Combining Disparate Information for Machine Learning

by

Ko-Jen Hsiao

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering: Systems)
in The University of Michigan
2014

For my dad, my mom, my brother and Linda.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table

ABSTRACT

Combining Disparate Information for Machine Learning

by

Ko-Jen Hsiao

Chair: Alfred O. Hero

This thesis considers information fusion for four different types of machine learning problems: anomaly detection, information retrieval, collaborative filtering and structure learning for time series, and focuses on a common theme – the benefit to combining disparate information resulting in improved algorithm performance.

In this dissertation, several new algorithms and applications to real-world datasets are presented. In Chapter II, a novel approach called Pareto Depth Analysis (PDA) is proposed for combining different dissimilarity metrics for anomaly detection. PDA is applied to video-based anomaly detection of pedestrian trajectories. Following a similar idea, in Chapter III we propose to use a similar Pareto Front method for a multiple-query information retrieval problem when different queries represent different semantic concepts. Pareto Front information retrieval is applied to multiple query image retrieval. In Chapter IV, we extend a recently proposed collaborative retrieval approach to incorporate complementary social network information, an approach we call Social Collaborative Retrieval (SCR). SCR is applied to a music recommendation system that combines both user history and friendship network information to improve recall and weighted recall performance. In Chapter V, we propose a framework that combines time series data at different time scales and offsets for more accurate estimation of multiple precision matrices. We propose a general fused graphical lasso approach to jointly estimate these precision matrices. The framework is applied to modeling financial time series data.

CHAPTER I

Introduction

1.1 Motivation

In this thesis we investigate the benefits of combining different types of information in order to improve the performance of machine learning tasks. We also provide design strategies to build systems that can utilize these different types of information. The idea of fusing disparate information can arise in many different situations in machine learning. Examples include: data collected from different modalities of sensors; fusion of totally different types of information such as a user's past behaviors and social network information; and how to find images which are simultaneously related to images of different semantic concepts. These examples all require combining disparate types of information.

1.2 Background and Contributions

We motivate the thesis by considering several relevant types of machine learning problems. Firstly consider a multi-criteria anomaly detection problem in which different types of information generate dissimilarities which must be combined to perform accurate detection. Secondly consider a multiple-query image retrieval problem where different query images are used to capture different semantic concepts that jointly form a query. On the surface, these seem like inherently distinct problems in machine learning. However they share a common characteristic: namely they both involve fusing disparate information in order to accomplish a task, i.e., anomaly detection or image retrieval. Specifically, the goals are to successfully aggregate different dissimilarity measures for anomaly detection and different query images for better retrieval results.

Consider two other machine learning problems: collaborative retrieval and partial correlation learning. For the collaborative retrieval problem, we combine behavioral and relational information simultaneously to increase the quality of retrieval results. In the partial correlation learning problem, we propose a model that fuses precision matrices corresponding to time series data at different time scales and offsets. These four problems all follow the same theme of this thesis which is to combine disparate information. We briefly introduce each of them and our contributions below.

1.2.1 Combining dissimilarities under different criteria

In Chapter II, we consider the problem of identifying patterns in a data set that exhibit anomalous behavior, often referred to as *anomaly detection*. In most anomaly detection algorithms, the dissimilarity between data samples is calculated by a single criterion, such as Euclidean distance. However, in many application domains there may not exist a single dissimilarity measure that captures all possible anomalous patterns. In such a case, multiple criteria can be defined, including non-metric criteria, and one can test for anomalies by scalarizing the multiple criteria using a non-negative linear combination of them. If the importance of the different criteria are not known in advance, as in many unsupervised anomaly detection applications, the anomaly detection algorithm may need to be executed multiple times with different choices of weights in the linear combination. In Chapter II, we propose a method for non-parametric anomaly detection using a novel *multi-criteria* dissimilarity measure, the Pareto depth. The proposed *Pareto depth analysis* (PDA) anomaly detection algorithm uses the concept of Pareto optimality to detect anomalies under multiple criteria without having to run an algorithm multiple times with different choices of weights. The proposed PDA approach scales *linearly* in the number of criteria and is provably better than using linear combinations of the criteria.

1.2.2 Retrieval for queries with different semantics

Following the same idea of the previous problem, we apply Pareto fronts to rank samples in a multiple-query information retrieval problem. We are particularly interested in an image retrieval problem. Most content-based image retrieval systems consider either one single query, or multiple queries that include the same object or represent the same semantic information. In Chapter III, we consider the content-based image retrieval problem for multiple query images corresponding to *different* image semantics. We propose a novel multiple-query information retrieval algorithm that combines the Pareto front method (PFM) with efficient manifold ranking (EMR). We show that our proposed algorithm outperforms state of the art multiple-query retrieval algorithms on real-world image databases. We attribute this performance improvement to concavity properties of the Pareto fronts, and also prove a theoretical result that characterizes the asymptotic concavity of the fronts and is related to theoretical results in Chapter II.

1.2.3 Combining relational and behavioral information

In Chapter IV we focus on socially-aided recommendation systems. Recommender systems have recently attracted significant interest and a number of studies have shown that social information can dramatically improve a system's predictions of user interests. Meanwhile, there are now many potential applications that involve aspects of both recommendation and information retrieval, and the task of *collaborative retrieval*—a combination of these two traditional problems—has recently been introduced. Successful collaborative retrieval requires overcoming severe data sparsity, making additional sources of information, such as social graphs, particularly valuable.

In Chapter IV, we propose a new model for collaborative retrieval, and show that our algorithm outperforms current state-of-the-art approaches by incorporating information from social networks. We also provide empirical analyses of the ways in which cultural interests propagate along a social graph using a real-world music dataset.

1.2.4 Fusion of multiscale time and spatial information

The last problem considered in this thesis is precision estimation for time series by fusing information at different time scales and offsets. Note that the elements of the precision matrix have an interpretation in terms of partial correlations and therefore different estimated precision matrices at different times can be viewed as a time-varying partial correlation network. Partial correlation networks are commonly applied to financial markets, where they can help to reveal graphical structure in historical equity prices. This structure can be used in a variety of ways to analyze and understand market dynamics. However, short-term correlation estimates tend to be noisy, while longer-term estimates are slow to respond to potentially important changes, and often smooth out interesting behavior.

In Chapter V, we aim to combine the desirable properties of these two extremes by performing joint estimation of precision matrices at multiple time scales and offsets, but encouraging them to share a sparsity pattern via the general fused graphical lasso. We develop the optimization machinery needed to fit our model to large data sets, and show empirically that this technique allows us to estimate partial correlation networks more robustly and accurately on both synthetic data and real-world time series.

1.3 Outline of the thesis

The thesis is organized as follows. Chapter 2 presents research on multi-criteria anomaly detection using Pareto depth analysis (PDA), which also motivates us to apply the Pareto method on another multiple-query problem. Some theoretical results of Pareto fronts are also presented in this chapter. In Chapter 3, we formulate a multiple-query retrieval problem in which queries might represent different semantic concepts and tackle this problem by using Pareto depths to rank items. Chapter 4 addresses another interesting retrieval problem which is a blend of recommendation, retrieval and social networks. In Chapter 5, we present an approach to learn multiple precision matrices for time series at different time scales and offsets.

1.4 List of publications

The publications that have come out of research presented in this dissertation are listed as follows.

Journals

[1] Ko-Jen Hsiao, Alex Kulesza, Alfred O. Hero. "Social Collaborative Retrieval". *Journal of Selected Topics in Signal Processing (J-STSP)*, Topic on "Signal Processing for Social Networks", 2014.

[2] Ko-Jen Hsiao, Kevin S. Xu, Jeff Calder, Alfred O. Hero. "Multi-criteria Anomaly Detection using Pareto Depth Analysis". *Transaction on Neural Networks and Learning Systems (TNNLS)*, Special Issue on "Learning in Non-(geo)metric Spaces", 2013, submitted.

[3] Ko-Jen Hsiao, Jeff Calder, Alfred O. Hero. "Pareto-depth for Multiple-query Image Retrieval". *Transaction on Image ProcessingI (TIP)*, 2014, submitted.

Articles in conference proceedings

[1] Ko-Jen Hsiao, Kevin S. Xu, Jeff Calder, Alfred O. Hero. "Multi-criteria Anomaly Detection using Pareto Depth Analysis". *Neural Information Processing Systems (NIPS)*, 2012.

[2] Ko-Jen Hsiao, Alex Kulesza, Alfred O. Hero."Social Collaborative Retrieval". *Web Search and Data Mining (WSDM)*, 2014.

[3] Ko-Jen Hsiao, Alex Kulesza, Alfred O. Hero. "Graphical Lasso Fusion across Multiple Time-sclaes". In preparation.

CHAPTER II

Multi-criteria Anomaly Detection using Pareto Depth Analysis

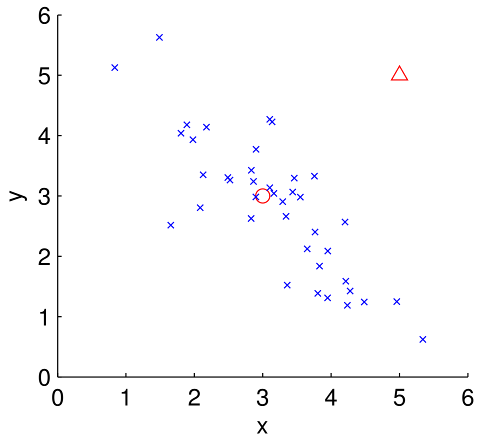
2.1 Introduction

The first machine learning problem considered here is to combine disparate information for anomaly detection. Identifying patterns of anomalous behavior in a data set, often referred to as *anomaly detection*, is an important problem with diverse applications including intrusion detection in computer networks, detection of credit card fraud, and medical informatics [28, 50]. Many methods for anomaly detection have been developed using both parametric and non-parametric approaches and typically involve the calculation of dissimilarities between data samples using a single criterion, such as Euclidean distance. However, in many application domains, such as those involving categorical data, it may not be possible or practical to represent data samples in a geometric space in order to compute Euclidean distances. Furthermore, *multiple* dissimilarity measures corresponding to different criteria may be required to detect certain types of anomalies. For example, consider the problem of detecting anomalous object trajectories in video sequences of different lengths. Multiple criteria, such as dissimilarities in object speeds or trajectory shapes, can be used to detect a greater range of anomalies than any single criterion. In order to perform anomaly detection using these multiple criteria, one could first combine the dissimilarities for each criterion using a non-negative linear combination then apply a (single-criterion) anomaly detection algorithm. However, in many applications, the importance of the different criteria are not known in advance. In an unsupervised anomaly detection setting, it is difficult to determine how much weight to assign to each criterion, so one may have to run the anomaly detection algorithm multiple times using multiple choices of weights selected by a grid search or similar method.

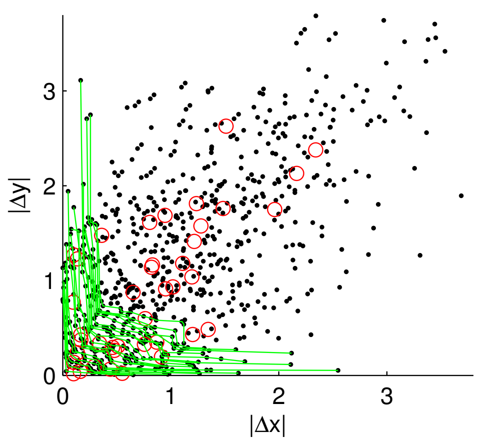
In this chapter we propose a novel non-parametric *multi-criteria* approach for unsupervised anomaly detection using *Pareto depth analysis* (PDA). PDA uses the concept of Pareto optimality, which is the typical method for defining optimality when there may be multiple conflicting criteria for comparing items. An item is said to be Pareto-optimal if there does not exist another item that is better or equal in *all* of the criteria. An item that is Pareto-optimal is optimal in the usual sense under some combination, not necessarily linear, of the criteria. Hence PDA is able to detect anomalies under multiple combinations of the criteria without explicitly forming these combinations.

The PDA approach involves creating *dyads* corresponding to dissimilarities between pairs of data samples under all of the criteria. Sets of Pareto-optimal dyads, called *Pareto fronts*, are then computed. The first Pareto front (depth one) is the set of non-dominated dyads. The second Pareto front (depth two) is obtained by removing these non-dominated dyads, i.e. peeling off the first front, and recomputing the first Pareto front of those remaining. This process continues until no dyads remain. In this way, each dyad is assigned to a Pareto front at some depth (see Fig. 2.1 for illustration).

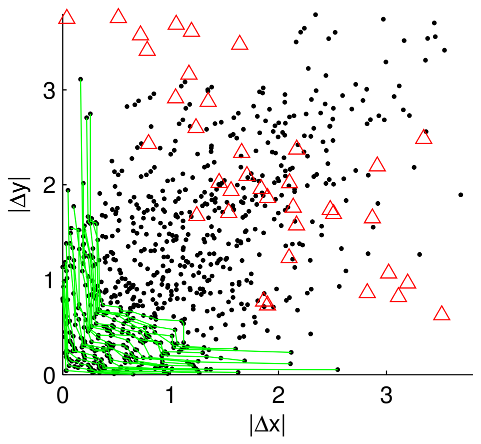
The *Pareto depth* of a dyad is a novel measure of dissimilarity between a pair of data samples under multiple criteria. In an unsupervised anomaly detection setting, the majority of the training samples are assumed to be nominal. Thus a nominal test sample would likely be similar to many training samples under some criteria, so most dyads for the nominal test sample would appear in shallow Pareto fronts. On the other hand, an anomalous test sample would likely be dissimilar to many training samples under many criteria, so most dyads for the anomalous test sample would be located in deep Pareto fronts. Thus computing the Pareto depths of the dyads corresponding to a test sample can discriminate between



(a)



(b)



(c)

Figure 2.1: (a) Illustrative example with 40 training samples (blue x's) and 2 test samples (red circle and triangle) in $\mathbb{R}^2$. (b) Dyads for the training samples (black dots) along with first 20 Pareto fronts (green lines) under two criteria: $|\Delta x|$ and $|\Delta y|$. The Pareto fronts induce a partial ordering on the set of dyads. Dyads associated with the test sample marked by the red circle concentrate around *shallow fronts* (near the lower left of the figure). (c) Dyads associated with the test sample marked by the red triangle concentrate around *deep fronts*.

nominal and anomalous samples.

Under the assumption that the multi-criteria dyads can be modeled as realizations from a K-dimensional density, we provide a mathematical analysis of properties of the first Pareto front relevant to anomaly detection. In particular, in the Scalarization Gap Theorem we prove upper and lower bounds on the degree to which the Pareto fronts are non-convex.

For any algorithm using non-negative linear combinations of criteria, non-convexities in the Pareto fronts contribute to an artificially inflated anomaly score, resulting in an increased false positive rate. Thus our analysis shows in a precise sense that PDA can outperform any algorithm that uses a non-negative linear combination of the criteria. Furthermore, this theoretical prediction is experimentally validated by comparing PDA to several state-of-the-art anomaly detection algorithms in two experiments involving both synthetic and real data sets. Finally, we note that the proposed PDA approach scales *linearly* in the number of criteria, which is a significant improvement compared to selecting multiple linear combinations via a grid search, which scales exponentially.

The rest of Chapter II is organized as follows. We discuss related work in Section 2.2. In Section 2.3 we provide an introduction to Pareto fronts and present a theoretical analysis of the properties of the first Pareto front. Section 2.4 relates Pareto fronts to the multi-criteria anomaly detection problem, which leads to the PDA anomaly detection algorithm. Finally we present three experiments in Section 2.5 to provide experimental support for our theoretical results and evaluate the performance of PDA for anomaly detection.

2.2 Related work

2.2.1 Multi-criteria methods for machine learning

Several machine learning methods utilizing Pareto optimality have previously been proposed; an overview can be found in [63]. These methods typically formulate supervised machine learning problems as multi-objective optimization problems over a potentially infinite set of candidate items where finding even the first Pareto front is quite difficult, often requiring multi-objective evolutionary algorithms. These methods differ from our use of Pareto optimality because we consider Pareto fronts created from a finite set of items, so we do not need to employ sophisticated algorithms in order to find these fronts. Rather, we utilize Pareto fronts to form a statistical criterion for unsupervised anomaly detection.

Finding the Pareto front of a finite set of items has also been referred to in the literature as the skyline query [15, 103] or the maximal vector problem [70]. Since research on skyline queries is more related to our multiple-query retrieval problem in Chapter III, we delay the discussion of skyline queries to next chapter (Section 3.2).

Hero and Fleury [48] introduced a method for gene ranking using multiple Pareto fronts that is related to our approach. The method ranks genes, in order of interest to a biologist, by creating Pareto fronts on the data samples, i.e. the genes. In this work, we consider Pareto fronts of *dyads*, which correspond to dissimilarities between *pairs* of data samples under multiple criteria rather than the samples themselves, and use the distribution of dyads in Pareto fronts to perform multi-criteria anomaly detection rather than gene ranking.

Another related area is multi-view learning [12, 98], which involves learning from data represented by multiple sets of features, commonly referred to as "views". In such a case, training in one view is assumed to help to improve learning in another view. The problem of view disagreement, where samples take on different classes in different views, has recently been investigated [30]. The views are similar to criteria in our problem setting. However, in our setting, different criteria may be orthogonal and could even give contradictory information; hence there may be severe view disagreement. Thus training in one view could actually worsen performance in another view, so the problem we consider differs from multi-view learning. A similar area is that of multiple kernel learning [45], which is typically applied to supervised learning problems, unlike the unsupervised anomaly detection setting we consider.

2.2.2 Anomaly detection

Many methods for anomaly detection have previously been proposed. Hodge and Austin [50] and Chandola et al. [28] both provide extensive surveys of different anomaly detection methods and applications. Nearest neighbor-based methods are related to the proposed PDA approach. Byers and Raftery [21] proposed to use the distance between a sample and its kth-nearest neighbor as the anomaly score for the sample; similarly, Angiulli and Pizzuti [2] and Eskin et al. [38] proposed to the use the sum of the distances between a sample and its k nearest neighbors. Breunig et al. [18] used an anomaly score based on the local density of the k nearest neighbors of a sample. Hero [47] and Sricharan and Hero [100] introduced non-parametric adaptive anomaly detection methods using geometric entropy minimization, based on random k-point minimal spanning trees and bipartite k-nearest neighbor (k-NN) graphs, respectively. Zhao and Saligrama [114] proposed an anomaly detection algorithm k-LPE using local p-value estimation (LPE) based on a k-NN graph. These k-NN anomaly detection schemes only depend on the data through the pairs of data points (dyads) that define the edges in the k-NN graphs. All of the aforementioned anomaly detection methods are designed for a single distance-based criterion, unlike the PDA anomaly detection algorithm that we propose in this chapter, which accommodates non-metric dissimilarities corresponding to multiple criteria.

2.3 Pareto depth analysis

Multi-criteria optimization and Pareto optimality have been studied in many application areas in computer science, economics and the social sciences. An overview can be found in [37]. The proposed PDA method in this Chapter utilizes the notion of Pareto optimality, which we now introduce.

Consider the following problem: given n items, denoted by the set $\mathcal{S}$, and K criteria for evaluating each item, denoted by functions $f_1, \ldots, f_K$, select $x \in \mathcal{S}$ that minimizes $[f_1(x), \ldots, f_K(x)]$. In most settings, it is not possible to find a single item x which simultaneously minimizes $f_i(x)$ for all $i \in \{1, \ldots, K\}$. Many approaches to the multi-criteria optimization problem reduce to combining all of the criteria into a single criterion, a process often referred to as *scalarization* [37]. A common approach is to use a non-negative linear combination of the f_i's and find the item that minimizes the linear combination. Different choices of weights in the linear combination yield different minimizers. In this case, one would need to identify a set of optimal solutions corresponding to different weights using, for example, a grid search.

A more robust and powerful approach involves identifying the set of *Pareto-optimal* items. An item x is said to *strictly dominate* another item x^* if x is no greater than x^* in each criterion and x is less than x^* in at least one criterion. This relation can be written as $x \succ x^*$ if $f_i(x) \leq f_i(x^*)$ for *each* i and $f_i(x) < f_i(x^*)$ for *some* i. The set of Pareto-optimal items, called the *Pareto front*, is the set of items in $\mathcal{S}$ that are not strictly dominated by another item in $\mathcal{S}$. It contains all of the minimizers that are found using non-negative linear combinations, but also includes other items that cannot be found by linear combinations. Denote the Pareto front by $\mathcal{F}_1$, which we call the first Pareto front. The second Pareto front can be constructed by finding items that are not strictly dominated by any of the remaining items, which are members of the set $\mathcal{S} \setminus \mathcal{F}_1$. More generally, define the ith Pareto front by

$$\mathcal{F}_i = \text{Pareto front of the set } \mathcal{S} \setminus \left(\bigcup_{j=1}^{i-1} \mathcal{F}_j \right).$$

For convenience, we say that a Pareto front $\mathcal{F}_i$ is *deeper* than $\mathcal{F}_j$ if $i > j$.

2.3.1 Mathematical properties of Pareto fronts

The distribution of the number of points on the first Pareto front was first studied by Barndorff-Nielsen and Sobel [7]. The problem has garnered much attention since. Bai et al. [5] and Hwang and Tsai[56] provide good surveys of recent results. We will be concerned here with properties of the first Pareto front that are relevant to the PDA anomaly detection algorithm and have not yet been considered in the literature.

Let $Y_1, \ldots, Y_n$ be independent and identically distributed (*i.i.d.*) on $\mathbb{R}^d$ with density function $f : \mathbb{R}^d \to \mathbb{R}$, and let $\mathcal{F}^n$ denote the first Pareto front of $Y_1, \ldots, Y_n$. In the general multi-criteria optimization framework, the points $Y_1, \ldots, Y_n$ are the images in $\mathbb{R}^d$ of n feasible solutions to some optimization problem under a vector of objective functions of length d. In the context of multi-criteria anomaly detection, each point Y_i is a dyad corresponding to dissimilarities between two data samples under multiple criteria, and $d = K$ is the number of criteria. A common approach in multi-objective optimization is linear scalarization [37], which constructs a new single criterion as a non-negative linear combination of the d criteria. It is well-known, and easy to see, that linear scalarization will only identify Pareto-optimal points on the boundary of the convex hull of

$$\mathcal{G}^n := \bigcup_{x\in\mathcal{F}^n} (x + \mathbb{R}^d_+),$$

where $\mathbb{R}^d_+ = \{x \in \mathbb{R}^d \mid \forall i,\ x_i \geq 0\}$. Although this is a common motivation for Pareto optimization methods, there are, to the best of our knowledge, no results in the literature regarding how many points on the Pareto front are missed by scalarization. We present such a result in this section, namely the Scalarization Gap Theorem.

We define

$$\mathcal{L}^n = \bigcup_{\alpha\in\mathbb{R}^d_+} \operatorname*{argmin}_{x\in S_n} \left\{ \sum_{i=1}^d \alpha_i x_i \right\}, \quad S_n = \{Y_1, \dots, Y_n\}.$$

The subset $\mathcal{L}^n \subset \mathcal{F}^n$ contains all Pareto-optimal points that can be obtained by some selection of of non-negative weights for linear scalarization. Let K_n denote the cardinality of $\mathcal{F}^n$, and let L_n denote the cardinality of $\mathcal{L}^n$. When $Y_1, \dots, Y_n$ are uniformly distributed on the unit hypercube, Barndorff-Nielsen and Sobel [7] showed that

$$E(K_n) = \frac{n}{(d-1)!} \int_0^1 (1-x)^{n-1} (-\log x)^{d-1}\, dx,$$

from which one can easily obtain the asymptotics

$$E(K_n) = \frac{(\log n)^{d-1}}{(d-1)!} + O((\log n)^{d-2}).$$

Many more recent works have studied the variance of K_n and have proven central limit theorems for K_n. All of these works assume that $Y_1, \dots, Y_n$ are uniformly distributed on $[0,1]^d$. For a summary, see *Bai et al.* [5] and *Hwang and Tsai* [56]. Other works have studied K_n for more general distributions on domains that have smooth "non-horizontal" boundaries near the Pareto front [8] and for multivariate normal distributions on $\mathbb{R}^d$ [57]. The "non-horizontal" condition excludes hypercubes. To the best of our knowledge there are no results on the asymptotics of K_n for non-uniformly distributed points on the unit hypercube. This is of great importance as it is impractical in multi-criteria optimization (or anomaly detection) to assume that the coordinates of the points are independent. Typically the coordinates of $Y_i \in \mathbb{R}^d$ are the images of the *same* feasible solution under several *different* criteria, which will not in general be independent.

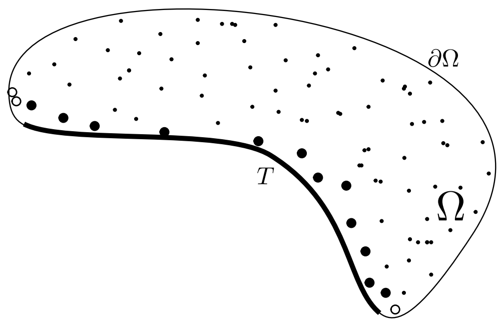
Here we develop results on the size of the gap between the number of items L_n discoverable by scalarization compared to the number of items K_n discovered on the Pareto front. The larger the gap, the more suboptimal scalarization is relative to Pareto depth analysis. Since $x \in \mathcal{L}^n$ if and only if x is on the boundary of the convex hull of $\mathcal{G}^n$, the size of $\mathcal{L}^n$ is related to the convexity (or lack thereof) of the Pareto front. There are several ways in which the Pareto front can be *non-convex*. First, suppose that $Y_1, \ldots, Y_n$ are distributed on some domain $\Omega \subset \mathbb{R}^d$ with a continuous density function $f : \overline{\Omega} \to \mathbb{R}$ that is strictly positive on $\overline{\Omega}$. Let $T \subset \partial\Omega$ be a portion of the boundary of Ω such that

$$\inf_{z \in T} \min(\nu_1(z), \ldots, \nu_d(z)) > 0,$$

and

$$\{y \in \overline{\Omega} \,:\, \forall i \;\; y_i \leq x_i\} = \{x\}, \;\; \text{for all} \;\; x \in T,$$

where $\nu : \partial\Omega \to \mathbb{R}^d$ is the unit inward normal to $\partial\Omega$. The conditions on T guarantee that a portion of the first Pareto front will concentrate near T as $n \to \infty$. If we suppose that T is contained in the interior of the convex hull of Ω, then points on the portion of the Pareto front near T cannot be obtained by linear scalarization, as they are on a non-convex portion of the front. Such non-convexities are a direct result of the geometry of the domain Ω and are depicted in Fig. 2.2a. In a preliminary version of this work, we studied the expectation of the number of points on the Pareto front within a neighborhood of T (Theorem 1 in



(a)

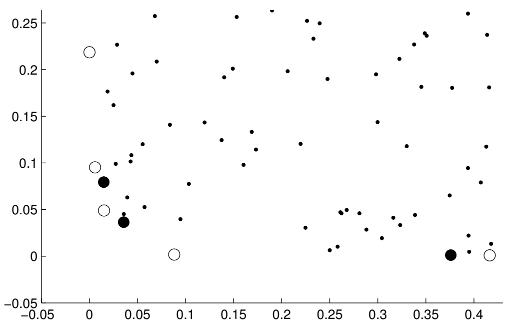

(b)

Figure 2.2: (a) Non-convexities in the Pareto front induced by the geometry of the domain Ω. (b) Non-convexities due to randomness in the points. In each case, the larger points are Pareto-optimal, and the large black points *cannot* be obtained by scalarization.

[53]). As a result, we showed that

$$E(K_n - L_n) \geq \gamma n^{\frac{d-1}{d}} + O(n^{\frac{d-2}{d}}),$$

as $n \to \infty$, where γ is a positive constant given by

$$\gamma = \frac{1}{d}(d!)^{\frac{1}{d}}\Gamma\left(\frac{1}{d}\right)\int\limits_T f(z)^{\frac{d-1}{d}}(\nu_1(z)\cdots\nu_d(z))^{\frac{1}{d}}dz.$$

It has recently come to our attention that a stronger result was proven previously by Baryshnikov and Yukich [8] in an unpublished manuscript.

In practice, it is unlikely that one would have enough information about f or Ω to compute the constant γ. In this work, we instead study a second type of non-convexity in the Pareto front. These non-convexities are strictly due to randomness in the positions of the points and occur even when the domain Ω is convex (see Fig. 2.2b for a depiction of such non-convexities). In the following, we assume that $Y_1, \ldots, Y_n$ are *i.i.d.* on the unit hypercube $[0,1]^d$ with a bounded density function $f : [0,1]^d \to \mathbb{R}^d$ which is continuous at the origin and strictly positive on $[0,1]^d$. Under these assumptions on f, it turns out that the asymptotics of $E(K_n)$ and $E(L_n)$ are independent of f. Hence our results are applicable to a wide range of problems without the need to know detailed information about the density

f.

Our first result is

Theorem II.1. *Assume* $f : [0,1]^d \to [\sigma, M]$ *is continuous at the origin, and* $0 < \sigma < M < \infty$. *Then*

$$E(K_n) \sim c_{n,d} := \frac{(\log n)^{d-1}}{(d-1)!} \text{ as } n \to \infty.$$

We give the proof of Theorem II.1 after some preliminary results. Our second result concerns $E(L_n)$. We are not able to get the exact asymptotics of $E(L_n)$, so we provide upper and lower asymptotic bounds.

Theorem II.2. *Assume* $f : [0,1]^d \to [\sigma, M]$ *is continuous at the origin, and* $0 < \sigma < M < \infty$. *Then*

$$\tfrac{d!}{d^d} c_{n,d} + o((\log n)^{d-1}) \leq E(L_n) \leq \tfrac{3d-1}{4d-2} c_{n,d} + o((\log n)^{d-1})$$

as $n \to \infty$.

The proof of Theorem II.2 is also given after some preliminary results. Theorem II.2 provides a significant generalization of a previous result (Theorem 2 in [53]) that holds only for uniform distributions in $d = 2$. Combining Theorems II.1 and II.2, we arrive at our main result:

Theorem II.3 (Scalarization Gap Theorem)**.** *Assume* $f : [0,1]^d \to [\sigma, M]$ *is continuous at the origin, and* $0 < \sigma < M < \infty$. *Then*

$$\begin{aligned} &\tfrac{d-1}{4d-2} c_{n,d} + o((\log n)^{d-1}) \\ &\qquad \leq E(K_n - L_n) \leq \left(1 - \tfrac{d!}{d^d}\right) c_{n,d} + o((\log n)^{d-1}), \end{aligned}$$

as $n \to \infty$.

The Scalarization Gap Theorem shows that the fraction of Pareto-optimal points that *cannot* be obtained by linear scalarization is at least $\frac{d-1}{4d-2}$. We provide experimental evidence supporting these bounds in Section 2.5.1.

2.3.2 Proofs

We first present a general result on the expectation of K_n. Let $F : [0,1]^d \to \mathbb{R}$ denote the cumulative distribution function of f, defined by

$$F(x) = \int_0^{x_1} \cdots \int_0^{x_d} f(y_1, \ldots, y_d)\, dy_1 \cdots dy_d.$$

Proposition II.4. *For any $n \geq 1$ we have*

$$E(K_n) = n \int_{[0,1]^d} f(x)\,(1 - F(x))^{n-1}\, dx.$$

Proof. Let E_i be the event that $Y_i \in \mathcal{F}^n$ and let χ_{E_i} be indicator random variables for E_i. Then

$$E(K_n) = E\left(\sum_{i=1}^n \chi_{E_i}\right) = \sum_{i=1}^n P(E_i) = nP(E_1).$$

Conditioning on Y_1 we obtain

$$E(K_n) = n \int_{[0,1]^d} f(x) P(E_1 \,|\, Y_1 = x) dx.$$

Noting that $P(E_1 \,|\, Y_1 = x) = (1 - F(x))^{n-1}$ completes the proof. □

The following simple proposition is essential in the proofs of Theorem II.1 and II.2:

Proposition II.5. *Let $0 < \delta \leq 1$ and $a > 0$. For $a \leq \delta^{-d}$ we have*

$$n \int_{[0,\delta]^d} (1 - a x_1 \cdots x_d)^{n-1}\, dx = \frac{c_{n,d}}{a} + O((\log n)^{d-2}), \tag{2.1}$$

and for $a \leq 1$ we have

$$n \int_{[0,1]^d \setminus [0,\delta]^d} (1 - a x_1 \cdots x_d)^{n-1}\, dx = O((\log n)^{d-2}). \tag{2.2}$$

Proof. We will give a sketch of the proof as similar results are well-known [5]. Assume $\delta = 1$ and let Q_n denote the quantity on the left hand side of (2.1). Making the change of variables $y_i = x_i$ for $i = 1, \ldots, d-1$ and $t = x_1 \cdots x_d$, we see that

$$Q_n = n \int_0^1 \int_t^1 \int_{\frac{t}{y_{d-1}}}^1 \cdots \int_{\frac{t}{y_2 \cdots y_{d-1}}}^1 \frac{(1-at)^{n-1}}{y_1 \cdots y_{d-1}} \, dy_1 \cdots dy_{d-1} dt.$$

By computing the inner $d-1$ integrals we find that

$$Q_n = \frac{n}{(d-1)!} \int_0^1 (-\log t)^{d-1} (1-at)^{n-1} dt,$$

from which the asymptotics (2.1) can be easily obtained by another change of variables $u = nat$, provided $a \leq 1$. For $0 < \delta < 1$, we make the change of variables $y = x/\delta$ to find that

$$Q_n = \delta^d n \int_{[0,1]^d} \left(1 - a\delta^d y_1 \cdots y_d\right)^{n-1} dy.$$

We can now apply the above result provided $a\delta^d \leq 1$. The asymptotics in (2.1) show that

$$n \int_{[0,1]^d} \left(1 - a x_1 \cdots x_d\right)^{n-1} dx$$
$$= n \int_{[0,\delta]^d} \left(1 - a x_1 \cdots x_d\right)^{n-1} dx + O((\log n)^{d-2}),$$

when $a \leq 1$, which gives the second result (2.2). $\square$

We now give the proof of Theorem II.1.

