodes corresponding to the positive entries in $\mathbf{u}^*$ to one side of the cut, and negative entries to the other side. Our algorithm is summarized in Algorithm 1.

Since our model encodes soft constraints as degree of belief, inconsistent constraints in $Q$ will not corrupt our algorithm. Instead, they are enforced implicitly by maximizing $\mathbf{u}^T \mathbf{Q} \mathbf{u}$. Note that if the constraint matrix $Q$ is generated from a partial labeling, then
Figure 2.4. The joint numerical range of the normalized graph Laplacian $\bar{L}$ and the normalized constraint matrix $\bar{Q}$, as well as the optimal solutions to unconstrained/constrained spectral clustering.
the constraints in $Q$ will always be consistent.

**Runtime analysis:** The runtime of our algorithm is dominated by that of the generalized eigendecomposition. In other words, the complexity of our algorithm is on a par with that of unconstrained spectral clustering in big-O notation, which is $O(kN^2)$, $N$ to be the number of data instances and $k$ to be the number of eigenpairs we need to compute. Here $k$ is a number large enough to guarantee the existence of feasible solutions. In practice we normally have $2 < k \ll N$.

### 2.6.2 Extension to $K$-Way Partition

Our algorithm can be naturally extended to $K$-way partition for $K > 2$, following what we usually do for unconstrained spectral clustering [55]: instead of only using the optimal feasible eigenvector $u^*$, we preserve top-$(K - 1)$ eigenvectors associated with positive eigenvalues, and perform $K$-means algorithm based on that embedding.

Specifically, the constraint matrix $Q$ follows the same encoding scheme: $Q_{ij} > 0$ if node $i$ and $j$ are believed to belong to the same cluster, $Q_{ij} < 0$ otherwise. To guarantee we can find $K - 1$ feasible eigenvectors, we set the threshold $\beta$ such that

$$\beta < \lambda_{K-1} \text{vol},$$

where $\lambda_{K-1}$ is the $(K-1)$-th largest eigenvalue of $\bar{Q}$. Given all the feasible eigenvectors, we pick the top $K - 1$ in terms of minimizing $v^T \bar{L} v$ \footnote{Here we assume the trivial solution, the eigenvector with all 1’s, has been excluded.}. Let the $K - 1$ eigenvectors form the columns of $V \in \mathbb{R}^{N \times (K-1)}$. We perform $K$-means clustering on the rows of $V$ and get the final clustering. Algorithm 2 shows the complete routine.

Note that $K$-means is only one of many possible discretization techniques that can derive a $K$-way partition from the relaxed indicator matrix $D^{-1/2}V^*$. Due to the orthogonality of the eigenvectors, they can be independently discretized first, e.g. we can replace Step 11 of Algorithm 2 with:

$$u^* \leftarrow \text{kmeans}(\text{sign}(D^{-1/2}V^*), K).$$

This additional step can help alleviate the influence of possible outliers on the $K$-means step in some cases.
Algorithm 2: Constrained Spectral Clustering for $K$-way Partition

**Input:** Affinity matrix $A$, constraint matrix $Q$, $\beta$, $K$;

**Output:** The cluster assignment indicator $u^*$;

1. $vol \leftarrow \sum_{i=1}^{N} \sum_{j=1}^{N} A_{ij}$, $D \leftarrow \text{diag}(\sum_{j=1}^{N} A_{ij})$;
2. $\bar{L} \leftarrow I - D^{-1/2} A D^{-1/2}$, $\bar{Q} \leftarrow D^{-1/2} Q D^{-1/2}$;
3. $\lambda_{\text{max}} \leftarrow$ the largest eigenvalue of $\bar{Q}$;
4. if $\beta \geq \lambda_{K-1} vol$ then
   5. return $u^* = \emptyset$;
else
   8. Solve the generalized eigenvalue system in Eq.(2.13);
   9. Remove eigenvectors associated with non-positive eigenvalues and normalize
the rest by $v \leftarrow \frac{v}{\|v\|} \sqrt{vol}$;
10. $V^* \leftarrow \arg\min_{V \in \mathbb{R}^{N \times (K-1)}} \text{trace}(V^T \bar{L} V)$, where the columns of $V$ are a subset of
the feasible eigenvectors generated in the previous step;
11. return $u^* \leftarrow \text{kmeans}(D^{-1/2} V^*, K)$;
end

Moreover, notice that the feasible eigenvectors, which are the columns of $V^*$, are
treated equally in Eq.(2.15). This may not be ideal in practice because these candidate
cuts are not equally favored by graph $G$, i.e. some of them have higher costs than the
other. Therefore, we can weight the columns of $V^*$ with the inverse of their respective
costs:

$$u^* \leftarrow \text{kmeans}(\text{sign}(D^{-1/2} V^* (V^* \bar{L} V^*)^{-1}), K).$$ (2.16)

#### 2.6.3 Using Constrained Spectral Clustering for Transfer Learning

The constrained spectral clustering framework naturally fits into the scenario of transfer
learning between two graphs. Assume we have two graphs, a source graph and a target
graph, which share the same set of nodes but have different sets of edges (or edge weights). The goal is to transfer knowledge from the source graph so that we can improve the cut on the target graph. The knowledge to transfer is derived from the source graph in the form of soft constraints.

Specifically, let $G_S(V, E_S)$ be the source graph, $G_T(V, E_T)$ the target graph. $A_S$ and $A_T$ are their respective affinity matrices. Then $A_S$ can be considered as a constraint matrix with only ML constraints. It carries the structural knowledge from the source graph, and we can transfer it to the target graph using our constrained spectral clustering formulation:

$$
\underset{v \in \mathbb{R}^N}{\text{argmin}} v^T \bar{L}_T v, \quad \text{s.t.} \quad v^T A_S v \geq \alpha, \quad v^T v = \text{vol}, \quad v \neq D_T^{1/2} 1.
$$

(2.17)

$\alpha$ is now the lower bound of how much knowledge from the source graph must be enforced on the target graph. To solution to this is similar to Eq.(2.13):

$$
\bar{L}_T v = \lambda (\bar{A}_S - \frac{\beta}{\text{vol}(G_T)} I) v
$$

(2.18)

Note that since the largest eigenvalue of $\bar{A}_S$ corresponds to a trivial cut, in practice we should set the threshold such that $\beta < \lambda_1 \text{vol}$, $\lambda_1$ to be the second largest eigenvalue of $\bar{A}_S$. This will guarantee a feasible eigenvector that is non-trivial.

### 2.6.4 A Non-Parametric Version of Our Algorithm

In some applications the constraint matrix $Q$ can be guaranteed to be positive semi-definite (PSD). For instance, $Q$ may be derived from an alternative kernel (e.g., Gaussian or cosine similarity) for the same set of instances. In that case, both $\bar{L}$ and $\bar{Q}$ are both PSD. Instead of lower-bounding $v^T \bar{Q} v$ using $\alpha$, we consider the following cost-satisfaction ratio for any cut $v$:

$$
f(v) \triangleq \frac{v^T \bar{L} v}{v^T \bar{Q} v}.
$$

Since $\bar{Q}$ is now guaranteed to be PSD, we have

$$
f(v) \in [0, \infty), \forall v \in \mathbb{R}^N.
$$
Algorithm 3: Non-Parametric Constrained Spectral Clustering (csp-n)

**Input:** $\mathbf{L}, \mathbf{Q}, K$;

**Output:** $\mathbf{u}$;

1. Solve the generalized eigenvalue problem $\mathbf{L} \mathbf{v} = \lambda \mathbf{Q} \mathbf{v}$;
2. Let $\mathcal{V} = \{ \mathbf{v}_i \}_{i=1}^N$ be the set of all generalized eigenvectors;
3. $\mathcal{V} \leftarrow \{ \}$;
4. for $i = 1$ to $K$ do
5. $\mathbf{v}^* = \arg\min_{\mathbf{v} \in \mathcal{V}} \frac{\mathbf{v}^T \mathbf{L} \mathbf{v}}{\mathbf{v}^T \mathbf{Q} \mathbf{v}}$;
6. Remove $\mathbf{v}^*$ from $\mathcal{V}$;
7. $\mathcal{V} \leftarrow [\mathcal{V}, \mathbf{v}^*]$;
8. end
9. return $\mathbf{u} \leftarrow K\text{means}(\mathcal{V}, K)$;

The general goal of constrained spectral clustering is to maximize $\mathbf{v}^T \mathbf{Q} \mathbf{v}$ and minimize $\mathbf{v}^T \mathbf{L} \mathbf{v}$, which is equivalent to minimize $f(\mathbf{v})$. Therefore $f(\mathbf{v})$ becomes a unified measure for the quality of the cut $\mathbf{v}$: smaller $f(\mathbf{v})$ means better cut.

Formally, the non-parametric version of our objective is:

$$\arg\min_{\mathbf{v}_i \in \mathbb{R}^N} \sum_{i=1}^K \frac{\mathbf{v}_i^T \mathbf{L} \mathbf{v}_i}{\mathbf{v}_i^T \mathbf{Q} \mathbf{v}_i}, \text{ s.t. } \mathbf{v}_i^T \mathbf{v}_i = 1, \forall i, \text{ and } \mathbf{v}_i \perp \mathbf{v}_j, \forall i \neq j. \quad (2.19)$$

Consider the generalized eigenvalue problem:

$$\mathbf{L} \mathbf{v} = \lambda \mathbf{Q} \mathbf{v}. \quad (2.20)$$

Since both $\mathbf{L}$ and $\mathbf{Q}$ are Hermitian and PSD, we will have $N$ real generalized eigenvectors [5], and they are the critical points for the generalized Rayleigh quotient $f(\mathbf{v})$ [27]. The complete algorithm is listed in Algorithm 3.

### 2.7 Testing and Innovative Uses of Our Work

We begin with three sets of experiments to test our approach on standard spectral clustering data sets. We then show that since our approach can handle large amounts
of soft constraints in a flexible fashion, this opens up two innovative uses of our work: encoding multiple metrics for translated document clustering and transfer learning for fMRI analysis.

We aim to answer the following questions with the empirical study:

- Can our algorithm effectively incorporate side information and generate semantically meaningful partitions?
- Does our algorithm converge to the underlying ground truth partition as more constraints are provided?
- How does our algorithm perform on real-world datasets, as evaluated against ground truth labeling, with comparison to existing techniques?
- How well does our algorithm handle soft constraints?
- How well does our algorithm handle large amounts of constraints?

Recall that in Section 2.1 we listed four different types of side information: explicit pairwise constraints, partial labeling, alternative metrics, and transfer of knowledge. The empirical results presented in this section are arranged accordingly.

### 2.7.1 Explicit Pairwise Constraints: Image Segmentation

We demonstrate the effectiveness of our algorithm for image segmentation using explicit pairwise constraints assigned by users.

We choose the image segmentation application for several reasons: 1) it is one of the applications where spectral clustering significantly outperforms other clustering techniques, e.g. K-means; 2) the results of image segmentation can be easily interpreted and evaluated by human; 3) instead of generating random constraints, we can provide semantically meaningful constraints to see if the constrained partition conforms to our expectation.

The images we used were chosen from the Berkeley Segmentation Dataset and Benchmark [47]. The original images are 480 × 320 grayscale images in jpeg format. For efficiency consideration, we compressed them to 10% of the original size, which is 48 × 32
Figure 2.5. Segmentation of the elephant image. The images are reconstructed based on the relaxed cluster indicator $u^*$. Pixels that are closer to the red end of the spectrum belong to one segment and blue the other. The labeled pixels are as bounded by the black and white rectangles.

pixels, as shown in Fig. 2.5(a) and 2.6(a). Then affinity matrix of the image was computed using the RBF kernel, based on both the positions and the grayscale values of the pixels. As a baseline, we used unconstrained spectral clustering [51] to generate a 2-segmentation of the image. Then we introduced different sets of constraints to see if they can generate expected segmentation. Note that the results of image segmentation vary with the number of segments. To save us from the complications of parameter tuning, which is irrelevant to the contribution of this work, we always set the number of segments to be 2.

The results are shown in Fig. 2.5 and 2.6. To visualize the resultant segmentation, we reconstructed the image using the entry values in the relaxed cluster indicator vector $u^*$. In Fig. 2.5(b), the unconstrained spectral clustering partitioned the elephant image into two parts: the sky (red pixels) and the two elephants and the ground (blue pixels). This
is not satisfying in the sense that it failed to isolate the elephants from the background (the sky and the ground). To correct this, we introduced constraints by labeling two $5 \times 5$ blocks to be 1 (as bounded by the black rectangles in Fig. 2.5(c)): one at the upper-right corner of the image (the sky) and the other at the lower-right corner (the ground); we also labeled two $5 \times 5$ blocks on the heads of the two elephants to be $-1$ (as bounded by the white rectangles in Fig. 2.5(c)). To generate the constraint matrix $Q$, a ML was added between every pair of pixels with the same label and a CL was added between every pair of pixels with different labels. The parameter $\beta$ was set to

$$\beta = \lambda_{\text{max}} \times \text{vol} \times (0.5 + 0.4 \times \frac{\# \text{ of constraints}}{N^2}),$$

(2.21)

where $\lambda_{\text{max}}$ is the largest eigenvalue of $\bar{Q}$. In this way, $\beta$ is always between $0.5\lambda_{\text{max}} \text{vol}$ and $0.9\lambda_{\text{max}} \text{vol}$, and it will gradually increase as the number of constraints increases. From Fig. 2.5(c) we can see that with the help of user supervision, our method successfully isolated the two elephants (blue) from the background, which is the sky and the ground (red). Note that our method achieved this with very simple labeling: four squared blocks.

To show the flexibility of our method, we tried a different set of constraints on
the same elephant image with the same parameter settings. This time we aimed to separate the two elephants from each other, which is impossible in the unconstrained case because the two elephants are not only similar in color (grayscale value) but also adjacent in space. Again we used two $5 \times 5$ blocks (as bounded by the black and white rectangles in Fig. 2.5(d)), one on the head of the elephant on the left, labeled to be 1, and the other on the body of the elephant on the right, labeled to be $-1$. As shown in Fig. 2.5(d), our method cut the image into two parts with one elephant on the left (blue) and the other on the right (red), just like what a human user would do.

Similarly, we applied our method on a human face image as shown in Fig. 2.6(a). The unconstrained spectral clustering failed to isolate the human face from the background (Fig. 2.6(b)). This is because the tall hat breaks the spatial continuity between the left side of the background and the right side. Then we labeled two $5 \times 3$ blocks to be in the same class, one on each side of the background. As we intended, our method assigned the background of both sides into the same cluster and thus isolated the human face with his tall hat from the background (Fig. 2.6(c)). Again, this was achieved simply by labeling two blocks in the image, which covered about 3% of all pixels. Alternatively, if we labeled a $5 \times 5$ block in the hat to be 1, and a $5 \times 5$ block in the face to be $-1$, the resultant clustering will isolate the hat from the rest of the image (Fig. 2.6(d)).

### 2.7.2 Explicit Pairwise Constraints: The Double Moon Dataset

We further examine the behavior of our algorithm on a synthetic dataset using explicit constraints that are derived from underlying ground truth.

