Metric Learning Revisited

New Approaches for Supervised and Unsupervised Metric Learning with Analysis and Algorithms

Karim Tamer Abou-Moustafa

Department of Electrical & Computer Engineering
McGill University
Montréal, Canada

December 2011

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

2011/12/14

Abstract

In machine learning one is usually given a data set of real high dimensional vectors $\mathcal{X}$, based on which it is desired to select a hypothesis θ from the space of hypotheses Θ using a learning algorithm. An immediate assumption that is usually imposed on $\mathcal{X}$ is that it is a subset from the very general embedding space $\mathbb{R}^p$ which makes the Euclidean distance $\|\cdot\|_2$ to become the default metric for the elements of $\mathcal{X}$. Since various learning algorithms assume that the input space is $\mathbb{R}^p$ with its endowed metric $\|\cdot\|_2$ as a (dis)similarity measure, it follows that selecting hypothesis θ becomes intrinsically tied to the Euclidean distance.

Metric learning is the problem of selecting a specific metric $d_{\mathcal{X}}$ from a certain family of metrics $\mathbb{D}$ based on the properties of the elements in the set $\mathcal{X}$. Under some performance measure, the metric $d_{\mathcal{X}}$ is expected to perform better on $\mathcal{X}$ than any other metric $d \in \mathbb{D}$. If the learning algorithm replaces the very general metric $\|\cdot\|_2$ with the metric $d_{\mathcal{X}}$, then selecting hypothesis θ will be tied to the more specific metric $d_{\mathcal{X}}$ which carries all the information on the properties of the elements in $\mathcal{X}$.

In this thesis I propose two algorithms for learning the metric $d_{\mathcal{X}}$; the first for supervised learning settings, and the second for unsupervised, as well as for supervised and semi-supervised settings. In particular, I propose algorithms that take into consideration the structure and geometry of $\mathcal{X}$ on one hand, and the characteristics of real world data sets on the other. However, if we are also seeking dimensionality reduction, then under some mild assumptions on the topology of $\mathcal{X}$, and based on the available *a priori* information, one can learn an embedding for $\mathcal{X}$ into a low dimensional Euclidean space $\mathbb{R}^{p_0}$, $p_0 \ll p$, where the Euclidean distance better reveals the similarities between the elements of $\mathcal{X}$ and their groupings (clusters). That is, as a by-product, we obtain dimensionality reduction together with metric learning.

In the supervised setting, I propose PARDA, or Pareto discriminant analysis for discriminative linear dimensionality reduction. PARDA is based on the machinery of multi-objective optimization; simultaneously optimizing multiple, possibly conflicting, objective functions. This allows PARDA to adapt to the class topology in the lower dimensional space, and naturally handles the class masking problem that is inherent in Fisher's discrim-

inant analysis framework for multiclass problems. As a result, PARDA yields significantly better classification results when compared with modern techniques for discriminative dimensionality reduction.

In the unsupervised setting, I propose an algorithmic framework, denoted by $\mathbb{X}$ (note the different notation), that encapsulates spectral manifold learning algorithms and gears them for metric learning. The framework $\mathbb{X}$ captures the local structure and the local density information from each point in a data set, and hence it carries all the information on the varying sample density in the input space. The structure of $\mathbb{X}$ induces two distance metrics for its elements, the Bhattacharyya-Riemann metric $d_{B\mathcal{R}}$ and the Jeffreys-Riemann metric $d_{J\mathcal{R}}$. Both metrics reorganize the proximity between the points in $\mathcal{X}$ based on the local structure and density around each point. As a result, when combining the metric space $(\mathbb{X}, d_{B\mathcal{R}})$ or $(\mathbb{X}, d_{J\mathcal{R}})$ with spectral clustering and Euclidean embedding, they yield significant improvements in clustering accuracies and error rates for a large variety of clustering and classification tasks.

Sommaire

Dans l'apprentissage de machine, on a généralement un ensemble de données réelles de vecteurs $\mathcal{X}$ à hautes dimensions, à partir duquel il est désiré de sélectionner une hypothèse θ parmi l'espace des hypothèses Θ à travers un algorithme d'apprentissage. Une supposition immédiate généralement imposée sur $\mathcal{X}$, est qu'il est un sous-ensemble de l'espace intégral $\mathbb{R}^p$ qui par défaut transforme la distance Euclidienne $\|\cdot\|_2$ pour devenir la métrique pour les éléments de $\mathcal{X}$. Puisque il y'a plusieurs algorithmes d'apprentissage qui supposent que l'espace d'entrée est $\mathbb{R}^p$ avec son métrique $\|\cdot\|_2$ comme une mesure de similarité (ou même une mesure de différence), donc la sélection de l'hypothèse θ devient liée à la distance Euclidienne intrinsèquement.

L'apprentissage de métrique est le problème de sélectionner une métrique spécifiques $d_{\mathcal{X}}$ à partir d'une certaine famille de métriques $\mathbb{D}$ basée sur les propriétés des éléments de l'ensemble $\mathcal{X}$. Sous certaines mesures de performance, la métrique $d_{\mathcal{X}}$ devrait performer sur $\mathcal{X}$ mieux que n'importe quelle autre métrique $d \in \mathbb{D}$. Si l'algorithme d'apprentissage remplace la métrique très générale $\|\cdot\|_2$ avec la métrique $d_{\mathcal{X}}$, la sélection de l'hypothèse sera liée à la métrique la plus spécifique $d_{\mathcal{X}}$ qui transporte toutes les informations sur les propriétés des éléments de $\mathcal{X}$.

Dans cette thèse, je propose deux algorithmes pour l'apprentissage de la métrique $d_{\mathcal{X}}$; le premier pour l'apprentissage supervisé, et le deuxième pour l'apprentissage non-supervisé, ainsi que pour l'apprentissage supervisé et semi-supervisé. En particulier, je propose des algorithmes qui prennent en considération la structure et la géométrie de $\mathcal{X}$ d'une part, et les caractéristiques des ensembles de données du monde réel d'autre part. Cependant, si on cherche également la réduction de dimension, donc sous certaines hypothèses légères sur la topologie de $\mathcal{X}$, et en même temps basé sur des informations disponibles a priori, on peut apprendre une intégration de $\mathcal{X}$ dans un espace Euclidien de petite dimension $\mathbb{R}^{p_0}$, $p_0 \ll p$, où la distance Euclidienne révèle mieux les ressemblances entre les éléments de $\mathbb{X}$ et leurs groupements (clusters). Alors, comme un sous-produit, on obtient simultanément une réduction de dimension et un apprentissage métrique.

Pour l'apprentissage supervisé, je propose PARDA, ou Pareto discriminant analysis,

pour la discriminante réduction linéaire de dimension. PARDA est basé sur le mécanisme d'optimisation à multi-objectifs; optimisant simultanément plusieurs fonctions objectives, éventuellement des fonctions contradictoires. Cela permet à PARDA de s'adapter à la topologie de classe dans un espace dimensionnel plus petit, et naturellement gère le problème de masquage de classe associé au discriminant Fisher dans le cadre d'analyse de problèmes à multi-classes. En conséquence, PARDA permet des meilleurs résultats de classification par rapport aux techniques modernes de réduction discriminante de dimension.

Pour l'apprentissage non-supervisés, je propose un cadre algorithmique, noté par $\mathbb{X}$, qui encapsule les algorithmes spectraux d'apprentissage formant an algorithme d'apprentissage de métrique. Le cadre $\mathbb{X}$ capture la structure locale et la densité locale d'information de chaque point dans un ensemble de données, et donc il porte toutes les informations sur la densité d'échantillon différente dans l'espace d'entrée. La structure de $\mathbb{X}$ induit deux métriques de distance pour ses éléments: la métrique Bhattacharyya-Riemann $d_{B\mathcal{R}}$ et la métrique Jeffreys-Riemann $d_{J\mathcal{R}}$. Les deux mesures réorganisent la proximité entre les points de $\mathcal{X}$ basé sur la structure locale et la densité autour de chaque point. En conséquence, lorsqu'on combine l'espace métrique $(\mathbb{X}, d_{B\mathcal{R}})$ ou $(\mathbb{X}, d_{J\mathcal{R}})$ avec les algorithmes de "spectral clustering" et "Euclidean embedding", ils donnent des améliorations significatives dans les précisions de regroupement et les taux d'erreur pour une grande variété de tâches de clustering et de classification.

DECLARATION

This thesis presents original scholarship by the author. The results, analysis, and views reported throughout the thesis reflect the work done largely by the author with assistance offered by Prof. Frank Ferrie (from McGill University) and Prof. Fernando De La Torre (from Carnegie Mellon University), as is reflected in the co-authorship of publications arising from this thesis described in Section 1.2. The primary contributor to the development of algorithms, the analysis of their correctness, and their implementations, as well as the technical descriptions, was the author of this thesis.