Proof. Let $\varepsilon > 0$ and choose $\delta > 0$ such that

$$f(0) - \varepsilon \le f(x) \le f(0) + \varepsilon \quad \text{for any } x \in [0, \delta]^d,$$

and $f(0) < \delta^{-d}$. Since $\sigma \le f \le M$, we have that $F(x) \ge \sigma x_1 \cdots x_d$ for all $x \in [0,1]^d$. Since f is a probability density on $[0,1]^d$, we must have $\sigma \le 1$. Since $\sigma > 0$, we can apply Proposition II.5 to find that

$$\begin{aligned} n \int\limits_{[0,1]^d \setminus [0,\delta]^d} & f(x)(1 - F(x))^{n-1}\, dx \\ & \le Mn \int\limits_{[0,1]^d \setminus [0,\delta]^d} (1 - \sigma x_1 \cdots x_d)^{n-1}\, dx \\ & = O((\log n)^{d-2}). \end{aligned} \tag{2.3}$$

For $x \in [0, \delta]^d$, we have

$$(f(0) - \varepsilon) x_1 \cdots x_d \le F(x) \le (f(0) + \varepsilon) x_1 \cdots x_d.$$

Combining this with Proposition II.5, and the fact that $f(0) - \varepsilon < \delta^{-d}$ we have

$$\begin{aligned} n \int\limits_{[0,\delta]^d} & f(x)(1 - F(x))^{n-1}\, dx \\ & \le (f(0) + \varepsilon) n \int\limits_{[0,\delta]^d} (1 - (f(0) - \varepsilon) x_1 \cdots x_d)^{n-1}\, dx \\ & = \frac{f(0) + \varepsilon}{f(0) - \varepsilon} \cdot c_{n,d} + O((\log n)^{d-2}). \end{aligned} \tag{2.4}$$

Combining (2.3) and (2.4) with Proposition (II.4) we have

$$E(K_n) \le \frac{f(0) + \varepsilon}{f(0) - \varepsilon} \cdot c_{n,d} + O((\log n)^{d-2}).$$

It follows that

$$\limsup_{n\to\infty} c_{n,d}^{-1} E(K_n) \le \frac{f(0)+\varepsilon}{f(0)-\varepsilon}.$$

By a similar argument we can obtain

$$\liminf_{n\to\infty} c_{n,d}^{-1} E(K_n) \ge \frac{f(0)-\varepsilon}{f(0)+\varepsilon}.$$

Since $\varepsilon > 0$ was arbitrary, we see that

$$\lim_{n\to\infty} c_{n,d}^{-1} E(K_n) = 1. \qquad \square$$

The proof of Theorem II.2 is split into the following two lemmas. It is well-known, and easy to see, that $x \in \mathcal{L}^n$ if and only if $x \in \mathcal{F}^n$ and x is on the boundary of the convex hull of $\mathcal{G}^n$ [37]. This fact will be used in the proof of Lemma II.6.

Lemma II.6. *Assume $f : [0,1]^d \to \mathbb{R}$ is continuous at the origin and there exists $\sigma, M > 0$ such that $\sigma \le f \le M$. Then*

$$E(L_n) \le \frac{3d-1}{4d-2} \cdot c_{n,d} + o((\log n)^{d-1}) \text{ as } n \to \infty.$$

Proof. Let $\varepsilon > 0$ and choose $0 < \delta < \frac{1}{2}$ so that

$$f(0) - \varepsilon \le f(x) \le f(0) + \varepsilon \text{ for any } x \in [0, 2\delta]^d, \tag{2.5}$$

and $3f(0) \le \delta^{-d}$. As in the proof of Proposition II.4 we have $E(L_n) = nP(Y_1 \in \mathcal{L}^n)$, so conditioning on Y_1 we have

$$E(L_n) = n \int_{[0,1]^d} f(x) P(Y_1 \in \mathcal{L}^n \,|\, Y_1 = x)\, dx.$$

As in the proof of Theorem II.1, we have

$$n \int\limits_{[0,1]^d\setminus[0,\delta]^d} f(x)P(Y_1 \in \mathcal{L}^n \mid Y_1 = x)\,dx$$
$$\leq n \int\limits_{[0,1]^d\setminus[0,\delta]^d} f(x)(1-F(x))^{n-1}\,dx$$
$$= O((\log n)^{d-2}),$$

and hence

$$E(L_n) = n \int\limits_{[0,\delta]^d} f(x)P(Y_1 \in \mathcal{L}^n \mid Y_1 = x)\,dx + O((\log n)^{d-2}). \tag{2.6}$$

Fix $x \in [0,\delta]^d$ and define $A = \{y \in [0,1]^d \ : \ \forall i,\ y_i \leq x_i\}$ and

$$B_i = \left\{ y \in [0,1]^d \ : \ \forall j \neq i,\ y_j < x_j \text{ and } x_i < y_i < 2x_i - \frac{x_i}{x_j}y_j \right\},$$

for $i = 1, \ldots, d$, and note that $B_i \subset [0, 2\delta]^d$ for all i. See Fig. 2.3 for an illustration of these sets for $d = 3$.

We claim that if at least two of $B_1, \ldots, B_d$ contain samples from $Y_2, \ldots, Y_n$, and $Y_1 = x$, then $Y_1 \notin \mathcal{L}^n$. To see this, assume without loss of generality that B_1 and B_2 are nonempty and let $y \in B_1$ and $z \in B_2$. Set

$$\tilde{y} = \left(y_1, 2x_2 - \frac{x_2}{x_1}y_1, x_3, \ldots, x_d\right)$$

$$\tilde{z} = \left(2x_1 - \frac{x_1}{x_2}z_2, z_2, x_3, \ldots, x_d\right).$$

By the definitions of B_1 and B_2 we see that $y_i \leq \tilde{y}_i$ and $z_i \leq \tilde{z}_i$ for all i, hence $\tilde{y}, \tilde{z} \in \mathcal{G}_n$.

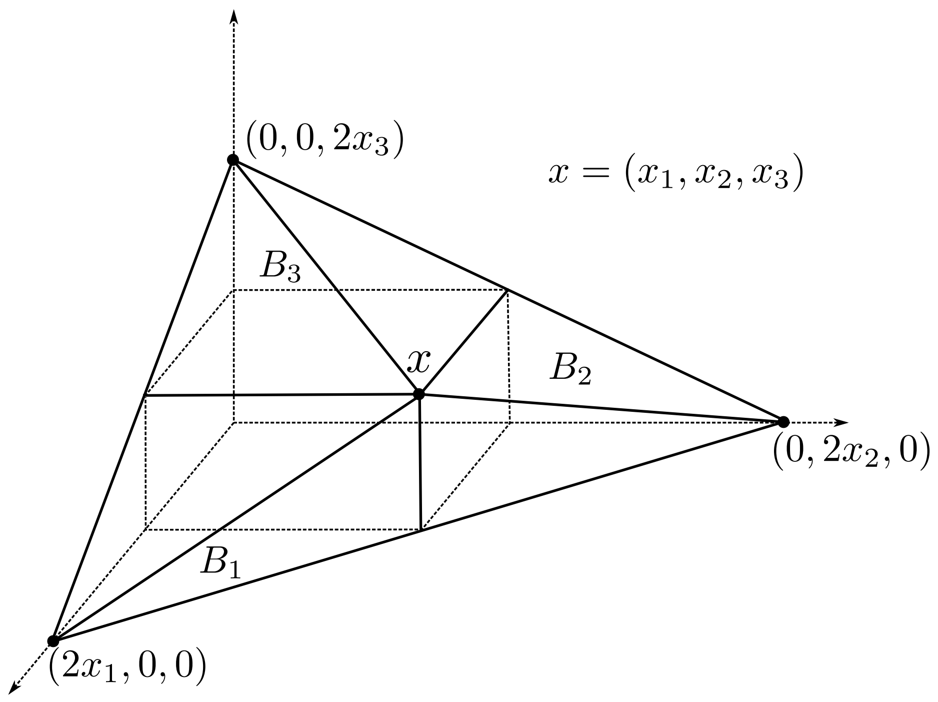


Figure 2.3: Depiction of the sets B_1, B_2 and B_3 from the proof of Lemma II.6 in the case that $d = 3$.

Let $\alpha \in (0,1)$ such that

$$\alpha y_1 + (1-\alpha)\left(2x_1 - \frac{x_1}{x_2} z_2\right) = x_1.$$

A short calculation shows that $x = \alpha\tilde{y} + (1-\alpha)\tilde{z}$ which implies that x is in the interior of the convex hull of $\mathcal{G}_n$, proving the claim.

Let E denote the event that at most one of $B_1, \ldots, B_d$ contains a sample from $Y_2, \ldots, Y_n$, and let F denote the event that A contains no samples from $Y_2, \ldots, Y_n$. Then by the observation above we have

$$P(Y_1 \in \mathcal{L}^n \,|\, Y_1 = x) \le P(E \cap F \,|\, Y_1 = x) = P(E \cap F). \tag{2.7}$$

For $i = 1, \ldots, d$, let E_i denote the event that B_i contains no samples from $Y_2, \ldots, Y_n$. It is not hard to see that

$$E = \bigcup_{i=1}^{d} \left(\bigcap_{j \neq i} E_j \setminus \bigcap_{j} E_j \right) \bigcup \left(\bigcap_{j} E_j \right).$$

Furthermore, the events in the unions above are mutually exclusive (disjoint) and $\cap_j E_j \subset \cap_{j\neq i} E_j$ for $i = 1, \ldots, d$. It follows that

$$\begin{aligned} &P(E \cap F) \\ &= \sum_{i=1}^{d} \left(P\left(\cap_{j\neq i} E_j \cap F\right) - P\left(\cap_j E_j \cap F\right)\right) + P\left(\cap_j E_j \cap F\right) \\ &= \sum_{i=1}^{d} P\left(\cap_{j\neq i} E_j \cap F\right) - (d-1) P\left(\cap_j E_j \cap F\right) \\ &= \sum_{i=1}^{d} \left(1 - F(x) - \int\limits_{\cup_{j\neq i} B_j} f(y)\,dy\right)^{n-1} \\ &\qquad\qquad - (d-1)\left(1 - F(x) - \int\limits_{\cup_j B_j} f(y)\,dy\right)^{n-1}. \end{aligned} \tag{2.8}$$

A simple computation shows that $|B_j| = \frac{1}{d} x_1 \cdots x_d$ for $j = 1, \ldots, d$. Since $A, B_i \subset [0, 2\delta]^d$, we have by (2.5) that

$$(f(0) - \varepsilon) x_1 \cdots x_d \leq F(x) \leq (f(0) + \varepsilon) x_1 \cdots x_d,$$

and

$$\frac{1}{d}(f(0) - \varepsilon) x_1 \cdots x_d \leq \int\limits_{B_j} f(y)\,dy \leq \frac{1}{d}(f(0) + \varepsilon) x_1 \cdots x_d.$$

Inserting these into (2.8) and combining with (2.7) we have

$$\begin{aligned}
&P(Y_1 \in \mathcal{L}^n \,|\, Y_1 = x) \\
&\qquad \le d\left(1 - \frac{2d-1}{d}(f(0)-\varepsilon)x_1\cdots x_d\right)^{n-1} \\
&\qquad\qquad - (d-1)\left(1 - 2(f(0)+\varepsilon)x_1\cdots x_d\right)^{n-1}.
\end{aligned}$$

We can now insert this into (2.6) and apply Proposition II.5 (since $3f(0) \le \delta^{-d}$) to obtain

$$\begin{aligned}
E(L_n) \le \left(\frac{d^2}{2d-1}\frac{f(0)+\varepsilon}{f(0)-\varepsilon} - \frac{d-1}{2}\frac{f(0)-\varepsilon}{f(0)+\varepsilon}\right) c_{n,d} \\
+ O((\log n)^{d-2}).
\end{aligned}$$

Since $\varepsilon > 0$ was arbitrary, we find that

$$\limsup_{n\to\infty}\ c_{n,d}^{-1}E(L_n) \le \left(\frac{d^2}{2d-1} - \frac{d-1}{2}\right) = \frac{3d-1}{4d-2}. \qquad \square$$

Lemma II.7. *Assume $f : [0,1]^d \to \mathbb{R}$ is continuous and there exists $\sigma, M > 0$ such that $\sigma \le f \le M$. Then*

$$E(L_n) \ge \frac{d!}{d^d} \cdot c_{n,d} + o((\log n)^{d-1}) \ \text{ as } \ n \to \infty.$$

Proof. Let $\varepsilon > 0$ and choose $0 < \delta < 1/d$ so that

$$f(0) - \varepsilon \le f(x) \le f(0) + \varepsilon \ \text{ for } \ x \in [0, d\delta]^d, \tag{2.9}$$

and

$$\frac{d^d}{d!}(f(0)+\varepsilon) \le \delta^{-d}. \tag{2.10}$$

As in the proof of Lemma II.6 we have

$$E(L_n) = n \int\limits_{[0,\delta]^d} f(x) P(Y_1 \in \mathcal{L}^n \,|\, Y_1 = x)\, dx \\ + O((\log n)^{d-2}). \tag{2.11}$$

Fix $x \in (0,\delta)^d$, set $\nu = \left(\frac{1}{x_1}, \dots, \frac{1}{x_d}\right)$ and

$$A = \left\{ y \in [0,1]^d \;\;|\;\; y \cdot \nu \le x \cdot \nu \right\}.$$

Note that A is a simplex with an orthogonal corner at the origin and side lengths $d{\cdot}x_1, \dots, d{\cdot} x_d$. A simple computation shows that $|A| = \frac{d^d}{d!} x_1 \cdots x_d$. By (2.9) we have

$$\int\limits_A f(y)\, dy \le (f(0) + \varepsilon)|A| = \frac{d^d}{d!}(f(0)+\varepsilon) x_1 \cdots x_d.$$

It is easy to see that if A is empty and $Y_1 = x$ then $Y_1 \in \mathcal{L}^n$, hence

$$\begin{aligned} P(Y_1 \in \mathcal{L}^n \,|\, Y_1 = x) &\ge \left(1 - \int\limits_A f(y)\, dy\right)^{n-1} \\ &\ge \left(1 - \frac{d^d}{d!}(f(0)+\varepsilon) x_1 \cdots x_d\right)^{n-1}. \end{aligned}$$

Inserting this into (2.11) and noting (2.10), we can apply Proposition II.5 to obtain

$$E(L_n) \ge \frac{d!}{d^d} \frac{f(0) - \varepsilon}{f(0) + \varepsilon} c_{n,d} + O((\log n)^{d-2}),$$

and hence

$$\limsup_{n \to \infty} \; c_{n,d}^{-1} E(L_n) \ge \frac{d!}{d^d}. \qquad \square$$

2.4 Multi-criteria anomaly detection

We now formally define the multi-criteria anomaly detection problem. Assume that a training set $\mathcal{X}_N = \{X_1, \ldots, X_N\}$ of unlabeled data samples is available. Given a test sample X, the objective of anomaly detection is to declare X to be an anomaly if X is significantly different from samples in $\mathcal{X}_N$. Suppose that $K > 1$ different evaluation criteria are given. Each criterion is associated with a measure for computing dissimilarities. Denote the dissimilarity between X_i and X_j computed using the dissimilarity measure corresponding to the lth criterion by $d_l(i, j)$. Note that $d_l(i, j)$ need not be geometric; in particular it is not necessary that $d_l(i, j)$ be a distance function over the sample space or that $d_l(i, j)$ satisfy the triangle inequality.

We define a *dyad* between a pair of samples $i \in \{1, \ldots, N\}$ and $j \in \{1, \ldots, N\} \setminus i$ by a vector $D_{ij} = [d_1(i, j), \ldots, d_K(i, j)]^T \in \mathbb{R}_+^K$. There are in total $\binom{N}{2}$ different dyads for the training set. For convenience, denote the set of all dyads by $\mathcal{D}$. By the definition of strict dominance in Section 2.3, a dyad D_{ij} strictly dominates another dyad $D_{i^*j^*}$ if $d_l(i, j) \leq d_l(i^*, j^*)$ for all $l \in \{1, \ldots, K\}$ and $d_l(i, j) < d_l(i^*, j^*)$ for some l. The first Pareto front $\mathcal{F}_1$ corresponds to the set of dyads from $\mathcal{D}$ that are not strictly dominated by any other dyads from $\mathcal{D}$. The second Pareto front $\mathcal{F}_2$ corresponds to the set of dyads from $\mathcal{D} \setminus \mathcal{F}_1$ that are not strictly dominated by any other dyads from $\mathcal{D} \setminus \mathcal{F}_1$, and so on, as defined in Section 2.3. Recall that we refer to $\mathcal{F}_i$ as a *deeper* front than $\mathcal{F}_j$ if $i > j$.

2.4.1 Pareto fronts on dyads

For each sample X_n, there are $N - 1$ dyads corresponding to its connections with the other $N - 1$ samples. Define the set of $N - 1$ dyads associated with X_n by $\mathcal{D}^n$. If most dyads in $\mathcal{D}^n$ are located at shallow Pareto fronts, then the dissimilarities between X_n and the other $N - 1$ samples are small under *some* combination of the criteria. Thus, X_n is likely to be a nominal sample. This is the basic idea of the proposed multi-criteria anomaly detection method using PDA.

We construct Pareto fronts $\mathcal{F}_1, \ldots, \mathcal{F}_M$ of the dyads from the training set, where the total number of fronts M is the required number of fronts such that each dyad is a member of a front. When a test sample X is obtained, we create new dyads corresponding to connections between X and training samples, as illustrated in Fig. 2.1. Similar to many other anomaly detection methods, we connect each test sample to its k nearest neighbors. k could be different for each criterion, so we denote k_l as the choice of k for criterion l. We create $s = \sum_{l=1}^{K} k_l$ new dyads, which we denote by the set $\mathcal{D}^{\text{new}} = \{D_1^{\text{new}}, D_2^{\text{new}}, \ldots, D_s^{\text{new}}\}$, corresponding to the connections between X and the union of the k_l nearest neighbors in each criterion l. In other words, we create a dyad between X and X_i if X_i is among the k_l nearest neighbors[1] of X in any criterion l. We say that D_i^{new} is *below* a front $\mathcal{F}_j$ if $D_i^{\text{new}} \succ D$ for some $D \in \mathcal{F}_j$, i.e. D_i^{new} strictly dominates at least a single dyad in $\mathcal{F}_j$. Define the *Pareto depth* of D_i^{new} by

$$e_i = \min\{j \mid D_i^{\text{new}} \text{ is below } \mathcal{F}_j\}.$$

Therefore if e_i is large, then D_i^{new} will be near deep fronts, and the distance between X and the corresponding training sample will be large under *all* combinations of the K criteria. If e_i is small, then D_i^{new} will be near shallow fronts, so the distance between X and the corresponding training sample will be small under *some* combination of the K criteria.

2.4.2 Anomaly detection using Pareto depths

In k-NN based anomaly detection algorithms such as those mentioned in Section 2.2, the *anomaly score* is a function of the k nearest neighbors to a test sample. With multiple criteria, one could define an anomaly score by scalarization. From the probabilistic properties of Pareto fronts discussed in Section 2.3.1, we know that Pareto optimization methods identify more Pareto-optimal points than linear scalarization methods and signifi-

[1]If a training sample is one of the k_l nearest neighbors in multiple criteria, then multiple copies of the dyad corresponding to the connection between the test sample and the training sample are created.

cantly more Pareto-optimal points than a single weight for scalarization[2].

This motivates us to develop a *multi-criteria anomaly score* using Pareto fronts. We start with the observation from Fig. 2.1 that dyads corresponding to a nominal test sample are typically located near shallower fronts than dyads corresponding to an anomalous test sample. Each test sample is associated with $s = \sum_{l=1}^{K} k_l$ new dyads, where the ith dyad D_i^{new} has depth e_i. The Pareto depth e_i is a *multi-criteria dissimilarity measure* that indicates the dissimilarity between the test sample and training sample i under multiple combinations of the criteria. For each test sample X, we define the anomaly score $v(X)$ to be the mean of the e_i's, which corresponds to the average depth of the s dyads associated with X, or equivalently, the average of the multi-criteria dissimilarities between the test sample and its s nearest neighbors. Thus the anomaly score can be easily computed and compared to a decision threshold ρ using the test

$$v(X) = \frac{1}{s}\sum_{i=1}^{s} e_i \underset{H_0}{\overset{H_1}{\gtrless}} \rho.$$

Recall that the Scalarization Gap Theorem provides bounds on the fraction of dyads on the first Pareto front that *cannot* be obtained by linear scalarization. Specifically, at least $\frac{K-1}{4K-2}$ dyads will be missed by linear scalarization on average. These dyads will be associated with deeper fronts by linear scalarization, which will artificially inflate the anomaly score for the test sample, resulting in an increased false positive rate for any algorithm that utilizes non-negative linear combinations of criteria. This effect then cascades to dyads in deeper Pareto fronts, which also get assigned inflated anomaly scores. We provide some evidence of this effect on a real data experiment in Section 2.5.3. Notice that when there are only few training samples and few associated Pareto points for constructing fronts, the anomaly scores defined above might be unstable. One possible way to overcome this problem is to use the median of the e_i's instead of the mean as the anomaly score. Finally, the

[2]Theorems II.1 and II.2 require *i.i.d.* samples, but dyads are not independent. However, there are $O(N^2)$ dyads, and each dyad is only dependent on $O(N)$ other dyads. This suggests that the theorems should also hold for the non-*i.i.d.* dyads as well, and it is supported by experimental results presented in Section 2.5.1.

```
Training phase:
 1: for $l = 1 \to K$ do
 2:     Calculate pairwise dissimilarities $d_l(i,j)$ between all training samples $X_i$ and $X_j$
 3: end for
 4: Create dyads $D_{ij} = [d_1(i,j), \ldots, d_K(i,j)]$ for all training samples
 5: Construct Pareto fronts on set of all dyads until each dyad is in a front
Testing phase:
 1: $nb \leftarrow [\,]$ {empty list}
 2: for $l = 1 \to K$ do
 3:     Calculate dissimilarities between test sample $X$ and all training samples in criterion $l$
 4:     $nb_l \leftarrow k_l$ nearest neighbors of $X$
 5:     $nb \leftarrow [nb, nb_l]$ {append neighbors to list}
 6: end for
 7: Create $s = \sum_{l=1}^{K} k_l$ new dyads $D_i^{\text{new}}$ between $X$ and training samples in $nb$
 8: for $i = 1 \to s$ do
 9:     Calculate depth $e_i$ of $D_i^{\text{new}}$
10: end for
11: Declare $X$ an anomaly if $v(X) = (1/s) \sum_{i=1}^{s} e_i > \sigma$
```

Figure 2.4: Pseudocode for PDA anomaly detection algorithm.

lower bound increases monotonically in K, which implies that the PDA approach gains additional advantages over linear combinations as the number of criteria increases.

2.4.3 PDA anomaly detection algorithm

Pseudocode for the PDA anomaly detector is shown in Fig. 2.4. The training phase involves creating $\binom{N}{2}$ dyads corresponding to all pairs of training samples. Computing all pairwise dissimilarities in each criterion requires $O(mKN^2)$ floating-point operations (flops), where m denotes the number of dimensions involved in computing a dissimilarity. The Pareto fronts are constructed by non-dominated sorting. We use the non-dominated sort of Deb et al. *Deb et al.* [36] that constructs all of the Pareto fronts using $O(KN^4)$ comparisons in the worst case, which is linear in the number of criteria K. For large N, the algorithm of Jensen *Jensen* [59] may be a faster alternative, using $O(N^2 \log^{K-1} N^2)$ comparisons in the worst case. Note that the number of training samples might affect the PDA algorithm. If the number of training dyads is small, the variance of constructed Pareto fronts might be high. For sensitivity analysis of fronts using finite dyads, we refer readers to the work [22, 23] by Jeff Calder, one of the main collaborators of this work.

The testing phase involves creating dyads between the test sample and the k_l nearest training samples in criterion l, which requires $O(mKN)$ flops. For each dyad D_i^{new}, we need to calculate the depth e_i. This involves comparing the test dyad with training dyads on multiple fronts until we find a training dyad that is dominated by the test dyad. e_i is the front that this training dyad is a part of. Using a binary search to select the front and another binary search to select the training dyads within the front to compare to, we need to make $O(K \log^2 N)$ comparisons (in the worst case) to compute e_i. The anomaly score is computed by taking the mean of the s e_i's corresponding to the test sample; the score is then compared against a threshold ρ to determine whether the sample is anomalous.

To handle multiple criteria, other anomaly detection methods, such as the ones mentioned in Section 2.2, need to be re-executed multiple times using different (non-negative) linear combinations of the K criteria. If a grid search is used for selection of the weights in the linear combination, then the required computation time would be exponential in K. Such an approach presents a computational problem unless K is very small. On the other hand, PDA scales *linearly* with K in both the training and test phases so it does not encounter this problem.

2.4.4 Selection of parameters

The parameters to be selected in PDA are $k_1, \ldots, k_K$, which denote the number of nearest neighbors in each criterion. The selection of such parameters in unsupervised learning problems is very difficult in general. For each criterion l, we construct a k_l-NN graph using the corresponding dissimilarity measure. We construct symmetric k_l-NN graphs, i.e. we connect samples i and j if i is one of the k_l nearest neighbors of j or j is one of the k_l nearest neighbors of i. We choose $k_l = \log N$ as a starting point and, if necessary, increase k_l until the k_l-NN graph is connected. This method of choosing k_l is motivated by asymptotic results for connectivity in k-NN graphs and has been used as a heuristic in other unsupervised learning problems, such as spectral clustering [107]. We find that this heuristic works well in practice, including on a real data set of pedestrian trajectories, which we present in Section 2.5.3.

2.5 Experiments

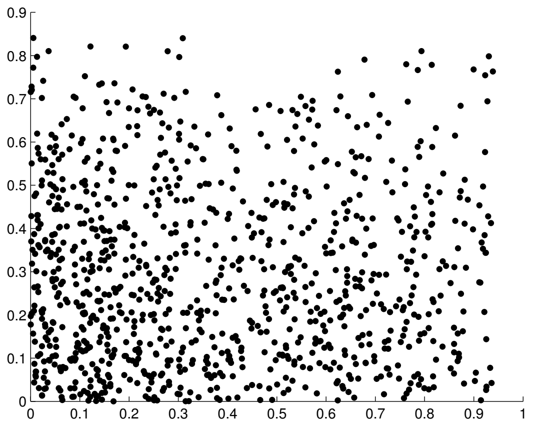
We first present an experiment involving the scalarization gap for dyads (rather than *i.i.d.* samples). Then we compare the PDA method with four nearest neighbor-based single-criterion anomaly detection algorithms mentioned in Section 2.2 on a simulated data set and a real data set. The four algorithms we present for comparison utilize the following anomaly scores:

- kNN: distance to the kth nearest neighbor [21].
- kNN sum: sum of the distances to the k nearest neighbors [2, 38].
- LOF: local density of the k nearest neighbors [18].
- k-LPE: localized p-value estimate using the k nearest neighbors [114].

For these methods, we use linear combinations of the criteria with different weights (linear scalarization) to compare performance with the proposed multi-criteria PDA method.

2.5.1 Scalarization gap for dyads

Independence of $Y_1, \ldots, Y_n$ is built into the assumptions of Theorems II.1 and II.2, and thus, the Scalarization Gap Theorem, but it is clear that dyads (as constructed in Section 2.4) are not independent. Each dyad D_{ij} represents a connection between two independent samples X_i and X_j. For a given dyad D_{ij}, there are $2(N-2)$ corresponding dyads involving X_i or X_j, and these are clearly not independent from D_{ij}. However, all other dyads are independent from D_{ij}. So while there are $O(N^2)$ dyads, each dyad is independent from all other dyads except for a set of size $O(N)$. Since the Scalarization Gap Theorem is an



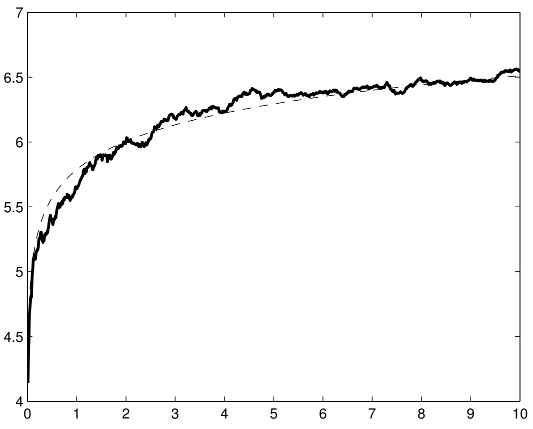


Figure 2.5: (a) 990 dyads constructed with the criteria $|\Delta x|, |\Delta y|$ from 45 samples uniformly distributed in $[0,1]^2$. (b) Sample means for $K_n - L_n$ versus number of dyads. Note the expected logarithmic growth. The dotted lines indicate the best fit curves described in this section.

asymptotic result, the above observation suggests it should hold for the dyads even though they are not *i.i.d.* In this subsection we present some experimental results which suggest that the Scalarization Gap Theorem does indeed hold for dyads.

We first draw samples uniformly in $[0,1]^2$ and construct dyads corresponding to the two criteria $|\Delta x|$ and $|\Delta y|$, which denote the absolute differences between the x and y coordinates, respectively. The domain of the resulting dyads is again the box $[0,1]^2$, as shown in Fig. 2.5a. In this case, the Scalarization Gap Theorem suggests that $E(K_n - L_n)$ should grow logarithmically. Fig. 2.5b shows the sample means of $K_n - L_n$ versus number of dyads and a best fit logarithmic curve of the form $y = \alpha \log n$, where $n = \binom{N}{2}$ denotes the number of dyads. We vary the number of dyads between 10^6 to 10^9 in increments of 10^6 and compute the size of $K_n - L_n$ after each increment. We compute the sample means over $1,000$ realizations. A linear regression on $y/\log n$ versus $\log n$ gives $\alpha = 0.3142$, which falls in the range specified by the Scalarization Gap Theorem.

We next explore the dependence of $K_n - L_n$ on the dimension d. Here, we generate $100,128$ dyads (corresponding to $N = 448$ points in $[0,1]^d$) in the same way as before, for dimensions $d = 2, \ldots, 7$. The criteria in this case correspond to the absolute differences in

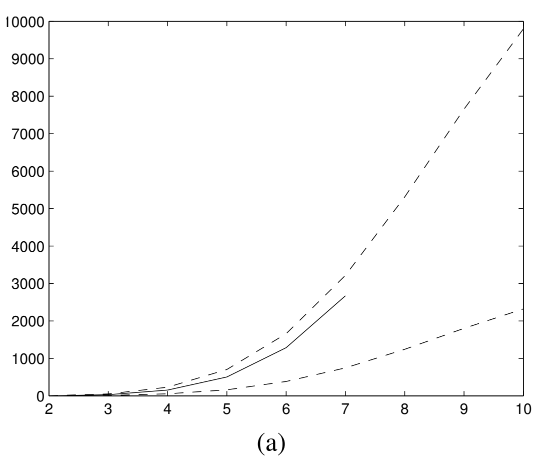


Figure 2.6: (a) Sample means for $K_n - L_n$ versus dimension and (b) $(K_n - L_n)/c_{n,d}$ versus dimension for $n = 100,128$ dyads. The upper and lower bounds on $E(K_n - L_n)$ established in the Scalarization Gap Theorem are given by the dotted lines in (a). Observing (b), we see that the fraction of points that cannot be obtained through linear scalarization is increasing, possibly to 1, as the dimension increases. We are only able to run the experiment up to dimension $d = 7$ due to the computational complexity of computing a convex hull, which is $O\big(n^{d/2}\big)$.

each dimension. Fig. 2.6a shows the sample means for $K_n - L_n$ versus the dimension d along with the asymptotic upper and lower bounds for $E(K_n - L_n)$ derived in the Scalarization Gap Theorem (shown as dotted lines). We see from the figure that the upper bound is reasonably tight. Recall from Theorem II.1 that

$$E(K_n) \sim c_{n,d} = \frac{(\log n)^{d-1}}{(d-1)!} \quad \text{as } n \to \infty.$$

In Fig. 2.6b we plot $E(K_n - L_n)/c_{n,d}$ versus dimension to show the fraction of Pareto-optimal points that cannot be obtained by scalarization. Based on the figure, one might conjecture that this fraction converges to 1 as $d \to \infty$. If this is true, it would essentially imply that linear scalarization is useless for identifying dyads on the first Pareto front when there are a large number of criteria. As before, we compute the sample means over $1,000$ realizations of the experiment.