We claim that our formulation is a natural extension to spectral clustering. The question to ask then is whether the output of our algorithm converges to that of spectral clustering. More specifically, consider the ground truth partition defined by performing spectral clustering on an ideal distribution. We draw an imperfect sample from the distribution, on which spectral clustering is unable to find the ground truth partition. Then we perform our algorithm on this imperfect sample. As more and more constraints are provided, we want to know whether or not the partition found by our algorithm would converge to the ground truth partition.
To answer the question, we used the Double Moon distribution. As shown in Fig. 2.1, spectral clustering is able to find the two moons when the sample is dense enough. In Fig. 2.7(a), we generated an under-sampled instance of the distribution with 100 data points, on which unconstrained spectral clustering could no longer find the ground truth partition. Then we performed our algorithm on this imperfect sample, and compared the partition found by our algorithm to the ground truth partition in terms of adjusted Rand index (ARI, [31]). ARI indicates how well a given partition conform to the ground truth: 0 means the given partition is no better than a random assignment; 1 means the given partition matches the ground truth exactly. For each random sample, we generated 50 random sets of constraints and recorded the average ARI. We repeated the process on 10 different random samples of the same size and reported the results in Fig. 2.7(b). We can see that our algorithm consistently converge to the ground truth result as more constraints are provided. Notice that there is performance drop when an extreme small number of constraints are provided (less than 10), which is expected because such small number of constraints are insufficient to hint a better partition, and consequentially lead to random perturbation to the results. As more constraints were provided, the results were quickly stabilized.

To illustrate the robustness of the our approach, we created a Double Moon sample with uniform background noise, as shown in Fig. 2.8. Although the sample is dense enough (600 data instances in total), spectral clustering fails to find the correctly identify
the two moons, due to the influence of background noise (100 data instances). However, with 20 constraints, our algorithm is able to recover the two moons in spite of the background noise.

2.7.3 Constraints from Partial Labeling: Clustering the UCI Benchmarks

Next we evaluate the performance of our algorithm by clustering the UCI benchmark datasets [3] with constraints derived from ground truth labeling.

We chose six different datasets with class label information, namely Hepatitis, Iris, Wine, Glass, Ionosphere and Breast Cancer Wisconsin (Diagnostic). We performed 2-way clustering simply by partitioning the optimal cluster indicator according to the sign: positive entries to one cluster and negative the other. We removed the setosa class from the Iris data set, which is the class that is known to be well-separated from the other two. For the same reason we removed Class 3 from the Wine data set, which is well-separated from the other two. We also removed data instances with missing values. The statistics of the data sets after preprocessing are listed in Table 2.2.

For each data set, we computed the affinity matrix using the RBF kernel. To generate constraints, we randomly selected pairs of nodes that the unconstrained spectral clustering wrongly partitioned, and fill in the correct relation in $Q$ according to ground truth labels. The quality of the clustering results was measured by adjusted Rand index. Since the constraints are guaranteed to be correct, we set the threshold $\beta$ such that there
Table 2.2. The UCI benchmarks

<table>
<thead>
<tr>
<th>Identifier</th>
<th>#Instances</th>
<th>#Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hepatitis</td>
<td>80</td>
<td>19</td>
</tr>
<tr>
<td>Iris</td>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>Wine</td>
<td>130</td>
<td>13</td>
</tr>
<tr>
<td>Glass</td>
<td>214</td>
<td>9</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>351</td>
<td>34</td>
</tr>
<tr>
<td>WDBC</td>
<td>569</td>
<td>30</td>
</tr>
</tbody>
</table>

will be only one feasible eigenvector, i.e. the one that best conforms to the constraint matrix $Q$.

In addition to comparing our algorithm (CSP) to unconstrained spectral clustering, we implemented two state-of-the-art techniques:

- Spectral Learning (SL) [34] modifies the affinity matrix of the original graph directly: $A_{ij}$ is set to 1 if there is a ML between node $i$ and $j$, 0 for CL.

- Semi-Supervised Kernel $K$-means (SSKK) [39] adds penalties to the affinity matrix based on the given constraints, and then performs kernel $K$-means on the new kernel to find the partition.

We also tried the algorithm proposed by Yu and Shi [68, 69], which encodes partial grouping information as a projection matrix, the subspace trick proposed in [20], and the affinity propagation algorithm proposed in [45]. However, we were not able to use these algorithms to achieve better results than SL and SSKK, hence their results are not reported. Xu et al. [67] proposed a variation of SL, where the constraints are encoded in the same way, but instead of the normalized graph Laplacian, they suggested to use the random walk matrix. We performed their algorithm on the UCI datasets, which produced largely identical results to that of SL.

We report the adjusted Rand index against the number of constraints used (ranging from 50 to 500) so that we can see how the quality of clustering varies when more constraints are provided. At each stop, we randomly generated 100 sets of constraints.
The mean, maximum and minimum ARI of the 100 random trials are reported in Fig. 2.9. We also report the ratio of the constraints that were satisfied by the constrained partition in Fig. 2.10. The observations are:

- Across all six datasets, our algorithm is able to effectively utilize the constraints and improve over unconstrained spectral clustering (Baseline). On the one hand, our algorithm can quickly improve the results with a small amount of constraints. On the other hand, as more constraints are provided, the performance of our algorithm consistently increases and converges to the ground truth partition (Fig. 2.9).

- Our algorithm outperforms the competitors by a large margin in terms of ARI (Fig. 2.9). Since we have control over the lower-bounding threshold $\alpha$, our algorithm is able to satisfy almost all the given constraints (Fig. 2.10).

- The performance of our algorithm has significantly smaller variance over different random constraint sets than its competitors (Fig. 2.9 and 2.10), and the variance quickly diminishes as more constraints are provided. This suggests that our algorithm would perform more consistently in practice.

- Our algorithm performs especially well on sparse graphs, i.e. Fig. 2.9(e)(f), where the competitors suffer. The reason is that when the graph is too sparse, it implies many “free” cuts that are equally good to unconstrained spectral clustering. Even after introducing a small number of constraints, the modified graph remains too sparse for SL and SSKK, which are unable to identify the ground truth partition. In contrast, these free cuts are not equivalent when judged by the constraint matrix $Q$ of our algorithm, which can easily identify the one cut that minimizes $v^T\bar{Q}v$. As a result, our algorithm outperforms SL and SSKK significantly in this scenario.

### 2.7.4 Constraints from Alternative Metrics: The Reuters Multilingual Dataset

We test our algorithm using soft constraints derived from alternative metrics of the same set of data instances.
Figure 2.9. The performance of our algorithm (CSP) on six UCI datasets, with comparison to unconstrained spectral clustering (Baseline) and the Spectral Learning algorithm (SL). Adjusted Rand index over 100 random trials is reported (mean, min, and max).
Figure 2.10. The ratio of constraints that are actually satisfied.
Table 2.3. The Reuters multilingual dataset

<table>
<thead>
<tr>
<th>Language</th>
<th>#docs</th>
<th>#words</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>18,758</td>
<td>21,531</td>
</tr>
<tr>
<td>French</td>
<td>26,648</td>
<td>24,839</td>
</tr>
<tr>
<td>German</td>
<td>29,953</td>
<td>34,279</td>
</tr>
<tr>
<td>Spanish</td>
<td>12,342</td>
<td>11,547</td>
</tr>
<tr>
<td>Italian</td>
<td>24,039</td>
<td>15,506</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Topics</th>
<th>#docs</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>C15</td>
<td>18,816</td>
<td>16.84%</td>
</tr>
<tr>
<td>CCAT</td>
<td>21,426</td>
<td>19.17%</td>
</tr>
<tr>
<td>E21</td>
<td>13,701</td>
<td>12.26%</td>
</tr>
<tr>
<td>ECAT</td>
<td>19,198</td>
<td>17.18%</td>
</tr>
<tr>
<td>GCAT</td>
<td>19,178</td>
<td>17.16%</td>
</tr>
<tr>
<td>M11</td>
<td>19,421</td>
<td>17.39%</td>
</tr>
</tbody>
</table>

**Dataset**  We used the Reuters RCV1/RCV2 Multilingual Dataset introduced by [2]. This dataset has been used by previous work [35, 40, 41] to evaluate the performance of multiview spectral clustering algorithm. The dataset contains documents originally written in five different languages, namely English (EN), French (FR), German (GR), Spanish (SP) and Italian (IT). Each document, originally written in one language, was translated to the other four languages using the Portage system [53]. The documents are categorized into six different topics. The statistics of the dataset is summarized in Table 2.3. More detail can be found on the dataset homepage \(^2\). The dataset is provided in the form of tf-idf vectors. We did not apply additional preprocessing to the data. No dimensionality reduction technique was applied. We used cosine similarity to construct the graph and the constraint matrix.

**Evaluation metrics**  We used two common evaluation metrics to measure the quality of clustering, namely Adjusted Rand Index (ARI) [31] and Normalized Mutual Information (NMI). Both of them indicate the similarity between a given clustering and the ground truth partition: higher value means better clustering; 1 means perfect match.

**Algorithm Implementations**  We implemented both the parametric (csp-p) and the non-parametric (csp-n) version of our algorithm. For the parametric version, we always set

\[
\alpha \leftarrow \lambda_{2K} (\bar{Q}),
\]

\(^2\)http://multilingreuters.iit.nrc.ca/ReutersMultiLingualMultiView.htm
where $\lambda_{2K}(\bar{Q})$ is the $2K$-th largest eigenvalue of $\bar{Q}$. In other words, we provide $2K$ feasible cuts and $\bar{L}$ will choose the top-$K$ with the lowest costs.

We also implemented five baseline algorithms in MATLAB to compare with:

- **orig**: Spectral clustering using the original view only.

- **trans**: Spectral clustering using the translated view only.

- **kersum**: The kernel summation method for multiview spectral clustering, which performs spectral clustering on the weighted sum of the two views’ kernels. Previous study [14] showed that this approach works very well in practice, even in comparison to much more sophisticated multiview learning techniques. We used equal weights for the two views in our experiments.

- **mrw**: The mixing random walk algorithm proposed in [71], which finds the stationary distribution of a mixing random walk in both graphs. We used equal weights for the two views in our experiments.

- **co-reg**: The co-regularization multiview spectral clustering algorithm proposed in [41]. We implemented the centroid based version and used the centroids to compute final clustering. This algorithm has one parameter, which is the weight for the regularizer. We set it to 0.01 in our experiments, as suggested in the original paper.

**Task Description**  We first pick a language pair, say EN-FR, which means documents that are originally written in English along with their French translation. To maximize the diversity between the data samples we use, in each trial we randomly sample 1200 documents, which is less than 10% of all available documents. We have 100 trials for each language pair. We apply our algorithm and the baseline algorithms to the sample and partition it into $K = 6$ clusters. We measure the resultant clusterings using both ARI and NMI. Since the last step of spectral clustering involves the $K$-means algorithm, in each trial, we repeat $K$-means algorithm 100 times with 100 random seeds and report the average performance.
Table 2.4. Average ARI of 7 algorithms on 8 language pairs

<table>
<thead>
<tr>
<th></th>
<th>orig</th>
<th>trans</th>
<th>kersum</th>
<th>mrw</th>
<th>co-reg</th>
<th>csp-p</th>
<th>csp-n</th>
</tr>
</thead>
<tbody>
<tr>
<td>GR-EN</td>
<td>0.276</td>
<td>0.261</td>
<td>0.303</td>
<td>0.296</td>
<td>0.303</td>
<td>0.314</td>
<td>0.289</td>
</tr>
<tr>
<td>GR-FR</td>
<td>0.277</td>
<td>0.250</td>
<td>0.284</td>
<td>0.279</td>
<td>0.274</td>
<td>0.303</td>
<td>0.281</td>
</tr>
<tr>
<td>GR-IT</td>
<td>0.279</td>
<td>0.301</td>
<td>0.294</td>
<td>0.286</td>
<td>0.303</td>
<td>0.304</td>
<td>0.287</td>
</tr>
<tr>
<td>GR-SP</td>
<td>0.274</td>
<td>0.224</td>
<td>0.280</td>
<td>0.273</td>
<td>0.271</td>
<td>0.293</td>
<td>0.276</td>
</tr>
<tr>
<td>EN-FR</td>
<td>0.168</td>
<td>0.163</td>
<td>0.195</td>
<td>0.172</td>
<td>0.174</td>
<td>0.203</td>
<td>0.194</td>
</tr>
<tr>
<td>EN-GR</td>
<td>0.171</td>
<td>0.149</td>
<td>0.192</td>
<td>0.172</td>
<td>0.167</td>
<td>0.197</td>
<td>0.192</td>
</tr>
<tr>
<td>EN-IT</td>
<td>0.168</td>
<td>0.170</td>
<td>0.178</td>
<td>0.161</td>
<td>0.183</td>
<td>0.184</td>
<td>0.179</td>
</tr>
<tr>
<td>EN-SP</td>
<td>0.169</td>
<td>0.163</td>
<td>0.180</td>
<td>0.164</td>
<td>0.182</td>
<td>0.182</td>
<td>0.175</td>
</tr>
</tbody>
</table>

For all language pairs, we report the average performance (Table 2.4), aggregated results over 100 trials (Figure 2.11 and 2.12), and the performance of each individual trial (Figure 2.13 and 2.14).

**Results and Analysis** Since we have 5 different languages, there are 20 possible original-translation language combinations. We show our results on 8 pairs, namely English to French (EN-FR), German (EN-GR), Italian (EN-IT), Spanish (EN-SP), and German to English (GR-EN), French (GR-FR), Italian (GR-IT), and Spanish (GR-SP). The conclusions we draw from these 8 pairs also hold for the other 12 pairs.

First we give an overview of the results in terms of average performance. In Table 2.4, we report the average ARI of 7 different algorithms on 8 different language pairs. Our approach (csp-p) shows consistent and significant (99% confidence level) improvement over the clustering on the original view only (orig) for all language pairs. Also, for all language pairs, csp-p has the highest average ARI.

More detailed results are reported in Figure 2.11 and 2.12, with box plots for all 7 algorithms and 8 language pairs, in terms of ARI and NMI, respectively. Besides of showing the advantage of our approach (csp-p), as we have seen in Table 2.4, Figure 2.11 and 2.12 illustrate the diversity of the data samples we used in different trials. Some data samples were easier to cluster and others more difficult. This demonstrates that
the effectiveness of our approach is not limited to a certain data distribution or a certain language.

Note that the performance of our non-parametric approach (\textit{csp-n}) is not as good as the parametric one (\textit{csp-p}). This is expected because \textit{csp-n} uses zero prior knowledge. On the other hand, as shown in Table 2.4, \textit{csp-n} managed to outperform \textit{orig} on all 8 language pairs. It also outperformed the multiview competitors on several language pairs. This result is non-trivial considering the approach is completely parameter-free.

To further demonstrate the consistency and reliability of our approach over different random samples, in Figure 2.13 and 2.14, we show the trial by trial breakdown of the performance gain of our approach (\textit{csp-p} and \textit{csp-n}) over the clustering on the original view only (\textit{orig}), as measured by ARI. We can see that for 800 random trials over 8 different language pairs, our algorithm achieved positive gain in most cases, and it rarely caused large performance loss. This means our approach is reliable in practice. Practitioners can apply our approach to a dataset, with peace of mind that it is very likely that our algorithm will improve the result, and it is very unlikely that our algorithm will lead to a great performance loss.