Acknowledgments

This thesis is the culmination of years of research work – from Sept. 2005 to Apr. 2011 – at McGill University. During that period, I received a lot of help, support, and encouragement from family, faculty, and friends. However, first and foremost, I would like to thank God for his guidance, blessings, and for his endless gifts in my life. Second, comes my family. I am indebted for my parents *Capt.* Tamer Abou-Moustafa & Nadia Sadek, Shady Abou-Moustafa, *Capt.* Aly Sadek, Souzan Traboulsi, and Sherif Sadek for their endless love, support and encouragement during all my life. Mom and Dad, it would have been impossible for me to get to this point in my life without your tremendous effort and sacrifices during all these years. Hala Moustafa, my wife, is my real partner throughout every aspect of this thesis. Hala offered me endless love and support during my years at McGill, as well as all the stability I needed to accomplish this work. Despite the stressful circumstances of a PhD student life, Hala was very successful in making those years very pleasant. Salma, your Mom and I are very grateful for every moment we spend in our life with you.

At McGill University, first of all, I would like to thank my advisor Frank P. Ferrie for his guidance, support, and confidence despite my research that took an orthogonal path to the main path of APL. Frank helped me accomplish the work in this thesis in various ways. He always tracked technical details while never losing the bird eye view for the main problems in computer vision. Frank established my understanding of how most computer vision and machine learning problems are in fact ill-posed inverse problems. He has always amazed me with his inherent ability to abstract and summarize complex scientific questions, and further make the clear distinction between these fundamental scientific questions and high level engineering problems. Every discussion between us served, and still continues to serve as a learning experience for me. Frank provided an excellent research environment without distractions, and he was always available whenever I needed him. Last but not least[1], I would like to thank him for bankrolling many years of graduate school, giving me the opportunity to attend the machine learning summer school at Purdue University, and supporting my application for the FQRNT postdoctoral fellowship.

I thank my thesis committee, James Clark and Greg Dudek for many helpful comments and discussions. I thank James for his suggestion to consider Riemannian manifolds and geometry, giving me his copy of information geometry (by Amari & Nagoka), and for supporting my application for the FQRNT International training award. I am also grateful for Xiao-Wen Chang from the School of Computer Science at McGill. Chang introduced me to the world of matrix

[1]Frank also convinced me to switch to the Mac world.

computations and numerical estimation which are fundamental for the research work presented here. I learned a lot from his classes, lecture notes, and subside discussions.

From Université Laval, QC, I would like to thank Denis Laurendeau for supporting my FQRNT International training award and orchestrating the whole process from its beginning till its end. Here, I can not forget Doina Precup and Kaleem Siddiqi from the School of Computer Science at McGill. Doina pointed me to the great importance of internships in the machine learning community and encouraged me to consider this option. She also suggested to talk to Kaleem for funding resources, who gratefully informed me on the availability of the FQRNT award. Thank you Kaleem.

It is now the turn to thank my mentor Fernando De La Torre from Carnegie Mellon University, Pittsburgh, PA, whom I stayed in his lab from April 2009 till October 2009. Fernando introduced me to linear dimensionality reduction and Fisher's discriminant analysis as the dual of the metric learning problem. He also pointed me to the masking problem in the mutliclass case which I address in Chapter 4 using the Pareto framework. I learned an enormous deal on component analysis from our daily discussions, and from our collaboration in research papers. Finally, I thank him for supporting my application for the FQRNT postdoctoral fellowship.

Special thanks to my colleague Mohak Shah, a former postdoctoral fellow at CIM and currently at Accenture Technology Labs, for all the interesting discussions related to machine learning, and to my research work. In particular, Mohak suggested that convolution kernels would be a more elegant entry point for the Bhattacharyya-Riemann and the Jeffreys-Riemann metrics in Chapter 5. He also helped in proofreading the proofs in Chapter 5.

From CIM, I thank my colleagues Prasun Lala, Shufei Fan, Prakash Patel, Andrew Phan, and Amin Abolhassani for their continuous help with my manuscripts and reports, discussions, and the good times we had at CIM. I also thank Catherine Laporte for proofreading various manuscripts, Maxime Boucher, Mathew Toews, John Harrison, Isabel Begin, Rola Hamrouch, Scott McCloskey, Peter Savadjiev, and more recently Sean Lawlor for all the interesting discussions at CIM. Life at CIM and McGill could not have been smoother without Cynthia Davidson and Marlene Gray on the administrative and managerial side. On the system administration side, I thank Jan Binder and Patrick McLean for their unmatched prompt replies to any issue related to the computing facilities at CIM.

Last but not least, I would like to thank Ching Y. Suen from Concordia University and Mohamed Cheriet from ETS Montréal for the valuable research experience during my Masters degree, Adam Krzyzak from Concordia University for his approach to statistical pattern recognition and machine learning, Yasser El-Sonbaty for my first project in pattern recognition, Magdy Saeb, Mahmoud El-Haddad, and Hisham Farouk Anan – all from the Arab Academy for Science and Technology, Alexandria, Egypt – for intriguing me about research in computer science. Our friends, Haytham, Iman, Hazem, Dina, Ziad and Dahlia, thank you for all the good times and cherishable moments we had together.

Contents

List of Figures

List of Tables

Notation

a, b, c or i, j, k	Non-bold lower-case letters represent scalar variables (a, b, c) or indexes (i, j, k).
$\mathbf{x}, \mathbf{y}, \mathbf{z}$	Bold lower-case letters are vectors.
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	Bold upper-case letters are matrices.
$\mathcal{X}, \mathcal{Y}, \mathcal{R}$	Calligraphic upper-case letters are usually sets or manifolds.
$\mathbb{R}, \mathbb{S}$	Double bold upper-case letters are spaces (or manifolds).
n	Number of samples.
p	Number of input features.
p_0	The intrinsic dimensionality of the data (defined below).
$\mathbb{R}^p$	The p-dimensional Euclidean space.
$\mathbf{I}$	The identity matrix.
$\mathbf{A} \succ 0$, $\mathbf{A} \succeq 0$	Matrix $\mathbf{A}$ is positive definite (PD) or positive semi-definite (PSD) respectively.

$\mathbb{S}_{++}^{p\times p}$	The space (or manifold) of symmetric PD matrices of dimension $p \times p$.
$\mathcal{G}(\ \cdot\ ; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	The multivariate Gaussian distribution with mean vector $\boldsymbol{\mu} \in \mathbb{R}^p$, and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{S}_{++}^{p\times p}$.
$\mathrm{tr}(\mathbf{A})$	$\mathrm{tr}(\mathbf{A}) = \sum_i a_{ii}$ is the trace of the square matrix $\mathbf{A} \in \mathbb{R}^{p\times p}$.
$\|\cdot\|_2$	The Euclidean norm.
$\|\cdot\|_{\mathbf{A}}$	Euclidean norm weighted by the symmetric and PD matrix $\mathbf{A}$. This is also known as the generalized quadratic distance (GQD). See below for more details.
$\langle\cdot,\cdot\rangle$	The dot product operator.

Chapter 1

Introduction

There are various applications of machine learning, pattern recognition, and computer vision in which one is faced with a data set of n objects $\mathscr{D} = \{D_1, \ldots, D_n\}$, based on which it is desired to select a hypothesis θ from the space of hypotheses Θ using a learning algorithm. For instance, θ can be a hypothesis for classification, clustering, or density estimation, while D_i can be an image, a video sequence, a text document, or a gene under some test condition, to mention a few. Using some domain knowledge, each object D_i is usually represented as a high dimensional feature vector $\mathbf{x}_i \in \mathbb{R}^p$, resulting in a data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n$[1]. Further, it is assumed that $\mathcal{D} \subset \mathcal{X}$, where $\mathcal{X}$ is known as the input space, and the elements of $\mathcal{X}$ are assumed to be independent and identically distributed (i.i.d) samples from an unknown probability distribution $\Pr(X = \mathbf{x})$.

Once this setting is established, an *immediate* assumption that is usually imposed on $\mathcal{D}$ is that $\mathcal{X}$ is a subset from $\mathbb{R}^p$, with the Euclidean distance $\|\cdot\|_2$ becoming the default metric for the elements of $\mathcal{X}$. Since various learning algorithms assume that the input space is $\mathbb{R}^p$ with its endowed metric $\|\cdot\|_2$ as a (dis)similarity measure, it follows that selecting hypothesis θ becomes intrinsically tied to the Euclidean distance. This is indeed the case for various learning algorithms such as the k nearest neighbour (k-NN) classifier [2], radial basis functions [3], logistic regression [4], the perceptron [5], neural networks [6], linear support vector machines [7, 8], k-means clustering [9], and many others.

[1] There are various other scenarios in which each object D_i is represented as a time-series (or sequential) pattern, a bag of features, or more generally, as a one set of vectors $\mathcal{S}_i$. This setting will be slightly covered in Chapter 6.