2.5.2 Simulated experiment with categorical attributes

In this experiment, we perform multi-criteria anomaly detection on simulated data with multiple groups of categorical attributes. These groups could represent different types of attributes. Each data sample consists of K groups of 20 categorical attributes. Let A_{ij} denote the jth attribute in group i, and let n_{ij} denote the number of possible values for this attribute. We randomly select between 6 and 10 possible values for each attribute with equal probability independent of all other attributes. Each attribute is a random variable described by a categorical distribution, where the parameters $q_1, \ldots, q_{n_{ij}}$ of the categorical distribution are sampled from a Dirichlet distribution with parameters $\alpha_1, \ldots, \alpha_{n_{ij}}$. For a nominal data sample, we set $\alpha_1 = 5$ and $\alpha_2, \ldots, \alpha_{n_{ij}} = 1$ for each attribute j in each group i.

To simulate an anomalous data sample, we randomly select a group i with probability p_i for which the parameters of the Dirichlet distribution are changed to $\alpha_1 = \cdots = \alpha_{n_{ij}} = 1$ for each attribute j in group i. Note that different anomalous samples may differ in the group that is selected. The p_i's are chosen such that $p_i/p_j = i/j$ with $\sum_{i=1}^{K} p_i = 0.5$, so that the probability that a test sample is anomalous is 0.5. The non-uniform distribution on the p_i's results in some criteria being more useful than others for identifying anomalies. The K criteria for anomaly detection are taken to be the dissimilarities between data samples for each of the K groups of attributes. For each group, we calculate the dissimilarity over the attributes using a dissimilarity measure for anomaly detection on categorical data proposed in *Eskin et al.* [38][3].

We draw 400 training samples from the nominal distribution and 400 test samples from a mixture of the nominal and anomalous distributions. We use $k = 6$ nearest neighbors in all cases. For the single-criterion algorithms, which we use as baselines for comparison, we use linear scalarization with multiple choices of weights. Since a grid search scales exponentially with the number of criteria K and is computationally intractable even for

[3]We obtain similar results with several other dissimilarity measures for categorical data, including the Goodall2 and IOF measures described in the survey paper by Boriah et al. *Boriah et al.* [14].

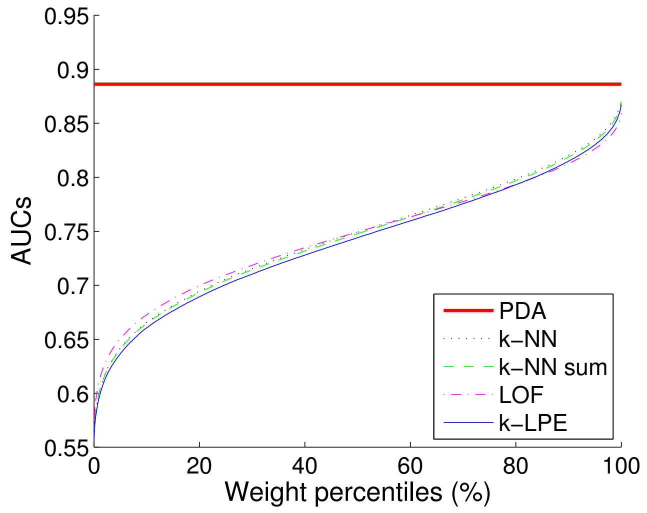


Figure 2.7: AUC of PDA compared to AUCs of single-criterion methods for the simulated experiment. The single-criterion methods use 600 randomly sampled weights for linear scalarization, with weights ordered from worst choice of weights (left) to best choice (right) in terms of maximizing AUC. The proposed PDA algorithm is a multi-criteria algorithm that does not require selecting weights. PDA outperforms all of the single-criterion methods, even for the best choice of weights, which is not known in advance.

moderate values of K, we instead uniformly sample weights from the $(K-1)$-dimensional simplex. In other words, we sample weights from a uniform distribution over all convex combinations of K criteria. Since PDA scales linearly in K, we uniformly sample $100K$ weights to present a fair comparison.

The different methods are evaluated using the receiver operating characteristic (ROC) curve and the area under the ROC curve (AUC). We first fix the number of criteria K to be 6. The mean AUCs over 100 simulation runs are shown in Fig. 2.7. Multiple choices of weights are used for linear scalarization for the single-criterion algorithms; the results are ordered from worst to best weight in terms of maximizing AUC. Table 2.1 presents a comparison of the AUC for PDA with the median and best AUCs over all choices of weights for scalarization. Both the mean and standard error of the AUCs over the 100 simulation runs are shown. Notice that *PDA outperforms even the best weighted combination* for each of the four single-criterion algorithms and significantly outperforms the combination resulting in the median AUC.

Table 2.1: Comparison of AUCs for both anomaly detection experiments. Best performer is shown in **bold**. PDA does not use weights so it has a single AUC. The median and best AUCs over all choices of weights are shown for the other methods.

Method	Simulated experiment		Pedestrian	
	Median	Best	Median	Best
PDA	**0.885 ± 0.002**		**0.933**	
k-NN	0.749 ± 0.002	0.872 ± 0.002	0.891	0.905
k-NN sum	0.747 ± 0.002	0.870 ± 0.002	0.881	0.912
LOF	0.749 ± 0.002	0.859 ± 0.002	0.779	0.929
k-LPE	0.744 ± 0.002	0.867 ± 0.002	0.881	0.912

Next we investigate the performance gap between PDA and scalarization as the number of criteria K varies from 2 to 10. The four single-criterion algorithms perform roughly equally, so we show scalarization results only for LOF. The ratio of the AUC for PDA to the AUCs of the best and median weights for scalarization are shown in Fig. 2.8. PDA offers a significant improvement compared to the median over the weights for scalarization. For small values of K, PDA performs roughly equally with scalarization under the best choice of weights. As K increases, however, PDA clearly outperforms scalarization, and the gap grows with K. We believe this is partially due to the inadequacy of scalarization for identifying Pareto fronts as described in the Scalarization Gap Theorem and partially due to the difficulty in selecting optimal weights for the criteria. A grid search may be able to reveal better weights for scalarization, but it is also computationally intractable. Thus we conclude that PDA is clearly the superior approach for large K.

2.5.3 Pedestrian trajectories

We now present an experiment on a real data set that contains thousands of pedestrians' trajectories in an open area monitored by a video camera [82]. We represent a trajectory with p time samples by

$$T = \begin{bmatrix} x_1 & x_2 & \dots & x_p \\ y_1 & y_2 & \dots & y_p \end{bmatrix},$$

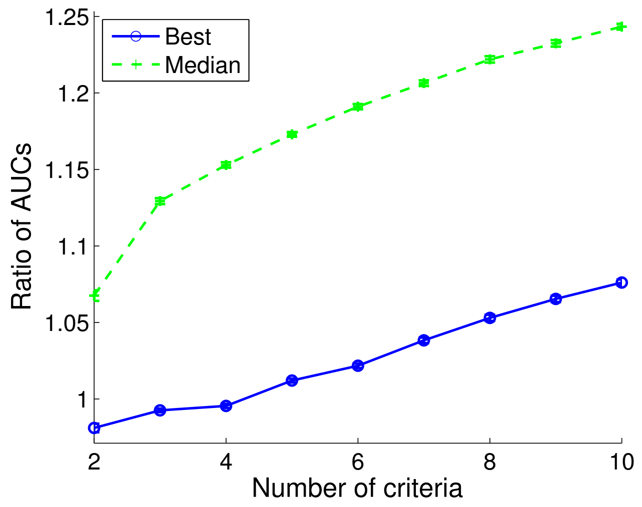
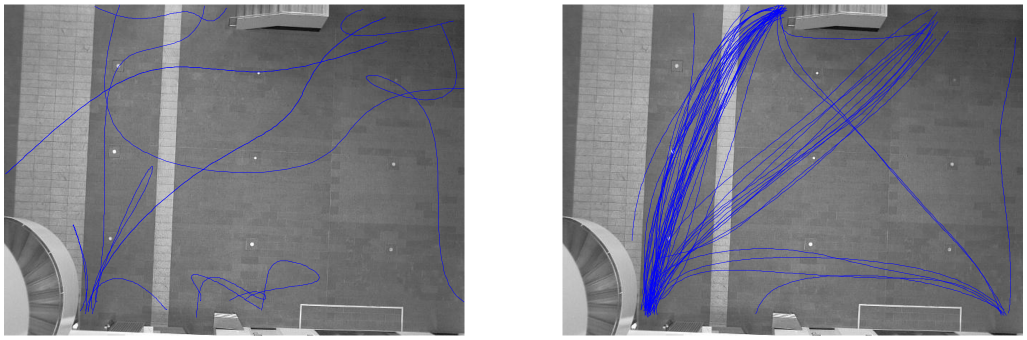


Figure 2.8: The ratio of the AUC for PDA compared to the best and median AUCs of scalarization using LOF as the number of criteria K is varied in the simulation experiment. $100K$ choices of weights uniformly sampled from the $(K-1)$-dimensional simplex are chosen for scalarization. PDA perfoms significantly better than the median over all weights for all K. For $K > 4$, PDA outperforms the best weights for scalarization, and the margin increases as K increases.

where $[x_t, y_t]$ denote a pedestrian's position at time step t. The pedestrian trajectories are of different lengths so we cannot simply treat the trajectories as vectors in $\mathbb{R}^p$ and calculate Euclidean distances between them. Instead, we propose to calculate dissimilarities between trajectories using two separate criteria for which trajectories may be dissimilar.

The first criterion is to compute the dissimilarity in *walking speed*. We compute the instantaneous speed at all time steps along each trajectory by finite differencing, i.e. the speed of trajectory T at time step t is given by $\sqrt{(x_t - x_{t-1})^2 + (y_t - y_{t-1})^2}$. A histogram of speeds for each trajectory is obtained in this manner. We take the dissimilarity between two trajectories S and T to be the *Kullback-Leibler (K-L) divergence* between the normalized speed histograms for those trajectories. K-L divergence is a commonly used measure of the difference between two probability distributions. The K-L divergence is asymmetric; to convert it to a dissimilarity we use the symmetrized K-L divergence $D_{KL}(S||T) + D_{KL}(T||S)$ as originally defined by Kullback and Leibler [69]. We note that, while the symmetrized K-L divergence is a dissimilarity, it does not, in general, sat-



(a) (b)

Figure 2.9: (a) Some anomalous pedestrian trajectories detected by PDA. (b) Trajectories with relatively low anomaly scores. The two criteria used are walking speed and trajectory shape. Anomalous trajectories could have anomalous speeds or shapes (or both), so some anomalous trajectories may not look anomalous by shape alone.

isfy the triangle inequality and is not a metric.

The second criterion is to compute the dissimilarity in *shape*. To calculate the shape dissimilarity between two trajectories, we apply a technique known as *dynamic time warping* (DTW) [94], which first non-linearly warps the trajectories in time to match them in an optimal manner. We then take the dissimilarity to be the summed Euclidean distance between the warped trajectories. This dissimilarity also does not satisfy the triangle inequality in general and is thus not a metric.

The training set for this experiment consists of 500 randomly sampled trajectories from the data set, a small fraction of which may be anomalous. The test set consists of 200 trajectories (150 nominal and 50 anomalous). The trajectories in the test set are labeled as nominal or anomalous by a human watching each individual trajectory. These labels are used as ground truth to evaluate anomaly detection performance. Fig. 2.9 shows some anomalous trajectories and nominal trajectories detected using PDA.

Fig. 2.10 shows the performance of PDA as compared to the other algorithms using 100 uniformly spaced weights for convex combinations. Again, we use $k = 6$ neighbors in all cases. Notice that PDA has higher AUC than the other methods under *all* choices of weights

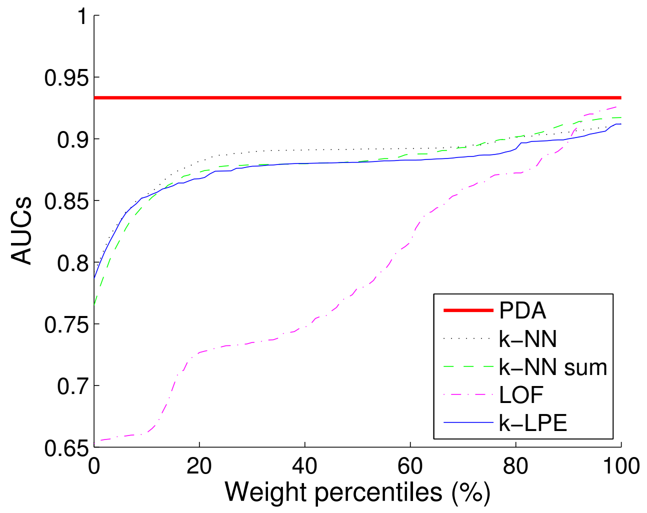


Figure 2.10: AUC of PDA compared to AUCs of single-criterion methods for the pedestrian trajectories experiment. The single-criterion methods use linear scalarization with 100 uniformly spaced weights; weights are ordered from worst (left) to best (right) in terms of maximizing AUC. PDA outperforms all of the single-criterion methods, even for the best choice of weights.

for the two criteria. The AUC for PDA is shown in Table 2.1 along with AUCs for the median and best choices of weights for the single-criterion methods. For the best choice of weights, LOF is the single-criterion method with the highest AUC, but it also has the lowest AUC for poor choices of weights. For a more detailed comparison, the ROC curve for PDA and the attainable region for LOF (the region between the ROC curves corresponding to weights resulting in the best and worst AUCs) is shown in Fig. 2.11. Note that the ROC curve for LOF can vary significantly based on the choice of weights. In the unsupervised setting, it is unlikely that one would be able to achieve the ROC curve corresponding to the weight with the highest AUC, so the expected performance gap between PDA and scalarization should be larger, as seen from the median AUCs in Table 2.1.

Many of the Pareto fronts on the dyads are *non-convex*, partially explaining the superior performance of the proposed PDA algorithm. The non-convexities in the Pareto fronts lead to inflated anomaly scores for linear scalarization. A comparison of a Pareto fronts with two convex fronts (obtained by scalarization) is shown in Fig. 2.12. The two convex fronts

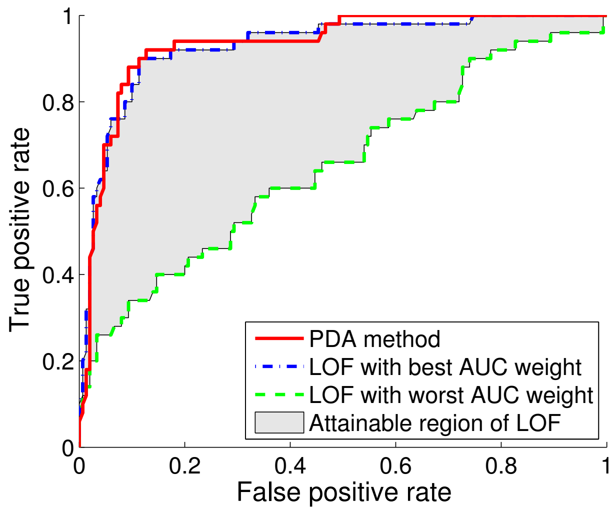


Figure 2.11: ROC curves for PDA and attainable region for LOF over 100 choices of weights for the pedestrian trajectories experiment. The attainable region denotes the possible ROC curves for LOF corresponding to different choices of weights for linear scalarization. The ROCs for linear scalarization vary greatly as a function of the weights yet even the best weights do not outperform the ROC of the proposed PDA method.

denote the shallowest and deepest convex fronts containing dyads on the illustrated Pareto front. The test samples associated with dyads near the middle of the Pareto fronts would suffer the aforementioned score inflation, as they would be found in deeper convex fronts than those at the tails. More examples showing how non-convexity property of Pareto fronts would affect the results are shown in the next chapter.

Finally we note that the proposed PDA algorithm does not appear to be very sensitive to the choices of the number of neighbors, as shown in Fig. 2.13. In fact, the heuristic proposed for choosing the k_l's in Section 2.4.4 performs quite well in this experiment. Specifically, the AUC obtained when using the parameters chosen by the proposed heuristic is very close to the maximum AUC over all choices of the number of neighbors $[k_1, k_2]$.

2.6 Conclusion

In this chapter we proposed a method for anomaly detection using a novel multi-criteria dissimilarity measure, the Pareto depth. The proposed method utilizes the notion of Pareto

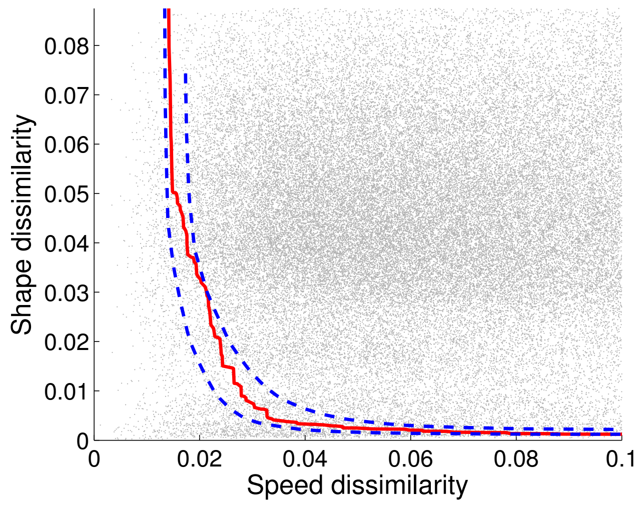


Figure 2.12: Comparison of a Pareto front (solid red line) on dyads (gray dots) with convex fronts (blue dashed lines) obtained by linear scalarization. The dyads towards the middle of the Pareto front are found in deeper convex fronts than those towards the edges. The result would be inflated anomaly scores for the samples associated with the dyads in the middle of the Pareto fronts when using linear scalarization.

optimality to detect anomalies under multiple criteria by examining the Pareto depths of dyads corresponding to the test sample. Dyads corresponding to an anomalous sample tended to be located at deeper fronts compared to dyads corresponding to a nominal sample. Instead of choosing a specific weighting or performing a grid search on the weights for dissimilarity measures corresponding to different criteria, the proposed method can efficiently detect anomalies in a manner that scales linearly in the number of criteria. The proposed Pareto depth analysis (PDA) approach is provably better than using linear combinations of criteria. Numerical studies validated our theoretical predictions of PDA's performance advantages compared to using linear combinations on simulated and real data.

Acknowledgements

We thank Zhaoshi Meng for his assistance in labeling the pedestrian trajectories. This work was supported in part by ARO grant W911NF-09-1-0310.

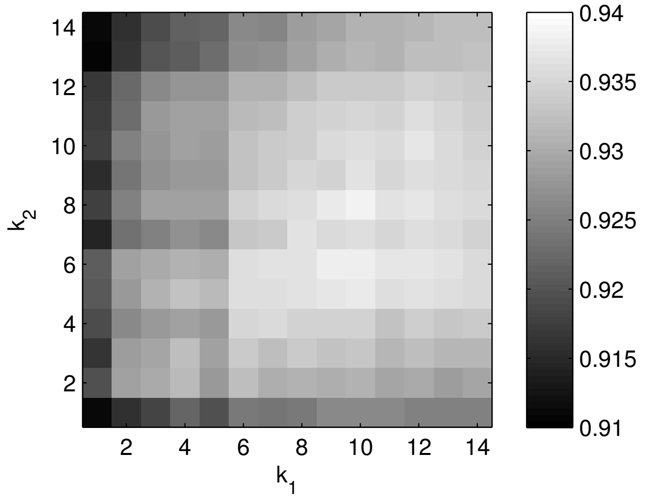


Figure 2.13: AUCs for different choices of $[k_1, k_2]$ in the pedestrian trajectories experiment. The AUC for the parameters chosen using the proposed heuristic $[k_1 = 6, k_2 = 7]$ is 0.933, which is close to the maximum AUC of 0.939 obtained by the parameters $[k_1 = 10, k_2 = 8]$.

CHAPTER III

Pareto-depth for Multiple-query Image Retrieval

3.1 Introduction

In this chapter we continue to use the idea of multiple Pareto fronts introduced in Chapter II on a fundamentally different problem – multiple-query image retrieval. In this problem different information come from different queries. In the past two decades content-based image retrieval (CBIR) has become an important problem in machine learning and information retrieval [42, 44, 106]. Several image retrieval systems for multiple queries have been proposed in the literature [3, 10, 62]. In most systems, each query image corresponds to the same image semantic concept, but may possibly have a different background, be shot from an alternative angle, or contain a different object in the same class. The idea is that by utilizing multiple queries of the same object, the performance of single-query retrieval can be improved. We will call this type of multiple-query retrieval *single-semantic-multiple-query* retrieval. Many of the techniques for single-semantic-multiple-query retrieval involve combining the low-level features from the query images to generate a single averaged query [3].

In this chapter we consider the more challenging problem of finding images that are relevant to multiple queries that represent different image semantics. In this case, the goal is to find images containing relevant features from *each and every* query. Since the queries correspond to different semantics, desirable images will contain features from several distinct images, and will not necessarily be closely related to any individual query. This makes the problem fundamentally different from single query retrieval, and from single-semantic-multiple-query retrieval. In this case, the query images will not have similar low level features, and forming an averaged query is not as useful.

Since relevant images do not necessarily have features closely aligned with any particular query, many of the standard retrieval techniques are not useful in this context. For example, bag-of-words type approaches, which may seem natural for this problem, require the target image to be closely related to several of the queries. Another common technique is to input each query one at a time and average the resulting similarities. This tends to produce images closely related to one of the queries, but rarely related to all at once. Many other multiple-query retrieval algorithms are designed specifically for the single-semantic-multiple-query problem [3], and again tend to find images related to only one, or a few, of the queries.

Multiple-query retrieval is related to the metasearch problem in computer science. In metasearch, the problem is to combine search results for the same query across multiple search engines. This is similar to the single-semantic-multiple-query problem in the sense that every search engine is issuing the same query (or semantic). Thus, metasearch algorithms are not suitable in the context of multiple-query retrieval with several distinct semantics.

In this chapter, we propose a novel algorithm for multiple-query image retrieval that combines the Pareto front method (PFM) with efficient manifold ranking (EMR). The first step in our PFM algorithm is to issue each query individually and rank all samples in the database based on their dissimilarities to the query. Several methods for computing representations of images, like SIFT and HoG, have been proposed in the computer vision literature, and any of these can be used to compute the image dissimilarities. Since it is very computationally intensive to compute the dissimilarities for every sample-query pair in large databases, we use a fast ranking algorithm called Efficient Manifold Ranking (EMR) [112] to compute the ranking without the need to consider all sample-query pairs. EMR can efficiently discover the underlying geometry of the given database and significantly reduces the computational time of traditional manifold ranking. Since EMR has been successfully applied to single query image retrieval, it is the natural ranking algorithm to consider for the multiple-query problem. The next step in our PFM algorithm is to use the ranking produced by EMR to create Pareto points, which correspond to dissimilarities between a sample and every query. Sets of Pareto-optimal points, called *Pareto fronts*, are then computed. Recall that we have introduced Pareto fronts in previous chapter. In Chapter II, a Pareto point corresponds to different dissimilarities between two samples. In other words one Pareto point corresponds to a pair of samples. However, in this chapter one Pareto point corresponds to one sample. Once we create Pareto points, we apply the same procedure, *non-dominated sorting*, to construct multiple Pareto fronts as in Chapter II.

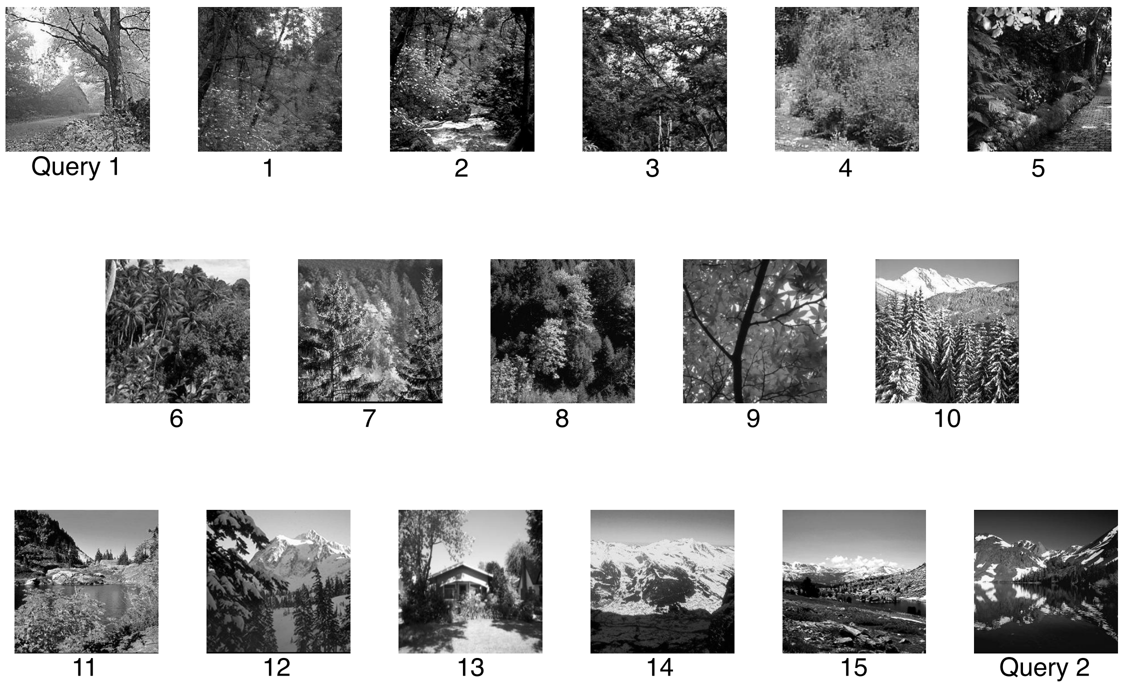


Figure 3.1: Images located on the first Pareto front when a pair of query images are issued. Images from the middle part of the front (images 10, 11 and 12) contain semantic information from *both* query images. The images are from **Stanford 15 scene** dataset.

A key observation in this work is that the middle of the Pareto front is of fundamental importance for the multiple-query retrieval problem. As an illustrative example, we show in Figure 3.1 the images from the first Pareto front for a pair of query images corresponding to a forest and a mountain. The images are listed according to their position within the front, from one tail to the other. The images located at the tails of the front are very close to one of the query images, and may not necessarily have any features in common with the other query. However, as seen in Figure 3.1, images in the middle of the front (e.g., images 10, 11 and 12) contain relevant features from *both* queries, and hence are very desirable for the multiple-query retrieval problem. It is exactly these types of images that our algorithm is designed to retrieve.

The Pareto front method is well-known to have many advantages when the Pareto fronts are non-convex [37]. We have presented some theoretical results about Pareto fronts in Chapter II. In this chapter, we present a new theorem that characterizes the asymptotic convexity (and lack thereof) of Pareto fronts as the size of the database becomes large. This result is based on establishing a connection between Pareto fronts and chains in partially ordered finite set theory. The connection is as follows: a data point is on the Pareto front of depth n if and only if it admits a maximal chain of length n. This connection allows us to utilize results from the literature on the longest chain problem, which has a long history in probability and combinatorics. Our main result (Theorem III.3) shows that the Pareto fronts are asymptotically convex when the dataset can be modeled as *i.i.d.* random variables drawn from a continuous separable log-concave density function $f : [0,1]^d \to (0,\infty)$. This theorem suggests that our proposed algorithm will be particularly useful when the underlying density is not log-concave. We give some numerical evidence (see Figure 3.2b) indicating that the underlying density is typically not even quasi-concave. This helps to explain the performance improvement obtained by our proposed Pareto front method.

We also note that our PFM algorithm could be applied to automatic image annotation of large databases. Here, the problem is to automatically assign keywords, classes or captioning to images in an unannotated or sparsely annotated database. Since images in the middle of first few Pareto fronts are relevant to all queries, one could issue different query combinations with known class labels or other metadata, and automatically annotate the images in the middle of the first few Pareto fronts with the metadata from the queries. This procedure could, for example, transform a single-class labeled image database into one with multi-class labels. Some major works and introductions to automatic image annotation can be found in [27, 60, 92].

The rest of this chapter is organized as follows. We discuss related work in Section 3.2. In Section 3.3, we introduce the Pareto front method and present a theoretical analysis of the convexity properties of Pareto fronts. In Section 3.4 we show how to apply the Pareto front method (PFM) to the multiple-query retrieval problem and briefly introduce Efficient Manifold Ranking. Finally, in Section 3.5 we present experimental results and demonstrate a graphical user interface (GUI) that allows the user to explore the Pareto fronts and visualize the partially ordered relationships between the queries and the images in the database.

3.2 Related work

3.2.1 Content-based image retrieval

Content-based image retrieval (CBIR) has become an important problem over the past two decades. Overviews can be found in [34, 77]. A popular image retrieval system is query-by-example (QBE) [49, 120], which retrieves images relevant to one or more queries provided by the user. In order to measure image similarity, many sophisticated color and texture feature extraction algorithms have been proposed; an overview can be found in [34, 77]. SIFT [78] and HoG [32] are two of most well-known and widely used feature extraction techniques in computer vision research. Several CBIR techniques using multiple queries have been proposed [3, 62]. Some methods combine the queries together to generate a query center, which is then modified with the help of relevance feedback. Other algorithms issue each query individually to introduce diversity and gather retrieved items scattered in visual feature space [62].

The problem of ranking large databases with respect to a similarity measure has drawn great attention in the machine learning and information retrieval fields. Many approaches to ranking have been proposed, including learning to rank [20, 76], content-based ranking models (BM25, Vector Space Model), and link structure ranking model [19]. Manifold ranking [116, 117] is an effective ranking method that takes into account the underlying geometrical structure of the database. Xu et al. [112] introduced an algorithm called Efficient Manifold Ranking (EMR) which uses an anchor graph to do efficient manifold ranking that can be applied to large-scale datasets. In this work, we use EMR to assign a rank to each sample with respect to each query before applying our Pareto front method.

3.2.2 Pareto method

As mentioned in Chapter II, there is wide use of Pareto-optimality in the machine learning community [63]. Many of these methods must solve complex multi-objective optimization problems, where finding even the first Pareto front is challenging. Like that in Chapter II, our use of Pareto-optimality differs as we generate *multiple* Pareto fronts from a finite set of items, and as such we do not require sophisticated methods to compute the fronts.

In Section 2.2, we have mentioned that the first Pareto front, which consists of the set of non-dominated points, is often called the Skyline in computer science. We give a brief introduction to Skyline here. Several sophisticated and efficient algorithms have been developed for computing the Skyline [15, 68, 88, 103]. Various Skyline techniques have been proposed for different applications in large-scale datasets, such as multi-criteria decision making, user-preference queries, and data mining and visualization [1, 52, 61]. Efficient and fast Skyline algorithms [68] or fast non-dominated sorting [59] can be used to find each Pareto front in our PFM algorithm for large-scale datasets.

Sharifzadeh and Shahabi[97] introduced Spatial Skyline Queries (SSQ) which is similar to the multiple-query retrieval problem. However, since EMR is not a metric (it doesn't satisfy the triangle inequality), the relation between the first Pareto front and the convex hull of the queries, which is exploited by Sharifzadeh and Shahabi[97], does not hold in our setting. Our method also differs from SSQ and other Skyline research because we use multiple fronts to rank items instead of using only Skyline queries. We also address the problem of combining EMR with the Pareto front method for multiple queries associated with *different* concepts, resulting in *non-convex* Pareto fronts. To the best of our knowledge, this problem has not been widely researched.

In Chapter II, we propose a multi-criteria anomaly detection algorithm utilizing Pareto depth analysis. This approach uses multiple Pareto fronts to define a new dissimilarity between samples based on their Pareto depth. In Chapter II's case, each Pareto point corresponds to a similarity vector between pairs of database entries under multiple similarity criteria. In this chapter, a Pareto point corresponds to a vector of dissimilarities between a single entry in the database and multiple queries.

A related field is metasearch [4, 86], in which one query is issued in different systems or search engines, and different ranking results or scores for each item in the database are obtained. These different scores are then combined to generate one final ranked list. Many different methods, such as Borda fuse and CombMNZ, have been proposed and are widely used in the metasearch community. The same methods have also been used to combine results for different representations of a query[10, 71]. However these algorithms are designed for the case that the queries represent the same semantics. In the multiple-query retrieval setting this case is not very interesting as it can easily be handled by other methods, including linear scalarization.