### 2.7.5 Transfer of Knowledge: Resting-State fMRI Analysis

Finally we apply our algorithm to transfer learning on the resting-state fMRI data.

An fMRI scan of a person consists of a sequence of 3D images over time. We can construct a graph from a given scan such that a node in the graph corresponds to a voxel in the image, and the edge weight between two nodes is (the absolute value of) the correlation between the two time sequences associated with the two voxels. Previous work has shown that by applying spectral clustering to the resting-state fMRI we can find the substructures in the brain that are periodically and simultaneously activated over time in the resting state, which may indicate a network associated with certain functions [54].

One of the challenges of resting-state fMRI analysis is instability. Noise can be easily introduced into the scan result, e.g. the subject moved his/her head during the scan, the subject was not at resting state (actively thinking about things during the scan), etc.
Figure 2.11. The box plot for 4 language pairs, 100 random trials each. Results are evaluated in terms of both ARI and NMI.
Figure 2.12. The box plot for another 4 language pairs, 100 random trials each. Results are evaluated in terms of both ARI and NMI.
Figure 2.13. The trial by trial breakdown of the performance gain of our technique (csp-p and csp-n) over the baseline (orig) on 4 language pairs.
Figure 2.14. The trial by trial breakdown of the performance gain of our technique (csp-p and csp-n) over the baseline (orig) on another 4 language pairs.
Consequently, the result of spectral clustering becomes instable. If we apply spectral clustering to two fMRI scans of the same person on two different days, the normalized min-cuts on the two different scans are so different that they provide little insight into the brain activity of the subject (Fig. 2.15(a) and (b)). To overcome this problem, we use our formulation to transfer knowledge from Scan 1 to Scan 2 and get a constrained cut, as shown in Fig. 2.15(c). This cut represents what the two scans agree on. The pattern captured by Fig. 2.15(c) is actually the default mode network (DMN), which is the network that is periodically activated at resting state (Fig. 2.15(d) shows the idealized DMN as specified by domain experts).

To further illustrate the practicability of our approach, we transfer the idealized DMN
Figure 2.16. The costs of transferring the idealized default mode network to the fMRI scans of two groups of elderly individuals.

In Fig. 2.15(d) to a set of fMRI scans of elderly subjects. The dataset was collected and processed within the research program of the University of California at Davis Alzheimer’s Disease Center (UCD ADC). The subjects were categorized into two groups: those diagnosed with cognitive syndrome (20 individuals) and those without cognitive syndrome (11 individuals). For each individual scan, we encode the idealized DMN into a constraint matrix (using the RBF kernel), and enforce the constraints onto the original fMRI scan. We then compute the cost of the constrained cut that is the most similar to the DMN. If the cost of the constrained cut is high, it means there is great disagreement between the original graph and the given constraints (the idealized DMN), and vice versa. In other words, the cost of the constrained cut can be interpreted as the cost of transferring the DMN to the particular fMRI scan.

In Fig. 2.16, we plot the costs of transferring the DMN to both subject groups. We can clearly see that the costs of transferring the DMN to people without cognitive syndrome tend to be lower than to people with cognitive syndrome. This conforms well to the observation made in a recent study that the DMN is often disrupted for people with the Alzheimer’s disease [12].
2.8 Summary

In this chapter we proposed a principled and flexible framework for constrained spectral clustering that can incorporate large amounts of both hard and soft constraints. The flexibility of our framework lends itself to the use of all types of side information: pairwise constraints, partial labeling, alternative metrics, and transfer learning. Our formulation is a natural extension to unconstrained spectral clustering and can be solved efficiently using generalized eigendecomposition. We demonstrated the effectiveness of our approach on a variety of datasets: the synthetic Two-Moon dataset, image segmentation, the UCI benchmarks, the multilingual Reuters documents, and resting-state fMRI scans. The comparison to existing techniques validated the advantage of our approach.
Chapter 3

Active Extension to Constrained Spectral Clustering

3.1 Introduction

Many real-world applications, such as image segmentation, social network analysis and data clustering can be abstracted into a graph partition problem: finding the normalized min-cut (Ncut) of a given graph. Although the Ncut problem is generally intractable, it is well-known that its relaxed form can be solved by spectral clustering. The seminal work by Shi and Malik [51] represents the first incarnation of spectral clustering which was passive and unsupervised. However, in many application domains, considerable domain expertise exists and encoding domain knowledge into clustering algorithms is important if the results are to be novel and actionable. To address this issue, user supervision in the form of pairwise relations between two nodes of the graph have been proposed [7]: Must-Link (they belong to the same side of the cut) and Cannot-Link (they belong to different sides of the cut) constraints. In Chapter 2, we have shown that ML and CL constraints, when used properly, can greatly improve the quality of the resultant clustering. Those work represents the second incarnation of spectral clustering: passive and semi-supervised.

In this chapter we make the natural and important progression to the third incarnation of spectral clustering: active and semi-supervised. In this formulation the constraints are provided incrementally after querying an oracle rather than a priori
in a batch before clustering begins. Active and semi-supervised spectral clustering has many natural problem settings that will see its wide-scale use:

- The constraints can be used as an explicit and elegant way to perform interactive transfer learning. In this setting the graph Laplacian is generated from the source domain and the constraints from the target domain. For example a given set of facial portraits may naturally cluster according to hair-style. The constraints (derived by a domain expert) are used to transfer this structure to a related but more complex domain such as identifying gender.

- The graph Laplacian can sometimes be noisy and inconsistent. This is particularly likely if it is constructed from complex objects such as images that are of poor quality, not completely aligned or taken with different cameras. The oracle then can be used to overcome these issues.

- Some problems in this domain naturally have a sequential aspect. Consider the clustering of a stream of video (Chapter 18, [7]), it makes most sense to specify constraints incrementally rather than all at once.

- Previous research [19] showed that in batch constrained clustering, not all given constraints are equally helpful/informative in terms of improving label purity in the clusters despite the constraints being generated from the same labels used to measure performance. We experimentally verify in section 3.4 that incrementally collecting constraints produces better results.

Therefore, it is a natural and important question to ask: instead of passively taking a given set of constraints, which may consist of both helpful and harmful constraints, is it possible for the algorithm to actively query/fetch only the constraints that are expected to be helpful? With the availability of an oracle, we can put constrained spectral clustering into the context of active learning [50]. Our goal is to maximally improve the quality of the resultant clustering (maximizing performance gain) while making as few queries as possible (minimizing cost). We propose an active learning
framework for constrained spectral clustering, or active spectral clustering for short. Our framework consists of two key components:

- A constrained spectral clustering algorithm: This component takes the graph and the constraints that have already been queried as input. It produces a Ncut of the graph which satisfies the constraints we currently have.

- A query strategy: This component evaluates the current cut of the graph as well as the constraints we have, then decides the best constraint to query next, based on the principle of maximally reducing the expected error between our current cluster assignment and the expected groundtruth assignment.

We apply these two components alternatively and the resultant clustering will be refined over iterations and eventually converge to the groundtruth result.

Our contributions are:

1. This is the first principled framework for active spectral clustering. It provides a cost-efficient solution to many real-world applications that need to find the Ncut of a graph: it actively selects the most helpful constraints to query from an oracle and thus minimizes the efforts required from the oracle/domain experts.

2. We propose a ready-to-use active spectral clustering algorithm as a realization of our framework. The proposed algorithm is highly flexible and can deal with both hard and soft constraints. This is of great practical importance because soft constraints are common in many applications, especially when the oracle is not a single human expert, but a group of users who may give inconsistent answers to the same query.

3. We address some important implementation issues for our method, such as dealing with outliers and runtime analysis. We also discuss the limitation of our method in its current form as well as future work.

4. We empirically show that our method significantly outperforms the baseline method typically used by practitioners. This is a significant result with practical implica-
tions. It means it is far better to incrementally and interactively specify constraints rather than getting them in one large batch.

The rest of the chapter is organized as follows: related work is discussed in Section 3.2; we propose our active spectral clustering framework in Section 3.3; it is evaluated empirically in Section 3.4; implementation issues are discussed in Section 3.5 and future directions are discussed in Section 3.6; we summarize this chapter in Section 3.7.

3.2 Related Work

Active clustering is a special sub-category of active learning algorithms [50]. The difference is that active clustering algorithms query pairwise relations between two nodes instead of the labels of individual nodes. Most existing active clustering methods [6, 19, 25, 30, 36, 46] were built upon hard clustering schemes such as *K*-means clustering and hierarchical clustering. Little attention has been paid to the active learning framework for *spectral clustering*, which is the most popular soft clustering scheme and a solution to many real-world applications.

Xu et al. [66] proposed an active spectral clustering method that examines the eigenvectors of the graph Laplacian to identify boundary points and sparse points, and then queries the oracle for constraints among these ambiguous points. Their work is limited mainly because they explicitly assume that the underlying clusters in the data set are nearly separated and it is the boundary points that cause the inaccuracy in the cluster assignment; it is unclear if the propose method would still work otherwise. Also, since the proposed method is built upon the KKM constrained spectral clustering method [34], it can only incorporate hard constraints, not soft ones.

Active spectral clustering is also related to the area of matrix perturbation analysis for spectral clustering [29, 52], which studies how much the resultant clustering will change when a perturbation is applied to the original graph Laplacian. The results from perturbation analysis can give us an idea of how stable the clustering is and how many constraints are needed for the clustering to change significantly. However, the bounds in matrix perturbation theory are typically of the form involving the norm of the matrix
3.3 Our Active Spectral Clustering Framework

In this section we present our active spectral clustering framework. We provide an overview of our framework in Section 3.3.1. In Section 3.3.2 we recall the constrained spectral clustering algorithm we proposed in Chapter 2, which is the first important component of our framework. Then in Section 3.3.3 we describe the second important component of our framework, an active query strategy based on maximum expected error reduction. For background knowledge on spectral clustering, please refer to Section 2.3. Important notations used throughout the chapter are listed in Table 3.1. Note that the superscript “∗” when attached to a symbol refers to the groundtruth answer typically only available to the oracle.

3.3.1 An Overview of Our Framework

In this work, we make the assumption that there is an oracle who has access to the groundtruth constraint matrix \( Q^* = u^*u^{*T} \), where \( u^* \in \mathbb{R}^N \) is the groundtruth cluster assignment. We assume that we can actively query an oracle about the value of any entry in \( Q^* \), one entry at a time. \( Q \) is the matrix that contains all the constraints we have queried so far (0 for unknown entries). Note that the nonzero entries in \( Q \) is always
a proper subset of the nonzero entries in $Q^*$. Our goal is to minimize the difference between $u = \mathcal{A}(L, Q)$ and $u^*$ using as few queries as possible.

We propose an iterative process to incrementally query the oracle about the constraint that can maximally reduce the expected error in our current result. We start with an empty constraint matrix $Q^{(0)}$ with all 0 entries, then we compute the current clustering using a constrained spectral clustering algorithm $\mathcal{A}$:

$$u^{(0)} \leftarrow \mathcal{A}(L, Q^{(0)}).$$

Note that $u^{(0)}$ should be the same as the clustering found by the unconstrained spectral clustering algorithm as we have no constraint so far. Then assume at iteration $t$ we already have

$$u^{(t)} \leftarrow \mathcal{A}(L, Q^{(t)}).$$

A query strategy $\mathcal{Q}$ will evaluate the current clustering $u^{(t)}$ and the current constraints $Q^{(t)}$ to decide what is the next entry in $Q^*$ we should query from the oracle. Let

$$(i, j) \leftarrow \mathcal{Q}(u^{(t)}, Q^{(t)}),$$

we update $Q^{(t)}$ to $Q^{(t+1)}$ by filling in $Q^{(t)}_{ij}$ and $Q^{(t)}_{ji}$ with the value of $Q^*_{ij}$ (since the constraint matrix is symmetric). Then we update the clustering by

$$u^{(t+1)} \leftarrow \mathcal{A}(L, Q^{(t+1)}).$$

We repeat this iteration until certain stopping criteria is met.

Our framework has two key components: the constrained clustering algorithm $\mathcal{A}$ and the query strategy $\mathcal{Q}$. Next we will discuss their realization in detail, respectively.

### 3.3.2 The Constrained Spectral Clustering Algorithm

The first key component of our framework is a constrained spectral clustering algorithm $\mathcal{A}$. $\mathcal{A}$ takes the graph Laplacian $L$ and a constraint matrix $Q$ as input and outputs a (relaxed) cluster indicator vector $u$. In general, our framework has no restriction on the realization of $\mathcal{A}$ as long as it satisfies the following property:
Property 3.3.1 (Convergence). As the constraint matrix $Q$ approaches $Q^*$, the output of the constrained clustering algorithm, $u$, will converge to the groundtruth cluster assignment $u^*$:

$$\lim_{Q \to Q^*} u = u^*.$$ 

This property is to ensure that our active learning framework will converge to the groundtruth cluster assignment as more constraints are revealed by the oracle. As trivial as this property may seem like, it is not automatically guaranteed by all constrained clustering algorithms. For example, some constrained $K$-means clustering algorithms are sensitive to the order in which the constraints are enforced and thus may not converge to the groundtruth clustering after all.

There are a number of possible candidates for $\mathcal{A}$ [13, 34, 45] in the literature. In this chapter we implement the framework using the constrained spectral clustering algorithm we proposed in Chapter 2. The main advantage of this approach is that it not only satisfies the convergence property but also is flexible enough to incorporate both hard and soft constraints. Recall that its objective function is as follows:

$$\arg\min_{u \in \mathbb{R}^N} u^T Lu,$$

s.t. $u^T Du = \text{vol}(G), \ u^T Q u \geq \alpha$. 