The main motivation for this thesis is to investigate the over simplified, and rather unjustified assumption that $\mathcal{X} \subset \mathbb{R}^p$, with the consequence that the Euclidean distance is the metric for the elements of $\mathcal{X}$. Although it is true that the elements of $\mathcal{X}$ are vectors in $\mathbb{R}^p$, the input space $\mathcal{X}$ is rarely Euclidean [10]. As will be shown in the next chapter, the input space $\mathcal{X}$ has more special properties than the very generic space $\mathbb{R}^p$, and these properties should be exploited for better hypothesis learning.

To see this from a different perspective, consider for instance two data sets from two different domains; one for face images, and one for genes under different test conditions. Let each element in each data set be represented by the most standard and widely acceptable features as a high dimensional vector. Now, given this setting, it is legitimate to ask the following questions: Is it possible that due to the unified representation for a face and a gene as vectors in $\mathbb{R}^{p_1}$ and $\mathbb{R}^{p_2}$ respectively, that the Euclidean distance measure is suitable for both data sets? Is the Euclidean distance a universal metric for any data set from any domain as long as it is represented as vectors in $\mathbb{R}^p$? What does the Euclidean distance between two genes, or two faces mean? In this thesis, I try to give answers to these questions based on the literature for metric learning [11, 12, 13, 14, 15, 16, 17], manifold learning algorithms, and spectral methods [18, 19, 20, 21, 22, 23, 24, 25, 26, 27].

In particular, I propose two new approaches for learning a low dimensional (semi-)metric space[2] $(\mathcal{M}, d_{\mathcal{M}})$ from $\mathcal{X}$; one in the supervised multiclass setting, and the other in the unsupervised learning setting, such that the similarities between the elements of $\mathcal{X}$, their structure, and their groupings (clusters) are revealed by the (semi-)metric $d_{\mathcal{M}}$. Both approaches are realized by algorithms: Pareto discriminant analysis (PARDA) for supervised multiclass dimensionality reduction [28], and an algorithmic framework – denoted for now by $\mathbb{X}$ – that encapsulates spectral learning algorithms, and gears them for unsupervised metric space learning [29, 30, 31]. Both approaches, and consequently the research here, are motivated by the following questions:

- When is the Euclidean distance a useful metric for the points in $\mathcal{X}$?
- If each $\mathbf{x}_i$ is associated with a class label y_i, where $y_i \in \mathcal{Y} = \{\mathcal{C}_1, \ldots, \mathcal{C}_c\}$ and c is

[2] See Appendix.

the number of classes in the data, can one learn a metric function specifically for $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Y}$ that better discriminates between points from different classes?

- A more challenging setting is the following – given only the data set $\mathcal{D} \subset \mathcal{X}$, and without labels nor side information, can one learn a metric function specifically for $\mathcal{X}$ in an unsupervised manner?

- Alternatively, since various learning algorithms assume that the input space is $\mathbb{R}^p$ and rely on the Euclidean distance as a metric, can one learn a low dimensional Euclidean embedding for $\mathcal{X}$, such that the natural clusters and groupings in the data are manifested by the Euclidean distance?

1.1 Thesis Organization

In Chapter 2, I motivate the problem of metric learning and show how it is strongly tied to machine learning algorithms. Next, I argue that the input space $\mathcal{X}$ should not be simply treated as a subset from the general Euclidean space $\mathbb{R}^p$. This is due to the topological structure and geometry of $\mathcal{X}$ on one hand, and the incoherence of real world data with the geometry of Euclidean spaces on the other. These arguments suggest that one should learn a metric $d_{\mathcal{X}}$ that takes the topology, geometry, and the characteristics of real world data into consideration. However, if only some mild assumptions are made on the topology and geometry of $\mathcal{X}$, and depending on the available *a priori* information, one can learn an embedding for $\mathcal{X}$ into a low dimensional Euclidean space $\mathbb{R}^{p_0}$, $p_0 \ll p$, where the Euclidean distance better reveals the similarities between the elements of $\mathcal{X}$ and their groupings (clusters). That is, as a by-product, we obtain dimensionality reduction together with metric learning. These are the main ideas underlying the algorithms in Chapters 4 and 5.

In Chapter 3, I briefly review the early ideas of metric learning that appeared in [11, 12, 13, 14, 15, 16, 17], followed by my initial work on local learning of a Mahalanobis metric for query based operations [32, 33, 34, 35], and I close the chapter with a quick review for the literature of spectral manifold learning algorithm [25, 26, 18, 19, 36, 27, 20, 21, 23, 24].

In Chapter 4, I consider the problem of learning a low dimensional embedding for the data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$ when the *a priori* information in the form of labels $y_i \in \mathcal{Y}$ are available for learning. This problem can be seen as a special case of the metric learning problem in which one learns an instance from the generalized quadratic distance (GQD): $d(\mathbf{x}, \mathbf{y}; \mathbf{A}) = \sqrt{(\mathbf{x}-\mathbf{y})^\top \mathbf{A}(\mathbf{x}-\mathbf{y})}$, where $\mathbf{A} \succ 0$, and $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$.

Here, we rely on the framework for Fisher's linear discriminant analysis (LDA) in the multiclass setting for learning a projection matrix $\mathbf{B} \in \mathbb{R}^{p \times p_0}$, where $p_0 \ll p$. I propose a new algorithm, namely Pareto discriminant analysis (PARDA) [28], for Fisher's LDA that is based on the machinery of multiobjective optimization [37, 38]. PARDA decomposes the multiclass problem into a set of pairwise objective functions representing the pairwise distance between different classes. Unlike existing extensions of Fisher's LDA to multiclass problems that typically maximize the sum of pairwise distances between classes, PARDA simultaneously maximizes each pairwise distance, encouraging the case where all classes are equidistant from each other in the lower dimensional embedding space. Solving PARDA is a multiobjective optimization problem – simultaneously optimizing multiple, possibly conflicting, objective functions – and the resulting solution is known to be "Pareto Optimal".

PARDA adapts to the class topology in the lower dimensional subspace, and hence it naturally overcomes the class masking problem that is inherent in Fishers' LDA for the multiclass setting. As a result, PARDA finds subspaces that improve the separation between classes, which finally results with lower error rates when compared with modern methods for discriminative linear dimensionality reduction methods. To the best of my knowledge, this is the first research to address the multiclass linear dimensionality reduction problem as a multiobjective minimization problem. Further, in Chapter 7, it will be shown that PARDA can define a general framework for learning discriminative linear dimensionality reduction models.

In Chapter 5, I consider the problem of learning a low dimensional metric space for the data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$ when no *a priori* information in the form of labels or side-information are available for learning. Here I propose a two-step algorithmic framework for learning a metric space, based on spectral methods, in an unsupervised manner. In the first step, the algorithm extracts local density information from each point $\mathbf{x}_i \in \mathcal{D}$ and

forms an augmented data set $\mathcal{D}_A = \{(\mathbf{x}_i, \mathbf{A}_i)\}_{i=1}^n$, where $\mathbf{A}_i \succ 0$. The augmented data set $\mathcal{D}_A$ is a subset from what is defined in Chapter 5 as the augmented data space $\mathbb{X}$, where $\mathbb{X} \subset \mathbb{R}^p \times \mathbb{S}_{++}^{p\times p}$, and $\mathbb{S}_{++}^{p\times p}$ is the space of symmetric positive definite (PD) matrices. The motivation for $\mathbb{X}$ is to accommodate the characteristics of real world data sets and the uneven sample distribution in the input space. As will be shown, the augmented data set $\mathcal{D}_A$ carries all the information on the varying sample density in $\mathcal{D}$.

In the second step, spectral embedding algorithms are used to embed the augmented data set $\mathcal{D}_A$ into a low dimensional Euclidean space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$. That is, unlike the traditional setting where spectral algorithms are directly applied on $\mathcal{D}$, here spectral methods are applied on the augmented data set $\mathcal{D}_A$ which carries the information on the varying density in the input space $\mathcal{X}$. However, to apply spectral methods on $\mathcal{D}_A$, a similarity or a distance measure needs to be defined over the 2-tuples $(\mathbf{x}_i, \mathbf{A}_i)$. Based on convolution kernels, I introduce the relaxed exponential kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ for the augmented space $\mathbb{X}$, which naturally induce two *corrected divergence measures* that adhere to the five metric axioms[3]; the Jeffreys-Riemann metric $d_{J\mathcal{R}}$, and the Bhattacharyya-Riemann metric $d_{B\mathcal{R}}$.

Due to the metric properties of $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$, I show, using the results of Young & Householder [39], and Gower & Legendre [40], that the metric spaces $(\mathbb{X}, d_{J\mathcal{R}})$ and $(\mathbb{X}, d_{B\mathcal{R}})$ can be embedded in a low dimensional Euclidean space using classical multidimensional scaling (MDS) [39, 41, 42]. Also, based on the results of Scheonberg [43], I show that the kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ can embed $(\mathbb{X}, d_{J\mathcal{R}})$ and $(\mathbb{X}, d_{B\mathcal{R}})$ in a low dimensional Euclidean space using Laplacian embedding [25, 26, 27, 20].