In contrast we study the problem where each query corresponds to a different image concept. In this case metasearch methods are not particularly useful, and are significantly outperformed by the Pareto front method. For example Borda fusion gives higher rankings to the tails of the fronts, and thus is similar to linear scalarization. CombMNZ gives a higher ranking to documents that are relevant to multiple-query aspects, but it utilizes a sum of all document scores, and as such is intimately related to linear scalarization with equal weights, which is equivalent to the Average of Multiple Queries (MQ-Avg) retrieval algorithm [3]. We show in Section 3.5 that our Pareto front method significantly outperforms

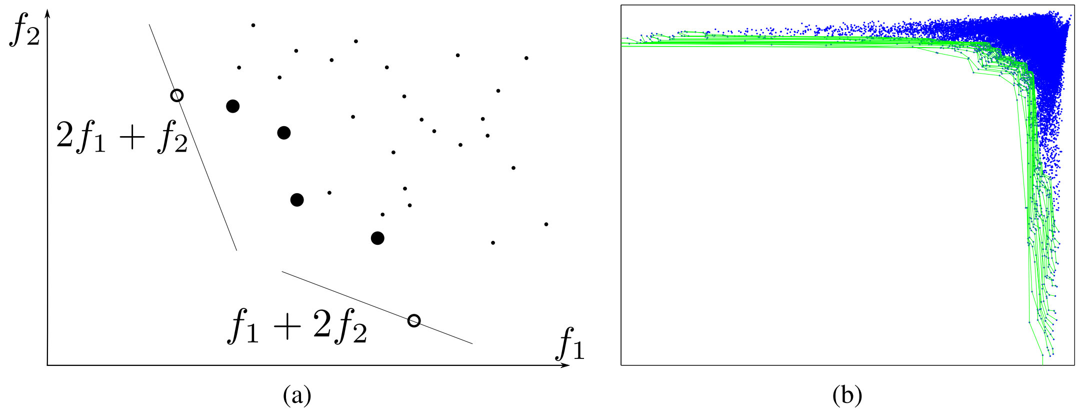


Figure 3.2: (a) Depiction of nonconvexities in the first Pareto front. The large points are Pareto-optimal, having the highest Pareto ranking by criteria f_1 and f_2, but only the hollow points can be obtained by linear scalarization. Here f_1 and f_2 are the dissimilarity values for query 1 and query 2, respectively. (b) Depiction of nonconvexities in the Pareto fronts in the real-world **Mediamill** dataset used in the experimental results in Section 3.5. The points on the non-convex portions of the fronts will be retrieved later by any scalarization algorithm, even though they correpsond to equally good images for the retrieval problem.

MQ-Avg, and all other multiple-query retrieval algorithms.

3.3 Pareto Front method

Although the meaning of Pareto points is fundamentally different from that in previous chapter, we can still use the same notion of a Pareto front as in Chapter II. Recall that the collection of Pareto-optimal feasible solutions is called the first *Pareto front*. It contains all solutions that can be found via linear scalarization, as well as other items that are missed by linear scalarization. The i^{th} Pareto front is again denoted by $\mathcal{F}_i$.

In this section we give another simple example in Figure 3.2a to show the advantage of using Pareto front methods for ranking. Here the number of criteria is $T = 2$ and the Pareto points $[f_1(x), f_2(x)]$, for $x \in S$, are shown in Figure 3.2a. In this figure the large points are Pareto-optimal, but only the hollow points can be obtained as top ranked items using linear scalarization. It is well-known, and easy to see in Figure 3.2a, that linear scalarization can only obtain Pareto points on the boundary of the convex hull of the Pareto front. The same observation holds for deeper Pareto fronts and has been shown in prevision section (Figure 2.12). Figure 3.2b shows Pareto fronts for the multiple-query retrieval problem using real data from the **Mediamill** dataset, introduced in Section 3.5. Notice the severe non-convexity in the shapes of the real Pareto fronts in Figure 3.2b. This is a key observation, and is directly related to the fact that each query corresponds to a different image semantic, and so there are few images that are very closely related to both queries. If two queries are strongly related to each other, associated point cloud will have a more convex shape. If queries are competing with each other and there are no images containing both semantic concepts in the database, a more severe non-convexity in the shapes of fronts will be observed.

3.3.1 Information retrieval using Pareto fronts

In this section we introduce the Pareto front method for the multiple-query information retrieval problem. Assume that a dataset $\mathcal{X}_N = \{X_1, \ldots, X_N\}$ of data samples is available. Given a query q, the objective of retrieval is to return samples that are related to the query. When multiple queries are present, our approach issues each query individually and then combines their results into one partially ordered list of Pareto-equivalent retrieved items at successive Pareto depths. For $T > 1$, denote the T-tuple of queries by $\{q_1, q_2, ..., q_T\}$ and the dissimilarity between q_i and the j^{th} item in the database, X_j, by $d_i(j)$. For convenience, define $d_i \in \mathbb{R}_+^N$ as the dissimilarity vector between q_i and all samples in the database. Given T queries, we define a *Pareto point* by $P_j = [d_1(j), \ldots, d_T(j)] \in \mathbb{R}_+^T, j \in \{1, \ldots, N\}$. Each Pareto point P_j corresponds to a sample X_j from the dataset $\mathcal{X}_N$. For convenience, denote the set of all Pareto points by $\mathcal{P}$. By definition, a Pareto point P_i strictly dominates another point P_j if $d_l(i) \leq d_l(j)$ for all $l \in \{1, \ldots, T\}$ and $d_l(i) < d_l(j)$ for some l. One can easily see that if P_i dominates P_j, then X_i is closer to every query than X_j. Therefore, the system should return X_i before X_j. The key idea of our approach is to return samples corresponding to which Pareto front they lie on, i.e., we return the points from $\mathcal{F}_1$ first, and then $\mathcal{F}_2$, and so on until a sufficient number of images have been retrieved. Since our goal is to find images related to each and every query, we start returning samples from the middle of the first Pareto front and work our way to the tails. The details of our algorithm are presented in Section 3.4.

3.3.2 Properties of Pareto fronts

In previous works we prove two theorems characterizing how many Pareto-optimal points are missed, on average and asymptotically, due to nonconvexities in the geometry of the Pareto point cloud, called *large-scale* non-convexities, and nonconvexities due to randomness of the Pareto points, called *small-scale* nonconvexities. In particular, we show that even when the Pareto point cloud appears convex, at least $1/6$ of the Pareto-optimal points are missed by linear scalarization in dimension $T = 2$.

We present here some new results on the asymptotic convexity of Pareto fronts. Let $X_1, \ldots, X_n$ be *i.i.d.* random variables on $[0,1]^d$ with probability density function $f : [0,1]^d \to \mathbb{R}$ and set $\mathcal{X}_n = \{X_1, \ldots, X_n\}$. Then $(\mathcal{X}_n, \leqq)$ is a partially ordered set, where $\leqq$ is the usual partial order on $\mathbb{R}^d$ defined by

$$x \leqq y \iff x_i \leq y_i \text{ for all } i \in \{1, \ldots, d\}.$$

Let $\mathcal{F}_1, \mathcal{F}_2, \ldots$ denote the Pareto fronts associated with $\mathcal{X}_n$, and let $h_n : [0,1]^d \to \mathbb{R}$ denote the Pareto depth function defined by

$$h_n(x) = \max\{i \in \mathbb{N} : \mathcal{F}_i \leqq x\}, \tag{3.1}$$

where for simplicity we set $\mathcal{F}_0 = \{(-1, \ldots, -1)\}$, and we write $\mathcal{F}_i \leqq x$ if there exists $y \in \mathcal{F}_i$ such that $y \leqq x$. The function h_n is a (random) piecewise constant function that "counts" the Pareto fronts associated with $X_1, \ldots, X_n$.

Recall that a *chain* of length ℓ in $\mathcal{X}_n$ is a sequence $x^1, \ldots, x^\ell \in \mathcal{X}_n$ such that $x^1 \leqq x^2 \leqq \cdots \leqq x^\ell$. Define $u_n : [0,1]^d \to \mathbb{R}$ by

$$u_n(x) = \max\{\ell \in \mathbb{N} : \exists\, x^1 \leqq \cdots \leqq x^\ell \leqq x \text{ in } \mathcal{X}_n\}. \tag{3.2}$$

The function $u_n(x)$ is the length of the longest chain in $\mathcal{X}_n$ with maximal element $x_\ell \leqq x$. We have the following alternative characterization of h_n:

Proposition III.1. $h_n(x) = u_n(x)$ *with probability one for all* $x \in [0,1]^d$.

Proof. Suppose that $X_1, \ldots, X_n$ are distinct. Then each Pareto front consists of mutually incomparable points. Let $x \in [0,1]^d$, $r = u_n(x)$ and $k = h_n(x)$. By the definition of $u_n(x)$, there exists a chain $x_1 \leqq \cdots \leqq x_r$ in $\mathcal{X}_n$ such that $x_r \leqq x$. Noting that each x_i must belong to a different Pareto front, we see there are at least r fronts $\mathcal{F}_i$ such that $\mathcal{F}_i \leqq x$. Note also that for $j \leq i$, $\mathcal{F}_i \leqq x \implies \mathcal{F}_j \leqq x$. It follows that $\mathcal{F}_i \leqq x$ for $i = 1, \ldots, r$ and $u_n(x) = r \leq h_n(x)$. For the opposite inequality, by definition of $h_n(x)$ there exists $x_k \in \mathcal{F}_k$ such that $x_k \leqq x$. By the definition of $\mathcal{F}_k$, there exists $x_{k-1} \in \mathcal{F}_{k-1}$ such that $x_{k-1} \leqq x_k$. By repeating this argument, we can find $x_1, \ldots, x_k$ with $x_i \in \mathcal{F}_i$ and $x_1 \leqq \cdots \leqq x_k$, hence we have exhibited a chain of length k in $\mathcal{X}_n$ and it follows that $h_n(x) = k \leq u_n(x)$. The proof is completed by noting that $X_1, \ldots, X_n$ are distinct with probability one. $\square$

It is well-known [37] that Pareto methods outperform more traditional linear scalarization methods when the Pareto fronts are non-convex. In previous chapter, we show that the Pareto fronts *always* have microscopic non-convexities due to randomness, even when the Pareto fronts appear convex on a macroscopic scale. Microscopic non-convexities only account for minor performance differences between Pareto methods and linear scalarization. Macroscopic non-convexities induced by the geometry of the Pareto fronts on a macroscopic scale account for the major performance advantage of Pareto methods.

It is thus very important to characterize when the Pareto fronts are macroscopically convex. We therefore make the following definition:

Definition III.2. Given a density $f : [0,1]^d \to [0,\infty)$, we say that f yields *macroscopically convex* Pareto fronts if for $X_1, \ldots, X_n$ drawn *i.i.d.* from f we have that the almost sure limit $U(x) := \lim_{n\to\infty} n^{-\frac{1}{d}} h_n(x)$ exists for all x and $U : [0,1]^d \to \mathbb{R}$ is quasiconcave.

Recall that U is said to be *quasiconcave* if the super level sets

$$\{x \in [0,1]^d \,:\, U(x) \geq a\}$$

are convex for all $a \in \mathbb{R}$. Since the Pareto fronts are encoded into the level sets of h_n, the asymptotic shape of the Pareto fronts is dictated by the level sets of the function U from Definition III.2. Hence the fronts are asymptotically convex on a macroscopic scale exactly when U is quasiconcave, hence the definition.

We now give our main result, which is a partial characterization of densities f that yield macroscopically convex Pareto fronts.

Theorem III.3. *Let $f : [0,1]^d \to (0,\infty)$ be a continuous, log-concave, and separable density, i.e., $f(x) = f_1(x_1)\cdots f_d(x_d)$. Then f yields macroscopically convex Pareto fronts.*

Proof. We denote by $F : [0,1]^d \to \mathbb{R}$ the cumulative distribution function (CDF) associated with the density f, which is defined by

$$F(x) = \int_0^{x_1} \cdots \int_0^{x_d} f(y_1,\ldots,y_d)\, dy_1 \cdots dy_d. \tag{3.3}$$

Let $X_1,\ldots,X_n$ be *i.i.d.* with density f, and let h_n denote the associated Pareto depth function, and u_n the associated longest chain function. By [23, Theorem 1] we have that for every $x \in [0,1]^d$

$$n^{-\frac{1}{d}} u_n(x) \longrightarrow U(x) \text{ almost surely as } n \to \infty, \tag{3.4}$$

where $U(x) = c_d F(x)^{\frac{1}{d}}$, and c_d is a positive constant. In fact, the convergence is actually uniform on $[0,1]^d$ with probability one, but this is not necessary for the proof. For a general non-separable density, the continuum limit (3.4) still holds, but the limit $U(x)$ is not given by $c_d F(x)^{\frac{1}{d}}$ (it is instead the viscosity solution of a Hamilton-Jacobi equation), and the proof is quite involved (see [23]). Fortunately, for the case of a separable density the proof is straightforward, and so we include it here for completeness.

Define $\Phi : [0,1]^d \to [0,1]^d$ by

$$\Phi(x) = \left(\int_0^{x_1} f_1(t)\,dt, \ldots, \int_0^{x_d} f_d(t)\,dt \right).$$

Since f is continuous and strictly positive, $\Phi : [0,1]^d \to [0,1]^d$ is a C^1-diffeomorphism. Setting $Y_i = \Phi(X_i)$, we easily see that $Y_1, \ldots, Y_d$ are independent and uniformly distributed on $[0,1]^d$. It is also easy to see that Φ preserves the partial order $\leqq$, i.e.,

$$x \leqq z \iff \Phi(x) \leqq \Phi(z).$$

Let $x \in [0,1]^d$, set $y = \Phi(x)$, and define $\mathcal{Y}_n = \Phi(\mathcal{X}_n)$. By our above observations we have

$$u_n(x) = \max\{\ell \in \mathbb{N} : \exists\, y_1 \leqq \cdots \leqq y_\ell \leqq y \text{ in } \mathcal{Y}_n\}.$$

Let $i_1 < \cdots < i_N$ denote the indices of the random variables among $Y_1, \ldots, Y_n$ that are less than or equal to y and set $Z_k = Y_{i_k}$ for $k = 1, \ldots, N$. Note that N is binomially distributed with parameter $p := F(x)$ and that $u_n(x)$ is the length of the longest chain among N uniformly distributed points in the hypercube $\{z \in [0,1]^d : z \leqq y\}$. By [13, Remark 1] we have $N^{-\frac{1}{d}} u_n(x) \to c_d$ almost surely as $n \to \infty$ where $c_d < e$ are dimensional constants. Since $n^{-1}N \to p$ almost surely as $n \to \infty$, we have

$$n^{-\frac{1}{d}} u_n(x) = \left(n^{-\frac{1}{d}} N^{\frac{1}{d}} \right) N^{-\frac{1}{d}} u_n(x) \to c_d p^{\frac{1}{d}}$$

almost surely as $n \to \infty$. The proof of (3.4) is completed by recalling Proposition III.1.

In the context of Definition III.2, we have $U(x) = c_d F(x)^{\frac{1}{d}}$. Hence U is quasiconcave if and only if the cumulative distribution function F is quasiconcave. A sufficient condition for quasiconcavity of F is log-concavity of f [89], which completes the proof. $\square$

Theorem III.3 indicates that Pareto methods are largely redundant when f is a log-concave separable density. As demonstrated in the **Mediamill** [99] dataset (see Figure 3.2b), the distribution of points in Pareto space is not quasiconcave, and hence not log-concave, for the multiple-query retrieval problem. This helps explain the success of our Pareto methods.

It would be very interesting to extend Theorem III.3 to arbitrary non-separable density functions f. When f is non-separable there is no simple integral expression like (3.3) for U, and instead U is characterized as the viscosity solution of a Hamilton-Jacobi partial differential equation [23, Theorem 1]. This makes the non-separable case substantially more difficult, since U is no longer an integral functional of f. See [24] for a brief overview of previous work on a continuum limit for non-dominated sorting [22, 23] by Jeff Calder, the main collaborator of this work.

3.4 Multiple-query image retrieval

For most CBIR systems, images are preprocessed to extract low dimensional features instead of using pixel values directly for indexing and retrieval. Many feature extraction methods have been proposed in image processing and computer vision. In this work we use the famous SIFT and HoG feature extraction techniques and apply spatial pyramid matching to obtain bag-of-words type features for image representation. To avoid comparing every sample-query pair, we use an efficient manifold ranking algorithm proposed by [112].

3.4.1 Efficient manifold ranking (EMR)

The traditional manifold ranking problem [117] is as follows. Let $\mathcal{X} = \{X_1, \ldots, X_n\} \subset \mathbb{R}^m$ be a finite set of points, and let $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a metric on $\mathcal{X}$, such as Euclidean distance. Define a vector $y = [y_1, \ldots, y_n]$, in which $y_i = 1$ if X_i is a query and $y_i = 0$ otherwise. Let $r : \mathcal{X} \to \mathbb{R}$ denote the ranking function which assigns a ranking score r_i to each point X_i. The query is assigned a rank of 1 and all other samples will be assigned smaller ranks based on their distance to the query along the manifold underlying the data. To construct a graph on $\mathcal{X}$, first sort the pairwise distances between all samples in ascending order, and then add edges between points according to this order until a connected graph G is constructed. The edge weight between X_i and X_j on this graph is denoted by w_{ij}. If there is an edge between X_i and X_j, define the weight by $w_{ij} = exp[-d^2(X_i, X_j)/2\sigma^2]$, and if not, set $w_{ij} = 0$, and set $W = (w_{ij})_{ij} \in \mathbb{R}^{n\times n}$. In the manifold ranking method, the cost function associated with ranking vector r is defined by

$$O(r) = \sum_{i,j=1}^{n} w_{ij} |\frac{1}{\sqrt{D_{ii}}} r_i - \frac{1}{\sqrt{D_{jj}}} r_j|^2 + \mu \sum_{i=1}^{n} |r_i - y_i|^2$$

where D is a diagonal matrix with $D_{ii} = \sum_{j=1}^{n} w_{ij}$ and $\mu > 0$ is the regularization parameter. The first term in the cost function is a smoothness term that forces nearby points have similar ranking scores. The second term is a regularization term, which forces the query to have a rank close to 1, and all other samples to have ranks as close to 0 as possible. The ranking function r is the minimizer of $O(r)$ over all possible ranking functions.

This optimization problem can be solved in either of two ways: a direct approach and an iterative approach. The direct approach computes the exact solution via the closed form expression

$$r^* = (I_n - \alpha S)^{-1} y \tag{3.5}$$

where $\alpha = \frac{1}{1+\mu}$, I_n is an $n \times n$ identity matrix and $S = D^{-1/2} W D^{-1/2}$. The iterative method is better suited to large scale datasets. The ranking function r is computed by repeating the iteration scheme$r(t+1) = \alpha S r(t) + (1-\alpha) y$, until convergence. The direct approach requires an $n \times n$ matrix inversion and the iterative approach requires $n \times n$ memory and may converge to a local minimum. In addition, the complexity of constructing the graph G is $O(n^2 \log n)$. Sometimes a kNN graph is used for G, in which case the complexity is $O(kn^2)$. Neither case is suitable for large-scale problems.

In [112], an efficient manifold ranking algorithm is proposed. The authors introduce an anchor graph U to model the data and use the Nadaraya-Watson kernel to construct a weight matrix $Z \in \mathbb{R}^{d\times n}$ which measures the potential relationships between data points in $\mathcal{X}$ and anchors in U. For convenience, denote by z_i the i-th column of Z. The affinity matrix W is then designed to be $Z^T Z$. The final ranking function r can then be directly computed by

$$r^* = (I_n - H^T(HH^T - \frac{1}{\alpha}I_d)^{-1}H)y, \tag{3.6}$$

where $H = ZD^{-\frac{1}{2}}$ and D is a diagonal matrix with $D_{ii} = \sum_{j=1}^{n} z_i^T z_j$. This method requires inverting only a $d \times d$ matrix, in contrast to inverting the $n \times n$ matrix used in standard manifold ranking. When $d \ll n$, as occurs in large databases, the computational cost of manifold ranking is significantly reduced. The complexity of computing the ranking function with the EMR algorithm is $O(dn + d^3)$. In addition, EMR does not require storage of an $n \times n$ matrix.

Notice construction of the anchor graph and computation of the matrix inversion [112] can be implemented offline. For out-of-sample retrieval, Xu et al. [112] provides an efficient way to update the graph structure and do out-of-sample retrieval quickly.

3.4.2 Multiple-query case

In [112], prior knowledge about the relevance or confidence of each query can be incorporated into the EMR algorithm through the choice of the initial vector y. For example, in the multiple-query information retrieval problem, we may have queried, say, X_1, X_2 and X_3. We could set $y_1 = y_2 = y_3 = 1$ and $y_i = 0$ for $i \geq 4$ in the EMR algorithm. This instructs the EMR algorithm to assign high ranks to X_1, X_2, and X_3 in the final ranking r^*. It is easy to see from (3.5) or (3.6) that r^* is equal to the scalarization $r_1^* + r_2^* + r_3^*$ where r_i^*, $i = 1, 2, 3$, is the ranking function obtained when issuing each query individually. The main contribution of this work is to show that our Pareto front method can outperform this standard linear scalarization method. Our proposed algorithm is given below.

Given a set of queries $\{q_1, q_2, ..., q_T\}$, we apply EMR to compute the ranking functions $r_1^* \dots r_T^* \in \mathbb{R}^N$ corresponding to each query. We then define the dissimilarity vector $d_i \in \mathbb{R}_+^N$ between q_i and all samples by $d_i = \mathbf{1} - r_i^*$ where $\mathbf{1} = [1, \dots, 1] \in \mathbb{R}^N$. We then construct the Pareto fronts associated to $d_1, \ldots, d_T$ as described in Section 3.3.1. To return relevant samples to the user, we return samples according to their Pareto front number, i.e., we return points on $\mathcal{F}_1$ first, then $\mathcal{F}_2$, and so on, until sufficiently many images have been retrieved. Within the same front, we return points in a particular order, e.g., for $T = 2$, from the middle first. In the context of this work, the middle part of each front will contain samples related to *all* queries. Relevance feedback schemes can also be used with our algorithm to enhance retrieval performance. For example one could use images labeled as relevant by the user as new queries to generate new Pareto fronts.

3.5 Experimental study

We now present an experimental study comparing our Pareto front method against several state of the art multiple-query retrieval algorithms. Since our proposed algorithm was developed for the case where each query corresponds to a different semantic, we use multi-label datasets in our experiments. By multi-label, we mean that many images in the dataset belong to more than one class. This allows us to measure in a precise way our algorithm's ability to find images that are similar to all queries.

3.5.1 Multiple-query performance metrics

We evaluate the performance of our algorithm using normalized Discounted Cumulative Gain (nDCG) [58], which is standard in the retrieval community. The nDCG is defined in terms of a relevance score, which measures the relevance of a returned image to the query. In single query retrieval, a popular relevance score is the binary score, which is 1 if the retrieved image is related to the query and 0 otherwise. In the context of multiple-query retrieval, where a retrieved image may be related to each query in distinct ways, the binary score is an oversimplification of the notion of relevance. Therefore, we define a new relevance score for performance assessment of multiple-query multiclass retrieval algorithms. We call this relevance score multiple-query unique relevance (mq-uniq-rel).

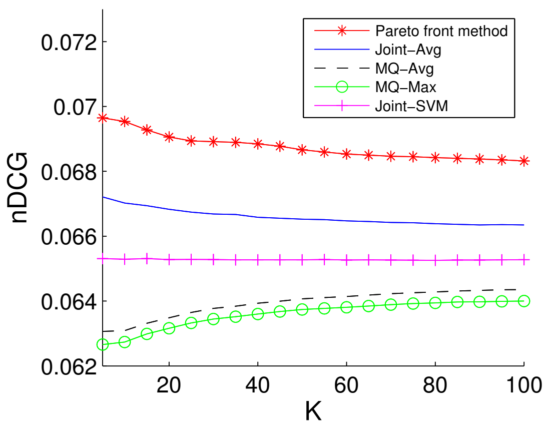
Roughly speaking, multiple-query unique relevance measures the fraction of query class labels that are covered by the retrieved object when the retrieved object is uniquely related to each query. When the retrieved object is not uniquely related to each query, the relevance score is set to zero. The importance of having unique relations to each query cannot be understated. For instance, in the two-query problem, if a retrieved image is related to one of the queries only through a feature common to both queries, then the image is effectively relevant to only one of the those queries in the sense it would likely be ranked highly by a single-query retrieval algorithm issuing only one of the queries. A more interesting and challenging problem, which is the focus of this work, is to find images that have different features in common with each query.

More formally, let us denote by C the total number of classes in the dataset, and let $\ell \in \{0,1\}^C$ be the binary label vector of a retrieved object X. Similarly, let y^i be the label vector of query q_i. Given two label vectors ℓ^1 and ℓ^2, we denote by the logical disjunction $\ell^1 \vee \ell^2$ (respectively, the logical conjunction $\ell^1 \wedge \ell^2$) the label vector whose j^{th} entry is given by $\max(\ell^1_j, \ell^2_j)$ (respectively, $\min(\ell^1_j, \ell^2_j)$). We denote by $|\ell|$ the number of non-zero entries in the label vector ℓ. Given a set of queries $\{q_1, \ldots, q_T\}$, we define the multiple-query unique relevance (mq-uniq-rel) of retrieved sample X having label ℓ to the query set by

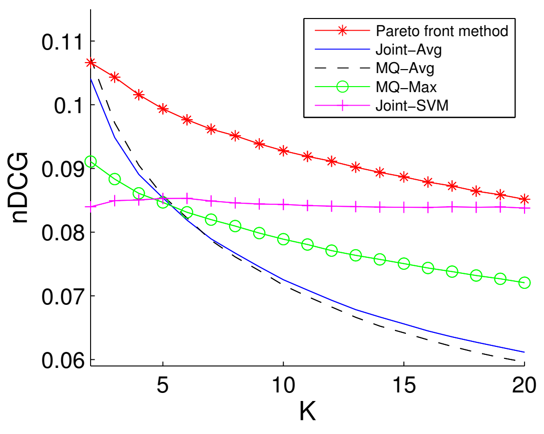
$$\text{mq-uniq-rel}(X) = \begin{cases} \dfrac{|\ell \wedge \beta|}{|\beta|}, & \text{if } \forall i, |\ell \wedge (y^i - \eta^i)| \neq 0, \\ 0, & \text{otherwise}, \end{cases} \tag{3.7}$$

where $\beta = y^1 \vee y^2 \vee \cdots \vee y^T$ is the disjunction of the label vectors corresponding to $q_1, \ldots, q_T$ and $\eta^i = \bigvee_{j \neq i} y^j \wedge y^i$. Multiple-query unique relevance measures the fraction of query classes that the retrieved object belongs to whenever the retrieved image has a unique relation to each query, and is set to zero otherwise.

For simplicity of notation, we denote by mq-uniq-rel$_i$ the multiple-query unique relevance of the i^{th} retrieved image. The Discounted Cumulative Gain (DCG) is then given



(a) **Mediamill** dataset



(b) **LAMDA** dataset

Figure 3.3: Comparison of PFM against state of the art multiple-query retrieval algorithms for **LAMDA** and **Mediamill** dataset with respect to the nDCG defined in (3.8). The proposed method significantly outperforms others on both datasets.

by

$$\text{DCG} = \text{mq-uniq-rel}_1 + \sum_{i=2}^{k} \frac{\text{mq-uniq-rel}_i^1}{\log_2(i)}, \tag{3.8}$$

The normalized DCG, or nDCG, is computed by normalizing the DCG by the best possible score which is $1 + \sum_{i=1}^{k} \frac{1}{\log_2(i)}$.

Note that, analogous to binary relevance score, we have $\text{mq-uniq-rel}_i = 1$ if and only if the label vector corresponding to the i^{th} retrieved object contains all labels from both queries and each query has at least one unique class. The difference is that multiple-query relevance is not a binary score, and instead assigns a range of values between zero and one, depending on how many of the query labels are covered by the retrieved object. Thus, mq-uniq-rel_i can be viewed as a generalization of the binary relevance score to the multiple-query setting in which the goal is to find objects uniquely related to all queries.

3.5.2 Evaluation on multi-label datasets

We evaluate our algorithm on the **Mediamill** video dataset [99], which has been widely used for benchmarking multi-class classifiers, and the **LAMDA** dataset, which is widely used in the retrieval community [118]. The **Mediamill** dataset consists of 29800 videos, and Snoek et al. [99] provide a manually annotated lexicon containing 101 semantic concepts and a large number of pre-computed low-level multimedia features. Each visual feature vector, X_i, is a 120-dimensional vector that corresponds to a specified key frame in the dataset. The feature extraction is based on the method of [105] and characterizes both global and local color-texture information of a single key frame, that is, an image. Each key frame is associated with a label vector $\ell \in \{0,1\}^C$, and each entry of ℓ corresponds to one of 101 semantic concepts. If X_i contains content related to the j^{th} semantic concept, then the j^{th} entry of ℓ_i is set to 1, and if not, it is set to 0.

The **LAMDA** database contains 2000 images, and each image has been labeled with one or more of the following five class labels: desert, mountains, sea, sunset, and trees. In total, 1543 images belong to exactly one class, 442 images belong to exactly two classes, and 15 images belong to three classes. Of the 442 two-class images, 106 images have the labels 'mountain' and 'sky', 172 images have the labels 'sunset' and 'sea', and the remaining label-pairs each have less than 40 image members, with some as few as 5. Zhou and Zhang [118] preprocessed the database and extracted from each image a 135 element feature vector, which we use to compute image similarities.

To evaluate the performance of our algorithm, we randomly generated 10000 query-pairs for **Mediamill** and 1000 for **LAMDA**, and ran our multiple-query retrieval algorithm on each pair. We computed the nDCG for different retrieval algorithms for each query-pair, and then computed the average nDCG over all query-pairs at different K. Since Efficient Manifold Ranking (EMR) uses a random initial condition for constructing the anchor graph, we furthermore run the entire experiment 20 times for **Mediamill** and 100 times for **LAMDA**, and computed the mean nDCG over all experiments. This ensures that we avoid any bias from a particular EMR model.

We show the mean nDCG for our algorithm and many state of the art multiple-query retrieval algorithms for **Mediamill** and **LAMDA** in Figures 3.3a and 3.3b, respectively. We compare against MQ-Avg, MQ-Max, Joint-Avg, and Joint-SVM [3]. Joint-Avg combines

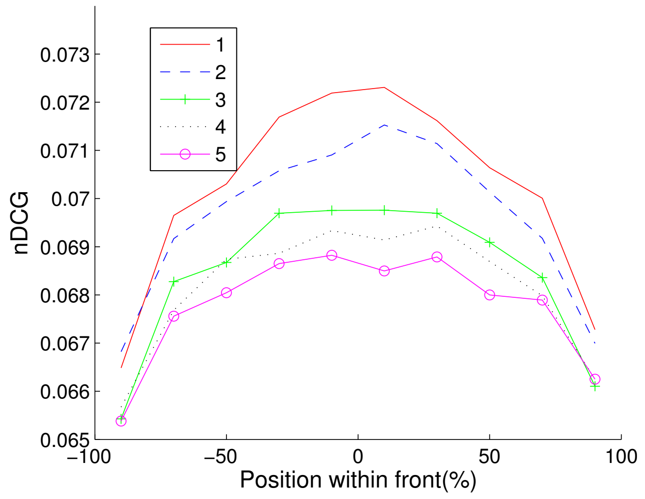


Figure 3.4: Average unique relevance scores at different regions along top five Pareto fronts. This plot validates our assumption that the middle part of first Pareto fronts contain more important samples that are uniquely related to both queries. Samples at deeper fronts and near the tails are less interesting.

histogram features of different queries to generate a new feature vector to use as an out-of-example query. A Joint-SVM classifier is used to rank each sample in response to each query. We note that Joint-SVM does not use EMR, while MQ-Avg and MQ-Max both do. Figures 3.3a and 3.3b show that our retrieval algorithm significantly outperforms all other algorithms.

We should note that when randomly generating query-pairs for **LAMDA**, we consider only the label-pairs (‘mountain’,‘sky’) and (‘sunset’,‘sea’), since these are the only label-pairs for which there are a significant number of corresponding two-class images. If there no multi-class images corresponding to a query-pair, then multiple-query retrieval is unnecessary; one can simply issue each query separately and take a union of the retrieved images.