The solution is provided by the generalized eigenvalue problem

$$Lu = \lambda(Q - \alpha I)u.$$ 

3.3.3 The Query Strategy

The second key component of our framework is the query strategy $\mathcal{Q}$. Our strategy evaluates the current cluster assignment $u$ and the constraints in $Q$ and decides what is the best entry of $Q^*$ to query next. The principle it uses is maximum expected error reduction, which means that for all unknown pairwise relations between two nodes, we compute the expected error between our current estimation of that value and its groundtruth value, and we pick the pair of nodes with largest expected error and query the oracle for their relation.
Formally, let $P^{(t)}_{ij}$ be our estimation of the pairwise relation between node $i$ and $j$ at time $t$. A straightforward way to compute $P^{(t)}_{ij}$ from the cluster assignment vector $u^{(t)}$ is:

$$
P^{(t)}_{ij} = u^{(t)}_i u^{(t)}_j. \quad (3.3)
$$

Let $d \in \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ be a distance function that measures the error between our current estimation and the groundtruth value:

$$
d(P^{(t)}_{ij}, Q^*_{ij}) = (P^{(t)}_{ij} - Q^*_{ij})^2.
$$

Since $Q^*_{ij}$ remains unknown until after we actually query it, we cannot compute the error $d(P^{(t)}_{ij}, Q^*_{ij})$ directly. Instead, we compute the mathematical expectation of the error over the two possible answers from the oracle:

$$
E(d(P^{(t)}_{ij}, Q^*_{ij})) = d(P^{(t)}_{ij}, 1) \Pr(Q^*_{ij} = 1) + d(P^{(t)}_{ij}, -1) \Pr(Q^*_{ij} = -1).
$$

Now the question becomes how we can estimate $\Pr(Q^*_{ij} = 1)$ and $\Pr(Q^*_{ij} = -1)$ based on the information we already have. Recall that we assumed $Q^* = u^* u^{*T}$, thus $Q^*$ is a rank-one matrix. If we treat the current constraint matrix $Q^{(t)}$ as an approximation to $Q^*$ with missing values, then it is a standard approach to use the rank-one approximation of $Q^{(t)}$ to recover the unknown entries in $Q^*$ [10]. Let $\bar{u}^{(t)}$ be the largest singular vector of $Q^{(t)}$, then

$$
\bar{Q}^t = \bar{u}^{(t)} \bar{u}^{(t)T}
$$

is the optimal rank-one approximation to $Q^{(t)}$ in terms of Frobenius norm [27]. Then we can compute $\Pr(Q^*_{ij} = 1)$ and $\Pr(Q^*_{ij} = -1)$ as follows:

$$
\Pr(Q^*_{ij} = 1) = \Pr(Q^*_{ij} = 1 | Q^{(t)}) = 1 + \min\{1, \max\{-1, \bar{Q}^t_{ij}\}\}/2,
$$

$$
\Pr(Q^*_{ij} = -1) = 1 - \Pr(Q^*_{ij} = 1).
$$

Finally, we query the entry that has the maximum expected error:

$$
\mathcal{Q}(u^{(t)}, Q^{(t)}) = \arg \max_{(i,j)|Q^{(t)}_{ij} = 0} E(d(P^{(t)}_{ij}, Q^*_{ij})). \quad (3.4)
$$
3.4 Empirical Results

To show the effectiveness of our approach, we evaluated its performance on several UCI benchmark data sets [3]. Our goal is to show that our framework can achieve better performance with a smaller number of actively selected constraints, as compared to a randomly selected constraint set. This effectively tests our active selection approach against the batch approach using the same number of constraints.

We compared our method (active) to a baseline method (random). Both methods used the exact same implementation of the constrained spectral clustering algorithm. The only difference was that the constraints used by active were actively selected using our query strategy, whereas the constraints used by random were randomly selected.

We used both hard and soft constraints in our experiments. For hard constraints, we chose five different data sets with groundtruth labels, namely Hepatitis, Iris, Wine, Glass, and Ionosphere. We performed 2-way partition on all data sets. We removed the setosa class from the Iris data set, which is the class that is known to be well-separated from the other two. For the same reason we removed Class 1 from the Wine data set. We also removed data instances with missing values. The statistics of the data sets after preprocessing are listed in Table 3.2. For each data set, we computed the affinity matrix $A$ using the RBF kernel (the edge weight between two nodes is the similarity between those two data instances).

For soft constraints, we chose a subset of the 20 Newsgroup data, as shown in Table 3.3. We randomly sampled about 350 documents from 6 groups. At the highest level, those groups can be divided into two topics: computer (comp) and recreation (rec). To generate soft constraints, if two articles are from different topics, we set the corresponding entry in $Q^*$ to $-1$; if two articles are from the same topic but different groups, we set the corresponding entry to 0.5; if they are from the same group, we set the entry to 1. The affinity matrix $A$ was generated from the similarity matrix based on inner-product (the edge weight between two nodes is the number of words those two articles have in common).

On all data sets, we started with no constraint and queried one constraint at a time.
Table 3.2. The UCI benchmarks

<table>
<thead>
<tr>
<th>Identifier</th>
<th>#Instances</th>
<th>#Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hepatitis</td>
<td>80</td>
<td>19</td>
</tr>
<tr>
<td>Iris</td>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>Wine</td>
<td>119</td>
<td>13</td>
</tr>
<tr>
<td>Glass</td>
<td>214</td>
<td>9</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>351</td>
<td>34</td>
</tr>
</tbody>
</table>

Table 3.3. The 20 Newsgroup data

<table>
<thead>
<tr>
<th>Group</th>
<th>Label</th>
<th>#Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>comp.os.ms-windows.misc</td>
<td>53</td>
</tr>
<tr>
<td>4</td>
<td>comp.sys.ibm.pc.hardware</td>
<td>60</td>
</tr>
<tr>
<td>5</td>
<td>comp.sys.mac.hardware</td>
<td>59</td>
</tr>
<tr>
<td>9</td>
<td>rec.motorcycles</td>
<td>65</td>
</tr>
<tr>
<td>10</td>
<td>rec.sport.baseball</td>
<td>64</td>
</tr>
<tr>
<td>11</td>
<td>rec.sport.hockey</td>
<td>51</td>
</tr>
</tbody>
</table>

from the oracle. Note that we did not use the transitive or entailment properties \[19\] to deduce more constraints based on the existing ones, which is impossible to do when the constraints are soft. To evaluate the accuracy of the resultant clustering at each time step, we used Rand index \[56\]. For each data set, we made up to \(2N\) queries, where \(N\) was the size of the data set. There is only one parameter in our method, which is \(\alpha\) for the constrained spectral clustering algorithm (see Eq. (3.2)). Throughout all experiments, we simply set it to \(\lambda_{max}/2\), where \(\lambda_{max}\) is the largest eigenvalue of \(Q\). In this way, we guarantee the existence of at least one feasible solution, while requiring that a reasonable amount constraints must be satisfied.

The results are shown in Fig. 3.1 and we can observe that:

- In most cases, our active method converged to the groundtruth clustering after a small number of queries.

- As a contrast, the baseline method often did not even show performance gain with
a randomly selected constraint set of the same size.

- In most cases, the performance of our active method consistently increased as more constraints had been queried. This suggested that the active query process utilized constraints in a helpful way.

- Our active approach outperformed the random approach in average by a large margin. In many cases, our active approach even outperformed the best-of-luck result from the random approach.

- There are also some dataset-specific observations. For example, the performance of our active method on the Iris data set was not as good when only a very small number of constraints had been queried. We contribute this to the existence of contextual outliers, as we discovered in our previous work [57], the influence of which misled the initial active query process. However, our active method recovered quickly after more constraints were queried. We also noticed that on the 20NG data set, the performance of the random method struggled hopelessly. This is because the data set contains six well-separated sub-clusters (each corresponding to a sub-topic). Randomly queried constraints did not work as effectively as the active method to help identify the two more generalized topics, which lead to the groundtruth 2-partition we are looking for.

The above observations can be explained by noting that our actively set of constraints complement each other whilst the randomly selected constraints may very well be contradictory. These results for the baseline approach are consistent with earlier work [19] which showed the randomly chosen constraint sets often hurt performance of the underlying algorithm when measured by the Rand index.

In our experiments, we also noticed that there were cases where our query strategy found more than one constraints with the same largest expected error. In this case, we used a randomized tie-breaking step to pick one of them to query. As a result, although our query strategy is designed to be deterministic, its output on certain data sets could
vary over many trials. However, for reasonably large data sets, the variation between
different trials appeared to be insignificant.

3.5 Implementation Issues

3.5.1 Outliers

Our query strategy $Q$ is an instance-based strategy. As a result, the existence of outliers
may cause a large number of additional queries. For example, imagine we have a graph
with one outlying node. Without constraints, the spectral clustering algorithm may
identify the outlying node as one cluster and the rest of the graph as another (especially
when the majority of the graph is a relatively well-connected component). According to
our query strategy, the outlying node will have the largest expected error, because the
resultant cluster assignment will be entirely different depending on whether or not this
node is a true outlier, or it is actually a cluster by itself. Our query strategy will keep
querying the pairwise relations between this outlying node and all the other nodes in
the rest of the graph until it reaches a conclusion.

On the one hand, we need to point out that this kind of intensive queries on a key
node is necessary without prior knowledge on the existence of outliers or the underlying
distribution of the data. On the hand other, if we do have prior information, e.g.
the minimum size of the potential clusters, we could remove obvious outliers during
the preprocessing step. This can help avoid initiating our method with a completely
wrong clustering, which inevitably would take much more queries to converge to the
groundtruth result.

We also found out that normalizing our estimation of the relation between node $i$
and $j$, which is $P_{ij}^{(t)}$ as shown in Eq.3.3, can help reduce the influence of potential
outliers in the graph. This can avoid a relatively large entry in $u^{(t)}$ from dominating the
query process. Specifically, we have:

$$ P_{ij}^{(t)} = \begin{cases} 
1 & \text{if } u_i^{(t)} u_j^{(t)} > 1 \\
-1 & \text{if } u_i^{(t)} u_j^{(t)} < -1 \\
u_i^{(t)} u_j^{(t)} & \text{otherwise}
\end{cases} $$
Figure 3.1. Results on six UCI data sets, with comparison between our method active and the baseline method random. Y-axis is Rand index, X-axis is the number of constraints queried. The maximum number of queries is 2N, where N is the size of the corresponding data set. For the random method, the max/average/min performance over 10 runs (each with a randomly generated constraint set) are reported, respectively.
3.5.2 Time Complexity

Our active learning method is an iterative process. The time complexity for each iteration is constant. Within the iteration, there are two main steps, the constrained spectral clustering process, and the query process. The runtime of the constrained spectral clustering algorithm we introduce is dominated by that of solving a generalized eigenvalue problem on an $N \times N$ matrix; the runtime of the query process is dominated by computing the rank-one approximation to an $N \times N$ matrix, which takes no longer than solving the eigenvalue problem. Therefore, the overall time complexity of our method is equal to the number of iterations times the time complexity of solving an eigenvalue problem on an $N \times N$ matrix, depending on which solver you choose to use, but $O(N^2)$ at least. In other words, the runtime of our method is mainly decided by 1) the size of the data set; 2) the number of iterations/queries.

Note that the time complexity of our method does not increase with the number of constraints we have queried or the number of constraints we query at each time. Thus in practice we can choose to query more than one constraint during each iteration. However, under the assumption that each query comes with a cost, this is essentially a tradeoff between the runtime and the cost of querying the oracle.

3.5.3 Stopping Criterion

A common consideration when implementing active learning algorithms is when to stop querying, because either 1) the result has converged and will no longer change with more constraints, or 2) the result is “good enough” thus further queries are no longer worth the cost. It is possible to find such a criterion when the learning task itself is supervised/semi-supervised and there is some kind of auxiliary information to measure the quality and/or stability of the result. However, it is less likely to find such a measure that works for unsupervised learning (clustering) in general, due to the absolute absence of groundtruth information. Moreover, as Burr Settles stated in his survey on active learning [50]:

“... in my own experience, the real stopping criterion for practical applications is based on economic or other external factors, which likely come well
before an intrinsic learner-decided threshold.”

Therefore, from the practical perspective, we suggest to make as many queries as possible/affordable, and our method will consistently improve the result unless it has already converged to the groundtruth.

3.6 Limitations and Extensions

Our query strategy as described in Section 3.3.3 assumes that $Q^*$ is, or can be well approximated by, a rank-one matrix. Only with this assumption, we can estimate the expected error using the rank-one approximation technique. However, we notice that there are real-world application scenarios where $Q^*$ may have a higher rank. For example, the $Q^*$ may be derived from a $K$-way partition of the data set ($K > 2$), then the rank of $Q^*$ would be $K − 1$. Or there might be one-sided oracles who only provide Must-Link or Cannot-Link constraints, but not both. To deal with these scenarios, we need to adopt more sophisticated method to estimate the expected error between our current cluster assignment and the groundtruth result.

Another natural extension for our method is $K$-way partition where $K > 2$. The spectral formulation of clustering is for cutting a graph and many applications of the approach only require a $K = 2$ clustering. Although it is possible to extend the formulation to $K$-way partition, some important theoretical properties will be lost after the extension, e.g. the result is no longer deterministic and it is difficult to interpret the result as a normalized min-cut. To modify our current algorithm for $K$-way partition, we need to modify the constrained spectral clustering algorithm $A$ to support $K$-way partition. Common practice is to, instead of only looking at one eigenvector, look at the top-$K$ eigenvectors all together and perform $K$-means clustering on the rows of the $N \times K$ matrix $[55]$. Note that now the output of $A(L,Q)$ would become the $N \times N$ matrix $P$ that directly encodes the pairwise relations between nodes, since a single indicator vector $u$ cannot encode $K$-way partition for $K > 2$. Then we need to modify the query strategy to deal with a $Q^*$ whose rank is now $K − 1$, as we mentioned above.
3.7 Summary

In this chapter, we proposed an active learning framework for spectral clustering. Its goal is to maximally improve the performance of a given constrained spectral clustering algorithm by using as few constraints as possible. We designed a query strategy that incrementally and iteratively picks the constraint with the largest expected error among all unknown constraints and then retrieves the groundtruth value for that constraint from an oracle. Our framework is not only principled, but also high flexible to work with both hard and soft constraints that may occur in real-world applications. We used several UCI benchmark data sets to validate the advantage of our approach, by comparing to the baseline method with randomly selected constraint set. Empirical results showed that our method can find the groundtruth cluster assignment by only using a small constraint set, and it outperformed the baseline method of specifying the same number of constraints as a batch by a large margin.
Chapter 4

Constrained Spectral Clustering and Label Propagation

4.1 Introduction

A common approach to data mining and machine learning is to take a set of instances and construct a corresponding graph where each node is an instance and the edge weight is the similarity between two nodes. A recent innovation is to add labels to a small set of nodes in the graph. This setting gives rise to semi-supervised learning, which studies how to use side information to label the unlabeled nodes. Depending on how the side information is encoded, there are two popular categories of approaches:

1. **Label Propagation**: Side information is kept in the form of node labels. The known labels are propagated to the unlabeled nodes. The prediction is usually evaluated by 0/1 loss functions (e.g. accuracy) directly on the labels.

2. **Constrained Spectral Clustering**: Side information is converted to pairwise constraints. A graph cut is computed by minimizing the cost of the cut while maximizing the constraint satisfaction. The cut is usually evaluated by 0/1 loss functions (e.g. Rand index) on the pairwise relations derived from the labels.

Both categories have been proven effective in their respective problem domains, but the relation between the two has not been explored [1, 72]. Exploring this topic leads to some interesting questions:
Given a set of labels as side information, should we use label propagation, or should we first convert the labels into pairwise constraints (which is common practice for many constrained clustering algorithms) and then use constrained clustering?

Since labels are more expressive than pairwise constraints (a set of pairwise constraints may not correspond to a unique labeling), is label propagation superior to constrained spectral clustering?

In the active learning setting, where we have the chance to query an oracle for ground truth, should we query labels (more expressive but difficult to acquire) or constraints (less expressive but easy to acquire)?

To address these questions and more, we need a unified view of label propagation and constrained spectral clustering.

In this chapter, we explore the relation between label propagation and constrained spectral clustering. We unify the two areas by presenting a novel framework called *stationary label propagation*. This framework gives us new insights into how side information contributes to recovering the ground truth labeling. It also enables us to propose a novel constraint construction technique which can benefit existing constrained spectral clustering algorithms.

Our contributions are:

- We establish equivalence between label propagation and constrained spectral clustering. Constrained spectral clustering using a non-negative and positive semi-definite constraint matrix is equivalent to finding a stationary labeling under label propagation (Section 4.4).