The metric spaces $(\mathbb{X}, d_{J\mathcal{R}})$ and $(\mathbb{X}, d_{B\mathcal{R}})$ reorganize the proximity between the points in $\mathcal{D}$ based on $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ respectively, which take the varying local density of the input space into consideration, and respect the geometry of $\mathbb{R}^p$ and $\mathbb{S}_{++}^{p\times p}$. This is unlike the GQD type measures, including the Euclidean distance, that are constant over the entire input space and do not take this varying density into consideration. This makes the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ more suitable for the characteristics of real world data sets, and the uneven sample distribution in the input space. As will be shown in Chapter 5, the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ significantly improve the performance of spectral clustering on data sets from various domains.

[3] See Appendix.

As an application of the developed metrics in Chapter 5, in Chapter 6 I consider the problem of learning a hypothesis (classification and clustering) over sets of vectors (SOVs), a.k.a bags of features, that appeared in the work of Kondor & Jebara [44], and Moreno *et al.* [45]. Interestingly, the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$, and the relaxed kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ naturally fit in this setting and will be used as distance and similarity measures for this type of data. I will show that these measures together with Laplacian and Euclidean embeddings, can be used for classification and clustering of SOVs, and they usually lead to better results than the measures proposed in [44] and [45]. This will be demonstrated using preliminary experiments for classification of human actions and clustering of human motion in video sequences.

1.2 Contributions

This thesis draws from the areas of metric learning, linear discriminant analysis, nonlinear dimensionality reduction, spectral and manifold learning algorithms, and kernel methods. Its contributions also lie across these areas. Specifically, this thesis advances the following developments:

- A new algorithm for discriminative linear dimensionality reduction using the framework of multiobjective optimization [28]. The algorithm takes in to consideration the class topology in the lower dimensional subspace, and hence it naturally leverage the class masking problem that is inherent in Fisher's LDA multiclass problems. PARDA was presented in CVPR 2010 [28], and its journal version is under preparation. Prof. De La Torre introduced me to Fisher's LDA as a method for metric learning, together with their class masking (merging) problem. I analyzed the problem, proposed the solution based on multiobjective optimization, developed all the algorithms, and carried all the experimental results. Prof. De La Torre and Prof. Ferrie helped in theoretical discussions and in writing the manuscripts.

- An algorithmic framework that encapsulates the metric spaces $(\mathbb{X}, d_{J\mathcal{R}})$, or $(\mathbb{X}, d_{B\mathcal{R}})$ with spectral manifold learning algorithms to learn an embedding for $\mathcal{X}$ into a low dimensional Euclidean space $\mathbb{R}^{p_0}$ [29, 30]. The framework overcomes the limitations of generalized quadratic distance type measures, and offers a means to handle the

uneven sample distribution in the input space. Further, the framework captures the local structure and the local density information for each point in the data set, which is finally manifested by the metrics $d_{J\mathcal{R}}$ & $d_{B\mathcal{R}}$.

- Two corrected divergence measures for Gaussian densities that adhere to all metric axioms; namely the Jeffreys-Riemann metric $d_{J\mathcal{R}}$ and the Bhattacharyya-Riemann metric $d_{B\mathcal{R}}$ [29, 30] . The metrics $d_{J\mathcal{R}}$ & $d_{B\mathcal{R}}$ give a new meaning for the distance between points based on the local structure and the local density around each point. That is, two points are close or similar to each other, when they are physically close to each other in the input space, and the local structure and density around each point are very similar. When the metric space $(\mathbb{X}, d_{B\mathcal{R}})$ is combined with spectral clustering, it yields significant improvements in clustering accuracy for a large variety of data sets. The research work on the metrics $d_{J\mathcal{R}}$ & $d_{B\mathcal{R}}$ appeared for the first time in [29], where we also derived their respective kernels. In this work, Prof. Ferrie and Prof. De La Torre helped in theoretical discussions and in writing the manuscript. Mohak Shah, the second co-author in [30], suggested that another entry point to obtain the metrics $d_{J\mathcal{R}}$ & $d_{B\mathcal{R}}$, can be via convolution kernels. Hence, we reintroduced the metrics $d_{J\mathcal{R}}$ & $d_{B\mathcal{R}}$ from that kernel perspective in [30]. The work on the augmented space $\mathbb{X}$ with the metrics $d_{J\mathcal{R}}$ & $d_{B\mathcal{R}}$ as a general framework that can encapsulate manifold learning algorithms is only presented in this thesis so far.

- A framework for unifying the representation for sets of vectors (or bags of features) based on the metrics $d_{J\mathcal{R}}$ & $d_{B\mathcal{R}}$ [31]. The framework has the following properties. (1) It allows any learning algorithm to be transparently applied on SOVs through their images residing in a low dimensional subspace. (2) The framework offers a reduction, by orders of magnitude, in the data's space complexity, which correlates directly with the computational complexity of the learning algorithm, resulting in significantly faster hypothesis learning. (3) The framework is unsupervised, and hence it does not require labels nor side-information. However, if labels or side-information are available, they can be naturally integrated into the framework. (4) The spectral embedding algorithm in the framework reveals the natural clusters in the sets of vector. (5) The framework has a well defined generalization to out-of-sample examples using the Nyström formula, and hence it does not require retraining the system whenever new data are available. This research work was presented in [31], and Prof.

Ferrie helped in theoretical discussions, and in writing the manuscript.

Chapter 2

Motivation

The problem of metric learning can be informally described as inferring the mutual distances between a set of objects. The inference process should take into consideration the nature of the objects in terms of their structure, and their relative differences such that the distance between similar objects should always be smaller than the distance between less or non similar objects. The problem of metric learning is strongly tied to machine learning algorithms. To see this, it is necessary to have a formal understanding of the definition of metric spaces, from which the problem of metric learning can be defined, as well as an understanding of how learning algorithms interact with metric spaces.

A metric space [46, p. 3] is an ordered pair $(\mathcal{X}, d)$, where $\mathcal{X}$ is a non-empty abstract set (of any objects/elements whose nature is left unspecified), and d is a distance function, or a metric, defined as: $d : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$, and $\forall\ a, b, c \in \mathcal{X}$, the following axioms hold:

1. $d(a, b) \geq 0$,
2. $d(a, a) = 0$,
3. $d(a, b) = 0$ iff $a = b$,
4. Symmetry : $d(a, b) = d(b, a)$, and
5. The triangle inequality : $d(a, c) \leq d(a, b) + d(b, c)$[1].

[1] A semi-metric distance satisfies Axioms (1), (2) and (4) only. That is, the triangle inequality need not hold for semi-metrics, and $d(a, b)$ can be zero for any a, b and $a \neq b$. For instance, $(\mathbb{R}^p, \|\cdot\|_2)$ is a metric space, while $(\mathbb{R}^p, \|\cdot\|_2^2)$ is a semi-metric space.

In metric learning, one is given the set $\mathcal{X}$ with the requirement of selecting a specific metric $d_{\mathcal{X}} \in \mathbb{D}$ based on the properties of the elements in the set $\mathcal{X}$, where $\mathbb{D}$ is a certain family of metrics (or semi-metrics). Under some performance measure, the metric $d_{\mathcal{X}}$ is expected to perform better on $\mathcal{X}$ than any other metric $d \in \mathbb{D}$. Given this understanding for metric learning and metric spaces, let us see how machine learning algorithms interact with $\mathcal{X}$.

2.1 Machine Learning Algorithms and Metric Spaces

In machine learning, there are two fundamental spaces; the input space $\mathcal{X} \sim \Pr(X)$, and the space of hypotheses Θ [10]. Learning algorithms select a hypothesis $\theta \in \Theta$ based on the training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Y}$ in the supervised learning case, or based on the set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$ in the unsupervised learning case. By selecting a hypothesis $\theta \in \Theta$ based on $\mathcal{D}$, the learning algorithm, either implicitly or explicitly, assumes that $\mathcal{X}$ and Θ are embedded in some space. Indeed, there are various algorithms such as the k-NN classifier [2], radial basis functions [3], logistic regression [4], the perceptron [5], neural networks [6], linear support vector machines [7, 8], and k-means clustering [9], that assume the data is embedded in the metric space $(\mathbb{R}^p, \|\cdot\|_2)$. Note that in theory, these algorithms just assume the existence of any metric space as defined above, and not necessarily the Euclidean space. However, as mentioned earlier, an object D is usually transformed by the feature extraction process into a vector $\mathbf{x} \in \mathbb{R}^p$, and in practice, it become *easier* to assume that $(\mathbb{R}^p, \|\cdot\|_2)$ is the data's embedding space. Therefore, selecting the hypothesis θ becomes intrinsically tied to the metric space $(\mathbb{R}^p, \|\cdot\|_2)$.