To visualize advantages of the Pareto front method, we show in Figure 3.4 the multiple-query unique relevance scores for points within each of the first five Pareto fronts, plotted from one tail of the front, to the middle, to the other tail. The relevance scores within each front are interpolated to a fixed grid, and averaged over all query pairs to give the

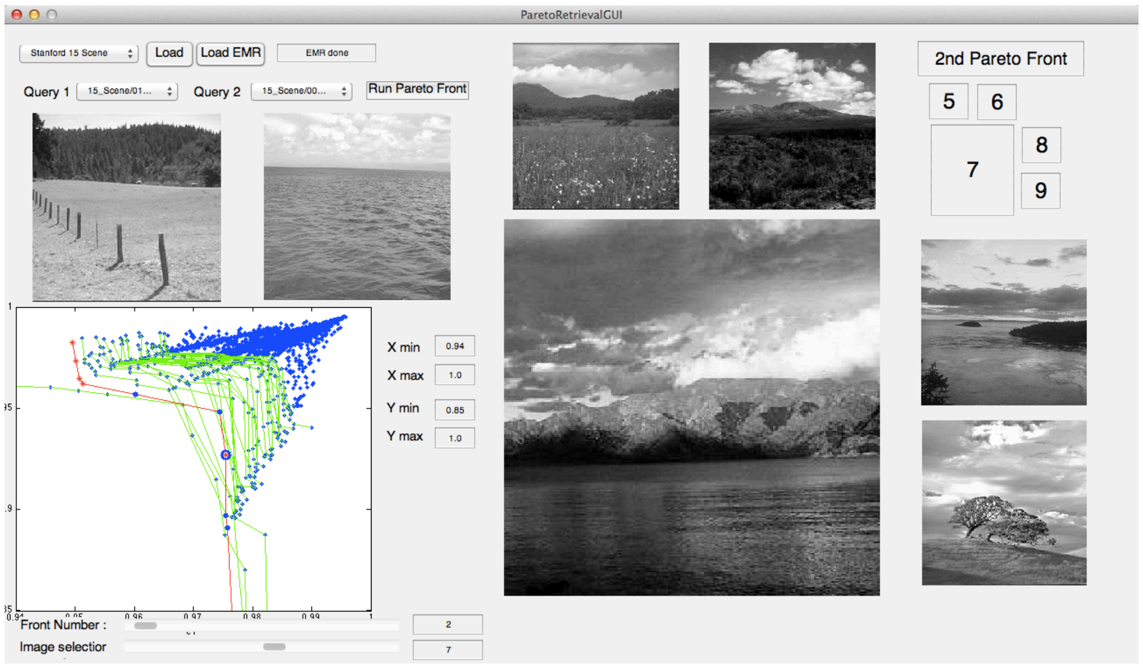


Figure 3.5: GUI screenshot. The two images on the upper left are two query images containing mountain and water, respectively. The largest image corresponds to the 7th Pareto point on the second Pareto front and the other four images correspond to adjacent points on the same front . Users can select the front and the specific relevance point using the two slider bars at the bottom.

curves in Figure 3.4. We used the **Mediamill** dataset to generate Figure 3.4; the results on **LAMDA** are similar. This result validates our assumption that the first front includes more important samples than deeper fronts, and that the middle of the Pareto fronts is fundamentally important for multiple-query retrieval.

We also note that the middle portions of fronts 2–5 contain samples with higher scores than those located at the tail of the first front. This phenomenon suggests a modified version of PFM which starts returning points around the middle of second front after returning only, say, d points from the first front. The same would hold for the second front and so on. We have carried out some experiments with such an algorithm, and have found that it can lead to even larger performance improvements, as suggested by Figure 3.4, for certain choices of d. However, it may be difficult to determine the best choice of d in advance since the label information is not available. Recall that label information is available only for testing and generating Figure 3.4 for validation. Therefore, we decided for simplicity to leave this simple modification of the algorithm to future work.

3.5.3 GUI for Pareto front retrieval

A GUI for a two-query image retrieval was implemented to illustrate the Pareto front method for image retrieval. Users can easily select samples from different fronts and visually explore the neighboring samples along the front. Samples corresponding to Pareto points at one tail of the front are similar to only one query, while samples corresponding to Pareto points at the middle part of front are similar to both queries. When the Pareto point cloud is non-convex, users can use our GUI to easily identify the Pareto points that cannot be obtained by any linear scalarization method. The screen shot of our GUI is shown in Figure 3.5. In this example, the two query images correspond to a mountain and an ocean respectively. One of the retrieved images corresponds to a point in the middle part of the second front that includes both a mountain and an ocean. The Matlab code of GUI can be downloaded from `http://tbayes.eecs.umich.edu/coolmark/pareto`.

3.6 Conclusions

We have presented a novel algorithm for content-based multiple-query image retrieval where the queries all correspond to different image semantics, and the goal is to find images related to *all* queries. This algorithm can retrieve samples which are not easily retrieved by other multiple-query retrieval algorithms and any linear scalarization method. We have presented theoretical results on asymptotic non-convexity of Pareto fronts that proves that the Pareto approach is better than using linear combinations of ranking results. Experimental studies on real-world datasets illustrate the advantages of the proposed Pareto front method. Note that one possible and interesting future work is to build a retrieval system that involves human-in-the-loop for defining semantic similarities between images with consistency. Properties of Pareto fronts constructed by these similarities are of interests.

CHAPTER IV

Social Collaborative Retrieval

4.1 Introduction

In previous two chapters we utilize Pareto fronts for two different types of machine learning problems that require combining different dissimilarity measures and different queries respectively. For those two problems one can apply trivial linear scalarization approaches to combine disparate information. In this chapter we focus on the case that disparate information can not be combined using any trivial approach. We are particularly interested in collaborative retrieval [109] which can be viewed as a blend of retrieval and collaborative filtering problems. We try to combine social networks and users' behavior information to improve the performance. These two types of information are totally different and can not be combined trivially. We first give a brief introduction of collaborative filtering in the following.

Collaborative filtering (CF) and related recommendation techniques, which aim to automatically predict the interests of users and make personal recommendations of music, movies, products, or other items, have been both intensively studied by researchers and successfully deployed in industry during the past decade [17, 73, 95, 101]. Recently, *Weston et al.* [109] proposed extending CF to a setting termed collaborative retrieval (CR), in which recommendations are made with respect to a particular *query context*; for instance, a user might be looking for music in a particular genre or movies similar to a recent favorite.

In these situations, otherwise accurate recommendations will become irrelevant. Similarly, a shopping website might want to deliver a list of recommended items to a user based on their browsing history. In this case the recently viewed pages act as a sort of query, and we would like recommendations that are specific both to the query and to the user. *Weston et al.* [109] proposed the first algorithm to solve CR problems, called latent collaborative retrieval (LCR).

However, several important issues remain. While it is well-known that CF models often contend with data sparsity problems since a large number of user-item pairs must be judged for compatibility based on a relatively small dataset, CR models suffer even more severely from sparsity, since the range of possible queries multiplies the number of judgments to be made. Viewed as a matrix completion problem, traditional CF requires filling out a user $\times$ item matrix, where each entry indicates the relevance of a specific item to a specific user. In the same light, CR can be seen as a tensor completion problem, where the goal is to fill out a (much larger) query $\times$ user $\times$ item tensor. Techniques like singular value decomposition and non-negative matrix factorization (NMF) [96], applied widely in CF, have recently begun to be extended to tensor models of this type [64, 102, 111, 115]; however, these methods typically do not accommodate the ranking losses used for CR, and sparsity remains a major concern. In this work, we propose to deal with data sparsity in CR by incorporating an additional (but often readily available) source of information: social networks.

In recent years social networking sites have become extremely popular, producing significant insights into the ways in which people connect themselves and interact. Information derived from these networks can be used to help address the sparsity problem faced by recommender systems, for instance by propagating information from a user's social connections to fill in gaps in the recommendation matrix. A variety of CF models augmented with social information have recently been proposed [26, 65, 67, 79, 81, 90]; these include state-of-the-art methods like Collaborative Topic Regression with social matrix factorization [90], which is based on LDA [11], and Probabilistic Matrix Factorization (PMF)

[79, 93]. There has also been interest in so-called *trust-aware* recommendation methods [9, 80, 83, 84, 87], which are similar in spirit but inherently limited compared with using real social networks [81]. However, social information has not yet been employed for collaborative retrieval, which arguably stands to benefit even more due to data sparsity. In this chapter we set out to fill this gap.

We propose an approach we call social collaborative retrieval (SCR), building on the latent collaborative retrieval (LCR) model of *Weston et al.* [109] by integrating social networking data. As in LCR, our algorithm sets out to optimize the top-ranked items retrieved for a given user and query, but we incorporate a regularization penalty that encourages a low dimensional embedding of each user to be similar to those of the user's nearest social neighbors. On a collaborative retrieval task using real-world artist ratings from the Last.fm music dataset, our proposed algorithm significantly outperforms LCR, as well as baseline CF methods based on non-negative matrix factorization and singular value decomposition, particularly when a smaller training set leads to increased data sparsity.

The rest of this chapter is organized as follows. In Section 4.2 we provide a brief overview of latent collaborative retrieval (LCR) [109], and then describe our proposed SCR algorithm in detail. Section 4.3 contains an empirical analysis of the Last.fm social networking dataset, and finally we present experimental results evaluating the performance of SCR in Section 4.4.

4.2 Collaborative retrieval

The goal of collaborative retrieval is to produce a ranked list of items that are of interest to a particular user given a particular query. While a natural approach to this problem might be to simply filter a set of unconstrained CF recommendations for the specified user using the query—or, conversely, to filter a set of generic search results for the query using the user's profile—these pipeline approaches fail to account for interactions between the query and the user. For instance, two users might have very different interpretations of the query “jazz”, despite having broadly similar preferences among artists. The idea of CR is to obtain more informative results through a unified approach.

We therefore formalize the problem by defining a single score function $f(q, u, a)$ to represent the relevance of a given item a with respect to both a query q and a user u. If we enumerate all users, queries, and items, we can think of this score function as specifying the values of a rating tensor $\mathbf{R} \in \mathbb{R}^{|\mathcal{Q}| \times |\mathcal{U}| \times |\mathcal{A}|}$, where $\mathcal{Q}$ is the set of all queries, $\mathcal{U}$ is the set of users, and $\mathcal{A}$ is the set of items. However, in practice we usually only care about the top k items retrieved (for some small constant k) for a given user and query, and our evaluation metrics will share this property as well. (We discuss specific error measures in Section 4.2.3.) Thus, learning a score function that can correctly *rank* items for a given user-query pair is more important than learning one which can correctly approximate the full set of entries in $\mathbf{R}$. The objectives that we use to learn the parameters of the score function will therefore involve a measure of error on such top-k ranked lists.

We next briefly review the existing latent collaborative retrieval model for this problem, and then introduce our model using social information. Finally, we discuss the optimization needed to learn the parameters of the model.

4.2.1 Latent collaborative retrieval

Latent collaborative retrieval (LCR) [109] was the first algorithm proposed to solve collaborative retrieval problems. The central idea is to embed users, queries, and items in a common n-dimensional space in which they can be compared using linear operations. (n is a hyperparameter that is typically tuned on a validation set.) Formally, LCR is parameterized by matrices $S \in \mathbb{R}^{n \times |\mathcal{Q}|}$, $V \in \mathbb{R}^{n \times |\mathcal{U}|}$, and $T \in \mathbb{R}^{n \times |\mathcal{A}|}$, which give the low-dimensional representations of queries, users, and items, respectively. Additionally, for each user u a matrix $U_u \in \mathbb{R}^{n \times n}$ encodes the user-specific relationship between queries and items. The scoring function f is then defined as

$$f(q, u, a) = S_q^\top U_u T_a + V_u^\top T_a \,, \tag{4.1}$$

where S_q is the column of S corresponding to query q, T_a is the column of T corresponding to item a, and V_u is the column of V corresponding to user u. Intuitively, the first term in Equation (4.1) measures the similarity between the query and the item under a linear transformation that is dependent on the user. The second term is independent of the query and can be viewed as a bias term which models user preferences for different items. Since for a given instance of a CR task the query and user are held fixed, there is no need for the scoring function to include a term like $S_q^\top \cdot V_u$, which would measure the compatibility of a user and a query. However, interactions between the user and the query that pertain to actual item recommendations can be expressed in the first term. If there are significant non-user-specific aspects of the compatibility between queries and items (i.e., a $S_q^\top \cdot T_a$ term), these can simply be absorbed into the first term and need not appear separately.

The parameters of the LCR scoring function are learned by optimizing a chosen error metric over a training set; we discuss some such metrics and other details in Section 4.2.3.

To aid in generalization, and to avoid the potentially prohibitive enumeration of queries, Equation (4.1) can be generalized using features. In this case $\Phi_Q(q)$, $\Phi_U(u)$ and $\Phi_A(a)$ are vector-valued feature maps for queries, users, and items, respectively, and S, T, and V are linear maps from feature vectors to the embedded n-dimensional space. The feature-based scoring function is given by

$$f(q, u, a) = \Phi_Q(q)^\top S^\top U_u T \Phi_A(a) + \Phi_U(u)^\top V^\top T \Phi_A(a) \ . \tag{4.2}$$

If the feature maps are simple characteristic vectors, with a unique feature for each query, user, and item, then we recover the simpler form of Equation (4.1). Features of this type can also be used for content-based recommendation models (see [109]). For our purposes, we simply note that this feature-based formulation can be easily extended to SCR, but for simplicity we focus on models of the type shown in Equation (4.1).

4.2.2 Social collaborative retrieval

In the real world, people often turn to their friends for recommendations of musics, movies, or products. Here, we apply this intuition to improve the performance of CR techniques on tasks where social information is available. Our approach, which we refer to as social collaborative retrieval (SCR), learns a scoring function using a combination of *behavioral* and *relational* error measures.

Behavioral measures encourage the model to respect implicit similarities between users, items, and queries that are revealed by the training data. For instance, the preferences of one user may be useful for recommending items to another user if the two users have expressed similar preferences in the past. This is the traditional mode of operation for collaborative filtering, as well as for CR.

Relational measures, on the other hand, take account of explicitly labeled connections that (hopefully) reveal underlying similarities. In this work, we employ a relational measure that encourages the scoring function to be smooth with respect to a social graph; that is, we assume that users who are social neighbors should, on average, have more similar preferences than those who are not. (We validate this assumption empirically in Section 4.3.) The hope is that this relational measure term provides complementary guidance to the system when little is known about the behavior of a user.

For simplicity, and to make the fairest comparisons later, we use the same parameterization of the scoring function as LCR (Equation (4.1)); we have n-dimensional representations of queries, users, and items in matrices S, V, and T, respectively, as well as user-specific transformations $U_u \in \mathbb{R}^{n\times n}$. We additionally assume that a social graph G is available, where $G(i,j)=1$ whenever users i and j are linked, and $G(i,j)=0$ otherwise. We will sometimes refer to users who are linked as "friends", though of course the social graph may encode relations with varying semantics. To bias the preferences of friends toward each other, we introduce a social error measure

$$\mathrm{err}_{\text{social}}(V,G) = \sum_{i,j,G(i,j)=1} \|\sigma(V_i^T V_j) - 1\|^2 \ , \tag{4.3}$$

where $\sigma(\cdot)$ is the sigmoid function

$$\sigma(x) = \frac{1}{1+e^{-cx}} \tag{4.4}$$

and c is a hyperparameter. This measure can be seen as a regularization penalty that is minimized when friends have identical, high-norm representations in V. Notice that we do not penalize similarity among non-friends, since users may have similar tastes even though they are not friends. Importantly, although we encourage friends to have similar representations V_u, we do not introduce such regularization for U_u matrices, as this would tend to force friends to always receive the same results. Intuitively, we expect that friends are likely to have similar taste in items, but we allow each their own particular querying "style".

Combining the relational measure in Equation (4.3) with a behavioral measure $\mathrm{err}_{\text{behavior}}$ that depends on the scoring function f and the training set X yields the SCR learning objective to be minimized:

$$\mathrm{err}_{\text{behavior}}(f,X) + w_s \mathrm{err}_{\text{social}}(V,G) \ , \tag{4.5}$$

where w_s is a regularization hyperparameter. In the following subsection we will discuss choices for $\mathrm{err}_{\text{behavior}}$, as well as optimization techniques used to find the parameters in practice.

Similarity-based error measures related to Equation (4.3) have been proposed by others, typically based not on a social graph but instead on measured similarities between users. For example, the measured Pearson correlation of item ratings can be used as a similarity measure $Sim(i,j)$ between users i and j, and this can be incorporated as in [81]:

$\sum_{i=1}^{|\mathcal{U}|} \left\| V_i - \frac{\sum_{j,G(i,j)=1} Sim(i,j) \times V_j}{\sum_{j,G(i,j)=1} Sim(i,j)} \right\|^2$. Through the chapter we denote by $\| \cdot \|$ the L2-norm of a vector. However, accurately estimating similarities from data is often unreliable due to sparsity, especially in the CR setting. Moreover, such measures make it difficult to easily recommend items to newer users; without a long history of ratings, we cannot know which established users they are similar to. On the other hand, SCR requires an external source of information in the form of a social graph. Social networks are increasingly ubiquitous, and, since they are by nature centralized, can often be reliable even when extensive training data for a specific CR task is not yet available.

SCR can be viewed as a blend of social networking, collaborative filtering, and information retrieval. As a side benefit, in addition to providing improved recommendations for users under particular query contexts, SCR can potentially be used in the inverse to recommend new social links between users with similar preferences. In this way SCR can strengthen the social network and improve its own predictions in the future.

4.2.3 Learning

The goal of SCR learning is to (efficiently) find parameters S, V, T, and U_u that minimize the objective in Equation (4.5). In this section we describe the formal learning setup, the specific behavioral measures used in our experiments, and the algorithm used to optimize the model parameters.

We assume we are given a training set X containing N training examples:

$$X = \{(q_i, u_i, a_i, w_i)\}_{i=1,2,\ldots,N} \ , \tag{4.6}$$

where $q_i \in \mathcal{Q}$ is a query, $u_i \in \mathcal{U}$ is a user, $a_i \in \mathcal{A}$ is an item, and $w_i \in \mathbb{R}_{>0}$ is a measure of relevance for the item a_i given the user u_i and the query q_i. Importantly, we assume that the weights w_i always have a positive connotation; that is, triples (q, u, a) that do not appear in the training set implicitly have a weight of zero, and are therefore dispreferred to triples that do appear. For instance, in our experiments, w_i will be derived from the number of times a user listens to a particular musical artist.

The behavioral part of the objective, which measures the compatibility of the scoring function f (defined by the model parameters) with the training set X, can take a variety of forms depending on the setting. As noted earlier, we will focus on top-k ranking losses that optimize the most important aspects of the model, rather than, say, filling out all entries of the tensor $\mathbf{R}$.

Following *Weston et al.* [109] we define the vector $\bar{f}(q,u)$, which contains predictions for all items in the database given query q and user u. The a^{th} entry of $\bar{f}(q,u)$, denoted $\bar{f}_a(q,u)$, is equal to $f(q,u,a)$.

With this notation, we can define the *Weighted Approximate-Rank Pairwise (WARP) Loss*, introduced in [108]:

$$\mathrm{err}_{\mathrm{WARP}}(f,X) = \sum_{i=1}^{N} L\left(\mathrm{rank}_{a_i}\left(\bar{f}(q_i,u_i)\right)\right) \ . \tag{4.7}$$

Here $\mathrm{rank}_{a_i}\left(\bar{f}(q_i,u_i)\right)$ is the margin-based rank of item a_i,

$$\mathrm{rank}_{a_i}\left(\bar{f}(q_i,u_i)\right) = \sum_{b \neq a_i} \mathbb{I}[1 + \bar{f}_b(q_i,u_i) \geq \bar{f}_{a_i}(q_i,u_i)] \ , \tag{4.8}$$

where $\mathbb{I}[\cdot]$ is the indicator function, and L is a loss function:

$$L(k) = \sum_{i=1}^{k} \alpha_i \tag{4.9}$$

$$\alpha_1 \geq \alpha_2 \geq \alpha_3 \geq \cdots \geq 0 \ , \tag{4.10}$$

with the values of α_r determining the additional penalty for each successive reduction in rank. We choose $\alpha_r = 1/r$, which gives a smooth weighting over positions while assigning large weights to top positions and rapidly decaying weights to lower positions.

Intuitively, the WARP loss prefers that the item a_i is always ranked highest. For each training example $i = 1, \ldots, N$, the positive item a_i is compared pairwise with all other (negative) items in the database. If the score of another item is less than a margin of one from the score of a_i, this pair incurs a cost. The WARP loss determines this cost based on the corresponding items' ranking positions and the choice of α parameters.

We use the WARP loss in our experiments for comparison with prior work. However, in our setting it ignores the relevance scores w_i that are part of the training set; this can be inefficient, since the optimization cannot focus on the most important training examples. We thus propose a modified behavioral measure that we refer to as the weighted WARP loss:

$$\mathrm{err}_{\mathrm{weighted}}(f, X) = \sum_{i=1}^{N} w_i L\left(\mathrm{rank}_{a_i}\left(\bar{f}(q_i, u_i)\right)\right) , \tag{4.11}$$

where rank and L are defined as before. In our results, we refer to models learned under this loss as SCR-weighted, while models trained under the standard WARP loss are referred to simply as SCR. We will derive the optimization procedure for the weighted WARP loss, since the standard WARP loss is a special case.

To minimize Equation (4.5), we employ stochastic gradient descent (SGD), choosing at each iteration a single training instance i uniformly at random from the training set. We then seek to optimize the objective for this single example; that is, we minimize

$$w_i L\left(\mathrm{rank}_{a_i}\left(\bar{f}(q_i, u_i)\right)\right) + w_s \sum_{v, G(u_i, v)=1} \|1 - \sigma(V_{u_i}^{\top} V_v)\|^2 . \tag{4.12}$$

Because it is expensive to compute the exact rank of an item a_i when the total number of items is very large, the optimization procedure includes a sampling process at each step, as introduced in [108]. For the training sample i chosen at the current iteration, negative items b are sampled uniformly at random from $\mathcal{A}$ until a pairwise violation is found—that is, until $1 + f(q_i, u_i, b) > f(q_i, u_i, a_i)$. If K steps are required to find such a b, then the rank of a_i can be approximated as

$$rank_{a_i}(\bar{f}(q_i, u_i)) \approx \left\lfloor \frac{|\mathcal{A}| - 1}{K} \right\rfloor , \tag{4.13}$$

where $\lfloor\cdot\rfloor$ is the floor function.

In each iteration of stochastic gradient descent, at most $|\mathcal{A}|-1$ sampling steps are required, since the right hand side of Equation (4.13) is constant (zero) for $K \geq |\mathcal{A}|-1$. Therefore at most $1+\min\left(\frac{|\mathcal{A}|-1}{rank_{a_i}(\bar{f}(q_i,u_i))}, |\mathcal{A}|-1\right)$ scores $f(q_i,u_i,b)$ must be computed. The worst case is when a_i has rank one; however, in our experiments most items do not have small ranks, particularly during the early stages of training when the model still has large errors. As a result, rank approximation dramatically speeds up SGD in practice. Note that SGD can also be parallelized to take advantage of multiple processors [31, 119].

Following *Weston et al.* [108], the single-instance objective becomes

$$w_i L\left(\left\lfloor\frac{|\mathcal{A}|-1}{K}\right\rfloor\right)\cdot|1-f(q_i,u_i,a_i)+f(q_i,u_i,b)| \\ + w_s \sum_{v,G(u_i,v)=1} \|1-\sigma(V_{u_i}^\top V_v)\|^2\,. \quad (4.14)$$

Rewriting Equation (4.14), we have

$$C_i(1+(S_{q_i}^\top U_{u_i}+V_{u_i}^\top)(T_b-T_{a_i})) \\ + w_s \sum_{v,G(u_i,v)=1} \|1-\sigma(V_{u_i}^\top V_v)\|^2\,, \quad (4.15)$$

where $C_i = w_i \cdot L(\lfloor\frac{|\mathcal{A}|-1}{K}\rfloor)$. To speed up each gradient step, we only update the variables associated with the current violation pair; that is, we only update S_{q_i}, V_{u_i}, T_{a_i}, T_b, and U_{u_i}. (In particular, we do not update the representations of u_i's friends V_v for $G(u_i,v)=1$.) Now we can simply take the gradient of (4.15) to perform an update.

The update for user u_i's low-dimensional embedding is

$$V_{u_i} \leftarrow V_{u_i} - \eta\left(C_i(T_b-T_{a_i}) \\ + w_s \sum_{v,G(u_i,v)=1}\left(-2c\sigma(V_{u_i}^T V_v)\left(1-\sigma(V_{u_i}^T V_v)\right)^2\right)\cdot V_j\right)\,, \quad (4.16)$$

or equivalently

$$V_{u_i} \leftarrow V_{u_i} - \eta C_i(T_b - T_{a_i}) + w'_s \sum_{v, G(u_i, v)=1} b_v \cdot V_v \ , \tag{4.17}$$

where $b_v = 2c\sigma(V_{u_i}^T V_v)(1 - \sigma(V_{u_i}^T V_v))^2 > 0$, $w'_s = \eta w_s$, and η is a learning rate parameter. (Recall that c is a hyperparameter for the sigmoid function.) Thus at each gradient step, the user's low-dimensional embedding is updated toward the weighted mean of his or her friends' embeddings.

Similarly, we have the following updates for the remaining parameters:

$$S_{q_i} \leftarrow S_{q_i} - \eta \left(C_i(U_{u_i}(T_b - T_{a_i})) \right) \tag{4.18}$$

$$T_{a_i} \leftarrow T_{a_i} - \eta \left(C_i(-U_{u_i}^T S_{q_i} - V_{u_i}) \right) \tag{4.19}$$

$$T_b \leftarrow T_b - \eta \left(C_i(U_{u_i}^T S_{q_i} + V_{u_i}) \right) \tag{4.20}$$

$$U_{u_i} \leftarrow U_{u_i} - \eta \left(C_i \left(S_{q_i}(T_b - T_{a_i})^T \right) \right) \ . \tag{4.21}$$

Finally, we constrain the parameters using

$$\|S_i\| \leq L_S \ , \quad i \in \{1, \ldots, |Q|\} \tag{4.22}$$

$$\|T_i\| \leq L_T \ , \quad i \in \{1, \ldots, |A|\} \tag{4.23}$$

$$\|V_i\| \leq L_V \ , \quad i \in \{1, \ldots, |\mathcal{U}|\} \tag{4.24}$$

and project the parameters back on to the constraints at each step. These constraints can be viewed as additional regularizers that bound the lengths of vectors in Q, A, and U with hyperparameters L_S, L_T, and L_V.

Once the SCR stochastic gradient training procedure converges, we take the learned parameters and apply them to predict scores for unseen test triples using Equation (4.1).

4.3 Social data analysis

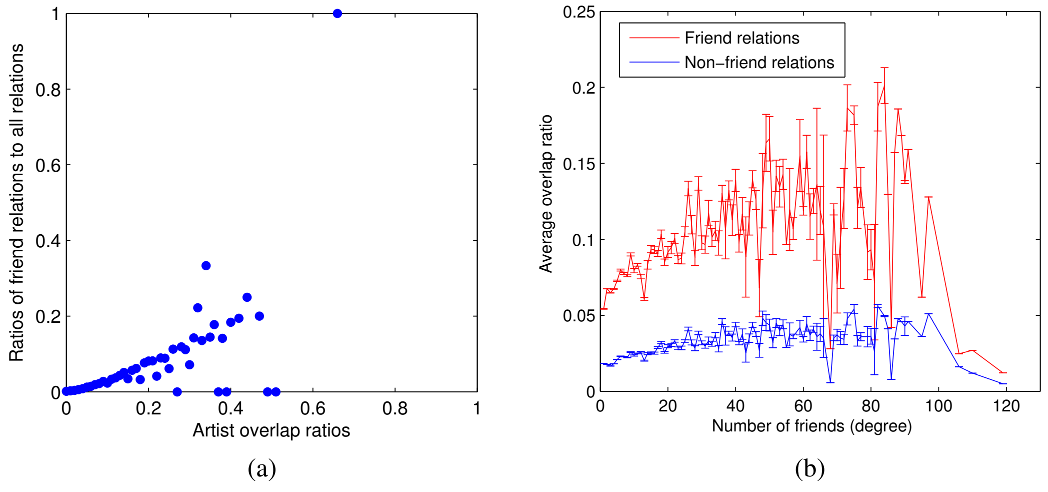


Figure 4.1: (a) For user pairs having listened artist overlap ratios within the same interval, the proportion of friend relations among these pairs is shown. (b) Average artist overlap ratios among friends (red) and non-friends (blue) for users having different numbers of friends (degree).

Before showing results that compare our proposed approach to existing state-of-the-art methods, we first experimentally validate the fundamental assumption that friends, on average, have more in common than non-friends.

4.3.1 Last.fm dataset

In our experiments we use a real-word music dataset obtained from the Last.fm music website in May of 2011, *hetrec2011-lastfm-2k* [25]. In this dataset each user is described by his or her listen counts for all musical artists in the database (items, in CR parlance), as well as any artist tags (queries) that might have been provided by each user.

While the data contain more than ten thousand unique tags across all users, the vast majority of tags are used by only one user. Typically these tags appear to be personal "notes" rather than widely used genre distinctions. To remove this noisy information, we throw out tags that are less frequent, keeping only the top 30 most common tags. These tags were all used by at least 165 unique users, and generally correspond to genres of music; for example, the top 5 most popular tags are *"rock"*, *"pop"*, *"alternative"*, *"electronic"* and *"indie"*. The Last.fm dataset contains listening histories for 1892 users and 17632 music artists. A social graph is also included; on average each user has 13.44 friends.

4.3.2 Shared musical interests

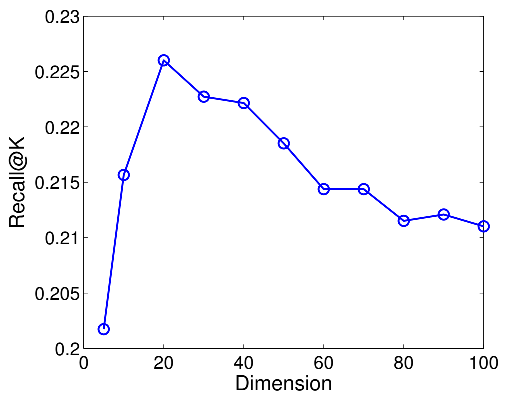
Do friends share more preferences than non-friends? This is a key question for our approach. If the answer is no, it may not be useful to include social networks as a predictor variable in recommendation systems. To estimate the similarity between two users' tastes for music, we compute the *listened artists overlap ratio*, defined as

$$\mathrm{Sim}(i,j) = \frac{|A_i \cap A_j|}{|A_i \cup A_j|} \in [0,1] \quad \forall\, i,j\ , \tag{4.25}$$

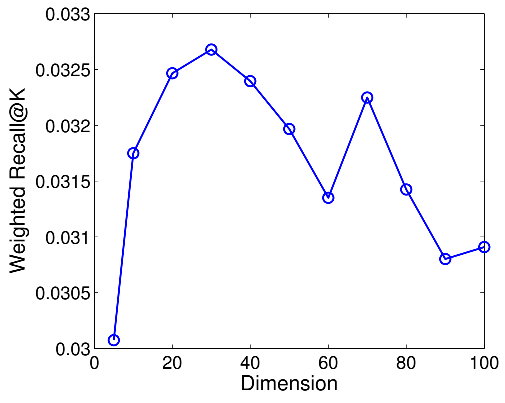
where A_i is the set of artists listened to by user i.

We compute these overlap ratios for all $\binom{|\mathcal{U}|}{2}$ user pairs. We then divide the range $[0,1]$ of possible ratios evenly into 100 intervals, and calculate the fraction of the user pairs falling in each interval that are friends in the social graph. Intuitively, we hope that users with greater similarity are more likely to be friends. The result is shown in Figure 4.1 (a). The percentage of realized *friend* relations increases sharply as the artist overlap ratio increases.

To reinforce this analysis, we also compute the average similarity between each user i and his or her friends, as well as the average similarity between user i and all other non-friend users, denoting the two numbers as β^i_{friend} and $\beta^i_{non-friend}$. Figure 4.1 (b) shows the values of β^i_{friend} and $\beta^i_{non-friend}$ averaged over users grouped by the number of friends they have in the social graph. We can see that, regardless of how well-connected a user is, on average he or she has more in common with friends than with non-friends; moreover, the size of this effect increases for users with more friends. Overall, these analyses support our assumptions regarding the use of social networks for recommendation and retrieval tasks on this dataset.



(a)



(b)

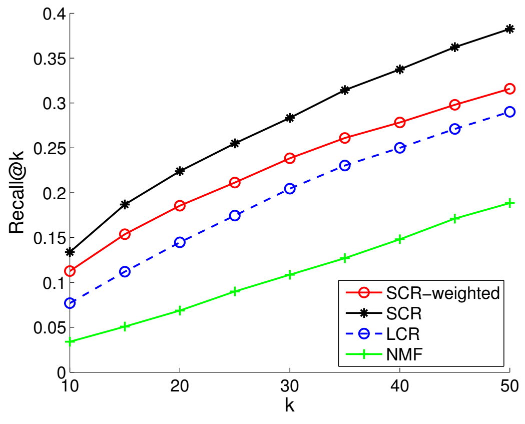
Figure 4.2: (a) Recall@30 of SCR-weighted using different embedding dimensions. (b) Weighted Recall@30 of SCR-weighted using different embedding dimensions.