- We propose a novel algorithm that combines label propagation and constrained spectral clustering. It uses propagated labels to generate a (better) constraint matrix, and then uses the constraint matrix for constrained spectral clustering (Section 4.5).
• We use empirical results to demonstrate the advantage of the newly proposed algorithm over a variety of techniques (Section 4.6). In particular we show that our method is not only more accurate (Fig. 4.4) but also more stable (Fig. 4.5) in terms of generating helpful constraint sets. This addresses the stability issue raised by Davidson et al. [19].

4.2 Related Work and Preliminaries

In this section, we give a brief review of existing work on label propagation and constrained spectral clustering.

4.2.1 Label Propagation

In this work we focus on two popular label propagation techniques, namely Gaussian Fields Harmonic Function (GFHF) [73, 74] and Learning with Local and Global Consistency (LLGC) [70].

**GFHF:** Given a graph $G$ with $N$ nodes, its affinity matrix is $A$. We assume the first $N_1$ nodes are labeled and the remaining $N_2 = N - N_1$ nodes are unlabeled. Let $y_l \in \mathbb{R}^{N_1}$ be the known labels. $P = D^{-1}A$ is the transition matrix of the graph, where $D$ is the degree matrix. We partition $P$ into blocks

$$P = \begin{bmatrix} P_{ll} & P_{lu} \\ P_{ul} & P_{uu} \end{bmatrix}$$

so that $P_{ul}$ is the transition probability from the labeled nodes to the unlabeled ones and $P_{uu}$ is the transition probability between unlabeled nodes. GFHF uses the following iterative propagation rule:

$$y^{t+1} = \begin{bmatrix} y_{l}^{t+1} \\ y_{u}^{t+1} \end{bmatrix} = \begin{bmatrix} I & 0 \\ P_{ul} & P_{uu} \end{bmatrix} \begin{bmatrix} y_{l}^{t} \\ y_{u}^{t} \end{bmatrix}.$$  \hspace{1cm} (4.1)

An illustration of the GFHF model is shown in Fig. 4.1. From Eq.(4.1) we can see that the given labels $y_l$ will not be changed during the propagation. Zhu and Ghahramani

\footnote{For the simplicity of notations, for now we limit ourselves to the 2-class/bi-partition problem. We use binary encoding for the class/cluster indicator vector, i.e. $y \in \{-1, 0, +1\}$, where 0 means unknown. After relaxation, $y \in \mathbb{R}^N$ and 0 is used as the boundary of the two classes.}
Figure 4.1. An illustration of the GFHF propagation model \((N = 5, N_1 = 1)\). Node 4 is labeled. The propagation from 4 to 3 and 5 is governed by \(P_{ul}\) (directed edges); the propagation between 1, 2, and 3 is governed by \(P_{uu}\) (undirected edges).

[73] showed that \(y^t\) converges to

\[
    f = \begin{bmatrix} f_l \\ f_u \end{bmatrix} = \lim_{t \to \infty} y^t = \begin{bmatrix} y_l \\ (I - P_{uu})^{-1} P_{ul} y_l \end{bmatrix},
\]

\(4.2\)

**LLGC:** A key difference between LLGC and GFHF is that LLGC allows the initially given labels to be changed during the propagation process. Assume we have a graph \(G\) whose affinity matrix is \(A\). \(\bar{A} = D^{-1/2} A D^{-1/2}\) is the normalized affinity matrix. Let \(y^0 \in \mathbb{R}^N\) be the initial labeling. LLGC uses the following iterative propagation rule:

\[
y^{t+1} = \alpha \bar{A} y^t + (1 - \alpha) y^0,
\]

\(4.3\)

\(\alpha \in (0, 1)\). An illustration of the LLGC model is shown in Fig. 4.2. Zhou et al. [70] showed that \(y^t\) converges to:

\[
f = \lim_{t \to \infty} y^t = (1 - \alpha)(I - \alpha \bar{A})^{-1} y^0.
\]

\(4.4\)

### 4.2.2 Constrained Spectral Clustering (CSC)

In Chapter 2 we proposed the following quadratic formulation for constrained spectral clustering:

\[
\operatorname{argmin}_{f \in \mathbb{R}^N} f^T L f,
\]

s.t. \(f^T \bar{Q} f \geq \beta,\)

\(f^T f = 1, f \perp D^{1/2} 1.\)

\(4.5\)
Figure 4.2. An illustration of the LLGC propagation model ($N = 5$). Node 4 is labeled. The propagation between nodes is governed by $\bar{A}_{ij}$ (all edges are undirected).

$\bar{L} = I - \bar{A}$ is the normalized graph Laplacian. $\bar{Q} \in \mathbb{R}^{N \times N}$ is the normalized constraint matrix. Generally speaking, a large positive $\bar{Q}_{ij}$ indicates that node $i$ and $j$ should belong to the same cluster, and conversely a large negative entry indicates they should be in different clusters. $f^T \bar{L} f$ is the cost of the cut $f$; $f^T \bar{Q} f$ measures how well $f$ satisfies constraints in $\bar{Q}$. $\beta$ is the threshold that lower bounds constraint satisfaction. To solve Eq. (4.5), we solve the following generalized eigenvalue problem:

$$\bar{L} f = \lambda (\bar{Q} - \beta I) f. \tag{4.6}$$

Then they pick the eigenvectors associated with non-negative eigenvalues, i.e. these eigenvectors satisfy the constraint $f^T \bar{Q} f \geq \beta$. Among the non-negative eigenvectors, the one that minimizes $f^T \bar{L} f$ is the solution to Eq. (4.5).

### 4.3 An Overview of Our Main Results

Here we overview the main results of this work and discuss the relationship between them:

1. We introduce the notion of **stationary label propagation** in Definition 4.4.1, Section 4.4.1. It can be viewed as label propagation from a latent graph to an observed graph (see Fig. 4.3 for an example).

2. The equivalence between CSC and stationary label propagation is established in Claim 4.4.2, Section 4.4.2. It states that the labeling derived from the constrained cut found by CSC is a stationary labeling, where the constraint
matrix $\bar{Q}$ serves as $P_{GH}$ in Fig. 4.3 and the affinity matrix $\bar{A}$ serves as $P_{GG}$. This insight provides us with better understanding of how and why constrained spectral clustering works (Section 4.4.3). In particular, any labeling that violates the constraints will become non-stationary under propagation.

3. LLGC is a special case of our stationary label propagation framework, where $P_{GH}$ in Fig. 4.3 is an identity matrix (Section 4.4.4).

4. Given the relationship between label propagation and CSC established above, we propose a novel constraint construction algorithm. This algorithm first propagates the labels, and then generates a constraint matrix which is a Gaussian kernel based on the propagated labels (Algorithm 4, Section 4.5).

5. Empirical evaluation of the new algorithm with comparison to CSC, GFHF, and LLGC is presented in Section 4.6. Experimental results indicate that the new algorithm yields better (Fig. 4.4) and more stable (Fig. 4.5) results when given the same side information.

### 4.4 The Equivalence Between Label Propagation and Constrained Spectral Clustering

In this section, we explore the relation between label propagation and constrained spectral clustering. We propose a novel framework called stationary label propagation based on a variation of the GFHF propagation framework. This new framework enables us to give the CSC algorithm in Chapter 2 a label propagation interpretation. We also define stationary label propagation under the LLGC framework.

#### 4.4.1 Stationary Label Propagation: A Variation of GFHF

We have a graph $G$ with $N$ unlabeled nodes. To establish the label propagation process, we construct a label-bearing latent graph $H$, whose nodes have a one-to-one correspondence to the nodes of $G$. $H$ has no in-graph edges, which means its affinity matrix is an identity matrix $I$. There are edges from $H$ to $G$, encoded by the transition matrix $P_{GH}$.
Figure 4.3. An illustration of our stationary label propagation model \((N = 5)\). \(G\) is the unlabeled graph we want to propagate labels to. \(H\) is a latent node-bearing graph whose node set matches the node set of \(G\). The label propagation inside \(G\) is governed by the transition matrix \(P_{GG}\) (undirected edges). The propagation from \(H\) to \(G\) is governed by the transition matrix \(P_{GH}\) (directed edges).

The edges between the nodes of \(G\) are encoded by the transition matrix \(P_{GG}\). Fig. 4.3 is an illustration of our model.

Under the GFHF propagation rule (Eq.(4.1)), we have:

\[
y^{t+1} = \begin{bmatrix} y_H^{t+1} \\ y_G^{t+1} \end{bmatrix} = \begin{bmatrix} I & 0 \\ P_{GH} & P_{GG} \end{bmatrix} \begin{bmatrix} y_H^t \\ y_G^t \end{bmatrix}.
\]  \(4.7\)

Note that \(y_H^t, y_G^t \in \mathbb{R}^N\), \(I, P_{GH}, P_{GG} \in \mathbb{R}^{N \times N}\). From [73] we know that the labeling of graph \(G\), which is \(y_G^t\) in Eq.(4.7), converges to:

\[
f_G = \lim_{t \to \infty} y_G^t = (I - P_{GG})^{-1} P_{GH} y_H.
\]  \(4.8\)

The traditional GFHF framework assumes that \(y_H\) is given so that we compute \(f_G\) from \(y_H\) using Eq.(4.8). However, we now propose a new setting for label propagation, where we do not have a set of known labels to start with, i.e. \(y_H\) is unknown. Instead, we want to find a labeling \(f\) which will not change after the propagation has converged, and we call \(f\) a stationary labeling under propagation. Formally:

**Definition 4.4.1** (Stationary Label Propagation). Let \(f \in \mathbb{R}^N\) be a labeling of both graph \(G\) and \(H\). Under the propagation rule described in Eq.(4.7), we call \(f\) a stationary...
labeling if
\[ f = \lambda (I - P_{GG})^{-1} P_{GH} f, \quad (4.9) \]
where \( \lambda \neq 0 \) is a constant.

Intuitively speaking, a stationary labeling \( f \) is such a labeling on \( H \) that after propagation, the labeling on \( G \) will be the same as \( H \). Recall the stationary distribution of a transition matrix \( P \) reflects the underlying structure of a random walk graph. Similarly, the stationary labeling of \( P_{GG} \) and \( P_{GH} \) reflects the underlying structure of \( G \) and the way labels are propagated from the latent graph \( H \) to \( G \).

By re-organizing Eq.(4.9), we can see that given \( P_{GG} \) and \( P_{GH} \), their stationary labeling \( f \) can be computed by solving the generalized eigenvalue problem \((I - P_{GG}) f = \lambda P_{GH} f\), excluding the eigenvector associated with \( \lambda = 0 \). When \( I - P_{GG} \) and \( P_{GH} \) are both positive semi-definite and Hermitian matrices, under some mild conditions, above generalized eigenvalue problem guarantees to have \( N \) real eigenpairs.

### 4.4.2 CSC and Stationary Label Propagation

Next we establish the equivalence between stationary label propagation and constrained spectral clustering.

Consider the propagation matrix in Eq.(4.7). We use \( \bar{Q} \) to replace \( P_{GH} \) and \( \bar{A} \) to replace \( P_{GG} \). \( \bar{A} = D^{-1/2} A D^{-1/2} \) is the normalized affinity matrix of \( G \). \( \bar{A} \) is non-negative and positive semi-definite. Therefore it can (with proper normalization) serve as the transition matrix within \( G \). In the meanwhile, we require the constraint matrix \( Q \) in the CSC framework (Eq.(4.5)) to be non-negative and positive semi-definite. As a result, \( \bar{Q} = D^{-1/2} Q D^{-1/2} \) is also non-negative and positive semi-definite, and it can serve as the transition matrix from \( H \) to \( G \).

With the replaced symbols, we rewrite Eq.(4.9) as follows:

\[ f = \lambda (I - \bar{A})^{-1} \bar{Q} f. \]

The stationary labeling \( f \) is now the eigenvector of the generalized eigenvalue problem \((I - \bar{A}) f = \lambda \bar{Q} f\). Since \( \bar{L} = I - \bar{A} \) (\( \bar{L} \) is called the normalized graph Laplacian), we
have

\[ \tilde{L}f = \lambda \tilde{Q}f. \]  

(4.10)

Comparing Eq. (4.10) to Eq. (4.6), we can see that they are equivalent when \( \beta = 0 \). Notice that since \( \tilde{Q} \) is now positive semi-definite, when \( \beta = 0 \) the constraint in CSC is trivially satisfied, i.e. \( f^T \tilde{Q}f \geq \beta = 0, \forall f \). Hence we have:

Claim 4.4.2. With \( \beta \) set to 0, CSC as formulated in Eq. (4.5) finds all the stationary labeling \( f \) for given \( A \) and \( Q \), among which the one with the lowest cost on \( G \) will be chosen as the solution to CSC.

Intuitively speaking, the constraint matrix \( \tilde{Q} \) in CSC governs how the labels are propagated from the latent graph \( H \) to \( G \). CSC uses the threshold \( \beta \) to rule out the stationary labelings that do not fit \( \tilde{Q} \) well enough. The solution to CSC is then chosen by graph \( G \) from the qualified labelings based on the cost function \( f^T \tilde{L}f \).

4.4.3 Why Constrained Spectral Clustering Works: A Label Propagation Interpretation

The equivalence between CSC and stationary label propagation provides us with a new explanation for why and how the CSC algorithm works. Assume we have a graph \( G \). According to some ground truth, node \( i \) and \( j \) should belong to the same cluster. However, in graph \( G \), node \( i \) and \( j \) are not connected. As a result, if we cut \( G \) without any side information, node \( i \) and \( j \) may be incorrectly assigned to different clusters. Now we assume we have a constraint matrix \( Q \), where \( Q_{ij} = 1 \). It encodes the side information that node \( i \) and \( j \) should belong to the same cluster. Under the stationary label propagation framework, \( Q_{ij} = 1 \) specifies that node \( i \) in the latent graph \( H \) will propagate its label to node \( j \) in graph \( G \). As a result, an incorrect cut \( f \) where \( f_i \neq f_j \) will become non-stationary under the propagation. Instead, the constrained spectral clustering will tend to find such an \( f \) that \( f_i = f_j \).

Take Fig. 4.3 for example, if we cut graph \( G \) by itself, the partition will be \{1, 2, 3\}4, 5\}. However, the constraint matrix (\( P_{GH} \) in the figure) specifies that node 3 and 4 should have the same label. As a result, the constrained cut will become \{1, 2\}3, 4, 5\}. 
4.4.4 LLGC and Stationary Label Propagation

Under the LLGC propagation scheme, the propagated labels converge to $f = (1 - \alpha)(I - \alpha \bar{A})^{-1}y^0$, where $\alpha \in (0, 1)$. The stationary labeling under LLGC is:

$$f = \lambda(I - \alpha \bar{A})^{-1}f,$$

where $\lambda \neq 0$ (the term $1 - \alpha$ is absorbed by $\lambda$).

Comparing Eq.(4.11) to Eq.(4.9), we have:

Claim 4.4.3. Stationary label propagation under the LLGC framework is a special case of Definition 4.4.1, where $P_{GG} = \alpha \bar{A}$ and $P_{GH} = I$.

In other words, when propagating labels from the latent graph $H$ to graph $G$ under the LLGC framework, the label of node $i$ in $H$ will only be propagated to the corresponding node $i$ in graph $G$.