However, $\|\cdot\|_2$ is the metric for the very general embedding space $\mathbb{R}^p$. If this general metric is replaced by $d_{\mathcal{X}}$, which is specifically for $\mathcal{X}$, then the learning algorithm will use the more specific metric space $(\mathcal{X}, d_{\mathcal{X}})$ for selecting $\theta \in \Theta$. In fact, the input space $\mathcal{X}$ has more specific properties than the very generic space $\mathbb{R}^p$; these properties should be exploited for better hypothesis learning and are discussed in the following sections.

Remark. In some applied domains such as text categorization, computer vision, bioinformatics, etc., researchers have proposed different (dis)similarity measures that were found to be better than the Euclidean distance. Indeed, this came through tremendous effort in terms of studying the nature of data in hand in each of these applied domains. However,

there are a few questions with regard to these (dis)similarity measures. A first question is whether these (dis)similarity measures define metric spaces as defined above in order to allow any learning algorithm to be transparently applied to these domains. Second, to what extent these (dis)similarity measures are coherent with the unknown $\Pr(X)$. In other words, it is a question whther these (dis)similarity measures make or do not make any assumptions on the data distribution, and whether the nonlinearity of the data and the varying density in the input space are taken into consideration.

2.1.1 The structure and geometry of $\mathcal{X}$

If $\mathcal{X} \sim \Pr(X)$ is considered as a collection of sets, which is known as the topological structure of a set, then $\mathcal{X}$ is rarely an Euclidean space $\mathbb{R}^p$ [10]. Take for instance the data space of intensity images which are usually matrices within a bounded range of intensity values. There is no real meaning for images outside that range which are obtained by addition or scalar multiplication. The same holds if one considers images represented by histograms of gradients, or short video clips represented by a histogram of orientations of optical flow vectors. These histograms always have positive values and lie within a certain range; there is no real meaning for histograms outside this range. A similar argument follows for the hypothesis space Θ. Take for instance, the set of weights for a neural network, the parameters of hidden Markov models, or the parameters for probabilistic models, such as Gaussians, exponentials, multinomials, etc. Therefore, topologically, $\mathcal{X}$ and Θ are not Euclidean.

Also note the geometric ramifications of considering $\mathcal{X}$ and Θ as an Euclidean space $\mathbb{R}^p$. The geometry of $\mathbb{R}^p$ is manifested through the Euclidean distance $\|\cdot\|_2$. When using this distance measure for $\mathcal{X}$ and Θ, we are actually imposing an artificial distance on these spaces [10]. This false imposition also holds even if the data or models are real vectors. Imposing the Euclidean distance is equivalent to detaching $\mathcal{D}$ from its context, ignoring that it is generated from a certain unknown probability distribution, and treating the points in $\mathcal{D}$ as general points in $\mathbb{R}^p$ with no common context or shared information.

2.1.2 Real world data sets and the geometry of $\mathbb{R}^p$

Another reason for avoiding the assumption that the input space $\mathcal{X}$ is directly embedded in $\mathbb{R}^p$ is the nature of real world data sets; their nature is incoherent with the geometry of

$\mathbb{R}^p$, and also incoherent with the Euclidean distance. A set of reasons for this incoherence is that data sets such as images, videos, documents, etc., are usually (i) highly structured, (ii) highly nonlinear, (iii) measured from various sources at different scales with various degrees of variability and correlation, and (iv) prone to various sources of noise that may largely deviate the measurements, raise outliers, and cause ambiguities in the data.

Another reason is the limited number of training samples. That is, the training set $\mathcal{D}$ is finite, and it is not known *a priori* if $\mathcal{D}$ is uniformly and sufficiently sampled from $\mathcal{X}$. Moreover, although it is assumed that the samples in $\mathcal{X}$ are i.i.d from $\Pr(X)$, it is not known to what extent this assumption really holds for real world data sets, given the above characteristics. As such, it is acceptable to assume that low probability areas for $\Pr(X)$ are poorly sampled, and hence poorly represented in the training set $\mathcal{D}$. These issues altogether result in what is known as the *uneven sample distribution in the input space* [47].

Let us consider the Geometry of $\mathbb{R}^p$, manifested by the Euclidean distance $\|\cdot\|_2$, under the characteristics of real world data sets and the uneven sample distribution problem. By expanding the squared Euclidean distance $\|\mathbf{x}-\mathbf{y}\|_2^2$ to $(\mathbf{x}-\mathbf{y})^\top\mathbf{I}(\mathbf{x}-\mathbf{y})$, one directly obtains an instance from the generalized quadratic distance (GQD) $d(\mathbf{x},\mathbf{y},\mathbf{A})$: $\sqrt{(\mathbf{x}-\mathbf{y})^\top\mathbf{A}(\mathbf{x}-\mathbf{y})}$, where $\mathbf{A}$ is a symmetric positive definite (PD) matrix, $\mathbf{I}$ is the identity matrix, and $\mathbf{x},\mathbf{y}\in\mathbb{R}^p$. From a statistical vantage point, the Euclidean distance is the optimal metric if the data are generated from a spherical Gaussian distribution with equal variances and zero correlations among the variables – *the spherical assumption*[2]. This is not only a hard to attain natural setting in real world data sets, it is at the other extreme from their characteristics described above.

Thus, an inherent limitation of the Euclidean distance, and more generally the GQD, is that they are constant over the entire input space $\mathcal{X}$, and hence, they do not take into consideration the uneven sample distribution problem. For the GQD, it implies that the matrix $\mathbf{A}$ is globally defined over the whole input space, which enforces a global Gaussian assumption for the data – *the ellipsoidal assumption*. This is an unjustified assumption since a large Gaussian distribution with a full covariance matrix does not yield a faithful approximation of the true distribution $\Pr(\mathbf{x})$.

[2]When $\mathbf{A}$ is the inverse of the sample covariance matrix, the GQD is known as the Mahalanobis distance.

2.2 Learning an Embedding From $\mathcal{X}$

The discussion above suggests that the Euclidean space $\mathbb{R}^p$ should not be imposed on $\mathcal{X}$, and hence hypothesis learning should not be immediately applied in $\mathbb{R}^p$ by default. It also suggests that one should learn a metric $d_{\mathcal{X}}$ that takes the topology, geometry, and the characteristics of real world data into consideration. However, if we are also seeking dimensionality reduction, then under some mild assumptions on the topology and geometry of $\mathcal{X}$, and depending on the available *a priori* information, one can learn an embedding for $\mathcal{X}$ into a lower dimensional Euclidean space $\mathbb{R}^{p_0}$, $p_0 \ll p$, where the Euclidean distance better reveals the structure in the data in terms of similarities and clusters.

In Chapter 4, I consider the problem of learning an embedding for the data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$ when the *a priori* information in the form of labels $y_i \in \mathcal{Y}$ are available for learning. In terms of metric learning, this can be seen as learning an instance of the GQD $\|\cdot\|_{\mathbf{A}}$, where $\mathbf{A} \in \mathbb{R}^{p\times p}$, and $\mathbf{A} \succ 0$. However, to achieve linear dimensionality reduction together with metric learning, the matrix $\mathbf{A}$ is required to be low rank; that is $\mathbf{A} \succeq 0$, and $\mathbf{A} = \mathbf{B}\mathbf{B}^\top$, where $\mathbf{B} \in \mathbb{R}^{p\times p_0}$, and $p_0 \ll p$. Moreover, since labels are available for learning, it is required that the matrix $\mathbf{B}$ linearly projects the data into a subspace that better discriminates between points from different classes.

Learning the matrix $\mathbf{B}$ becomes an instance from Fisher's linear discriminant analysis (LDA) for dimensionality reduction. However, the difference here is that this is a multiclass setting in which each class is explicitly modelled as a multivariate Gaussian distribution, with different means and covariance matrices (or heteroscedastic LDA). Note here the joint usage of the labels (*a priori* information) together with the simple assumption that each class is a multivariate Gaussian distribution. As will be shown, the proposed algorithm learns a projection matrix $\mathbf{B}$ that maximizes a separation measure between these Gaussians in a low dimensional subspace.

In Chapter 5, I consider the more challenging setting when $\mathcal{Y}$ is not available for learning. Here, the realm of Euclidean geometry is abandoned in favour of a more flexible and richer class of geometries. To consider this new geometry, one mild assumption needs to be made about $\mathcal{X}$, and this is *local smoothness*. Based on this assumption, the space $\mathcal{X}$

can be considered as a smooth differentiable manifold which is *locally Euclidean*. Manifolds are the natural generalization of Euclidean spaces to locally Euclidean spaces, and differentiable manifolds are their smooth counterparts [10]. Studying such smooth locally Euclidean spaces is the subject of Riemannian geometry [48] which is partially used in this research. Note that these assumptions are only required to hold locally around each point, and not globally on the whole data.