4.4 Experiments

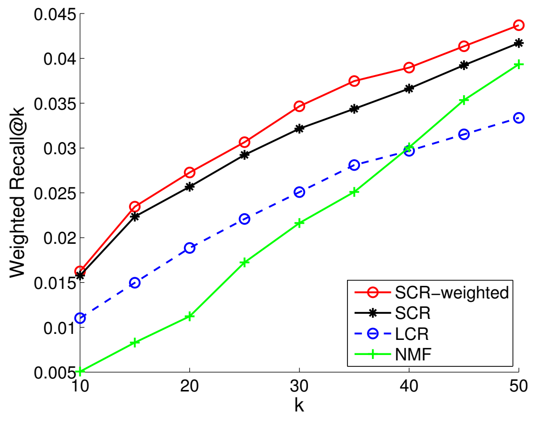
We next compare the SCR approach with other state-of-the-art algorithms. Recall that, for the Last.fm dataset described in the previous section, a *query* $\times$ *user* $\times$ *item* tensor entry corresponds to a *genre* $\times$ *user* $\times$ *artist* triple, where genres are obtained from the set of filtered user tags. We preprocess the data set in two ways to obtain listening counts for each such triple/tensor entry. First, if a user u has listened to an artist a and assigned multiple genres, for example *rock* and *indie*, then u's listening counts for a are evenly distributed to triples $(\textit{rock}, u, a)$ and $(\textit{indie}, u, a)$. If the user has not assigned any genre to an artist, the genres of a are those assigned to a by other users, and the listening count of a is distributed to each triple according to how frequently the genre appears. If no user has ever assigned any genre to a, the genres of a are defined as the genres used by u for any artist, and the listening count is again prorated to the triples.

Second, since we are interested in ranking artists given a particular user and query, we normalize listening counts of triples having the same u and q so that their weights sum to 1. In the end we have 389,405 data points of the form (q, u, a, w), where w is the normalized listen count of artist a by user u in genre q.

Since our main goal is to show how social information can help compensate for data



(a)

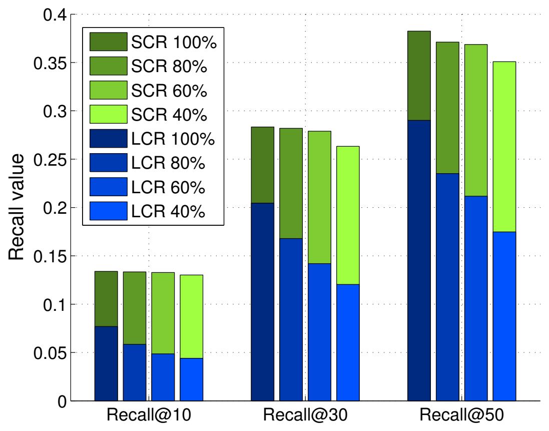


(b)

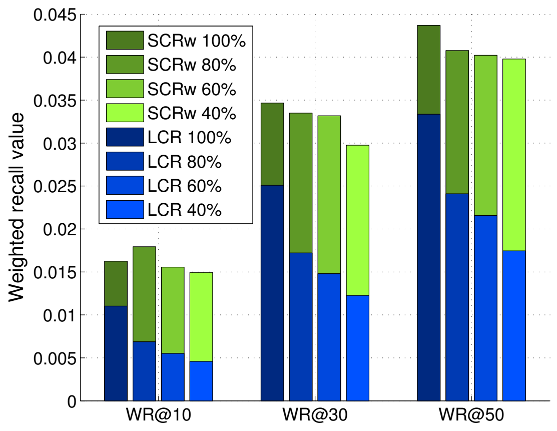
Figure 4.3: (a) Recall at different values of k. (b) Weighted recall at different values of k.

sparsity in a collaborative retrieval task, we identify a series of subsets of the Last.fm data that correspond to increasingly less compact social networks. We use a standard implementation of hierarchical clustering on the complete social adjacency matrix to select subsets of users that exhibit significant internal social structure; the number of users in these sets varies from 200 to 1000 (see Table 4.2). For each user set, the corresponding set of items contains all artists listened to by one or more of the selected users. In this way, the number of artists grows organically with the number of users. As in Section 4.3, we use the 30 most frequent genre tags as our query set.

The resulting datasets are referred to as Compact-*lastfm-N*, where N denotes the number of users in the dataset. Their statistics are shown in Table 4.2. By construction, users in the smaller datasets have higher average numbers of within-set friends. This means that the smaller sets are more tightly connected, which may make them more amenable to social regularization. Conversely, the larger datasets are sparser and may be more representative of large-scale social networks. Note that the *density* of social links falls with the number of users even if the average number of within-set friends stays constant, thus the largest datasets are in fact quite a bit more sparse (relatively speaking) than the smallest ones. We will show how the performance of SCR changes as these qualities are varied.



(a)



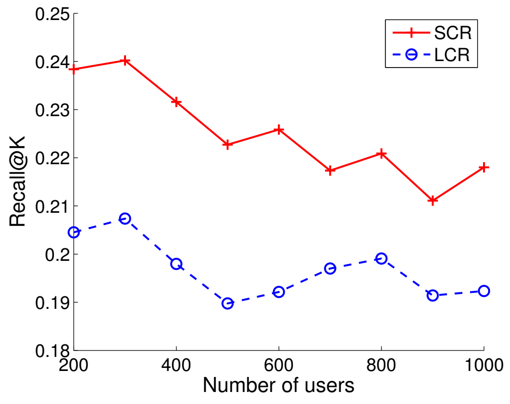
(b)

Figure 4.4: (a) Recall for training data of different reduced sizes. (b) Weighted Recall for training data of different reduced sizes.

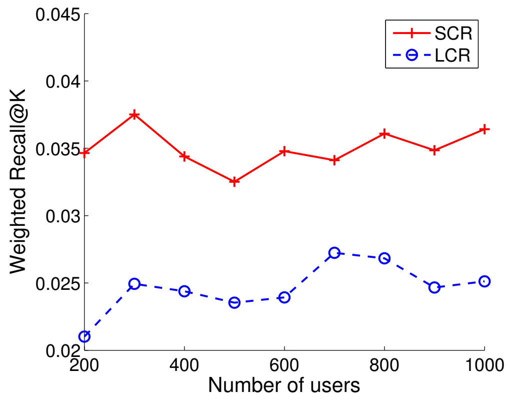
4.4.1 Evaluation

We compare SCR with state-of-the-art algorithms used for collaborative retrieval as well as traditional collaborative filtering. Popular matrix factorization methods such as SVD and NMF are often used for collaborative filtering; these methods optimize the deviation of the rating matrix from entries supplied in the training set. However, standard SVD and NMF techniques are not directly applicable to tensors. Instead, we perform NMF on the $|\mathcal{Q}|$ different user $\times$ artist matrices to compute the rank of a among all artists given q and u. We also compare to latent collaborative retrieval (LCR).

The dimension of the embeddings for all methods is chosen to be 30; as shown in Figure 4.2, this choice yields approximately optimal performance for SCR-weighted; however, the results are not qualitatively different for other choices of embedding dimension. The hyperparameters w_s, η, and c, along with constraint parameters L_S, L_T and L_V, are chosen separately for each method (as applicable) using a validation set (see below). Since matrix factorization approaches are not specially designed for tensors and typically show worse performance than LCR [109], we only present results for NMF, which performed the best. We use the NMF implementation from `http://www.csie.ntu.edu.tw/~cjlin/`



(a)



(b)

Figure 4.5: (a) Recall@30 for datasets of different sizes. (b) Weighted Recall@30 for datasets of different sizes.

`nmf/`. For each experiment, 60% of the samples are used for training (or less; see below), 20% are used for validation, and 20% are used for testing.

To evaluate the performance of each algorithm, for a given test sample (q, u, a, w) we first compute $f(q, u, i)$ for $i = 1, \ldots, |\mathcal{A}|$ and sort the artists in descending order of score. We then measure recall@k, which is 1 if artist a appears in the top k, and 0 otherwise, and report the mean *recall@*k over the whole test dataset. As a secondary measure we report *weighted recall@*k, which is the relevance score w if artist a appears in the top k, and 0 otherwise. Mean weighted recall@k thus not only measures how many triples are ranked in the top k, but the quality of these test triples.

4.4.2 Results

We begin with results for the smallest datasat, Compact-*lastfm-200*, which is small enough to be practical for all methods. The resulting recall@k and weighted recall@k for different k are shown in Figure 4.3. SCR (weighted or unweighted) outperforms the baselines on this top k ranking problem; note that SCR-weighted outperforms SCR under the weighted recall criterion, which makes sense since it incorporates relevance scores in the loss function.

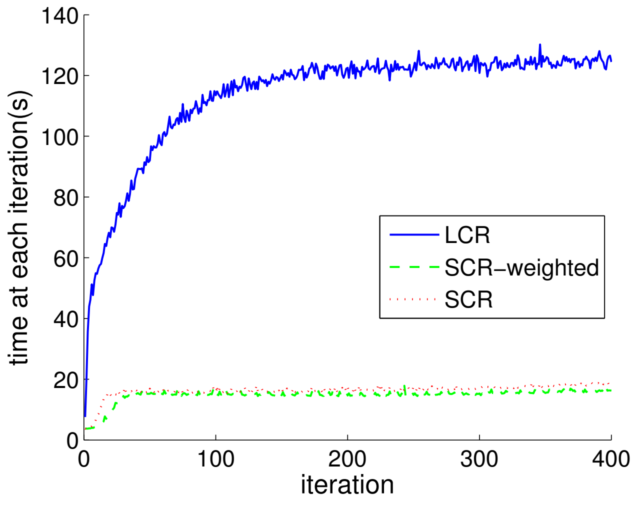


Figure 4.6: Runtime required for each training iteration of LCR and two versions of SCR on the Compact-*lastfm-200* dataset.

Since we expect social information to be particularly useful when data are sparse, we also show results of recall@k and weighted recall@k for different amounts of training data $(100\%, 80\%, 60\%$ and 40% of the total training data$)$ in Figure 4.4. Notice that the performance gap between SCR-weighted and LCR becomes larger as the number of available training examples is reduced, suggesting that our proposed algorithm can be especially useful for predicting the interests of new users or infrequent users when social network information is available.

Moving to the larger datasets, computation time increasingly becomes an issue for the matrix factorization approaches, so we focus on only two algorithms: SCR-weighted and LCR. Figure 4.5 shows recall results for all of the compact datasets. Note that the performance gap between SCR and LCR narrows slightly but remains significant even as the size of system becomes larger and the density of social links decreases. It may be counterintuitive that performance decreases (at least for unweighted recall) as the size of the dataset grows; however, since the number of artists grows with the number of users, the prediction problem is becoming more difficult at the same time. These results suggest that, while a dense social network may improve the relative performnce of SCR, it retains significant

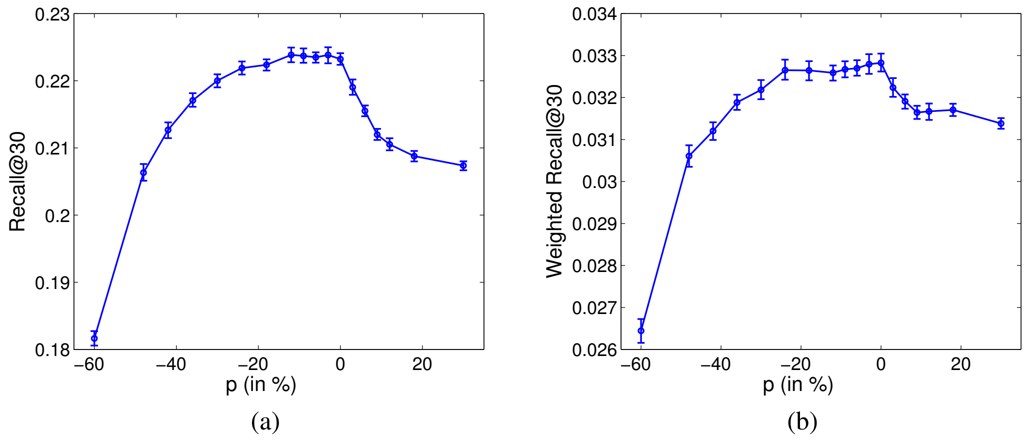


Figure 4.7: (a) Recall@30 for different levels of random friend addition/deletion noise with 95% confidence intervals. (b) Weighted Recall@30 for different levels of random friend addition/deletion noise with 95% confidence intervals.

advantages even in larger, sparser settings.

Finally, we show in Figure 4.6 the runtimes for each stochastic gradient training iteration of SCR and LCR on the Compact-*lastfm-200* dataset; SCR is dramatically faster, despite using essentially similar optimization techniques. This is because the runtime is dominated by the sampling procedure used to estimate the rank function. LCR promotes the observed items to high positions quickly, thus making subsequent iterations quite slow. SCR, on the other hand, has additional regularization that appears to prevent this situation. Combined with the performance improvements discussed above, this is a significant practical advantage.

4.4.3 Sensitivity

Since real-world social networks are subject to various sources of noise, we test the robustness of SCR on a series of datasets in which edges in the social graph have been randomly added or removed. Specifically, in these experiments we begin with the Compact-*lastfm-500* dataset, and then, for a user with F friends, randomly add Fp friend relations for a noise parameter p. If p is negative, then we instead remove Fp edges from the original

Table 4.1: Last.fm dataset statistics

Dataset	users	items (artists)	queries (tags)	samples
lastfm-2k	1892	17632	11946	186479
Compact-*lastfm-200*	200	2392	30	29850
Compact-*lastfm-300*	300	3299	30	44318
Compact-*lastfm-400*	400	4091	30	58098
Compact-*lastfm-500*	500	4928	30	72125
Compact-*lastfm-600*	600	5765	30	85522
Compact-*lastfm-700*	700	6454	30	98367
Compact-*lastfm-800*	800	7071	30	111062
Compact-*lastfm-900*	900	7782	30	124005
Compact-*lastfm-1000*	1000	8431	30	137518

Table 4.2: Last.fm dataset statistics (continue)

Dataset	data sparsity (%)	average # of friends
lastfm-2k	99.9999	13.44
Compact-*lastfm-200*	99.7920	28.54
Compact-*lastfm-300*	99.8507	29.79
Compact-*lastfm-400*	99.8817	29.10
Compact-*lastfm-500*	99.9024	27.81
Compact-*lastfm-600*	99.9176	26.32
Compact-*lastfm-700*	99.9274	24.90
Compact-*lastfm-800*	99.9346	23.57
Compact-*lastfm-900*	99.9410	22.23
Compact-*lastfm-1000*	99.9456	21.01

graph. Figure 4.7 shows the results of SCR learning using these noisy datasets for various values of p, averaged over 25 random trials. Table 4.3 shows the average number of friends added or removed per user for a given p.

The results reveal an interesting, asymmetric behavior. When friends are added at random ($p > 0$), performance begins to drop quickly, presumably due to the fact that non-friends can have significantly different preferences, as shown in Section 4.3. Luckily, the creation of spurious non-friend edges in the social graph seems relatively unlikely in the real world, where links must typically be confirmed by both parties.

On the other hand, removing edges ($p < 0$) seems to have a relatively small impact

Table 4.3: Average number of friends added/removed per user.

p	Friends removed	p	Friends added
-0.03	1.59	0	0
-0.06	3.27	0.03	1.62
-0.09	4.84	0.06	3.35
-0.12	6.27	0.09	5.04
-0.18	9.10	0.12	6.68
-0.24	11.71	0.18	10.00
-0.30	14.25	0.30	16.67
-0.36	16.33		
-0.42	18.38		
-0.48	20.18		
-0.60	23.25		

on performance unless a significant proportion of the links are removed. This may be because friends are often linked by multiple short paths though other mutual friends, thus the removal of a single link only slightly diminishes connectivity. Moreover, groups of users linked in cliques tend to influence each other strongly, and such cliques cannot be broken up by removing only a few edges. This type of noise, though presumably common, has only a limited impact when using social networks for collaborative retrieval.

To visualize these patterns, we select a random subset of 200 users and plot the original social graph as well as the social graphs obtained for different values of p in Figure 4.8. When $p = -0.3$, the basic structure of the network is still visible, and almost every user is still connected to all of his or her original friends through short paths in the reduced graph. However, when $p = -0.6$, the structure has begun to break down, and many former friends are now totally disconnected. In this case the performance of SCR degrades approximately to the level of LCR. When p is positive, we see a different kind of degredation, where many new edges connect together subgroups that were originally only sparsely connected.

4.5 Conclusions and future work

We proposed SCR, a new model blending social networking, information retrieval, and collaborative filtering. SCR uses a social graph to improve performance on collaborative

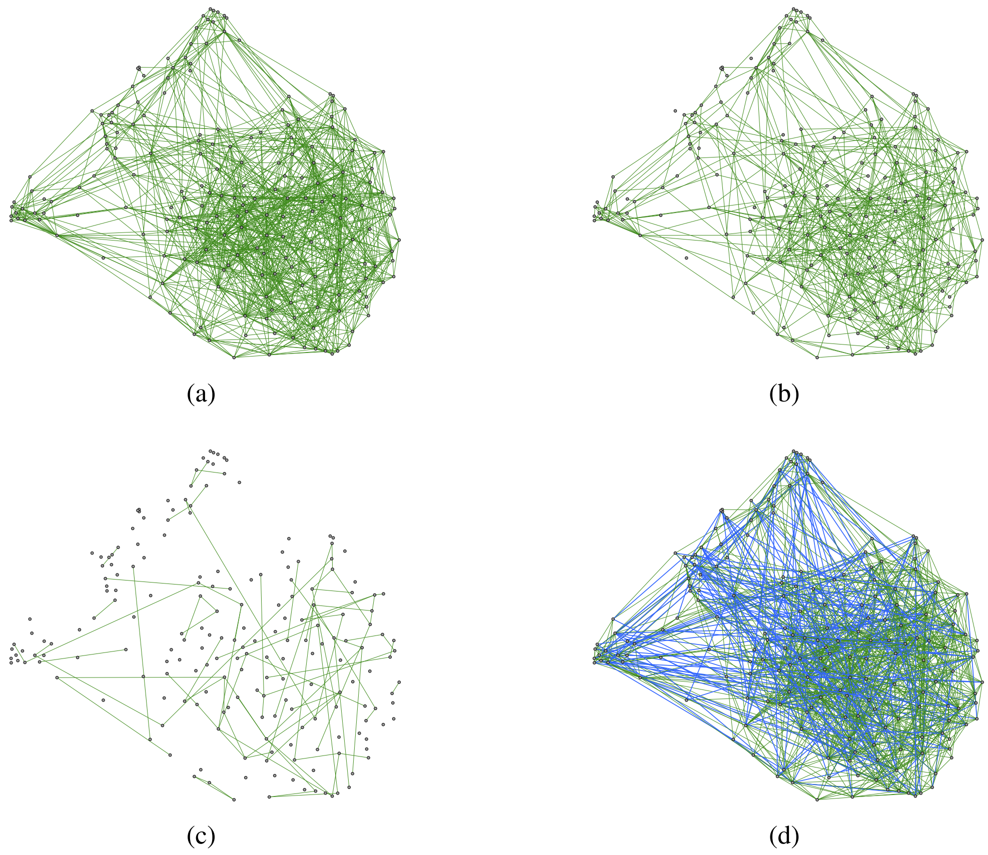


Figure 4.8: Visualizations of 200-user social graphs with different levels of random friend addition/deletion noise. (a) Original social graph ($p = 0$). (b) Reduced social graph ($p = -0.30$). (c) Reduced social graph ($p = -0.60$). (d) Augmented social graph ($p = 0.09$), with added edges shown in blue. The arrangement of the vertices was generated using the ForceAtlas2 algorithm.

retrieval problems, which we believe are increasingly important in practice, outperforming state-of-the-art CR and CF approaches. We also showed that users tend to share interests with friends. Going forward we hope to develop a two-pass version of the SCR algorithm that helps predict interest commonalities between friends, and can be used to prune out edges on the social graph that may work against achieving good performance.

CHAPTER V

Graphical Lasso Fusion across Multiple Time-sclaes

5.1 Introduction

The last information fusion problem we consider in this dissertation is inverse covariance (precision) estimation for multivariate time series. We aim to combine information at different time scales and offsets to improve the estimation. Like the problem in Chapter IV, different information in this chapter can not be combined using trivial linear scalarization approach as in Chapter II and III. A general framework for joint estimation of multiple precision matrices is proposed in this chapter. We first briefly give an introduction of correlation and partial correlation networks and then introduce our proposed fused method for combining information at different time scales and offsets.

Correlation networks that expose graphical structure in multivariate time series—for example, in historical stock prices—have been widely applied to analyze the dynamics of financial markets. *Partial* correlation networks have become especially popular in recent years, since they can more easily answer questions about which equities act as the primary drivers for the rest of the market. In these networks, the edge weight between two equities represents the partial correlation of their prices; an edge is omitted if they are conditionally uncorrelated given all of the other data. A partial correlation matrix can be viewed as a normalized version of inverse covariance matrix; thus for Gaussian graphical models, the network defined in this way is minimal.

Through this type of analysis we can identify equities that are the most influential, in the sense that a small subset can often explain large parts of the market [66]. Moreover, by estimating correlation networks from a moving window of recent data we can try to predict future market dynamics. For example, previous studies have observed that the dominant influences on specific equities are usually persistent across time [66]. Observations of this kind can even provide deeper understanding of market collapses and other anomalous phenomena.

In this chapter we propose a general fused graphical lasso objective for estimating the network structure of data containing multiple time series, under the assumption that the variables follow a multivariate Gaussian distribution. The key to our approach is that we jointly learn networks at multiple distinct time scales and at different offsets; since these networks all involve the same set of variables, we can regularize them towards each other under the assumption that they tend to exhibit similar structure. This will tend to be true, for example, if the dynamics of the relevant real-world processes change smoothly over time. Our approach estimates precision matrices that balance likelihood and per-network sparsity with a penalty that encourages precision matrices at different time scales and offsets to share a common sparsity pattern.

As we demonstrate empirically, estimates over short time scales are responsive to rapidly changing dynamics, but are often noisy due to a relatively small number of available samples. On the other hand, longer time scales allow the integration of many more data points, reducing variance, but smooth out potentially important short-term behaviors. By combining these approaches using the general fused graphical lasso, we aim to obtain the best properties from each. We show that this technique allows us to estimate partial correlation networks more robustly and accurately on both synthetic data and real-world time series.

5.2 Related Work

Although partial correlation networks for financial analysis have become widespread [43, 66], only recently has estimation of partial correlation or inverse covariance matrices has been studied in the context of practical applications [29, 35, 55, 91]. Several general estimation methods have been proposed for empirically estimating inverse covariance matrices (also called precision matrices) for Gaussian graphical models in such a way that the resulting estimate is sparse [6, 40, 40, 54, 72, 74]. Graphical lasso, which generates a sparse precision matrix by maximizing log likelihood with an L-1 or lasso penalty [40, 104], is one of the most popular approaches.

When several datasets are available whose precision matrices are expected to be similar, the estimation of these matrices can be porformed jointly, as in [33, 46, 113]. In [33], the authors propose to add fused lasso penalties [104] and group lasso penalties [41, 85] between all pairs of precision matrices. In [113], the precision matrices are ordered and the authors propose to add fused lasso penalties for adjacent precision matrices. All these approaches, which are related to our work, try to make multiple precision matrices estimates share a common sparsity pattern.

In [33], the authors applied an *alternating directions method of multipliers* (ADMM) method [16] to solve the joint graphical lasso with fused lasso penalties. In this work we apply a similar procedure to solve our optimization problem. Since we use fused lasso penalties, ADMM requires a subroutine for solving fused lasso signal approximator (FLSA) problems. Several methods have been proposed to solve FLSA problems [39, 51, 75]. We extend a path algorithm by [51] to solve a more general type of FLSA problem.

5.3 Structure Fusion across Different Time-scales

The goal of fusing information is to improve joint estimates of precision matrices at different time scales and offsets. Suppose the time series are not stationary, that is, the relationships between different variables vary over time. One could divide the whole time series into different years and divide the time series within the same year intro four different quarters. If one would like to estimate the precision matrices at each quarter of a year, a simple approach is to estimate the precision matrix using each quarter's data. However, the number of observations might not be enough for a good estimate when the number of variables is large. Alternately, one could use the whole year's data to overcome this problem, but that will smooth out local information at different quarters. Although we assume that true precision varies along time, the key assumption of this work is that different precision matrices at different times or different time scales share similar structure.

We first briefly introduce some notation used throughout the chapter. We let K denote the total number of different precision matrices we want to estimate and let Σ_k^{-1} and Σ_k denote the true precision and covariance, respectively. These K different precision or covariance matrices represent relationships of time series at different times and at different time scales. The estimate of Σ_k^{-1} is denoted as $\Theta^{(k)}$, $k = 1, \cdots, K$. An example of hierarchical structure of these precision estimates is shown in Figure 5.1. In this example, $\Theta_{1,1}$ is the precision estimate of multiple time series through the whole year and $\Theta_{2,i}$, $i = 1, \cdots, 4$, are precision estimates for different quarters. Finally, $\Theta_{3,i}, i = 1, \cdots, 12$, correspond to twelve different months. Our approach is to formulate a convex optimization problem which seeks better estimates for different times and different time scales by introducing convex penalties that encourage them to share similar sparsity patterns. For convenience, we denote by $G(\Theta, E)$ the graph which comprises a set $\{\Theta\}$ of precision matrices together with a set E of edges that relate them. $G(\Theta, E)$ is undirected in this work. Since a precision matrix is sometimes viewed as a representation of a graph describing the relationships between variables, $G(\Theta, E)$ can be viewed as a *graph of graphs*. For convenience, we will index all different precision matrices using $k = 1, \cdots, K$ and entries of matrices using $i = 1, \cdots, p$, $j = 1, \cdots, p$ where p is the number of time series. In this example, $\{\Theta_{1,1}, \Theta_{2,1}, \cdots, \Theta_{2,4}, \Theta_{3,1}, \cdots, \Theta_{3,12}\}$ are indexed by $\{\Theta^{(1)}, \cdots, \Theta^{(17)}\}$, respectively. Notice that in this example, the graph structure is a tree. Different structures are possible for different applications. We have tested different structures such as a line structure connecting precision matrices across time offsets and a line structures connecting matrices across time scales. The proposed tree structure outperformed simple line structures.

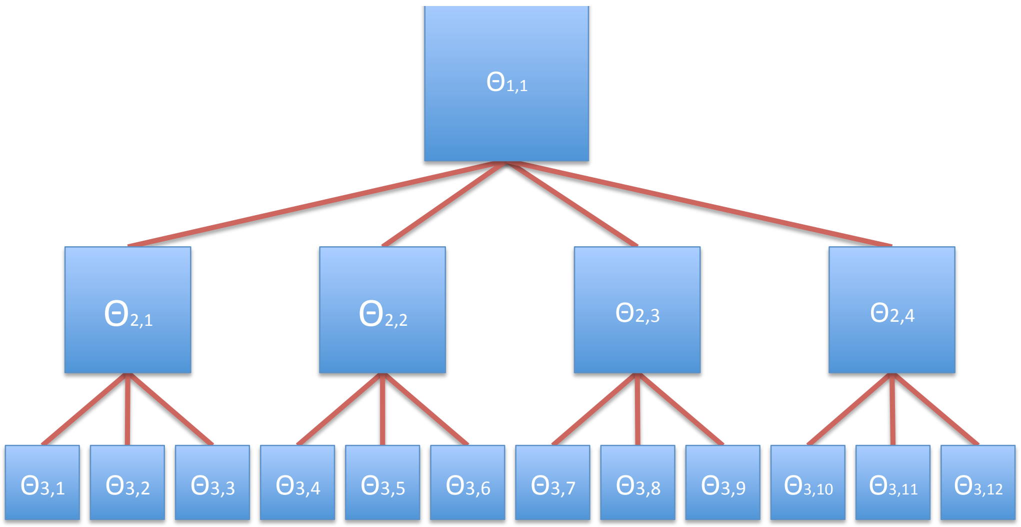


Figure 5.1: An example of graph structure for multiple precisions at different times and different time scales. At each node of the graph a precision matrix $\Theta_{i,j}$ governs the joint distribution of the measurements at that node. When there exists an edge between two nodes, the precision matrices at these nodes are constrained to be similar.

5.3.1 Graphical Lasso and Joint Graphical Lasso

Before introducing our general framework for fusing precision matrices, a brief introduction to Graphical Lasso and Joint Graphical Lasso (JGL) is presented in this section.

Assume we have K datasets, $Y^{(1)}, \cdots, Y^{(K)}$, with $K \geq 2$. $Y^{(k)}$ is a $n_k \times p$ matrix consisting of n_k observations with measurements on p features that are common across different datasets. Assume these $\sum_{k=1}^{K} n_k$ observations are all independent and the observations within each dataset are identically distributed following a Gaussian distribution : $y_i^{(k)} \sim N(\mu_k, \Sigma_k^{-1})$, $i = 1, \cdots, n_k$. Denote by $S^{(k)}$ the empirical covariance matrix for $Y^{(k)}$. In a Gaussian graphical model, a natural way to estimate these inverse covariance matrices or precision matrices $\{\Sigma_k^{-1}\}$, is by a maximum likelihood approach. The joint log-likelihood for all data can then be represented in the following form (up to a constant):

$$\ell(\{\Theta\}) = \frac{1}{2}\sum_{k=1}^{K} n_k \left(\log \det\Theta^{(k)} - \text{trace}(S^{(k)}\Theta^{(k)})\right). \tag{5.1}$$

Without any additional penalized term, the maximum likelihood estimates which maximize the above objective function are $(S^{(1)})^{-1}, \cdots, (S^{(K)})^{-1}$. These maximum likelihood estimates of precision matrices are usually not satisfactory and the inverse of a covariance matrix sometimes does not exist or may have very large variance. In the recent past, several methods have been proposed to estimate Σ^{-1} in such a way that the resulting estimate is sparse in the high-dimensional case ($p \gg n$). Instead of maximizing Eq.(5.1), many of them solve the following optimization problem (Eq.5.2) which maximizes a penalized log-likelihood for each dataset or class.

$$\text{maximize}_{\boldsymbol{\Theta}} n_k \left(\log \det \Theta^{(k)} - \text{trace}(S^{(k)}\Theta^{(k)})\right) - \lambda\|\Theta\|_1. \tag{5.2}$$

The solution of this maximization problem is often referred to as the *graphical lasso*(GL) [40].

[33] propose to use the joint graphical lasso (JGL) for inverse covariance estimation across multiple classes. The authors use the graphical lasso approach for estimation of multiple precision matrices with the assumption that these precision matrices share a similar sparsity pattern. Each precision matrix corresponds to the structure of a graphical model for a class. The overall penalized log-likelihood optimization problem is then written as the following:

$$\text{maximize}_{\boldsymbol{\Theta}} \sum_{k=1}^{K} n_k(\log \det \Theta^{(k)} - \text{trace}(S^{(k)}\Theta^{(k)})) - P(\{\boldsymbol{\Theta}\}), \tag{5.3}$$

where $P(\{\boldsymbol{\Theta}\})$ is a convex penalty function . In [33], the authors propose to use a penalty function P which encourages the estimates of precision matrices $\hat{\Theta}^1, \ldots, \hat{\Theta}^K$ to be sparse and share certain characteristics. For example, they propose *fused graphical lasso* (FGL) and *group graphical lasso* (FGL) which are solutions to the problem (5.3) with penalties shown in (5.4) and (5.5) respectively.

$$P_{\text{FGL}}(\{\mathbf{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\Theta_{ij}^{(k)}| + \lambda_2 \sum_{k<k'} \sum_{i,j} |\Theta_{ij}^{(k)} - \Theta_{ij}^{(k')}|. \tag{5.4}$$

$$P_{\text{GGL}}(\{\mathbf{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\Theta_{ij}^{(k)}| + \lambda_2 \sum_{i \neq j} \sqrt{\sum_{k=1}^{K} (\Theta_{ij}^{(k)})^2}. \tag{5.5}$$

Both penalties encourage similarity across the K estimated precision matrices.

5.3.2 General Fused Graphical Lasso (GFGL) for multiple time series at different time scales and offsets

To combine information from different time scales, we propose to use Gaussian graphical models to model multiple time series at different time scales. For example, given a time series x_t, $t = 1, \ldots, T$, one could define the m-*block average* process $y_{t'}$ on indices $t' = 1, 2, \ldots, \lfloor \frac{T}{m} \rfloor$ by

$$y_{t'} = \frac{1}{m} \sum_{i=1}^{m} x_{(t'-1)m+i}.$$

The y values are the averages of non-overlapping groups of m consecutive x values. In this example, one could obtain weekly and monthly time series by choosing m to be 7 and 30. For convenience, we denote x^ℓ as a time series x at time scale ℓ. Multiple time series at different time scales can be denoted as $\mathbf{X}^\ell$ and we follow the notations used in the previous subsection.