Combining Claim 4.4.2 and 4.4.3, we can further establish the equivalence between CSC and LLGC:

Claim 4.4.4. With $\beta$ set to 0 and $\bar{Q}$ set to $I$ (i.e. a zero-knowledge constraint matrix), CSC finds all the stationary labeling for given $\alpha \bar{A}$ under the LLGC framework, among which the one with the lowest cost will be chosen as the solution to CSC.

4.5 Generating Pairwise Constraints via Label Propagation

Inspired by the equivalence between label propagation and constrained spectral clustering, in this section we propose a novel algorithm that combines the two techniques for semi-supervised graph partition (summarized in Algorithm 4).

First we propagate the known labels $y_l$ to the entire graph using the GFHF method. We choose GFHF for the label propagation step instead of LLGC since in practice the side information is often from domain experts or ground truth. We do not want them to be changed during the propagation process. Let $y \in \mathbb{R}^N$ be the propagated labels. Then $Q$ can be encoded as follows:

$$Q_{ij} = \exp\left(-\frac{\|y_i - y_j\|^2}{2\sigma^2}\right).$$

(4.12)
It is easy to see that $Q$ is now non-negative and positive semi-definite. Since $y$ can be viewed as the semi-supervised embedding of the nodes, $Q$ is essentially the similarity matrix (or a kernel) for the nodes under the new embedding.

After constructing $Q$, we normalize it to get $\bar{Q}$. Then we solve the generalized eigenvalue problem in Eq.(4.10) to get all the stationary labelings. We pick the one that maximally satisfies the given constraints:

$$f^* = \arg \max_f f^T \bar{Q} f.$$ 

To derive a bi-partition from $f^*$, we simply assign nodes corresponding to the positive entries in $f^*$ to one cluster and negative entries the other. Notice that our algorithm, unlike the original CSC algorithm in Chapter 2, is parameter-free.

4.6 Empirical Study

Our empirical study aims to answer the following questions:

1. In terms of performance, does label propagation dominate constrained spectral clustering or vice versa?

2. How does our algorithm compare to existing label propagation and constrained spectral clustering techniques in terms of accuracy?

3. Does our algorithm yield more stable results, i.e. is it able to generate more helpful constraint set?

4.6.1 Experiment Setup

We used six different UCI benchmark datasets, namely Hepatitis, Iris, Wine, Glass, Ionosphere and Breast Cancer Wisconsin (Diagnostic). We removed the setosa class from the Iris data set and Class 3 from the Wine data set. After preprocessing, all datasets have two classes with ground truth labels. We used the RBF kernel to construct the affinity matrix of the graph.

We implemented five different techniques:
Algorithm 4: CSC+GFHF

Input: Initial labeling $y_l$, affinity matrix $A$, degree matrix $D$;
Output: $f$;

1 $P \leftarrow D^{-1}A$;
2 $y_u \leftarrow (I - P_{uu})^{-1}P_{ul}y_l$;
3 $y \leftarrow \begin{bmatrix} y_l \\ y_u \end{bmatrix}$;
4 for $i = 1$ to $N$ do
5 for $j = 1$ to $N$ do
6 $Q_{ij} \leftarrow \exp(-\frac{||y_i-y_j||^2}{2\sigma^2})$;
7 end
8 end
9 $D_Q \leftarrow \text{diag}(\sum_{i=1}^{N} Q_{ii}, \ldots, \sum_{i=1}^{N} Q_{iN})$;
10 $\bar{Q} \leftarrow D_Q^{-1/2}QD_Q^{-1/2}$;
11 $\bar{L} \leftarrow I - D^{-1/2}AD^{-1/2}$;
12 Solve the generalized eigenvalue problem $\bar{L}f = \lambda\bar{Q}f$;
13 Let $\mathcal{F}$ be the set of all generalized eigenvectors;
14 Remove the generalized eigenvector associated with $\lambda = 0$ from $\mathcal{F}$;
15 $f \leftarrow \arg\max_{f \in \mathcal{F}} f^T\bar{Q}f$;
16 return $f$;

- **Spectral**: Spectral clustering [51] without side information. This serves as the baseline performance.

- **GFHF [74]**: We propagate the known labels ($y_l$) on the graph, and partition the graph based on the propagated labels ($f$) by assigning nodes with positive label values to one cluster and negative the other.

- **LLGC [70]**: The regularization parameter in LLGC was set to $\alpha = 0.5$ throughout the experiments.
• **CSC**: The original constrained spectral clustering algorithm in Chapter 2. The constraint matrix was generated directly from given labels where $Q_{ij} = 1$ if the two nodes have the same label, $-1$ if the two nodes have different labels, 0 otherwise.

• **CSC+GFHF**: The constrained spectral clustering algorithm where the constraint matrix is constructed from propagated labels (Algorithm 4).

For each trial, we randomly revealed a subset of ground truth labels as side information. We applied the above algorithms to find a bi-partition, using the given label set. We evaluated the clustering results against the ground truth labeling using adjusted Rand index. Adjust Rand index equal to 0 means the clustering is as good as a random partition and 1 means the clustering perfectly matches the ground truth. For each dataset, we varied the size of the known label set from 5% to 50% of the total size. We randomly sampled 100 different label sets for each size.

### 4.6.2 Results and Analysis

**The accuracy of our algorithm:** We report the average adjusted Rand index of all five techniques on the UCI benchmark datasets in Fig. 4.4. The $x$-axis is the number of known labels. Existing constrained spectral clustering (CSC) and label propagation (GFHF and LLGC) algorithms managed to improve over the baseline method (Spectral) only on three of six datasets. They failed to find a better partition on Wine, Glass, and Breast Cancer, even with a large number of known labels. In contrast, our approach (CSC+GFHF) was able to outperform the baseline method on all six datasets with a small number of labels. More importantly, our algorithm consistently outperformed its competitors (CSC, GFHF and LLGC) on all datasets.

**The stability of our algorithm:** To examine the stability of our algorithm as compared to existing approaches, we computed their performance gain over the baseline method (Spectral). Specifically, we counted out of 100 random trials how many times the four techniques (GFHF, LLGC, CSC, and CSC+GFHF) can outperform the baseline, respectively. Note that previous work on constrained clustering [19] showed that a given constraint set could contribute either positively or negatively to the clustering, therefore
Figure 4.4. The average adjusted Rand index over 100 randomly chosen label sets of varying sizes.
Figure 4.5. The percentage of randomly chosen label sets that lead to positive performance gain with respect to the spectral clustering baseline.
being able to generate constraints that are more likely to be helpful is crucial in practice. In Fig. 4.5 we report the percentage of trials with positive performance gain for all four techniques. We can see that CSC+GFHF consistently outperformed its competitors on all but one dataset (all four techniques performed comparably on the Hepatitis dataset). It is especially the case when we start with a very small number of labels, which means the constraint matrix for CSC is very sparse and unstable. Label propagation mitigated the problem by constructing a dense constraint matrix (CSC+GFHF). As the number of known labels increased, the results of the two algorithms eventually converged. Fig. 4.5 suggests that the label propagation step indeed helped to generate better constraint sets.

4.7 Summary

In this chapter we explored the relationship between two popular semi-supervised graph learning techniques: label propagation, which uses node labels as side information, and constrained spectral clustering, which uses pairwise constraints as side information. We related the two approaches by introducing a new framework called stationary label propagation, under which either node labels or pairwise constraints can be encoded as the transition matrix from a latent graph to the observed unlabeled graph. A stationary labeling will then simultaneously capture the characteristics of the observed graph and the side information. Inspired by this new insight, we proposed a new constraint construction algorithm. Empirical results suggested that our algorithm is more accurate at recovering the ground truth labeling than the state-of-the-art label propagation and constrained spectral clustering algorithms. More importantly, its performance is also more stable over randomly chosen label sets.
Chapter 5

Multi-View Spectral Clustering

5.1 Introduction

Traditional spectral clustering only applies to a single graph/view/dataset [51, 55]. However, in a wide range of applications, the same dataset can be simultaneously characterized by multiple graphs, which are often constructed from heterogeneous sources. The most common setting, multi-view spectral clustering, is an extension of spectral clustering to multi-view datasets and it is still a developing area.

Previous work on multi-view spectral clustering typically combines different views so as a single objective function is optimized. This inherently makes the assumption that the different views are compatible to each other. Previous work also requires the users to set a parameter that regularizes the combination and thus implicitly decides the outcome of the algorithm.

In this chapter, we explore an alternative and more natural formulation that treats the problem as a multi-objective problem. Given two views, we create a bi-criteria objective function (see Eq. 5.1) that simultaneously considers the quality of a single cut on both graphs. This cut can be considered as a tradeoff between the two views/objectives. To solve the multi-objective problem, we use the classic Pareto optimization framework, which allows multiple objectives to compete with each other in deciding the optimal tradeoffs.

Our multi-objective spectral clustering formulation has several benefits and makes
the following contributions to the field:

- We can solve the multi-objective problem using Pareto optimization. The Pareto frontier captures all possible good cuts that are preferred by one or more objectives. (Section 5.4)

- We present a novel algorithm that reduces the search space from infinite number of possible cuts (since a cut in the relaxed sense is just a real vector) to a small set of mutually orthogonal cuts, so that the Pareto frontier can be computed efficiently. (Section 5.4.1)

- We provide an approximation bound on how good the solution in the reduced space is. The bound states how much better an optimal solution in the full search space can be than the one in the reduced search space. (Section 5.4.3)

- The Pareto optimal cuts can be interpreted either individually as alternative clusterings or collectively as a Pareto embedding of the dataset. (Section 5.4.2)

- The effectiveness of our approach is justified on benchmark datasets with comparison to state-of-the-art multi-view spectral clustering techniques (Section 5.5). We also demonstrate a novel application of our algorithm for resting-state fMRI analysis, where one graph represents the ground truth, and the other the observed data. (Section 5.6)

5.2 Related work

To our knowledge no work exists on multi-objective spectral clustering with the closest work being multi-view clustering. Previous work on multi-view (spectral) clustering relies on a fundamental assumption that all the views are compatible to each other, i.e. different views are generated from the same underlying distribution [9], or different views agree on a consensus partition that reflects a hidden ground truth [44]. This assumption is then exploited to convert multi-view spectral clustering into a single objective problem, which either maximizes the agreement between the partitions generated by different
views [40, 42], or combines multiple views into one view with the anticipation that the combined view is a better representation of the underlying distribution [21, 71]. In contrast our multi-objective formulation allows the two graphs to be incompatible and compete with each other based on their own preferences. The most preferred cuts will be captured by the Pareto frontier, which represents a range of alternative yet optimal ways to partition the dataset. Pareto optimization is popular in many computer science areas (see [33] for a review), since it provides a principled way of optimizing tradeoffs between competing objectives.

5.3 The Pareto Optimization Framework for Multi-View Spectral Clustering

In this section, we propose our multi-objective formulation for spectral clustering and show how to solve it in the context of Pareto optimization. We follow the standard formulation and notations of spectral clustering [51, 55] (see Table 5.1). We start with the two-view case, then later discuss its extension to more than two views (Section 5.4.4).

5.3.1 A Multi-Objective Formulation

A two-view dataset can be represented by two graphs that share the same set of $N$ nodes but have different sets of edges, namely $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$. Our goal is to find a shared cut that simultaneously cuts both graphs with minimal cost. This leads us to a
natural extension of spectral clustering. Here instead of finding the normalized min-cut on one graph, we find the normalized min-cut over the two graphs simultaneously:

\[
\arg\min_{v \in \Omega} \{v^T \tilde{L}_1 v, v^T \tilde{L}_2 v\},
\]

(5.1)

where

\[
\Omega \equiv \{v \in \mathbb{R}^N \mid v^T v = 1, v \perp_{\tilde{L}_1} D_1^{1/2} \mathbf{1}, v \perp_{\tilde{L}_2} D_2^{1/2} \mathbf{1}\}
\]

(5.2)

is the set of all nontrivial cuts. The notation \(v \perp_X v'\) means \(v^T X v' = 0\). \(v \in \Omega\) means that \(v\) is normalized and it is orthogonal to the trivial cut \(1\) (w.r.t. \(\tilde{L}_1\) and \(\tilde{L}_2\)).

Note that Eq. (5.1) can be reduced to spectral clustering if we replace \(\tilde{L}_2\) with \(\tilde{L}\) and \(\tilde{L}_1\) with the identity matrix \(I\). In other words, spectral clustering on a single graph is covered as a special case of our model where we combine the given graph with a zero-knowledge graph (whose normalized graph Laplacian is \(I\)).

### 5.3.2 Joint Numerical Range and Pareto Optimality

Rather than converting the two objectives in Eq. (5.1) to a single objective, we solve them simultaneously using Pareto optimization. Because we find a single cut for both graphs, we can consider finding this cut as a competition with each graph giving the cut a “score” (the cut quality). We enumerate all possible cuts by their costs on the respective graphs, which constitute the joint numerical range [27] of the two graphs. Each point in the joint numerical range represents a tradeoff between the two graphs in terms of cut cost. Then, we compute the Pareto frontier of the joint numerical range, which corresponds to the cuts that are optimal in terms of Pareto improvement: its cost on one graph cannot be improved (decrease) without making the cost on the other graph worse (increase).

The joint numerical range of \(\mathcal{G}_1\) and \(\mathcal{G}_2\) is defined as follows:

\[
\mathcal{J}(\mathcal{G}_1, \mathcal{G}_2) \equiv \{(v^T \tilde{L}_1 v, v^T \tilde{L}_2 v) \mid v \in \Omega\},
\]

(5.3)

where \(\Omega\) is defined as in Eq. (5.2). Essentially, \(\mathcal{J}(\mathcal{G}_1, \mathcal{G}_2)\) is the set of the costs of all nontrivial cuts over \(\mathcal{G}_1\) and \(\mathcal{G}_2\).
Recall in spectral clustering, we evaluate the quality of any two cuts by comparing their costs on a single graph. We say \( v \) is better than \( v' \) if \( v \) has a lower cost than \( v' \) does. Now consider the joint numerical range of two graphs. When we evaluate the quality of a cut, we must consider its cost on both graphs. Specifically, we need to introduce the notion of Pareto improvement:

**Definition 5.3.1. (Pareto Improvement)** Given two different cuts \( v \in \Omega \) and \( v' \in \Omega \) over two graphs \( G_1 \) and \( G_2 \), we say \( v \) is a Pareto improvement over \( v' \) if and only if one of the following two conditions holds:

\[
v^T \tilde{L}_1 v < v'^T \tilde{L}_1 v', \quad v^T \tilde{L}_2 v \leq v'^T \tilde{L}_2 v',
\]

or

\[
v^T \tilde{L}_1 v \leq v'^T \tilde{L}_1 v', \quad v^T \tilde{L}_2 v < v'^T \tilde{L}_2 v'.
\]

When \( v \) is a Pareto improvement over \( v' \), we say \( v \) dominates \( v' \), or \( v' \) is dominated by \( v \), and we use the following notation:

\[
v \prec_{(G_1, G_2)} v'.
\]

In terms of Pareto improvement, the optimal solution to Eq. (5.1) is the Pareto frontier of \( J(G_1, G_2) \).