Based on the smoothness and locally Euclidean assumption of $\mathcal{X}$, I extract from the data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n$ a new augmented data set $\mathcal{D}_A = \{(\mathbf{x}_i, \mathbf{A}_i)\}_{i=1}^n$ that carries the local density information from the neighbourhood around each $\mathbf{x}_i$, where $\mathbf{A}_i \in \mathbb{S}_{++}^{p\times p}$. To define a (dis)similarity measure between the 2-tuples $(\mathbf{x}_i, \mathbf{A}_i)$ and $(\mathbf{x}_j, \mathbf{A}_j)$, I rely on Riemannian geometry to define the Riemannian metric $d_{\mathcal{R}}$ for the elements of $\mathbb{S}_{++}^{p\times p}$, which when combined with convolution kernels, define the relaxed exponential kernels $K_{J\mathcal{R}}$ & $K_{B\mathcal{R}}$, and the corrected divergence measures $d_{J\mathcal{R}}$ & $d_{B\mathcal{R}}$. The non-empty set $\mathcal{D}_A$ together with the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ define the metric spaces $(\mathcal{D}_A, d_{J\mathcal{R}})$ and $(\mathcal{D}_A, d_{B\mathcal{R}})$ respectively, which are embedded in the low dimensional metric space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$, using Laplacian and Euclidean embedding.

In the following chapter, I will review some background material that is complimentary and necessary for the remaining chapters. This includes, briefly reviewing some metric learning algorithms, spectral manifold learning algorithms, and finally my initial work on local learning of a Mahalanobis metric for query based operations.

Chapter 3

Methods of Metric Learning

In this chapter I briefly review the literature on metric learning, my previous work on local learning of a Mahalanobis metric for query based operations [32, 33, 34, 35], and finally spectral manifold learning algorithms.

3.1 Metric Learning

The literature on metric learning can be categorized according to three dimensions [49, 50]; 1) supervised, unsupervised or semi–supervised 2) local or global, and 3) linear or non–linear. The supervised approach is further categorized based on the type of labels which can be either in the form of class labels, pairwise distances, or pairwise constraints. The latter constraints are also known as equivalence ($+ve$) and inequivalence ($-ve$) constraints, or side information [11]. If the data are only partially labelled with any kind of the previous labels, then the algorithm that learns the metric is considered to be semi–supervised. In the following, I present a brief literature review for various metric learning algorithms that combine the different aforementioned dimensions to form groupings of like algorithms.

3.1.1 Supervised local metric learning using class labels

The earliest work on metric learning in this category dates back to 1981 with the work of Short and Fukunaga [51] where they try to minimize the difference between the finite sample nearest neighbour (NN) classification error and the asymptotic NN error (or the twice Bayes error bound). Assuming a smooth posterior and conditional densities around points,

the distance between a query point and its neighbours is weighted by the gradient of the posterior probability with respect to the query point, given the labels of the nearest neighbours. This should give a larger weight to features that are relevant to the classification task (a.k.a local feature relevance). Friedman [52] reuses the idea of local feature relevance combined with recursive partitioning of the space, in a similar spirit to decision trees, to achieve a flexible nearest neighbour metric that is adapted to each point and its neighbourhood. Hastie and Tibshirani [53] generalize the work of Short and Fukunaga by defining local linear discriminant analysis (LDA) for each query point and its neighborhood. Their neighbourhoods are in the form of ellipsoids stretched along decision boundaries between classes. Domeniconi and Gunopulos [54] use support vector machines (SVMs) to compute locally flexible metrics where the maximum margin of SVMs decides the most discriminating features (or directions) over the query point's neighborhood, and hence provides weights for each feature. In a similar vein, Domeniconi *et al.* [55] replace SVMs by Chi–squared distance analysis while Peng *et al.* [56] replace SVMs by quasiconformal kernels to achieve the same purpose. By changing class labels to fully or partially side–information, Chang and Yeung [57] learn a metric through local linear transformations of neighbourhoods. The metric is learned independently for each point and its neighbourhood through a regularized moving least squares framework with closed form solutions.

3.1.2 Supervised and semi–supervised global metric learning

In supervised and semi–supervised global metric learning using class labels or side–information, most algorithms learn a metric through the general family of Mahalanobis distances $||\mathbf{x}-\mathbf{y}||_{\mathbf{A}} = (\mathbf{x}-\mathbf{y})'\mathbf{A}(\mathbf{x}-\mathbf{y})$, where $\mathbf{A} \in \mathbb{S}_{++}^{d\times d}$ and $\mathbb{S}_{++}^{d\times d}$ is the space of square and symmetric positive definite (SPD) matrices. The differences between these algorithms are due to the context and constraints defining each metric.

Supervised global metric learning using class labels

Goldberg *et al.* [14] define a differentiable probability function (softmax) using $||\mathbf{x}-\mathbf{y}||_{\mathbf{A}}$ with $\mathbf{A}$ as its parameter. This function is optimized to maximize the probability of correct classification using the labels in the training set. In an extended work, Globerson and Roweis [15] use the same objective function to map all points that belong to the same class into a single point, i.e. collapsing the class to a single point. Weinberger *et al.* [16] search for

a matrix $\mathbf{A}$ that defines a linear transformation such that k nearest neighbours of the same class are always kept together while samples from other classes are separated by a large margin, hence the name large margin nearest neighbour classifier (LMNN). They formulate their problem as a semi–definite program with constraints derived from each point in the training set and its k nearest neighbours and solve it using convex optimization.

Supervised global metric learning using side–information

Schultz and Joachims [12] define their constraints in the form of triplet relative comparisons; i.e. for samples $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ the relative comparison information is in the form of: $\mathbf{x}$ is closer to $\mathbf{y}$ than $\mathbf{x}$ is closer to $\mathbf{z}$. They induce the initial distance from a domain specific similarity measure then search for a matrix $\mathbf{A}$ with minimum trace that will decode and respect these constraints as follows: $||\mathbf{x}-\mathbf{y}||_{\mathbf{A}} < ||\mathbf{x}-\mathbf{z}||_{\mathbf{A}}$. The major drawback of this approach is that the number of constraints scales, at best, quadratically with the number of samples since these are required to define triplet constraints for each sample with all other samples in the data set.

Xing *et al.* [11] use $+ve$ and $-ve$ constraints to find a matrix $\mathbf{A}$ that will keep points in the $+ve$ constraints set close to each other, while points in the $-ve$ constraints set far from each other. Bar–Hillel *et al.* [13], in a simpler and a faster algorithm which they call relevant component analysis (RCA), rely only on $+ve$ constraints to define the metric. Hoi *et al.* [58], motivated by RCA, encapsulate Xing's [11] setting in an LDA framework that they call it discriminant component analysis (DCA) and minimize the ratio of determinants between the covariance of $+ve$ constraints and the covariance of $-ve$ constraints. Xiang *et al.* [17] in a variant of this framework minimize the ratio of traces for speed and efficiency purposes, while Tsang *et al.* [59] develop a kernelized version of RCA. The major advantage of Xing and RCA algorithms is that they can be used when partial side–information is available, however their generalization performance depends on the available amount of partially labelled data. In the same category, an online pseudo–metric learning for $\mathbf{A}$ was proposed by Shalev–Shwartz *et al.* [60] using $+ve$ and $-ve$ constraints, and Tsang and Kwok [61] encapsulate the $+ve$ and $-ve$ constraints sets in the context of idealized kernels and formulate the metric learning as a ν–SVM type training.

3.2 Local Learning of a Mahalanobis Metric

The research work presented in this thesis started from my previous work on the minimum volume ellipsoid metric (MVEM) [32, 33, 34, 35]. There, the objective was to learn a metric for query based operation in an unsupervised manner. By query based operations it is meant to find the nearest neighbour or neighbours for a query point $\mathbf{x}_q$. Similar to the augmented space $\mathbb{X}$ in Chapter 5, the MVEM takes into consideration the uneven sample distribution in the input space [47], and the fact that the data lie on or near a smooth low dimensional manifold $\mathcal{M}$.

To see this, consider a data set $\mathcal{X} = \{\mathbf{x}_i \mid 1 \leq i \leq m,\ \mathbf{x}_i \in \mathbb{R}^p\}$ that is drawn from a probability distribution $\Pr(X)$. Let $\mathbf{x}_q$ be defined as a *"query point"* such that, either $\mathbf{x}_q \in \mathcal{X}$, or $\mathbf{x}_q \sim \Pr(X)$; i.e. a new point that is drawn from $\Pr(X)$. We are interested in learning a metric for each query point $\mathbf{x}_q$ that is based on the information in a small neighbourhood $\mathcal{N}(\mathbf{x}_q) \subset \mathcal{X}$ around $\mathbf{x}_q$. That is, the metric is defined independently for each point. Using a flexible definition of $\mathcal{N}(\cdot)$, the metric tries to preserve the local information in $\mathcal{N}(\mathbf{x}_q)$ using a regularized covariance matrix $\mathbf{\Sigma}_q$ of the neighbourhood $\mathcal{N}(\mathbf{x}_q)$. This covariance matrix is then used to define a Mahalanobis distance that can measure the distance between $\mathbf{x}_q$ and any other point $\mathbf{x} \sim \Pr(X)$. Although the objective was to define the MVEM in an unsupervised manner, the fact that it was used for query based operations led to supervised training for learning the neighbourhood size for each data set, and the regularization parameter for the covariance matrices.