Motivated by joint graphical lasso, we propose to use the following objective function:

$$\text{maximize}_{\mathbf{\Theta}} \sum_{\ell=1}^{L} w_\ell (\text{log det } \Theta^{(\ell)} - \text{trace}(S^{(\ell)} \Theta^{(\ell)})) - P(\{\mathbf{\Theta}\}), \tag{5.6}$$

where w_ℓ is the weight for time scale ℓ and $P(\{\mathbf{\Theta}\})$ is the modified penalty function:

$$P(\{\mathbf{\Theta}\}) = \lambda_1 \sum_{\ell=1}^{L} \sum_{i \neq j} |\Theta_{ij}^{(\ell)}| + \lambda_2 \sum_{\ell=1}^{L-1} \sum_{i,j} |\Theta_{ij}^{(\ell)} - \Theta_{ij}^{(\ell+1)}|. \tag{5.7}$$

In other words, each graphical model for a specific time scale is only constrained with those models for neighboring time scales. Recall G is defined as the graph of precision matrices. In this example, G is a simple line structure of precision matrices across time scales.

Instead of modeling Gaussian graphical models only at different time scales, another interesting and useful approach is to construct graphical models across time. Recall that one could apply a moving window approach to estimate the relationship of variables for successive short time periods, for example, one precision estimate for each month. Since there are not enough samples within each window, the partial correlation network or precision matrix estimate will be noisy and have high variance. Although we assume that the time series are not stationary, we expect that naturally they share some common characteristics. We expect that they are similar but not identical and the differences between these graphical models may be of interest for discovering the dynamics of networks. In this situation, we could apply the same penalties described in Eq.(5.7) but now different $\Theta^{(\ell)}$ correspond to different graphical models at different times.

Different setups of the graph G for multiple graphical models are also possible. The main idea is to combine information at different time scales or at different time offsets to make the estimation of either partial correlation networks or inverse covariance matrices more accurate and robust. Our approach can be viewed as a fusion approach and other methods such as graphical lasso without fused penalties at non-fusion approaches. Recall that a more interesting and hierarchical tree structure has been mentioned at the beginning of this section and shown in Figure 5.1. In this example, neighboring precision estimates across times share sparsity via precision estimates at upper scales. In the context of this work, JGL used a complete graph in which there are edges between all pairs of precision estimates.

In addition to using a general graph instead of a complete graph in JGL, we also introduce edge weights on G such that one can apply different fusion strengths for different pairs of precision matrices. Take the structure in Figure 5.1, for example; if one would like to use the estimates $\Theta_{1,1}, \Theta_{2,4}$ or $\Theta_{3,12}$ for predictive purposes in the next year, different edge weights can be applied such that $e_{\Theta_{1,1},\Theta_{2,i}} < e_{\Theta_{1,1},\Theta_{2,j}}, i < j$ to make the estimates more accurate.

The overall generalized penalty function can be written as follows:

$$P(\{\mathbf{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\Theta_{ij}^{(k)}| + \lambda_2 \sum_{p<q,(p,q)\in E} e_{pq} \sum_{i,j} |\Theta_{ij}^{(p)} - \Theta_{ij}^{(q)}|, \tag{5.8}$$

where E is the edge set of graph G. From now on, we refer to the solution of the optimization problem (5.6) with generalized penalty function (5.8) as *general fused graphical Lasso* (GFGL).

5.3.3 ADMM Algorithm for GFGL

To solve the problem in (5.6), we use an *alternating directions method of multipliers* (ADMM) algorithm as in JGL[33]. For more details about ADMM, its convergence rate and recent applications, we refer the reader to [16].

The problem can be rewritten as

$$\underset{\mathbf{\Theta},\mathbf{Z}}{\text{minimize}} \; -\sum_{k=1}^{K} w_k(\log\det \Theta^{(k)} - \text{trace}(\mathbf{S}^{(k)}\Theta^{(k)})) + P(\{\mathbf{Z}\}) \tag{5.9}$$

$$\begin{aligned} \text{subject to} \quad & \Theta^{(k)} > 0, \forall k = 1, \cdots, K \\ & \Theta^{(k)} = Z^{(k)}, \forall k = 1, \cdots, K \end{aligned} \tag{5.10}$$

where $\Theta^{(k)} \succ 0$ means $\Theta^{(k)}$ is positive-definite and $\{\mathbf{Z}\} = \{Z^{(1)}, \cdots, Z^{(K)}\}$. The scaled augmented Lagrangian [16] for this problem is then given by

$$L_\rho(\{\mathbf{\Theta}\}, \{\mathbf{Z}\}, \{\mathbf{U}\}) = -\sum_{k=1}^{K} w_k(\log \det \Theta^{(k)} - \text{trace}(S^{(k)}\Theta^{(k)})) \\ + P(\{Z\}) + \frac{\rho}{2}\sum_{k=1}^{K} \|\Theta^{(k)} - Z^{(k)} + U^{(k)}\|, \quad (5.11)$$

where $\{\mathbf{U}\} = \{U^{(k)}, \cdots, U^{(K)}\}$ are dual variables and ρ is a scalar greater than 0. An ADMM algorithm solves this problem by iterating the following three simple steps at i-th iteration:

(i)$\{\mathbf{\Theta}_{(i)}\} \leftarrow$ arg $\min_{\{\mathbf{\Theta}\}}\{L_\rho(\{\mathbf{\Theta}\}, \{\mathbf{Z}_{(i-1)}\}, \{\mathbf{U}_{(i-1)}\})\}$.

(ii)$\{\mathbf{Z}_{(i)}\} \leftarrow$ arg $\min_{\{\mathbf{Z}\}}\{L_\rho(\{\mathbf{\Theta}_{(i)}\}, \{\mathbf{Z}\}, \{\mathbf{U}_{(i-1)}\})\}$.

(iii)$\{\mathbf{U}_{(i)}\} \leftarrow \{\mathbf{U}_{(i-1)}\} + \{\mathbf{\Theta}_{(i)}\} - \{\mathbf{Z}_{(i)}\}$.

Step (iii) is trivial while step (i) and (ii) can be solved as follows. Let $\mathbf{VDV}^T$ be the eigendecomposition of $S^{(k)} - \rho Z^{(k)}_{(i-1)}/w_k + \rho U^{(k)}_{(i-1)}/w_k$. The solution of step (i) can be computed by $\mathbf{V\tilde{D}V}^T$, where $\tilde{\mathbf{D}}$ is the diagonal matrix with the j-th element $\tilde{\mathbf{D}}_{jj} = \frac{w_k}{2\rho}\left(-D_{jj} + \sqrt{D_{jj}^2 + 4\rho/w_k}\right)$ [33, 110]. For step(ii), the minimization will depend on the convex penalty function P. When P is the *fused graphical lasso* shown in Equation (5.4), the problem can be viewed as a special case of a *Fused Lasso Signal Approximator* (FLSA) problem [104]. In [33], the authors solve this minimization problem using an efficient path algorithm for FLSA proposed in [51]. However when P is a generalized fused graphical lasso penalty (5.8), that is, the weights of different pair of $\Theta^{(k)}$s are different and defined by $G(\Theta, E)$, the path algorithm [51] has to be extended to a more general setting. More details on solving the optimization problem in step (ii) are shown in Section 5.4.

5.4 General Fused Lasso Signal Approximator (FLSA) with different edge weights

The minimization problem in step (ii) of ADMM in the previous section can be written as follows:

$$\underset{\{\mathbf{Z}\}}{\text{minimize}}\ \frac{\rho}{2}\sum_{k=1}^{K}\|Z^{(k)}-A^{(k)}\|_F^2+\lambda_1\sum_{k=1}^{K}\sum_{i\neq j}|Z_{ij}^{(k)}|+\lambda_2\sum_{p<q,(p,q)\in E}e_{pq}\sum_{i,j}|Z_{ij}^{(p)}-Z_{ij}^{(q)}|, \tag{5.12}$$

where $A^{(k)}=\Theta_{(t)}^{(k)}+U_{(t-1)}^{(k)}$ at the t-th iteration. Notice that (5.12) is separable with respect to each entry of matrix. One can solve the following problem for each (i,j).

$$\underset{Z_{ij}^{1},\cdots,Z_{ij}^{K}}{\text{minimize}}\ \frac{\rho}{2}\sum_{k=1}^{K}\|Z_{ij}^{(k)}-A_{ij}^{(k)}\|_F^2+\lambda_1\mathbf{1}_{i\neq j}\sum_{k=1}^{K}|Z_{ij}^{(k)}|+\lambda_2\sum_{p<q,(p,q)\in E}e_{pq}|Z_{ij}^{(p)}-Z_{ij}^{(q)}|. \tag{5.13}$$

When $e_{pq}=1$, for all p,q, the above problem can be viewed as a FLSA problem. With some variable changes and letting $\lambda_1\leftarrow\lambda_1/\rho$, $\lambda_2\leftarrow\lambda_2/\rho$, and $e_{pq}=1,\forall p,q$, we can rewrite the objective function of this general FLSA problem as follows.

$$L(\beta)=\frac{1}{2}\sum_{i=1}^{n}(y_i-\beta_i)^2+\lambda_1\sum_{i=1}^{n}|\beta_i|+\lambda_2\sum_{i<j,(i,j)\in E}|\beta_i-\beta_j|, \tag{5.14}$$

where β_i is an estimate and y_i is the given measurement. Suppose $\lambda_2>0$, $\lambda_1=0$ and the solution is obtained. When $\lambda_1>0$, the solution can be derived through soft thresholding by λ_1 [39]. Therefore we can first let $\lambda_1=0$ and general FLSA requires solving the following convex optimization problem:

$$L(\beta)=\frac{1}{2}\sum_{i=1}^{n}(y_i-\beta_i)^2+\lambda_2\sum_{i<j,(i,j)\in E}|\beta_i-\beta_j|. \tag{5.15}$$

One of the contributions of this work is to extend an efficient path algorithm [51] to the general FLSA problem with different edge weights, that is, to minimize the following *more* *general* convex objective function.

$$L(\beta) = \frac{1}{2}\sum_{i=1}^{n} w_i(y_i - \beta_i)^2 + \lambda_2 \sum_{i<j,(i,j)\in E} e_{ij}|\beta_i - \beta_j|. \tag{5.16}$$

Due to space limitations, we leave the details of this extended and efficient path algorithm which minimizes (5.16) to the appendix section. We also refer the reader to [51] for details of the original algorithm that solves the original FLSA problems in (5.15).

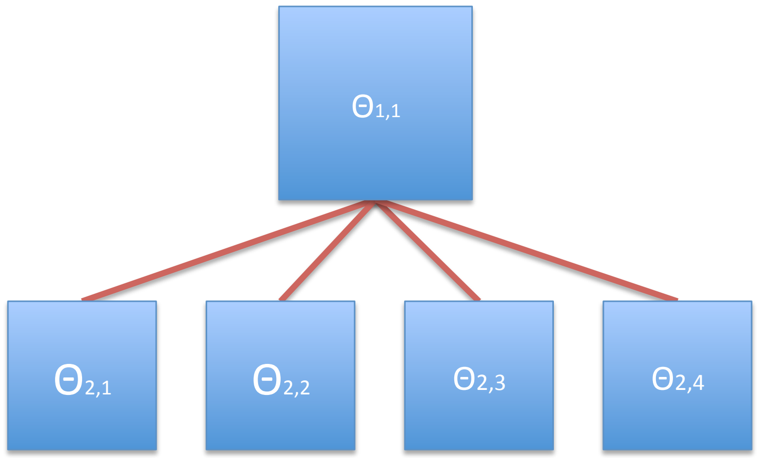


Figure 5.2: The graph structure for multiple precision matrices at different time scales and offsets for simulations and experiments.

5.5 Experiments

We compare our fusion approach with other non-fusion approaches on both synthetic and real-world datasets. The goal of this work is to show that fusing information at different times and scales can be helpful. When the fusion parameter λ_2 and sparsity parameter λ_1 in Eq.(5.8) are both zero, the maximum likelihood estimates are simply the inverse of sample covariance. While $\lambda_2 = 0$ and $\lambda_1 > 0$, the penalized maximum likelihood estimates are often known as graphical lasso as mentioned in previous sections. In this section we would like to show that when the GFGL approach starts fusing information, that is, when

λ_2 becomes non-zero, the precision estimates become more accurate. In addition, GFGL has another advantage. While introducing a sparsity penalty in graphical lasso can force some entries in precision matrices to be zero and help identify paris of variables that are unconnected in the graphical model, fusing different precision matrices can help identify edge difference across time scales and different times.

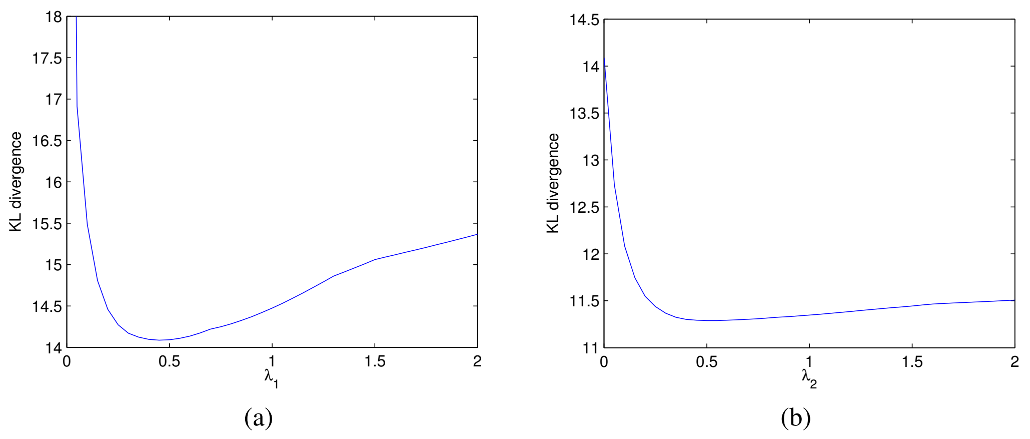


Figure 5.3: (a) shows KL divergence between graphical lasso estimate and true precision matrix with different λ_1. The minimum KL divergence is achieved when $\lambda_1 = 0.45$. Using the same λ_1, (b) shows that our GFGL can achieve lower KL divergence with the same λ_1 and $\lambda_2 = 0.55$.

5.5.1 Synthetic Simulations

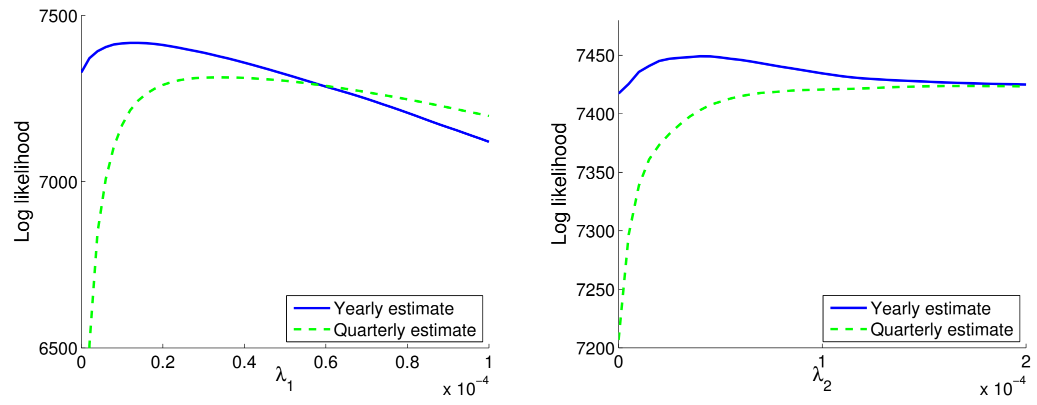
For synthetic simulations we first generate multiple synthetic precision matrices that are related to each other under a two-level hierarchical structure which is shown in Figure 5.2. In these experiments, we first generate a synthetic 5×5 sparse precision matrix $\Theta_{1,1}$ with density 0.3. To generate a lower level synthetic precision matrix from $\Theta_{1,1}$, we let the non-zero entries change to zero with probability 0.25 and let zero entries change to non-zero values with probability 0.1. Using this procedure, we generate $\Theta_{2,i}, i = 1, 2, 3, 4$ from $\Theta_{1,1}$. Given these bottom level synthetic precision matrices $\Theta_{2,i}, i = 1, \cdots, 4$, we generate 5 time series of length 20 under Gaussian distributions with zero mean and precision matrices $\{\Theta_{2,i}\}$, and add white noise. For example, the first 5 days of data are generated with

Table 5.1: Kullback-Liebler (KL) divergence between estimated and ground truth distributions for the simulation example explained in the text. Our fusion approach (GFGL) always yields lower KL divergence relative to the ground truth precision matrices.

	GL	GFGL
1	14.09	11.29
2	13.24	11.04
3	11.56	11.27
4	14.45	14.06
5	15.11	12.75
6	13.83	12.77
7	13.04	11.43
8	12.89	11.84
9	14.37	12.09
10	12.64	11.98

$\Theta_{2,1}$, the data from the 6-th to the 10-th day are with $\Theta_{2,2}$ and so on. Given these 20-day time series, we estimate the precision matrix for each 5-day period using graphical lasso and our GFGL approach with the same structure in Figure 5.2. The goal of this simulation is to see if fusing precision estimates from different time scales can improve the estimate for each 5-day period compared with a non-fusion approach. Since we have the ground truth of all precision matrices, we can compute the sum of KL divergences between $\Theta_{2,i}$, $i = 1, \cdots, 4$ and corresponding precision estimates to evaluate the performance of different approaches. For graphical lasso (GL), we do a dense grid search for λ_1 and report the lowest KL divergences for 10 different experimental runs. For GFGL, to find optimal parameters, one straightforward approach is to do a grid search over λ_1 and λ_2. However, doing a grid search is very time-consuming, so instead we apply a greedy approach, that is, to search λ_2 with the best λ_1 for graphical lasso and then report these suboptimal results. In this simulation, we intend to show that our fusion approach can give a more accurate estimate and report these oracle results for both fusion and non-fusion approaches. In real-world applications, parameter selection can be done through cross validation or by an approximation of the Akaike Information Criterion (AIC). In the next section, we use cross-validation to select parameters for experiments on a real-world dataset.

The results of this simulation are shown in Table 5.1. Our fusion approach GFGL obviously outperforms the non-fusion approach GL, increasing the accuracy. One example of performance under different parameters is given in Figure 5.3. In Figure 5.3(a), graphical lasso achieves the lowest KL divergence 14.09 when $\lambda_1 = 0.45$. Using this λ_1 and sweeping through λ_2, our fusion approach achieves the lowest KL divergence 11.29 when $\lambda_2 = 0.55$ shown in Figure 5.3(b). Note that a grid search over λ_1 and λ_2 would give a lower KL divergence, that is, a better estimate of precision matrices.



(a) (b)

Figure 5.4: (a) shows log likelihood on test data with different λ_1 given precision estimates by graphical lasso. The maximum log-likelihood is achieved when $\lambda_1 = 1.2 \times 10^{-5}$. Using the same λ_1, (b) shows that our GFGL can achieve better performance with the same λ_1 and $\lambda_2 = 4 \times 10^{-5}$.

5.5.2 S&P 500 Dataset

In this section we present the performance of fusion and non-fusion approaches on a real-world dataset. We collect stock prices of companies which belong to the $S\&P500$ component list at the end of 2013 and existed over the past 25 years from 1989 to 2013. All data are downloaded from *Yahoo Finance*. We first compute daily returns of all stocks and distribute all stocks to 10 different sectors based on public market information. In the end we obtained a total of 220 stocks and each sector has 16 – 33 stocks except the sector *Telecommunications Services (TS)* which has less than 5 companies. Due to space limitations we only present 10 years' results on all sectors except the TS sector from 2004 to 2013. In this experiment we use daily returns of all stocks as multiple time series as in previous work. Assuming these time series follow a multivariate Gaussian distribution, we want to estimate precision matrices at different years and also precision matrices at different quarters in each year. These sparse precision estimates can further be used in many applications such as portfolio construction, stock relationship visualization, and market analysis.

The hierarchical structure we use for this experiment is the same as that in Figure 5.2. The top level corresponds to the whole year precision matrix while the four bottom level precision matrices correspond to four different quarters. Because no one has ground truth of precision matrices for real-world stock data, instead of evaluating the KL divergence as in simulations, we compare log-likehoods of test data given different precision estimates. The experimental setup is as follows. For each year, we use each quarter's third month's data as test data. In other words we suppose data on March, June, September and December are missing. Different methods unitize data from the remaining months to estimate precision matrices corresponding to the whole year and four quarters. For all methods, we use the sample mean computed from the first two months' data of each quarter as the mean estimate for each quarter and denote m^i as the sample mean in quarter i. For convenience, denote by $\tilde{\mathbf{\Theta}}_{GL}^{year}$ and $\tilde{\mathbf{\Theta}}_{GL}^{i}$ the precision estimate for the whole year and the quarter i respectively of the graphical lasso method. The same notation is used for our fusion method (GFGL) and for a simple inverse approach (INV) which uses the inverse of sample covariance as the precision estimate.

For convenience we index missing data by X_i and define $Q(i)$ as the function which returns the quarter in which X_i is. For example, if X_i is a data from March, $Q(i) = 1$ and if X_i is a data from September, $Q(i) = 3$. Denote $\ell(X_i; m, \tilde{\Theta})$ as the Gaussian log-likelihood of X_i given mean m and precision matrix $\tilde{\Theta}$. We then compute the log likelihood of all missing data given different precision estimates obtained by different methods with parameters chosen by cross validation. The details of parameter selection are shown later. In all our experiments, a simple inverse approach which uses the inverse of sample covariance as the precision estimate does not outperform graphical lasso and our fusion approach. To compare our fusion approach with the non-fusion approach, we compute the difference between the log likelihoods of both approaches as follows:

$$d_i^{year} = \ell(X_i; m^{Q(i)}, \tilde{\Theta}_{GFGL}^{year}) - \ell(X_i; m^{Q(i)}, \tilde{\Theta}_{GL}^{year}), \forall i, \tag{5.17}$$

$$d_i^{Q} = \ell(X_i; m^{Q(i)}, \tilde{\Theta}_{GFGL}^{Q(i)}) - \ell(X_i; m^{Q(i)}, \tilde{\Theta}_{GL}^{Q(i)}), \forall i. \tag{5.18}$$

While d_i^{year} measures the performance difference between two methods using yearly estimates, d_i^Q measures the performance difference using quarterly estimates. The means and 95% confidence intervals of d_i^{year} and d_i^Q are reported. Results of all different sectors over the past 10 years are shown in Table 5.2 and Table 5.3. We observe a similar performance advantage of our method for all past 25 years. Due to space limitation we only show results over the past 10 years.

For parameter selection, we use cross validation to select both λ_1 and λ_2. For each year, we first run graphical lasso with different λ_1 on 90% of training data and choose the one that gives the highest log-likelihood for the remaining data. With this selected λ_1 we select λ_2 in the same manner. Notice that a dense grid search on both λ_1 and λ_2 would give an even higher log-likelihood on the cross validation data. We have also found that for some years and for some sectors our fusion approach tends to choose λ_2 to be 0. In this case, our method is equivalent to the graphical lasso approach using the same λ_1. This situation also suggests that the market at different quarters are so different that fusing information would not be good in this case and therefore our fusion method chooses λ_2 to be 0.

From the above two tables we have shown that our fusion approach can improve estimates of precision matrices. One really interesting thing is that fusion approach can achieve a log-likelihood that the non-fusion approach can never achieve. We take the sector *Consumer Discretionary* in 2013 as an example. In Figure 5.4(a), we show log-likelihood using graphical lasso's yearly estimate and quarterly estimates with different λ_1. By using graphical lasso, that is, a non-fusion approach, one can achieve the highest log likehood 7417 using yearly estimate when $\lambda_1 = 1.25 \times 10^{-5}$. Using the same λ_1, we increase λ_2 from zero to start fusion and are able to obtain higher log-likelihoods that can never be obtained by non-fusion approach.

5.6 Conclusion and Future Work

In this chapter we propose a framework that estimates multiple precision matrices at different time scales and offsets. Our method introduces a more general types of fused graphical lasso penalty to a maximum likelihood estimation problem. Our experimental studies show that combining information at different time scales and offsets by our fusion approach can result a more accurate estimation on both synthetic data and real-world time series. There are some interesting future works for this problem. One example is how to apply the proposed method for portfolio construction and market analysis. We believe that proposed fused method can possibly be applied to construct portfolios that outperform those constructed from non-fusion approaches. Another interesting future work is to use different structures and test performances under different structures.

Table 5.2: Differences of log-likehoods between our GFGL approach and graphical lasso(GL) using yearly estimates for different financial sectors during 2004$\sim$2013. Abbreviations at the first row stand for *Consumer Discretionary*, *Consumer Staples*, *Energy*, *Financials*, *Health Care*, *Industrials*, *Information Technology*, *Materials*, *Utilities*. Notice that at some years for some sector, the differences are zeros. This means that the fusion approach chooses λ_2 to be zero for those cases and is equivalent to GL. The results show that using the proposed fusion approach yields precision estimates with higher likelihood for missing months.

	CD	CS	EN	FI	HC	IN	IT	MA	UT
4	**0.283** ± 0.091	**0.039** ± 0.173	-0.013 ± 0.043	**0.287** ± 0.07	-0.797 ± 1.917	-0.072 ± 0.256	**0.203** ± 0.068	-0.028 ± 0.125	**0.172** ± 0.071
5	**0.033** ± 0.052	**0.114** ± 0.144	**0.082** ± 0.052	0 ± 0	**0.173** ± 0.145	-0.016 ± 0.323	**0.245** ± 0.08	0 ± 0	-0.01 ± 0.083
6	**0.08** ± 0.105	**0.059** ± 0.147	0 ± 0	-0.032 ± 0.307	**0.041** ± 0.25	**0.41** ± 0.125	**0.168** ± 0.126	**0.207** ± 0.068	0 ± 0
7	**0.047** ± 0.284	0 ± 0	0 ± 0	**0.146** ± 0.256	**0.243** ± 0.094	**0.114** ± 0.192	**0.185** ± 0.119	**0.064** ± 0.112	**0.343** ± 0.064
8	0 ± 0	**0.214** ± 0.153	**0.23** ± 0.16	**0.222** ± 0.151	**0.218** ± 0.256	**0.001** ± 0.404	**0.109** ± 0.108	-0.017 ± 0.435	**0.055** ± 0.624
9	-0.119 ± 0.182	**0.132** ± 0.102	-0.055 ± 0.098	**0.069** ± 0.281	**0.362** ± 0.509	**0.13** ± 0.235	0 ± 0	**0.126** ± 0.16	0 ± 0
10	-0.007 ± 0.098	**0.023** ± 0.079	-0.075 ± 0.185	**0.159** ± 0.126	-0.211 ± 0.743	**0.215** ± 0.193	-0.141 ± 0.324	**0.111** ± 0.07	**0.031** ± 0.103
11	-0.001 ± 0.129	**0.335** ± 0.083	**0.127** ± 0.056	**0.101** ± 0.259	**0.246** ± 0.073	**0.092** ± 0.193	**0.034** ± 0.112	**0.184** ± 0.16	-0.043 ± 0.176
12	**0.34** ± 0.098	**0.137** ± 0.112	**0.039** ± 0.038	**0.041** ± 0.212	**0.152** ± 0.095	-0.137 ± 0.278	**0.053** ± 0.115	**0.153** ± 0.137	**0.032** ± 0.112
13	**0.116** ± 0.046	0 ± 0	**0.084** ± 0.058	**0.112** ± 0.088	-0.021 ± 0.221	**0.193** ± 0.302	**0.092** ± 0.152	**0.222** ± 0.1	**0.135** ± 0.056

Table 5.3: Differences of log-likehoods between our GFGL approach and graphical lasso(GL) using quarterly estimates for different financial sectors during 2004 2013. Notice that the performance gaps between our approach and GL are much larger.

	CD	CS	EN	FI	HC	IN	IT	MA	UT
4	**4.088** ± 2.203	**3.335** ± 1.586	**0.484** ± 0.4	**1.157** ± 0.508	**8.307** ± 10.722	**8.245** ± 2.555	**1.723** ± 0.574	**2.832** ± 1.431	**1.745** ± 0.559
5	**0.765** ± 0.859	**6.997** ± 2.208	-0.071 ± 0.193	0 ± 0	**4.479** ± 1.573	**2.262** ± 1.081	**6.824** ± 2.546	0 ± 0	**0.889** ± 0.34
6	**1.972** ± 0.801	**5.255** ± 2.069	0 ± 0	**2.167** ± 1.45	**1.756** ± 1.693	**8.757** ± 1.994	**3.181** ± 1.296	**1.822** ± 0.732	0 ± 0
7	**1.429** ± 1.036	0 ± 0	0 ± 0	**1.231** ± 0.798	**8.223** ± 2.638	**4.646** ± 1.315	**6.113** ± 1.591	**1.145** ± 0.461	**0.527** ± 0.324
8	0 ± 0	**1.708** ± 0.91	**1.818** ± 1.015	**2.525** ± 1.681	**3.375** ± 2.868	**3.581** ± 1.163	**0.441** ± 0.467	**0.614** ± 0.717	**3.114** ± 2.309
9	-0.014 ± 0.781	**4.292** ± 2.204	**1.775** ± 0.929	-0.28 ± 0.691	**6.726** ± 9.045	**3.922** ± 1.537	0 ± 0	**2.291** ± 1.116	0 ± 0
10	**1.763** ± 0.967	-0.76 ± 0.455	**0.968** ± 0.722	**2.284** ± 0.687	**4.106** ± 4.297	**8.456** ± 1.636	**3.907** ± 2.664	**0.893** ± 0.448	**1.929** ± 1.541
11	**1.169** ± 0.782	**2.06** ± 1.061	**2.812** ± 0.972	**9.218** ± 2.843	**1.963** ± 1.23	**2.831** ± 0.949	**1.572** ± 1.331	**3.897** ± 1.991	**2.362** ± 1.682
12	**2.033** ± 1.068	**1.291** ± 0.646	**0.482** ± 0.321	**6.485** ± 1.511	**0.742** ± 0.426	**8.016** ± 3.014	**1.392** ± 0.691	**3.817** ± 1.11	**1.226** ± 0.731
13	**1.116** ± 1.609	0 ± 0	**1.161** ± 0.575	**4.294** ± 1.483	**4.167** ± 1.466	**8.229** ± 3.833	**3.373** ± 2.387	**1.06** ± 0.432	**0.642** ± 0.66

CHAPTER VI

Conclusion and Future Work

This thesis investigated information fusion in several different machine learning settings. We proposed novel and efficient algorithms to tackle down these problems and presented experimental results on synthetic and real-world datasets.

In Chapter II we proposed a new multi-criteria anomaly detection method. To better detect anomalies possibly under different criteria, multiple dissimilarity measures need to be considered at the same time. While linear scalarization methods require choosing a specific weights for these different dissimilarity measures and scales exponentially in the number of criteria, the proposed method, which uses Pareto depth analysis to compute anomaly scores of test samples, can detect anomalies and scales linearly. We also present theoretical results showing that Pareto approach is asymptotically better than using linear scalarization for multiple criteria. Experimental results on both synthetic and real-world datasets show that combining information under disparate criteria improve performances of anomaly detection and our proposed method outperforms other state-of-arts approaches.

In Chapter III we continued using Pareto depth and present a novel algorithm for multiple-query image retrieval. Since different query images might contain different semantic concepts, linear scalarization methods and other multiple-query retrieval approaches can not easily retrieve some samples that are related to all queries at the same time. Some theoretical results on asymptotic non-convexity of Pareto fronts are also shown in this chapter and shows that the proposed Pareto approach can outperform methods using linear combinations of ranking results. We also present experimental studies on real-world datasets to illustrated the advantages of the proposed algorithm. In this problem disparate information come from different query images. By issuing query images of different semantic concepts, users can retrieve images of more interests.

In Chapter IV we considered a collaborative retrieval problem which can be viewed as a blend of recommendation and retrieval task. By combing behavior information which is users' listening history and relation information which is social connectivity of users, we successfully shows that combing different types of information can improve the performance of collaborative retrieval task on a real-world music dataset. Since

We follow the theme of this thesis and research problems which attempt to estimate multiple precision matrices for multiple time series by combining information at different time scales and offsets. In Chapter V an information fusion approach, general fused graphical lasso (GFGL), is proposed to fuse multiple precision matrices. This approach obtains penalized graphical lasso estimates with a more general version of FGL penalty. In our experimental studies on both synthetic and real-world stock datasets, we show that proposed fusion approach can estimate multiple precision matrices across times more accurately.

To sum up, this thesis presents efficient algorithms that combine disparate information for different types of machine learning problems involving anomaly detection, information retrieval, collaborative retrieval, and multi-resolution time series analysis. We successfully show that combining disparate information can improve performances and proposed methods can outperform other approaches with and without other methods to combine disparate information.