**Definition 5.3.2. (Pareto Frontier)** Define:

\[
\mathcal{F}(G_1, G_2) \triangleq \{(v^T \tilde{L}_1 v, v^T \tilde{L}_2 v) \mid v \in P\}.
\]

\( \mathcal{F}(G_1, G_2) \) is the Pareto frontier of \( J(G_1, G_2) \) if \( P \) satisfies:

1. \( P \subset \Omega \);
2. (Optimality) \( \forall v \in P, \not\exists v' \in \Omega \), such that \( v' \prec_{(G_1, G_2)} v \);
3. (Completeness) \( \forall v \in \Omega \setminus P, \exists v' \in P \), such that \( v' \prec_{(G_1, G_2)} v \).

We say \( v \) lies on the Pareto frontier of \( J(G_1, G_2) \) if \( v \in P \).
We call $P$ the set of Pareto optimal cuts. Intuitively speaking, $P$ satisfies the following properties: 1) any cut in $P$ is better than cuts that are not in $P$ (completeness); 2) any two cuts in $P$ are equally good; 3) for any cut in $P$, it is impossible to reduce its cost on one graph without increasing its cost on the other graph (optimality). Therefore, our Pareto optimization framework captures the complete set of equally good cuts (in terms of Pareto optimality) that are superior to any other possible cuts.

We summarize our approach as follows:

1. Given the two graphs $G_1$ and $G_2$, construct their joint numerical range $J(G_1, G_2)$.
2. Compute the Pareto frontier of $J(G_1, G_2)$, which is $F(G_1, G_2)$.
3. Output $P$, the set of Pareto optimal cuts.

5.4 Algorithm Derivation

In this section, we present an efficient approximation algorithm to compute the Pareto frontier. We also discuss how to interpret the Pareto optimal cuts and convert them into clusterings in practice. Our algorithm is summarized in Algorithm 5.

5.4.1 Computing the Pareto Frontier via Generalized Eigendecomposition

Recall that $F(G_1, G_2) \subset J(G_1, G_2)$ is the Pareto frontier and $P \subset \Omega$ is the set of Pareto optimal cuts. Our goal is to compute $F(G_1, G_2)$, or equivalently $P$. However, $\Omega$ consists of an infinite number of different cuts (in the relaxed form), which map to an infinite number of points in $J(G_1, G_2)$. To the best of our knowledge, there is no efficient way to compute $F(G_1, G_2)$ in closed form.

Nevertheless, although $\Omega$ consists of an infinite number of cuts, many of those cuts are effectively identical to each other. For instance, one cut might just be another cut plus a small perturbation. From a practical point of view, those two cuts will lead to exactly the same partition. Therefore, we introduce an additional constraint to narrow down our search space: we only focus on a subset of cuts that are distinctive to each other, namely they must be mutually orthogonal. Consequently, instead of dealing with
Algorithm 5: Multi-View Spectral Clustering via Pareto Optimization

**Input:** Two graph Laplacians: $\bar{L}_1, \bar{L}_2$

**Output:** The set of Pareto optimal cuts $\tilde{P}$

1. Solve the generalized eigenvalue problem: $\bar{L}_1 v = \lambda \bar{L}_2 v$;
2. Normalize all $v$’s such that $v^T v = 1$;
3. Let $\tilde{P}$ be the set of all eigenvectors, excluding the two associated with eigenvalue 0 and $\infty$;
4. foreach $v \in \tilde{P}$ do
   5. foreach $v' \in \tilde{P}, v' \neq v$ do
      6. if $v \prec_{(G_1, G_2)} v'$ then // $v$ dominates $v'$
         7. Remove $v'$ from $\tilde{P}$;
         8. continue;
      9. end
   10. if $v' \prec_{(G_1, G_2)} v$ then // $v'$ dominates $v$
       11. Remove $v$ from $\tilde{P}$;
       12. break;
       13. end
   14. end
5. (Optional) Consolidate the cuts in $\tilde{P}$ into a single clustering $u$ (see Section 5.4.2);

a continuous vector space $\Omega$, we only consider the set of vectors, $\hat{\Omega}$, which comprise an orthogonal basis of $\Omega$.

Formally, we define

$$\hat{\Omega} \triangleq \{ v \in \Omega \mid \forall v \neq v', v \perp_{\bar{L}_1} v', v \perp_{\bar{L}_2} v' \}$$

$$\tilde{\mathcal{J}}(G_1, G_2) \triangleq \{ (v^T \bar{L}_1 v, v^T \bar{L}_2 v) \mid v \in \hat{\Omega} \}.$$  

Under a mild assumption that the null space of $\bar{L}_1$ and $\bar{L}_2$ do not overlap, $(\bar{L}_1, \bar{L}_2)$ is a Hermitian definite matrix pencil [5]. Then $\hat{\Omega}$ is the set of $N$ ($N$ is the number of nodes)
eigenvectors of the generalized eigenvalue problem [24]:

\[ L_1 v = \lambda L_2 v \]  \hspace{1cm} \text{(5.4)}

less the principal eigenvector of $\bar{L}_1$, which is $D_1^{1/2}1$, and the principal eigenvector of $\bar{L}_2$, which is $D_2^{1/2}1$. The generalized eigenvalue problem in Eq. (5.4) can be solved efficiently in closed form.

Now since $\tilde{J}(G_1, G_2)$ only consists of $N-2$ points, corresponding to the $N-2$ mutually orthogonal cuts in $\tilde{\Omega}$, we can efficiently find the Pareto frontier (see Algorithm 5). We define:

\[ \tilde{F}(G_1, G_2) \triangleq \{(v^T \bar{L}_1 v, v^T \bar{L}_2 v) \mid v \in \tilde{P}\}. \]

$\tilde{F}(G_1, G_2)$ is an approximation to $F(G_1, G_2)$. We call $\tilde{F}(G_1, G_2)$ the orthogonal Pareto frontier and $\tilde{P}$ the orthogonal Pareto optimal cuts. We will provide a bound for this approximation in Section 5.4.3.

The runtime of our algorithm is dominated by that of generalized eigendecomposition, which is on par with that of spectral clustering in big-$O$ notation.

**Example** We use the UCI Wine dataset to demonstrate how our algorithm works. It consists of 119 instances. Each instance has 13 features (attributes). We construct one view using the first 6 features and the other view using the remaining 7 features. After applying our algorithm, we find 117 points in $\tilde{J}(G_1, G_2)$ which correspond to 117 nontrivial orthogonal cuts of the graph, as shown in Fig. 5.1 (+’s). Among the 117 cuts, three lie on the Pareto frontier $\tilde{F}(G_1, G_2)$ (the circled points). We visualize the clusterings derived from the three Pareto optimal cuts in Fig. 5.2. Note that Cut 3 (Fig. 5.2(c)) captures the clustering derived from the original labels of the Wine dataset (Fig. 5.2(d)).

5.4.2 Interpreting and Using the Pareto Optimal Cuts

The Pareto optimal cuts in $\tilde{P}$ can be interpreted either individually as alternative clusterings or collectively as a Pareto embedding of the dataset.

Specifically, if the two views are compatible with each other, then by definition, they would agree on a single cut that is Pareto optimal. In this case, our algorithm
Figure 5.1. The joint numerical range of the Wine dataset. The +’s correspond to points in $\tilde{J}(G_1, G_2)$. The ◦’s are the Pareto optimal cuts found by our algorithm, which is $\tilde{F}(G_1, G_2)$.

will produce a unique clustering that is optimal. If the two views are not compatible (which is the case for the Wine dataset in Fig. 5.1), the cardinality of $\tilde{P}$ will be greater than 1. In this case, the Pareto optimal cuts can be interpreted as a set of alternative clusterings. On the one hand, these cuts are alternative to each other in terms of orthogonality. On the other hand, as shown in Fig. 5.2, different Pareto optimal cuts correspond to different ways to partition the dataset: (a) separates three outliers from the rest of the data points, (b) partitions the points vertically, and (c) partitions the points horizontally. These three alternative clusterings are all informative and could all make sense, depending on the users’ needs.

In practice, $|\tilde{P}|$ is usually small. Hence it is feasible to submit $\tilde{P}$ directly to domain experts for further review. We argue that it is more intuitive and much easier for domain
experts to choose among a small number plausible clusterings than assigning a parameter \textit{a priori} that implicitly decides the outcome of the algorithm.

Sometimes the application demands one single partition as output. In this case, we can interpret the Pareto optimal cuts in $\tilde{P}$ collectively using the classic spectral embedding technique \cite{4, 8}. Specifically, let $V$ be a $N \times |\tilde{P}|$ matrix, whose columns are the Pareto optimal cuts in $\tilde{P}$. If we look at the $i$-th rows of $V$, it can be considered as an embedding of the $i$-th node of the graph in a $|\tilde{P}|$-dimensional subspace, spanned by the mutually orthogonal generalized eigenvectors (Fig. 5.2 is the Pareto embedding of the Wine dataset). To derive a single clustering, we perform $K$-means on the Pareto
embedding of all nodes, which is also common practice.

In addition, we used in our experiments a simple but effective unsupervised weighting scheme that can further improve the result. We assigned each Pareto optimal cut a weight that is inversely proportional to the squared sum of its costs on respective graphs. In other words, all cuts being Pareto optimal, we assign higher weights to those with lower overall costs.

5.4.3 Approximation Bound for Our Algorithm

In our algorithm, we compute the orthogonal Pareto frontier $\tilde{F}(G_1, G_2)$ as an approximation to the Pareto frontier $F(G_1, G_2)$. Here we create an upper bound on how far a point in the Pareto frontier can be to the orthogonal Pareto frontier. This effectively bounds the difference between the costs of the cuts on the Pareto frontier and those on the orthogonal Pareto frontier.

Let $\text{co}(\tilde{J}(G_1, G_2))$ be the convex hull of $\tilde{J}(G_1, G_2)$. It is a convex polygon that lies in $\mathcal{J}(G_1, G_2)$ (see Fig. 5.1). Let $\text{ext}(\tilde{J}(G_1, G_2))$ be its extreme points (“corners” of the convex polygon). Let

$$B \triangleq \text{ext}(\tilde{J}(G_1, G_2)) \cap \tilde{F}(G_1, G_2).$$

$B$ is nonempty (e.g. the leftmost and lowest points in $\tilde{J}(G_1, G_2)$ are both in $B$). First, it is obvious that any points in $\text{co}(\tilde{J}(G_1, G_2))$ cannot dominate points in $B$. Then, we examine the chance that any points in $\mathcal{J}(G_1, G_2)$ dominate points in $B$.

Let $\tilde{\Omega} = \{\tilde{v}_i\}_{i=1}^{N-2}$ and $\tilde{V} = (\tilde{v}_1, \ldots, \tilde{v}_{N-2})$. Any $v \in \Omega$ can be represented by a linear combination of $\tilde{v}_i$’s: $v = \tilde{V}a$, $a = (a_1, \ldots, a_{N-2})^T$. We define $f(v) : \Omega \mapsto \mathcal{J}(G_1, G_2)$:

$$f(v) = (v^T\bar{L}_1v, v^T\bar{L}_2v)$$  \hspace{1cm} (5.5)

$$= \left( \sum_{i=1}^{N-2} a_i\tilde{v}_i^T\bar{L}_1 \sum_{j=1}^{N-2} a_j\tilde{v}_j, \sum_{i=1}^{N-2} a_i\tilde{v}_i^T\bar{L}_2 \sum_{j=1}^{N-2} a_j\tilde{v}_j \right)$$  \hspace{1cm} (5.6)

$$= \left( \sum_{i=1}^{N-2} \sum_{j=1}^{N-2} a_i a_j \tilde{v}_i^T\bar{L}_1 \tilde{v}_j, \sum_{i=1}^{N-2} \sum_{j=1}^{N-2} a_i a_j \tilde{v}_i^T\bar{L}_2 \tilde{v}_j \right)$$  \hspace{1cm} (5.7)

$$= \sum_{i=1}^{N-2} a_i^2 \tilde{v}_i^T\bar{L}_1 \tilde{v}_i, \sum_{i=1}^{N-2} a_i^2 \tilde{v}_i^T\bar{L}_2 \tilde{v}_i$$  \hspace{1cm} (5.8)
∥·∥ is 2-norm. The transition from Eq. (5.7) to (5.8) is due to the fact that, for \(i \neq j\), \(\tilde{v}_i\) and \(\tilde{v}_j\) are mutually orthogonal with respect to \(L_1\) and \(L_2\), according to the definition of \(\tilde{Ω}\). Eq. (5.10) simply replaces the two items in Eq. (5.9) with shorter notation.

Since \(\sum_{i=1}^{N-2} \frac{a_i^2}{\|a\|_2^2} = 1\), \((x, y)\) is a convex combination of points in \(\tilde{J}(\mathcal{G}_1, \mathcal{G}_2)\), therefore \((x, y) \in \text{co}(\tilde{J}(\mathcal{G}_1, \mathcal{G}_2))\). In other words, for any \(v \in Ω\), \(f(v)\) can be represented by a point in \(\tilde{J}(\mathcal{G}_1, \mathcal{G}_2)\) multiplied by a scaling factor \(\|a\|_2^2\). If \(\|a\|_2^2 = 1\), then \(f(v) = (x, y) \in \text{co}(\tilde{J}(\mathcal{G}_1, \mathcal{G}_2))\) and it cannot dominate any point in \(B\). If \(\|a\|_2^2 > 1\), then \(f(v)\) is dominated by \((x, y)\), therefore, it cannot dominate any point in \(B\).

On the other hand, we can derive a lower-bound for \(\|a\|_2^2\). We have:

\[
1 = \|v\| = \|\tilde{V}a\| \leq \|\tilde{V}\|\|a\| = \sigma_{\max}(\tilde{V})\|a\|,
\]

where \(\sigma_{\max}(\tilde{V})\) is the largest singular value of \(\tilde{V}\). Consequently we have:

\[
\|a\|_2^2 \geq \frac{1}{\sigma_{\max}^2(\tilde{V})}.
\]  \hspace{1cm} (5.11)

\(1/\sigma_{\max}^2(\tilde{V})\) effectively bounds how far \(f(v)\) can be from the point \((x, y)\), which is in \(\text{co}(\tilde{J}(\mathcal{G}_1, \mathcal{G}_2))\). The larger \(1/\sigma_{\max}^2(\tilde{V})\) is, the closer \(f(v)\) is to \((x, y)\), thus the better \(\text{co}(\tilde{J}(\mathcal{G}_1, \mathcal{G}_2))\) approximates \(J(\mathcal{G}_1, \mathcal{G}_2)\) and the more likely \(B\) coincides with \(F(\mathcal{G}_1, \mathcal{G}_2)\).

The equality in Eq. (5.11) holds when \(v\) is the largest right singular vector of \(\tilde{V}\). As shown in Fig. 5.1, \(\Delta\) is \(f(v) = \|a\|_2^2(x, y)\) when \(\|a\|_2^2\) reaches the lower-bound on the Wine dataset; \(\Box\) is the corresponding \((x, y)\). Note that \(\|a\|_2^2 < 1\) is a necessary but not sufficient condition for \(f(v)\) to dominate any point in \(B\). For example, in Fig. 5.1, although \(f(v)\) lies outside the convex hull of \(\tilde{J}(\mathcal{G}_1, \mathcal{G}_2)\), it does not dominate any point in \(B\), which are the three circled points.