The MVEM has some limitations inherent from its definition. Note that the metric is defined locally with respect to a small neighbourhood $\mathcal{N}(\mathbf{x}_q)$ around the query point $\mathbf{x}_q$ which is the basic ingredient for local learning algorithms [47]. This neighbourhood defines the local covariance matrix $\mathbf{\Sigma}_q$ which in turn defines a local subspace spanned by its principal eigenvectors. The Mahalanobis distance defined by the MVEM is then the Euclidean distance between $\mathbf{x}_q$ and any other point $\mathbf{y}$ that is projected on this local subspace. This implies that for two points $\mathbf{x}$ and $\mathbf{y}$, each with its own neighbourhood and covariance matrix, $\mathbf{\Sigma}_\mathbf{x}$ and $\mathbf{\Sigma}_\mathbf{y}$ respectively, the distance between the two points is not comparable, and the global symmetry of the distance can not be established.

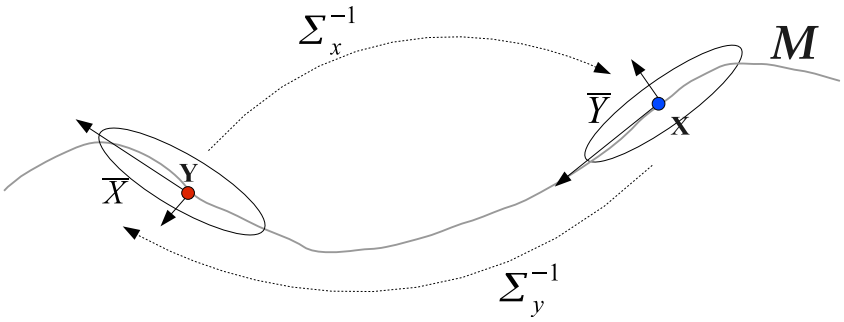


Fig. 3.1 Manifold $\boldsymbol{M}$ has two points on it, X and Y with their neighbourhoods defined by the ellipses $\boldsymbol{\Sigma_x}$ and $\boldsymbol{\Sigma_y}$ respectively, with their major and minor axes pointing along and orthogonal to the manifold respectively. $\boldsymbol{\Sigma_x^{-1}}$ maps every point, Y for instance, in the global space defined by the data set to another point $\bar{Y}$ in the local subspace spanned by its eigenvectors.

Figure 3.1 illustrates this limitation. Consider two points $\mathbf{x}$ and $\mathbf{y}$ from the data set $\mathcal{X}$, not necessarily far away from each other, and each point is defined by its own neighbourhood, $\mathcal{N}(\mathbf{x})$ and $\mathcal{N}(\mathbf{y})$, which define the covariance matrices $\boldsymbol{\Sigma_x}$ and $\boldsymbol{\Sigma_y}$ respectively. The distance between the two points can be defined in two different ways; $||\mathbf{x}-\mathbf{y}||_{\Sigma_\mathbf{x}^{-1}}$ and $||\mathbf{x}-\mathbf{y}||_{\Sigma_\mathbf{y}^{-1}}$, which makes both distance measures different since the weighting matrix is different. Therefore, globally, the MVEM is not symmetric and does not satisfy the triangle inequality, and as a result, distances between different points are not comparable.

Due to this particular setting, the MVEM is suitable for query based operations since it defines a Mahalanobis metric for each query point. However, the MVEM can not define a global metric on $\mathcal{X}$ due its definition as mentioned above. A simple remedy to its shortcoming is to define the distance between $\mathbf{x}$ and $\mathbf{y}$ as :

$$d(\mathbf{x},\mathbf{y}) = \tfrac{1}{2}(||\mathbf{x}-\mathbf{y}||_{\Sigma_\mathbf{x}^{-1}} + ||\mathbf{x}-\mathbf{y}||_{\Sigma_\mathbf{y}^{-1}}).$$

As it will be shown in Chapter 5, this distance is the first term of the symmetric Kullback-Leibler (KL) divergence between two Gaussian densities with different means and different covariance matrices. Hence, $d(\mathbf{x},\mathbf{y})$ misses the second term which is a dissimilarity measure between covariance matrices. This second term penalizes the distance between $\mathbf{x}$ and $\mathbf{y}$ when the second moments of the local density around each point are not similar. This interpretation gives a new meaning for the distance between points based on

the proximity of points in the input space, and the similarity of the local density around each point encoded in the second moments of the variables in each neighbourhood.

At this point, to achieve global metric learning in a supervised and unsupervised manner, the research work was split into two different directions; 1) supervised metric learning and linear dimensionality reduction based on Fisher's linear discriminant analysis introduced in Chapter 4, and 2) unsupervised metric learning based on spectral manifold learning algorithms introduced in Chapter 5.

3.3 Spectral Manifold Learning Algorithms

Manifold learning algorithms [18, 19, 20, 21, 23, 24] address a longstanding problem at the intersection of geometry and statistics: *"Compute a low dimensional embedding of high dimensional data sampled (with noise) from an underlying manifold"* [62]. This objective is not new when the embedding is assumed to be linear. For instance, under a linear embedding assumption, principal component analysis (PCA) [63] and multi–dimensional scaling (MDS) [39, 41, 64, 65, 66, 42] are the canonical forms of (linear) dimensionality reduction. When the linearity assumption does not hold, or when it is expected that nonlinear embedding will reveal more on the structure in the data, PCA and MDS are no longer valid solutions. Therefore, the novelty in recent manifold learning algorithms is their assumption of a nonlinear embedding process.

Similar to PCA and MDS, manifold learning algorithms are unsupervised nonparametric techniques for dimensionality reduction that rely on the machinery of eigensolvers. Hence, their optimization algorithms do not suffer from local minima and can scale well with large and high dimensional data sets thanks to state-of-the-art eigensolvers.

Various algorithms were proposed to recover the low dimensional manifold of the data; local linear embedding (LLE) by Saul and Roweis [18], ISOMAP by Tenenbaoum, De Silva, and Langford [19], Laplacian eigenmaps by Belkin and Nyiogi [20], Hessian eigenmaps by Donoho and Grimes [21], local tangent space alignment (LTSA) by Zhang and Zha [23], and maximum variance unfolding (MVU) by Weinberger and Saul [24]. There are also various algorithms for spectral clustering which are mainly motivated by graph cuts and random

walks on graphs. This includes the work of Y. Weiss on segmentation using eigenvectors [26], normalized cuts by Shi and Malik [25], random walks view for spectral segmentation by Meila and Shi [36], and spectral clustering by Ng, Jordan, and Weiss [27].

Despite the different names and motivations for all the above algorithms, they all share the use of an eigendecomposition step to obtain a lower dimensional embedding for the data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$. The eigendecomposition step characterizes the nonlinear manifold $\mathcal{M}^{p_0}$ on the hyperplane $\mathbb{R}^{p_0} \subset \mathbb{R}^p$, where $p_0 \ll p$, on which the data set $\mathcal{D}$ would lie. During this characterization, spectral manifold learning methods perform two simultaneous tasks; dimensionality reduction, and the characterization of non-spherical, non-compact clusters which are intimately related to nonlinear manifolds (both are regions of high densities). Therefore, both tasks, spectral clustering and manifold learning are linked since the clusters captured by spectral clustering can be arbitrary curved manifolds (as long as there is enough data to locally capture the curvature of the manifold) [67].

> *"Dimensionality reduction is an interesting alternative to feature selection. Like feature selection, it yields a low dimensional representation which helps to build lower capacity predictors in order to improve generalization. However, unlike feature selection it may preserve information from all the original input variables. If the data truly lies on a low dimensional manifold, it may preserve almost all of the original information while representing it in a way that eases learning" [67].*

3.3.1 A formal definition for manifold learning

We begin our discussion with a general, formal definition for manifold learning [68]. We are given a set of high dimensional points $\mathbb{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^p$ where n is the number of points and p is the dimensionality of the input space. It is assumed that the data points lie on, or near, an underlying smooth nonlinear manifold $\mathcal{M}$ of dimension p_0. Further, $\mathcal{M}^{p_0}$ is assumed to be an immersed sub–manifold of the ambient Euclidean space $\mathbb{R}^p$, where $p \gg p_0$. Let $\mathbf{y}_i$ denote the coordinate of a point on $\mathcal{M}^{p_0}$ corresponding to the point $\mathbf{x}_i$, so that we have the map $\mathbf{y}_i \to \mathbf{x}_i,\ 1 \leq i \leq n$. Given this setting, the problem of manifold learning can be stated as follows:

Manifold learning: *Given a set of the natural coordinates* $\mathbb{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ *of points on the manifold* $\mathcal{M}^{d_0}$*, find a single global coordinate system or a set of parameterized representations* $\mathbb{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$.