There are many interesting directions for future work. Some potential directions are outlined below.

(1) Pareto depth analysis can be possibly applied to a classification problem such as multiple kernel learning problems. In Chapter II we use multiple Pareto fronts to define anomaly scores of samples for anomaly detection problem. It is possible to apply same idea for multiple kernel learning (MKL). Instead of learning weights for different kernels

in (MKL), one could use Pareto depths as new dissimilarities. However normalization of Pareto depths need to be researched for construction of a valid kernel.

(2) In Chapter III we use Pareto front method for ranking samples in the database given multiple queries. As we mentioned in the introduction section of Chapter III, this approach can be further applied to automatic image annotation of a large database. One could issue different query combinations with know class labels or other metadata, and automatically annotate the images in the middle of the first few Pareto fronts with the metadata from the queries. This problem becomes more relevant in the current era when millions of unlabeled images become available every few days.

(3) In Chapter III we successfully apply our model to combine social information with users' listening history to improve the performance of collaborative retrieval on a music dataset. An interesting future work might be to apply similar approach to different types of datasets. Do friends share more shopping preferences? Do friends like to read articles on similar topics? Different social information analysis methods might be needed for different types of datasets with social network information.

(4) In Chapter IV we show that our approach that fuses precision estimates at different time scales and offsets can have a more accurate estimation on synthetic data and give higher likelihoods of test data on real-world dataset. One interesting question is whether these more accurate estimates can give better performance for real-world portfolio construction. Since the performance of financial portfolio construction depends severely on how accurate the mean estimate of daily returns, one would need to also consider this phenomenon before comparing the performance of portfolio constructions using different precision estimates.

APPENDIX

APPENDIX A

Graphical Lasso Fusion across Multiple Time-sclaes : Supplementary Materials

A.1 General FLSA with different edge weights

The general *fused lasso signal approximator* (FLSA) [51, 104] is to minimize the following objective function with respect to β.

$$L(\beta) = \frac{1}{2}\sum_{i=1}^{n}(y_i - \beta_i)^2 + \lambda_2 \sum_{i<j,(i,j)\in E} |\beta_i - \beta_j| \tag{A.1}$$

where β_i is an estimate and y_i is given measurement. Notice that we ignore the sparsity regularization $\lambda_1 \sum_i |\beta_i|$ here since the solution of the whole problem with this L-1 penalty can be obtained from the solution of above optimization problem by soft-thresholding.

Since in this work we introduce a more general version of fused graphical lasso problem, we need to solve a more general version of general FLSA problem in one of sub steps at each iteration of ADMM algorithm. This more general version of FLAS problem is to minimize the following objective function with respect to β:

$$L(\beta) = \frac{1}{2}\sum_{i=1}^{n} w_i(y_i - \beta_i)^2 + \lambda_2 \sum_{i<j,(i,j)\in E} e_{ij}|\beta_i - \beta_j| \tag{A.2}$$

Basically we add different weights $w_i's$ for data fitting terms and different edge weights e_{ij} for different pairs of $\beta_i's$. In this supplementary material we present the details of extending the path algorithm in [51] to solve this more general version of general FLSA.

A.1.1 Definition of group and gradient of β_{F_i}

Following the same notations used in [51], denote $n_F(\lambda_2)$ as the number of sets of fused variables for penalty parameter λ_2 and F_i as the sets of fused variables. Suppose we have the minimizer of the loss function (A.2) for the penalty parameter λ_2 and denote it by $\beta_{F_i}(\lambda_2)$. Under this definition, if $k, l \in F_i$, $\beta_k(\lambda_2) = \beta_l(\lambda_2)$ and if $k \in F_i, l \in F_j, i \neq j$ and $kl \in E$, then $\beta_k(\lambda_2) \neq \beta_l(\lambda_2)$. For convenience, let $\beta_k = \beta_{F_i}, \forall k \in F_i$ and denote the condition that $kl \in E, k \in F_i, l \in F_j$ as $kl \in E_{i,j}$. The loss function in (A.2) can now expressed as follows:

$$L_{F,\lambda_2}(\beta) = \frac{1}{2}\sum_{i=1}^{n_F(\lambda_2)}(\sum_{j\in F_i} w_j(y_j - \beta_{F_i})^2) + \lambda_2 \sum_{i<j}(\sum_{kl\in E, k\in F_i, l\in F_j} e_{kl})|\beta_{F_i} - \beta_{F_j}| \tag{A.3}$$

Since we assume that β is the minimizer, the gradient of $L_{F,\lambda_2}(\beta)$ with respect to β_{F_i} is zero and can be expressed as the following:

$$\frac{\partial L_{F,\lambda_2}(\beta)}{\partial \beta_{F_i}} = \sum_{j\in F_i} w_j \cdot \beta_{F_i} - \sum_{j\in F_i} y_j w_j + \lambda_2 \sum_{j\neq i}(\sum_{kl\in E_{ij}} e_{kl}) sign(\beta_{F_i} - \beta_{F_j}) = 0 \tag{A.4}$$

For small changes of λ_2, as long as the sign of $\beta_{F_i} - \beta_{F_j}$ does not change for all possible i, j, it can be easily seen that β_{F_i} is piece-wise linear with respect to λ_2 and the gradient of β_{F_i} w.r.t. λ_2 can then be written as the following :

$$\frac{\partial \beta_{F_i}}{\partial \lambda_2} = -\frac{\sum_{j\neq i}\left((\sum_{kl\in E_{ij}} e_{kl}) sign(\beta_{F_i} - \beta_{F_j})\right)}{\sum_{j\in F_i} w_j} \tag{A.5}$$

The key idea of this path algorithm is to update β_k according its set's gradient with respect to λ_2 until some sets need to be split or some sets need to be fused to one set. To determine the time of fusing sets, one could easily check the hitting time for different pairs of sets and determine the minimum as the earliest time. The hitting time can defined as in [51] as follows:

$$h_{ij}(\lambda_2) = \begin{cases} (\beta_{F_i} - \beta_{F_j}) / \left(\frac{\partial \beta_{F_j}}{\partial \lambda_2} - \frac{\partial \beta_{F_i}}{\partial \lambda_2} \right) + \lambda_2, \\ \text{if } \exists\, kl \in E_{ij}. \\ \infty, o.w. \end{cases} \tag{A.6}$$

$$h(\lambda_2) = \min_{h_{ij} > \lambda_2} h_{ij}(\lambda_2) \tag{A.7}$$

For the time of splitting some fused sets, it is more complicate and shown in the next subsection.

A.1.2 Subgradient of $L(y, \beta)$ with respect to β_k

To determine the time of splitting some fused sets, [51] solves a max flow problem to determine when the optimal condition would be broken. We modify the max flow problem's capacity for our case. Since $L_{\lambda_2}(\beta)$ is not differentiable everywhere, we follow the same procedure in [51] and use subgradients. For the subgradients, a necessary and sufficient condition for β_k to be optimal is that

$$\frac{\partial L_{\lambda_2}(\beta)}{\partial \beta_k} = w_k(\beta_k - y_k) + \lambda_2 \sum_{k<l, kl \in E} t_{kl} = 0, \forall k = 1, \ldots, n, \tag{A.8}$$

where

$$t_{kl} = \begin{cases} e_{kl} sign(\beta_k - \beta_l) & , \beta_k \neq \beta_l. \\ \in [-e_{kl}, e_{kl}] & , \beta_k = \beta_l. \end{cases} \tag{A.9}$$

Notice that no matter $\beta_k = \beta_l$ or not, $t_{kl} = -t_{kl}$. For $k \in F_i$, the subgradient can be written as follows

$$\frac{\partial L_{\lambda_2}(\beta)}{\partial \beta_k} = w_k(\beta_k - y_k) + \lambda_2 \sum_{j \neq i} \sum_{l \in F_j} t_{kl} + \lambda_2 \sum_{l \in F_i} t_{kl} = 0 \tag{A.10}$$

For convenience, define $\tau_{kl} = \lambda_2 t_{kl}$ and also define P_k as

$$P_k = -w_k(\beta_k - y_k) - \lambda_2 \sum_{j \neq i} \sum_{l \in F_j} t_{kl}, \forall k = 1 \dots n. \tag{A.11}$$

Basically we will have the following condition for all β_k:

$$\sum_{l \in F_i} \tau_{kl} = P_k \tag{A.12}$$

Therefore for each fused set F_i, one could solve a max flow problem to determine values of $\tau_{kl}, k, l \in F_i$.

A.1.3 The max-flow problem

Recall the conditions in (A.9) and $\tau_{kl} = \lambda_2 t_{kl}$. We need to ensure that $\tau_{kl}(\lambda_2) \in [-\lambda_2 e_{kl}, \lambda_2 e_{kl}]$, we have the restrictions

$$\frac{\partial \tau_{kl}}{\partial \lambda_2} \in \begin{cases} (-\infty, \infty) & , \tau_{kl} \in (-\lambda_2 e_{kl}, \lambda_2 e_{kl}) \\ (-\infty, 1] & , \tau_{kl} = \lambda_2 e_{kl} \\ [1, \infty) & , \tau_{kl} = -\lambda_2 e_{kl} \end{cases} \tag{A.13}$$

Therefore, the capacity for the link $kl \in E, k, l \in F_i$ can be set as

$$(c_{kl}, c_{lk}) = \begin{cases} (\infty, \infty) & , \tau_{kl} \in (-\lambda_2 e_{kl}, \lambda_2 e_{kl}) \\ (1, \infty) & , \tau_{kl} = \lambda_2 e_{kl} \\ (\infty, 1) & , \tau_{kl} = -\lambda_2 e_{kl} \end{cases} \tag{A.14}$$

A.1.4 The splitting time v

The splitting time is defined by the solutions of the maximum flow problem. We could determine how large λ_2' can be such that for all $\tau_{kl}(\lambda_2') \in [-\lambda_2', \lambda_2']$. Recall that the flow solution f_{kl} is actually the derivative of τ_{kl} w.r.t. λ_2. Therefor we require the following conditions to be hold.

For $f_{kl} > 0$ and $|f_{kl}| > e_{kl}$,

$$\tau_{kl}(\lambda_2) + f_{kl}(\lambda_2' - \lambda_2) \leq e_{kl}\lambda_2' \tag{A.15}$$

or equivalently

$$\lambda_2' \leq \lambda_2 + \frac{e_{kl}\lambda_2 - \tau_{kl}(\lambda_2)}{f_{kl} - e_{kl}}. \tag{A.16}$$

For $f_{kl} < 0$ and $|f_{kl}| > e_{kl}$,

$$\tau_{kl}(\lambda_2) + f_{kl}(\lambda_2' - \lambda_2) \geq -e_{kl}\lambda_2' \tag{A.17}$$

or equivalently

$$\lambda_2' \leq \lambda_2 + \frac{e_{kl}\lambda_2 + \tau_{kl}(\lambda_2)}{-f_{kl} - e_{kl}}. \tag{A.18}$$

Therefore, the general form of violation time can be defined as

$$v_{kl}(\lambda_2) = \begin{cases} \lambda_2 + \frac{e_{kl}\lambda_2 - sign(f_{kl})\tau_{kl}(\lambda_2)}{|f_{kl}| - e_{kl}} & \text{,if } |f_{kl}| > e_{kl}, \\ \infty & , otherwise. \end{cases} \tag{A.19}$$

The splitting time is then defined by

$$v(\lambda_2) = \min_{k,l} v_{kl}(\lambda_2). \tag{A.20}$$

Therefore we can at least increase λ_2 by $\triangle$ without changing the sign of $\beta_{F_i} - \beta_{F_j}, \forall i, j$ and the sets $F_i, \forall i$ where $\triangle$ is defined as follows:

$$\triangle = \min\left(h(\lambda_2), v(\lambda_2)\right) - \lambda_2. \tag{A.21}$$

We refer readers to [51] for details of the original path algorithm. In this appendix we only present key changes of equations used in the algorithm. The theorems shown in [51] can also be easily extended to generalized versions for this extended path algorithm.

Bibliography

[1] Agrawal, R., and E. L. Wimmers (2000), A framework for expressing and combining preferences, in *ACM SIGMOD Record*, vol. 29, pp. 297–306, ACM.

[2] Angiulli, F., and C. Pizzuti (2002), Fast outlier detection in high dimensional spaces, in *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, pp. 15–27, Springer.

[3] Arandjelovic, R., and A. Zisserman (2012), Multiple queries for large scale specific object retrieval, in *British Machine Vision Conference*.

[4] Aslam, J. A., and M. Montague (2001), Models for metasearch, in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 276–284, ACM.

[5] Bai, Z.-D., L. Devroye, H.-K. Hwang, and T.-H. Tsai (2005), Maxima in hypercubes, *Random Structures & Algorithms*, *27*(3), 290–309.

[6] Banerjee, O., L. El Ghaoui, and A. d'Aspremont (2008), Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data, *The Journal of Machine Learning Research*, *9*, 485–516.

[7] Barndorff-Nielsen, O., and M. Sobel (1966), On the distribution of the number of admissible points in a vector random sample, *Theory of Probability and its Applications*, *11*(2), 249–269.

[8] Baryshnikov, Y., and J. E. Yukich (2005), Maximal points and Gaussian fields, preprint.

[9] Bedi, P., H. Kaur, and S. Marwaha (2007), Trust based recommender system for semantic web, in *proceedings of the 2007 International Joint Conferences on Artificial Intelligence*, pp. 2677–2682.

[10] Belkin, N. J., C. Cool, W. B. Croft, and J. P. Callan (1993), The effect of multiple query representations on information retrieval system performance, in *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 339–346, ACM.

[11] Blei, D. M., A. Y. Ng, and M. I. Jordan (2003), Latent dirichlet allocation, *the Journal of machine Learning research*, *3*, 993–1022.

[12] Blum, A., and T. Mitchell (1998), Combining labeled and unlabeled data with co-training, in *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pp. 92–100, ACM.

[13] Bollobás, B., and P. Winkler (1988), The longest chain among random points in Euclidean space, *Proceedings of the American Mathematical Society*, *103*(2), 347–353.

[14] Boriah, S., V. Chandola, and V. Kumar (2008), Similarity measures for categorical data: A comparative evaluation, *red*, *30*(2), 3.

[15] Börzsönyi, S., D. Kossmann, and K. Stocker (2001), The Skyline operator, in *Proceedings of the 17th International Conference on Data Engineering*, pp. 421–430, IEEE.

[16] Boyd, S., N. Parikh, E. Chu, B. Peleato, and J. Eckstein (2011), Distributed optimization and statistical learning via the alternating direction method of multipliers, *Foundations and Trends® in Machine Learning*, *3*(1), 1–122.

[17] Breese, J. S., D. Heckerman, and C. Kadie (1998), Empirical analysis of predictive algorithms for collaborative filtering, in *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pp. 43–52, Morgan Kaufmann Publishers Inc.

[18] Breunig, M. M., H.-P. Kriegel, R. T. Ng, and J. Sander (2000), LOF: Identifying density-based local outliers, in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 93–104, ACM.

[19] Brin, S., and L. Page (1998), The anatomy of a large-scale hypertextual web search engine, *Comp. networks and ISDN sys.*, *30*(1), 107–117.

[20] Burges, C., T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender (2005), Learning to rank using gradient descent, in *Proc. of the 22nd ICML*, pp. 89–96, ACM.

[21] Byers, S., and A. E. Raftery (1998), Nearest-neighbor clutter removal for estimating features in spatial point processes, *Journal of the American Statistical Association*, *93*(442), 577–584.

[22] Calder, J., S. Esedoḡlu, and A. O. Hero (2013), A PDE-based approach to non-dominated sorting, *arXiv preprint:1310.2498*.

[23] Calder, J., S. Esedoglu, and A. O. Hero (2014), A Hamilton-Jacobi equation for the continuum limit of non-dominated sorting, *To appear in the SIAM Journal on Mathematical Analysis*.

[24] Calder, J., S. Esedoglu, and A. O. Hero (2014), A continuum limit for non-dominated sorting, in *To appear in the Proceedings of the Information Theory and Applications (ITA) Workshop*.

[25] Cantador, I., P. Brusilovsky, and T. Kuflik (2011), Second workshop on information heterogeneity and fusion in recommender systems (hetrec2011), in *Proceedings of the fifth ACM conference on Recommender systems*, pp. 387–388, ACM.

[26] Carmel, D., N. Zwerdling, I. Guy, S. Ofek-Koifman, N. Har'El, I. Ronen, E. Uziel, S. Yogev, and S. Chernov (2009), Personalized social search based on the user's social network, in *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 1227–1236, ACM.

[27] Carneiro, G., A. B. Chan, P. J. Moreno, and N. Vasconcelos (2007), Supervised learning of semantic classes for image annotation and retrieval, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *29*(3), 394–410.

[28] Chandola, V., A. Banerjee, and V. Kumar (2009), Anomaly detection: A survey, *ACM Computing Surveys*, *41*(3), 15.

[29] Chen, L., S. Zheng, et al. (2009), Studying alternative splicing regulatory networks through partial correlation analysis, *Genome Biol*, *10*(1), R3.

[30] Christoudias, C., R. Urtasun, and T. Darrell (2008), Multi-view learning in the presence of view disagreement, in *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, pp. 88–96.

[31] Chu, C., S. K. Kim, Y.-A. Lin, Y. Yu, G. Bradski, A. Y. Ng, and K. Olukotun (2007), Map-reduce for machine learning on multicore, *Advances in Neural Information Processing Systems*, *19*, 281.

[32] Dalal, N., and B. Triggs (2005), Histograms of oriented gradients for human detection, in *Proc. CVPR IEEE*, vol. 1, pp. 886–893, IEEE.

[33] Danaher, P., P. Wang, and D. M. Witten (2013), The joint graphical lasso for inverse covariance estimation across multiple classes, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*.

[34] Datta, R., D. Joshi, J. Li, and J. Wang (2008), Image retrieval: Ideas, influences, and trends of the new age, *ACM Comp. Surveys*, *40*(2), 5.

[35] De La Fuente, A., N. Bing, I. Hoeschele, and P. Mendes (2004), Discovery of meaningful associations in genomic data using partial correlation coefficients, *Bioinformatics*, *20*(18), 3565–3574.

[36] Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan (2002), A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.

[37] Ehrgott, M. (2005), *Multicriteria optimization*, 2nd ed., Springer.

[38] Eskin, E., A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo (2002), A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data, in *Applications of Data Mining in Computer Security*, chap. 4.

[39] Friedman, J., T. Hastie, H. Höfling, R. Tibshirani, et al. (2007), Pathwise coordinate optimization, *The Annals of Applied Statistics*, *1*(2), 302–332.

[40] Friedman, J., T. Hastie, and R. Tibshirani (2008), Sparse inverse covariance estimation with the graphical lasso, *Biostatistics*, *9*(3), 432–441.

[41] Friedman, J., T. Hastie, and R. Tibshirani (2010), A note on the group lasso and a sparse group lasso, *arXiv preprint arXiv:1001.0736*.

[42] Frome, A., Y. Singer, and J. Malik (2007), Image retrieval and classification using local distance functions, in *Advances in Neural Information Processing Systems: Proceedings of the 2006 Conference*, vol. 19, p. 417, The MIT Press.

[43] Ganeshapillai, G., J. Guttag, and A. Lo (2013), Learning connections in financial time series, in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp. 109–117.

[44] Gia, G., F. Roli, et al. (2004), Instance-based relevance feedback for image retrieval, in *Advances in neural information processing systems*, pp. 489–496.

[45] Gönen, M., and E. Alpaydın (2011), Multiple kernel learning algorithms, *Journal of Machine Learning Research*, *12*, 2211–2268.

[46] Guo, J., E. Levina, G. Michailidis, and J. Zhu (2011), Joint estimation of multiple graphical models, *Biometrika*, *98*(1), 1–15.

[47] Hero III, A. O. (2006), Geometric entropy minimization (GEM) for anomaly detection and localization, in *Advances in Neural Information Processing Systems 19*, pp. 585–592.

[48] Hero III, A. O., and G. Fleury (2004), Pareto-optimal methods for gene ranking, *The Journal of VLSI Signal Processing*, *38*(3), 259–275.

[49] Hirata, K., and T. Kato (1992), Query by visual example, in *Adv. in Database Technology-EDBT'92*, pp. 56–71, Springer.

[50] Hodge, V. J., and J. Austin (2004), A survey of outlier detection methodologies, *Artificial Intelligence Review*, *22*(2), 85–126.

[51] Hoefling, H. (2010), A path algorithm for the fused lasso signal approximator, *Journal of Computational and Graphical Statistics*, *19*(4), 984–1006.

[52] Hristidis, V., N. Koudas, and Y. Papakonstantinou (2001), Prefer: A system for the efficient execution of multi-parametric ranked queries, *ACM SIGMOD Record*, *30*(2), 259–270.

[53] Hsiao, K.-J., K. S. Xu, J. Calder, and A. O. Hero III (2012), Multi-criteria anomaly detection using Pareto Depth Analysis, in *Advances in Neural Information Processing Systems 25*, pp. 854–862.

[54] Hsieh, C.-J., M. A. Sustik, I. S. Dhillon, and P. D. Ravikumar (2011), Sparse inverse covariance matrix estimation using quadratic approximation., in *NIPS*, pp. 2330–2338.

[55] Huang, S., J. Li, L. Sun, J. Liu, T. Wu, K. Chen, A. Fleisher, E. Reiman, and J. Ye (2009), Learning brain connectivity of alzheimer's disease from neuroimaging data., in *NIPS*, vol. 22, pp. 808–816.

[56] Hwang, H.-K., and T.-H. Tsai (2010), Multivariate records based on dominance, *Electronic Journal of Probability*, *15*, 1863–1892.

[57] Ivanin, V. M. (1975), An asymptotic estimate of the mathematical expectation of the number of elements of the Pareto set, *Kibernetika*, *11*(1), 97–101.

[58] Järvelin, K., and J. Kekäläinen (2002), Cumulated gain-based evaluation of ir techniques, *ACM Transactions on Information Systems (TOIS)*, *20*(4), 422–446.

[59] Jensen, M. T. (2003), Reducing the run-time complexity of multiobjective EAs: The NSGA-II and other algorithms, *IEEE Transactions on Evolutionary Computation*, *7*(5), 503–515.

[60] Jeon, J., V. Lavrenko, and R. Manmatha (2003), Automatic image annotation and retrieval using cross-media relevance models, in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pp. 119–126, ACM.

[61] Jin, W., J. Han, and M. Ester (2004), Mining thick skylines over large databases, in *Knowledge Discovery in Databases: PKDD 2004*, pp. 255–266, Springer.

[62] Jin, X., and J. French (2005), Improving image retrieval effectiveness via multiple queries, *Multimedia Tools and Applications*, *26*(2), 221–245.

[63] Jin, Y., and B. Sendhoff (2008), Pareto-based multiobjective machine learning: An overview and case studies, *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, *38*(3), 397–415.

[64] Karatzoglou, A., X. Amatriain, L. Baltrunas, and N. Oliver (2010), Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering, in *Proceedings of the fourth ACM conference on Recommender systems*, pp. 79–86, ACM.

[65] Kautz, H., B. Selman, and M. Shah (1997), Referral web: combining social networks and collaborative filtering, *Communications of the ACM*, *40*(3), 63–65.

[66] Kenett, D. Y., M. Tumminello, A. Madi, G. Gur-Gershgoren, R. N. Mantegna, and E. Ben-Jacob (2010), Dominating clasp of the financial sector revealed by partial correlation analysis of the stock market, *PloS one*, *5*(12), e15,032.

[67] Konstas, I., V. Stathopoulos, and J. M. Jose (2009), On social networks and collaborative recommendation, in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 195–202, ACM.

[68] Kossmann, D., F. Ramsak, and S. Rost (2002), Shooting stars in the sky: an online algorithm for skyline queries, in *Proceedings of the 28th International Conference on Very Large Data Bases*, pp. 275–286.

[69] Kullback, S., and R. A. Leibler (1951), On information and sufficiency, *Ann. Math. Stat.*, *22*(1), 79–86.

[70] Kung, H. T., F. Luccio, and F. P. Preparata (1975), On finding the maxima of a set of vectors, *Journal of the ACM*, *22*(4), 469–476.

[71] Lee, J. H. (1997), Analyses of multiple evidence combination, in *ACM SIGIR Forum*, vol. 31, pp. 267–276, ACM.

[72] Li, L., and K.-C. Toh (2010), An inexact interior point method for l 1-regularized sparse covariance selection, *Mathematical Programming Computation*, *2*(3-4), 291–315.

[73] Linden, G., B. Smith, and J. York (2003), Amazon. com recommendations: Item-to-item collaborative filtering, *Internet Computing, IEEE*, *7*(1), 76–80.

[74] Liu, H., K. Roeder, and L. Wasserman (2010), Stability approach to regularization selection (stars) for high dimensional graphical models, in *Advances in Neural Information Processing Systems*, pp. 1432–1440.

[75] Liu, J., L. Yuan, and J. Ye (2010), An efficient algorithm for a class of fused lasso problems, in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 323–332, ACM.

[76] Liu, T. (2009), Learning to rank for information retrieval, *Foundations and Trends in Information Retrieval*, *3*(3), 225–331.

[77] Liu, Y., D. Zhang, G. Lu, and W. Ma (2007), A survey of content-based image retrieval with high-level semantics, *Pattern Recognition*, *40*(1), 262–282.

[78] Lowe, D. (2004), Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vision*, *60*(2), 91–110.

[79] Ma, H., H. Yang, M. R. Lyu, and I. King (2008), Sorec: social recommendation using probabilistic matrix factorization, in *Proceedings of the 17th ACM conference on Information and knowledge management*, pp. 931–940, ACM.

[80] Ma, H., I. King, and M. R. Lyu (2009), Learning to recommend with social trust ensemble, in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 203–210, ACM.

[81] Ma, H., D. Zhou, C. Liu, M. R. Lyu, and I. King (2011), Recommender systems with social regularization, in *Proceedings of the fourth ACM international conference on Web search and data mining*, pp. 287–296, ACM.

[82] Majecka, B. (2009), Statistical models of pedestrian behaviour in the forum, Master's thesis, School of Informatics, University of Edinburgh.

[83] Massa, P., and P. Avesani (2004), Trust-aware collaborative filtering for recommender systems, in *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE*, pp. 492–508, Springer.

[84] Massa, P., and P. Avesani (2007), Trust-aware recommender systems, in *Proceedings of the 2007 ACM conference on Recommender systems*, pp. 17–24, ACM.

[85] Meier, L., S. Van De Geer, and P. Bühlmann (2008), The group lasso for logistic regression, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *70*(1), 53–71.

[86] Meng, W., C. Yu, and K.-L. Liu (2002), Building efficient and effective metasearch engines, *ACM Computing Surveys (CSUR)*, *34*(1), 48–89.

[87] O'Donovan, J., and B. Smyth (2005), Trust in recommender systems, in *Proceedings of the 10th international conference on Intelligent user interfaces*, pp. 167–174, ACM.

[88] Papadias, D., Y. Tao, G. Fu, and B. Seeger (2003), An optimal and progressive algorithm for skyline queries, in *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pp. 467–478, ACM.

[89] Prekopa, A. (1973), On logarithmic concave measures and functions, *Acta Sci. Math.(Szeged)*, *34*, 335–343.

[90] Purushotham, S., Y. Liu, and C.-C. J. Kuo (2012), Collaborative topic regression with social matrix factorization for recommendation systems, *arXiv preprint arXiv:1206.4684*.

[91] Reverter, A., and E. K. Chan (2008), Combining partial correlation and an information theory approach to the reversed engineering of gene co-expression networks, *Bioinformatics*, *24*(21), 2491–2497.

[92] Russell, B. C., A. Torralba, K. P. Murphy, and W. T. Freeman (2008), Labelme: a database and web-based tool for image annotation, *International journal of computer vision*, *77*(1-3), 157–173.

[93] Salakhutdinov, R., and A. Mnih (2008), Probabilistic matrix factorization, *Advances in neural information processing systems*, *20*, 1257–1264.

[94] Sankoff, D., and J. B. Kruskal (1983), *Time warps, string edits, and macromolecules: the theory and practice of sequence comparison*, Addison-Wesley.

[95] Sarwar, B., G. Karypis, J. Konstan, and J. Riedl (2001), Item-based collaborative filtering recommendation algorithms, in *Proceedings of the 10th international conference on World Wide Web*, pp. 285–295, ACM.

[96] Seung, D., and L. Lee (2001), Algorithms for non-negative matrix factorization, *Advances in neural information processing systems*, *13*, 556–562.

[97] Sharifzadeh, M., and C. Shahabi (2006), The spatial skyline queries, in *Proceedings of the 32nd international conference on Very large data bases*, pp. 751–762, VLDB Endowment.

[98] Sindhwani, V., P. Niyogi, and M. Belkin (2005), A co-regularization approach to semi-supervised learning with multiple views, in *Proceedings of the ICML Workshop on Learning with Multiple Views*, pp. 74–79.

[99] Snoek, C., M. Worring, J. Van Gemert, J. Geusebroek, and A. Smeulders (2006), The challenge problem for automated detection of 101 semantic concepts in multimedia, in *Proc. of the 14th ACM int. conf. on Multimedia*, pp. 421–430.

[100] Sricharan, K., and A. O. Hero III (2011), Efficient anomaly detection using bipartite k-NN graphs, in *Advances in Neural Information Processing Systems 24*, pp. 478–486.

[101] Su, X., and T. M. Khoshgoftaar (2009), A survey of collaborative filtering techniques, *Advances in Artificial Intelligence*, *2009*, 4.

[102] Symeonidis, P., A. Nanopoulos, and Y. Manolopoulos (2008), Tag recommendations based on tensor dimensionality reduction, in *Proceedings of the 2008 ACM conference on Recommender systems*, pp. 43–50, ACM.

[103] Tan, K.-L., P.-K. Eng, and B. C. Ooi (2001), Efficient progressive skyline computation, in *Proceedings of the 27th International Conference on Very Large Data Bases*, pp. 301–310.

[104] Tibshirani, R., M. Saunders, S. Rosset, J. Zhu, and K. Knight (2005), Sparsity and smoothness via the fused lasso, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(1), 91–108.

[105] van Gemert, J., J. Geusebroek, C. Veenman, C. Snoek, and A. Smeulders (2006), Robust scene categorization by learning image statistics in context, in *Proc. CVPR Workshop IEEE*, pp. 105–105.

[106] Vasconcelos, N., and A. Lippman (1999), Learning from user feedback in image retrieval systems., in *NIPS*, pp. 977–986.

[107] von Luxburg, U. (2007), A tutorial on spectral clustering, *Statistics and Computing*, *17*(4), 395–416.

[108] Weston, J., S. Bengio, and N. Usunier (2010), Large scale image annotation: learning to rank with joint word-image embeddings, *Machine learning*, *81*(1), 21–35.

[109] Weston, J., C. Wang, R. Weiss, and A. Berenzweig (2012), Latent collaborative retrieval, *arXiv preprint arXiv:1206.4603*.

[110] Witten, D. M., and R. Tibshirani (2009), Covariance-regularized regression and classification for high dimensional problems, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *71*(3), 615–636.

[111] Xiong, L., X. Chen, T.-K. Huang, J. Schneider, and J. G. Carbonell (2010), Temporal collaborative filtering with bayesian probabilistic tensor factorization, in *Proceedings of SIAM Data Mining*, vol. 2010.

[112] Xu, B., J. Bu, C. Chen, D. Cai, X. He, W. Liu, and J. Luo (2011), Efficient manifold ranking for image retrieval, in *Proc. of the 34th international ACM SIGIR conf. on Research and development in Information*, pp. 525–534.

[113] Yang, S., Z. Pan, X. Shen, P. Wonka, and J. Ye (2012), Fused multiple graphical lasso, *arXiv preprint arXiv:1209.2139*.

[114] Zhao, M., and V. Saligrama (2009), Anomaly detection with score functions based on nearest neighbor graphs, in *Advances in Neural Information Processing Systems 22*, pp. 2250–2258.

[115] Zheng, V. W., B. Cao, Y. Zheng, X. Xie, and Q. Yang (2010), Collaborative filtering meets mobile recommendation: A user-centered approach, in *Proceedings of the 24rd AAAI Conference on Artificial Intelligence*.

[116] Zhou, D., O. Bousquet, T. Lal, J. Weston, and B. Schölkopf (2004), Learning with local and global consistency, *Adv. in neural information proc. sys.*, *16*, 321–328.

[117] Zhou, D., J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf (2004), Ranking on data manifolds, *Adv. in neural information proc. sys.*, *16*, 169–176.

[118] Zhou, Z.-H., and M.-L. Zhang (2006), Multi-instance multi-label learning with application to scene classification, *Advances in Neural Information Processing Systems*, *12*, 1609–1616.

[119] Zinkevich, M., M. Weimer, L. Li, and A. J. Smola (2010), Parallelized stochastic gradient descent, in *Advances in Neural Information Processing Systems*.

[120] Zloof, M. (1975), Query by example, in *Proc. of the national comp. conf. and exposition*, pp. 431–438, ACM.