5.4.4 Extension to Multiple Views

It is possible to extend our framework to \(M\) views. Given a finite number of cuts across \(M\) views, it is not difficult to compute the \(M\)-dimensional Pareto frontier. The
challenge is to discretize the joint numerical range of $M$ graphs, since the generalized eigenvalue system can only accommodate two graphs. To cope with the limitation, we can combine each view with the average of the other $M - 1$ views, respectively. We can apply generalized eigendecomposition to compute the orthogonal joint numerical range of those two graphs. We repeat this process $M$ times and will get $M(N - 2)$ cuts. Then we can compute the Pareto frontier of the $M(N - 2)$ cuts. This approach ensures that a good cut will be preserved as long as it is preferred by at least one view.

## 5.5 Empirical Study

In this section, we use the UCI benchmark datasets [3] and the 20 Newsgroups dataset\(^1\) to empirically evaluate the effectiveness of our approach. We aim to answer the following questions:

- How does our algorithm perform on datasets with incompatible views? (Table 5.3, Fig 5.3).
- How does it perform on datasets with compatible views? (Table 5.4, Fig 5.3).
- How does it compare to spectral clustering baselines and the state-of-the-art multi-view spectral clustering techniques? (Tables 5.3 and 5.4, Fig 5.3).

\(^1\)http://cs.nyu.edu/~roweis/data.html
The short answer to these questions is (see Fig. 5.3) that our technique performs comparably to other multi-view clustering techniques on datasets with compatible views, but outperforms all techniques by a large margin on datasets with incompatible views. This is a significant result since testing if views are compatible or incompatible is an open research problem.

We chose six UCI datasets, namely Hepatitis, Iris, Wine, Glass, Ionosphere, and Breast Cancer. To construct the two views, we divided the features into two disjoint subsets. We divided them in such a way that the two views tend to be incompatible, e.g. we put different types of features into opposite views. The graph Laplacians were computed using RBF kernel. We also used the 20 Newsgroups dataset that contains documents from four high-level categories: comp, rec, sci, and talk. These categories were used as ground truth labels. The features of the dataset are 100 representative words. To construct the two views, we randomly divided the features into two subsets, each with 50 words. Therefore, for this dataset, the two views tend to be compatible. The graph Laplacians were computed using inner-product kernel, based on the word-frequency vectors. The statistics of the datasets after preprocessing are summarized in Table 5.2.

For our algorithm, we first computed the Pareto optimal cuts, then used their Pareto embedding to find a clustering. We evaluated this clustering against ground truth labels using adjusted Rand index [31]: 0 means the partition is as good as a random assignment and 1 means the partition perfectly matches the ground truth.

To make comparisons, we implemented several state-of-the-art multi-view spectral clustering algorithms (which all use a single objective). MM is the Markov mixture algorithm proposed in [71], where the two views are combined using a mixing random walk on both graphs. KerAdd is kernel addition algorithm that combines the two views by averaging their graph Laplacians. Though simplistic, this method has been shown to be very effective when two views are compatible [14, 40, 71] and outperform many more complicated alternatives. CoReg is the co-regularization multi-view spectral clustering algorithm proposed in [42]. We implemented the centroid based version and used
Table 5.3. The adjusted Rand index of various algorithms on six UCI datasets with incompatible views. Bold numbers are best results. The number in the parenthesis is the performance gain of our approach (Pareto) over the best competitor. Our method performs the best on the majority of datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>View1</th>
<th>View2</th>
<th>Concat.</th>
<th>MM</th>
<th>KernAdd</th>
<th>CoReg</th>
<th>Pareto</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hepatitis</td>
<td>-.109</td>
<td>.247</td>
<td>.193</td>
<td>-.091</td>
<td>-.111</td>
<td>.247</td>
<td>.360(+.113)</td>
</tr>
<tr>
<td>Iris</td>
<td>.136</td>
<td><strong>.808</strong></td>
<td>.485</td>
<td>.430</td>
<td>.430</td>
<td>.404</td>
<td><strong>.808(+.000)</strong></td>
</tr>
<tr>
<td>Wine</td>
<td>-.015</td>
<td>.869</td>
<td>-.015</td>
<td>.869</td>
<td><strong>.933</strong></td>
<td><strong>.933</strong></td>
<td><strong>.933(+.000)</strong></td>
</tr>
<tr>
<td>Glass</td>
<td><strong>.510</strong></td>
<td>.041</td>
<td>.413</td>
<td>.474</td>
<td>.448</td>
<td><strong>.510</strong></td>
<td>.490(-.020)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>.209</td>
<td>-.043</td>
<td>-.043</td>
<td>.209</td>
<td><strong>.257</strong></td>
<td>.209</td>
<td>.209(-.048)</td>
</tr>
<tr>
<td>Breast</td>
<td>.005</td>
<td>.005</td>
<td>.112</td>
<td>.005</td>
<td>.002</td>
<td>.297</td>
<td><strong>.368(+.071)</strong></td>
</tr>
</tbody>
</table>

Table 5.4. The adjusted Rand index of various algorithms on the 20 Newsgroups dataset with compatible views. Bold numbers are best results. The number in the parenthesis is the performance gain of our approach (Pareto) over the best competitor. Note our method is comparable to other methods. The best performing method MM, performs poorly in Table 5.3.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>View1</th>
<th>View2</th>
<th>Concat.</th>
<th>MM</th>
<th>KernAdd</th>
<th>CoReg</th>
<th>Pareto</th>
</tr>
</thead>
<tbody>
<tr>
<td>comp-rec</td>
<td>.697</td>
<td>.719</td>
<td>.747</td>
<td><strong>.758</strong></td>
<td>.747</td>
<td>.741</td>
<td>.747(-.011)</td>
</tr>
<tr>
<td>comp-sci</td>
<td>.520</td>
<td>.506</td>
<td>.700</td>
<td>.702</td>
<td><strong>.717</strong></td>
<td>.688</td>
<td>.684(-.033)</td>
</tr>
<tr>
<td>comp-talk</td>
<td>.837</td>
<td>.702</td>
<td>.939</td>
<td>.939</td>
<td>.939</td>
<td>.939</td>
<td><strong>.957(+.018)</strong></td>
</tr>
<tr>
<td>rec-sci</td>
<td>.533</td>
<td>.605</td>
<td><strong>.640</strong></td>
<td>.633</td>
<td><strong>.640</strong></td>
<td>.626</td>
<td><strong>.640(+.000)</strong></td>
</tr>
<tr>
<td>rec-talk</td>
<td>.684</td>
<td>.681</td>
<td>.754</td>
<td><strong>.764</strong></td>
<td>.748</td>
<td>.748</td>
<td><strong>.725(-.039)</strong></td>
</tr>
<tr>
<td>sci-talk</td>
<td>-.011</td>
<td>.520</td>
<td>.558</td>
<td><strong>.566</strong></td>
<td>.559</td>
<td>.393</td>
<td><strong>.542(-.024)</strong></td>
</tr>
</tbody>
</table>

the centroids to compute final clustering. As a baseline, we also report the results of performing spectral clustering on each single view (View 1, View 2), as well as the concatenation of all features (Concat.).

The results are reported in Table 5.3 and 5.4. Our approach (Pareto) outperformed three spectral clustering baselines (View 1, View 2, Concat.) in most of the cases. This suggests that our approach is effective in combining the two views in a constructive way. When comparing to existing multi-view clustering techniques, our approach outperformed any single one of them. Across all 12 datasets, our approach achieved highest
ARI on 6 and second highest on 3.

More importantly, our approach performed more reliable than its competitors when the two views were constructed to be incompatible. Across 6 UCI datasets (Table 5.3), our approach achieved highest performance on 4 and second highest on the other 2. This justifies the advantage of our multi-objective framework over the single-objective framework used by previous methods. On the other hand, for the 20 Newsgroups datasets (Table 5.4) where the two views are constructed to be compatible, the advantage of our approach was less significant. Nevertheless, it was not outperformed by its competitors by a large margin.

To better demonstrate our approach’s consistent performance with both compatible and incompatible views, we compute the relative difference (in terms of adjusted Rand index) of each technique’s performance with respect to the best-performance approach per dataset. Then we compute the mean relative difference for each technique on the UCI datasets and the 20 Newsgroups dataset, respectively. Since no technique was always the best, the mean relative difference of all techniques is always less than zero.
However, in Fig. 5.3, we can clearly see that our algorithm is the only technique that performed consistently well in both cases (compatible and incompatible). In contrast, although the Concat., MM, and KerAdd performed very well on compatible views, they performed poorly on incompatible views.

5.6 Application: Automated fMRI Analysis

In this section we explore an application of our work where incompatible views naturally occur: resting-state fMRI analysis. A resting-state fMRI scan is a series of 3D brain images over time of a person at resting state. We can construct a graph for each scan, where each node corresponds to a voxel in the brain image, and the edge weight corresponds to the correlation between the activity of two voxels over time. If we partition this graph into two parts, one will comprise regions in the brain that share the same functionality (called a cognitive network), the other background. For our application, we are interested in a particular network, called the Default Mode Network (DMN) (see Fig. 5.4(a)), which is periodically activated when the person is in resting state. A weakened DMN has been related to the Alzheimer’s disease [12]. Our goal is to elicit the DMN from a given scan and determine its strength.

The challenge of this task is that fMRI scans are notoriously noisy. Many factors, such as equipment calibration, head positioning, and the mental state of the subject, can introduce a significant amount of noise into the scan. As a result, the same person scanned twice over the period of a month (as our data is) will produce two incompatible scans which suggest two very different clusterings. Combining two incompatible scans is not desirable because the noise in one scan can dominate the other scan. In effort to overcome this, we use our algorithm to simultaneously cut two scans: an exemplar scan and a target scan. The exemplar scan is a scan verified by domain experts that exhibits a strong DMN pattern. We pair this exemplar scan with a target scan, which may or may not be compatible, to detect the DMN therein.

Figure 5.4(a) shows what a DMN should look like. Note that it only illustrates the general shape of the DMN based on the average of a large number of scans. The actual
Figure 5.4. The results of applying our algorithm to resting-state fMRI scans. Illustrated is a horizontal slice of the scan (eyes are on the right-hand side). We use an exemplar scan (View 1) to induce the Default Mode Network (the red/yellow pixels in the figures) in a set of target scans (View 2). Our algorithm produced consistent partitions across different target scans.

DMN differs from individual to individual. Figure 5.4(b) shows the DMN exhibited by an exemplar scan from a young healthy person.

Given the exemplar scan and a target scan, our algorithm finds the set of Pareto optimal cuts. We compare each Pareto optimal cut to the DMN cut exhibited by the exemplar scan and choose the most similar one (as shown in Fig. 5.4(c) and (d)) as the induced DMN cut for the target scan. The induced DMN cut can be considered as
the target scan’s best effort (in terms of Pareto optimality) to accommodate the exemplar DMN cut. We then record the cost of the induced DMN cut for the target scan, which can be naturally used as an indicator for the strength of the DMN in the target scan. The lower the cost is, the more the target scan prefers the DMN cut, thus the stronger the DMN is in the target scan.

The dataset we used was collected and processed within the research program of UC Davis Alzheimer’s Disease Center. The exemplar scans were chosen by domain experts from a group of young healthy individuals. The target scans were from 31 elderly individuals: 11 diagnosed as Healthy, 10 Mild Cognitive Impairment (MCI), and 10 Dementia.

We observed that, despite the ubiquitous noise in fMRI scans, our algorithm managed to induce the DMN cut across all target scans, i.e. the candidate set always included a cut that is highly similar to the exemplar DMN in Figure 5.4(b). In Figure 5.4(c) and (d), we illustrate two induced DMN cuts from two different target scans. This demonstrated that our formulation can accommodate incompatible views and avoid destructive knowledge combination. Then we studied the costs of the induced cuts on three different sub-populations, namely Healthy, MCI, and Dementia. As shown in Figure 5.5(a), as the cognitive symptom develops, the costs of the induced cuts tend to increase, which means the strength of the DMN tends to decrease. To verify this, we tried a different exemplar scan and had similar results (Figure 5.5(b)). This observation provided direct support to the claim made in previous study [12] that the DMN diminishes as the Alzheimer’s disease progresses.

Existing multi-view techniques do not work well for this task since they assume compatible views. However, the two views, the exemplar and the target scan, are often incompatible due to not only the noise but also the fact that they are from different individuals. Consequently, existing methods suffer from destructive combination as indicated by earlier results (see Table 5.3). Moreover, the pattern we are interested in, the DMN, is often not the dominant pattern in the exemplar scan. This makes it much more difficult, if possible, for single-objective based techniques to find the DMN pattern.
in all the target scans.

5.7 Summary

In this chapter we explored multi-view spectral clustering using a multi-objective formulation. The search space of our objective is the joint numerical range of two graphs. We use Pareto optimization to find the optimal solutions, which is the Pareto frontier of the joint numerical range. To the best of our knowledge, we are the first to use Pareto optimization for multi-objective multi-view spectral clustering. We also proposed an efficient approximation algorithm to compute the Pareto frontier, which reduces the search space from an infinite number of cuts to a finite set of mutually orthogonal cuts.

We compared our work against a variety of algorithms in the multi-view setting. The pragmatic benefits of our approach over existing single-objective techniques are: 1) the users do not need to specify the weights for different views a priori; 2) the views need not to be compatible (a difficult-to-test property); 3) it efficiently enumerates plausible and alternative clusterings. We also explored using our multi-objective formulation in the setting where one objective captures the adherence to the ground truth and the other the adherence to the observed data.
Chapter 6

Conclusions

Heterogeneous datasets from real-world applications call for more complex graph models than a single graph. In this dissertation, we systematically study the extension of spectral clustering to a variety of complex settings. We propose a constrained optimization framework to incorporate pairwise constraints into the normalized min-cut objective. We show how to solve our new objective using generalized eigenvalue decomposition. Our framework can be used to encode different types of side information in practice, such as alternative metric distance, partial labeling, multiple views, etc. Our framework works not only when the side information is passively available, it can also actively acquire knowledge from an oracle with minimum cost. The effectiveness of our approach is theoretically justified by its equivalence to a special setting of label propagation. Our proposed algorithms were empirically tested on several benchmark datasets, where they significantly outperformed existing state-of-the-art techniques. We applied our algorithms to two real-world problems: 1) improving the accuracy of document clustering by utilizing side information from automated machine translation; 2) identifying cognitive networks in human brains by partitioning resting-state fMRI scans.

Our work made contributions to several areas of data mining and machine learning: constrained clustering, spectral clustering, graph-based semi-supervised learning, transfer learning, multi-view learning, and active learning. There are some future directions worth pursuing. For example, our constrained framework can be viewed as a transfer of knowledge between different data sources. However, this is no guarantee that
the transferred knowledge will actually improve the result. How to tell when destructive knowledge transfer happened and how to fix it using active learning are important questions to answer for critical real-world applications. Another important topic is how to scale our algorithms to massive datasets. Scalability is always an issue for spectral clustering. Nevertheless, the connection we draw between constrained spectral clustering and label propagation may give us some insights into designing more efficient algorithms.

Acknowledgments

This dissertation was partly supported by ONR grants N00014-09-1-0712, N00014-11-1-0108, and NSF Grant NSF IIS-0801528. The fMRI study was partly supported by NIH grants AG10220, AG10129, AG030514, AG031252, and AG021028.
References


-108-