It is intuitive to develop the manifold $\mathcal{M}^{p_0}$ on the hyperplane $\mathbb{R}^{p_0} \subset \mathbb{R}^p$, and this what most of manifold learning algorithms do. The mapping in that case may be isometric (preserve distances), conformal (preserve angles), or follow some weaker conditions such as locally isometric, locally conformal, or a combination of both. Based on this formal setup, I will briefly review some well known spectral manifold learning algorithms. These are: metric or classical multidimensional scaling (cMDS) [39, 41, 64, 65, 66, 42], ISOMAP [19], Laplacian eigenmaps (LAPMAP) [20] and spectral clustering (SC) [27], and local linear embedding (LLE) [18]. For further details and justification of the algorithms, the reader is kindly requested to refer to the original papers of these algorithms.

3.3.2 Skeleton of a general spectral manifold learning algorithm

The algorithms that will be discussed in the following can all be cast in a common framework which computes an embedding for the training data using an eigendecomposition of a symmetric similarity matrix $\mathbf{M}$. The embedding is nothing more than the coordinates of the leading eigenvectors of the matrix $\mathbf{M}$. This framework can be described as follows:

1. Given a data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n$, construct a similarity matrix based on a neighbourhood graph or a fully connected graph for the data set $\mathcal{D}$. Let $K_\mathcal{D}$ denote the kernel function that produces $\mathbf{M}$ by $\mathbf{M}_{ij} = K_\mathcal{D}(\mathbf{x}_i, \mathbf{x}_j)$. Note that $K_\mathcal{D}$ should be a symmetric PSD kernel. Note also that $K_\mathcal{D}$ might not only depend on $\mathbf{x}_i$ and $\mathbf{x}_j$, but on all the data set $\mathcal{D}$ and hence the notation $K_\mathcal{D}$.

2. Optionally, transform $\mathbf{M}$, yielding a processed symmetric matrix $\mathbf{M}^*$. This transformation can include normalization, scaling, centering, extracting the Laplacian, trace maximization, etc. Note that this is equivalent to having another kernel $K^*_\mathcal{D}$ that fills the entries $\mathbf{M}^*_{ij}$.

3. Compute the d_0 largest eigenvalues λ_j and their corresponding eigenvectors $\mathbf{v}_j \in \mathbb{R}^n$ of matrix $\mathbf{M}^*$, where $1 \leq j \leq d_0$.

4. The embedding of each example $\mathbf{x}_i$ is the vector $\mathbf{y}_i = [y_{i1}, y_{i2}, \ldots, y_{id_0}]^\top$, with y_{ij} is the ith element of the jth eigenvector of $\mathbf{M}^*$. Alternatively, as in cMDS and ISOMAP, the embedding $y_{ij} = \sqrt{\lambda_j} v_{ij}$.

Classical multidimensional scaling (cMDS)

cMDS starts from a distance matrix constructed from the Euclidean distance between every pair of points in $\mathcal{D}$. Note that cMDS works only with the notion of a metric as pointed out in [40]. As a transformation, cMDS transforms the Euclidean distance matrix into a similarity matrix using the double centering formula, which transforms distances to dot products:

$$\mathbf{M}^*_{ij} = -\frac{1}{2}\left(\mathbf{M}_{ij} - \frac{1}{n}s_i - \frac{1}{n}s_j + \frac{1}{n^2}s_i s_j\right), \tag{3.1}$$

where $\mathbf{M}_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|_2$, and $s_i = \sum_{j=1}^{n} \mathbf{M}_{ij}$. The embedding y_{ij} is given by $\sqrt{\lambda_j} v_{ij}$.

ISOMAP

ISOMAP generalizes MDS to nonlinear manifolds. The algorithm starts by defining a k nearest neighbour graph for the set $\mathcal{D}$. After constructing the neighborhood graph, unlike cMDS, ISOMAP replaces the Euclidean distance between two points with a discrete approximation of the geodesic distance between the points on the manifold. This approximated geodesic distance is computed using Djikstra's shortest path algorithm between the two points. Once the new distance matrix between the points is obtained, the algorithm proceeds exactly as cMDS.

LLE

Similar to ISOMAP, LLE start by defining the k-NN graph of the data set $\mathcal{D}$. The basic assumption of LLE is that each point can be faithfully reconstructed from its neighbourhood; i.e. each point is a linear combination from the points in its neighbourhood. Hence LLE preserves local distances and angles between points in the neighbourhood. The algorithm proceeds as follows. First, a sparse matrix of local predictive weights $\mathbf{W}_{ij}$ is computed, such that $\sum_j \mathbf{W}_{ij} = 1$ if point $\mathbf{x}_j$ is a neighbour for point $\mathbf{x}_i$, and $\sum_j \mathbf{W}_{ij} = 0$ if not. The predictive weights for each local neighbourhood are estimated by minimizing

$\sum_j = \mathbf{W}_{ij}(\mathbf{x}_j - \mathbf{x}_i)^2$, which is equivalent to a constrained system of linear equations. Then, the matrix $\mathbf{M} = (\mathbf{I} - \mathbf{W})^\top(\mathbf{I} - \mathbf{W})$ is formed with possibly an additional regularization term on the diagonal of $\mathbf{M}$. Finally, the embedding is obtained by the smallest eigenvectors of $\mathbf{M}$ (except for the first eigenvector with a zero eigenvalue).

LAPMAP and SC

Laplacian eigenmaps (LAPMAP) also starts by defining a k-NN graph over the data set $\mathcal{D}$. The similarity on each edge of the graph is approximated by the Gaussian kernel instead of the Laplacian operator. From the new similarity matrix (or gram matrix obtained by the Gaussian kernel), the Laplacian operator is computed and it becomes the new similarity matrix for the data points. The Laplacian optimally preserves the local geometry for each points, and hence LAPMAP and LLE are similar in that regard. The justification of the graph Laplacian is motivated from the role of the Laplace Beltrami operator in providing an optimal embedding of the continuous manifold. Hence, the continuous manifold is approximated by the neighbourhood graph of the data, and the Laplace Beltrami operator is approximated by the graph Laplacian [69]. The final embedding is obtained by the eigenvectors corresponding to the smallest eigenvalues (except the smallest one) of the following generalized eigenvalue problem (GEP):

$$\mathbf{L}\mathbf{v} = \lambda\mathbf{S}\mathbf{v}, \tag{3.2}$$

where $\mathbf{L} = \mathbf{S} - \mathbf{M}$ is the Laplacian operator, $\mathbf{S}$ is a a diagonal matrix with $s_{ii} = \sum_{j=1}^{n} \mathbf{M}_{ij}$, and λ and $\mathbf{v}$ are the generalized eigenvalues and eigenvectors of $\mathbf{L}$. Note that the generalized eigenvectors of $\mathbf{L}$ are equivalent to the eigenvectors of $\mathbf{I} - \mathbf{S}^{-1}\mathbf{M}$. That is, the difference here is in the normalization by $\mathbf{S}^{-1}$ which is known as divisive normalization, and it is closely related to a random walk over the data neighbourhood graph [36].

SC was proposed earlier than LAPMAP, and it proceeds exactly as LAPMAP except for two steps. The first is the normalization of the Laplacian, where [27] define the Laplacian as $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{M}\mathbf{D}^{-1/2}$. The second is that the embedded vectors $\mathbf{y}_i$ are normalized to have unit length; i.e. the embedded points are projected on the unit sphere. After that, k-Means clustering is applied on the embedded normalized vectors.

H–LLE, LTSA, and MVU

Hessian eigenmaps or Hessian LLE (H–LLE) [21] and local tangent space alignment (LTSA) [23] follow the same steps of LLE, however they operate on the tangent space defined at each neighbourhood. Hence, H–LLE and LTSA are also locally isometric and locally conformal. In a similar vein, Maximum variance unfolding (MVU) [24] tries to maximize the variance in the data under the constraint of being locally isometric and conformal. The alignment matrix $\mathbf{M}$ in their approach is the gram matrix of the data and the transformation applied on $\mathbf{M}$ is the trace maximization process under the constraints of preserving the local geometry. Finally, similar to all other algorithms, the final embedding is found via the solution of the eigensystem using the top eigenvectors of the optimized gram matrix.

Discussion

ISOMAP finds low dimensional coordinates that preserve the geodesic distance between high dimensional points. It assumes that the high dimensional data are generated by lifting low dimensional points that lie in a convex set through an isometric lifting. Donoho and Grimes [21] consider the particular case of images and point out that imaging processes are more accurately represented by local isometry, and that the location of multiple objects in a scene cannot be represented by a convex low dimensional set. They presented H-LLE to handle these special conditions. LLE finds a conformal mapping that preserves the affine relationship between high dimensional points in local neighbourhoods. Similar to LLE, LAPMAP and MVU preserve a notion of local geometry; the proximity of points in a neighbourhood weighted by local distance metrics for LAPMAP, and local isometry for MVU [70].

Despite the differences between these algorithms, it was pointed put in [71] that if the points do not densely sample the manifold, the local neighbourhood structure of the manifold becomes difficult to estimate, and these algorithms recover low dimensional points that do not exhibit the desired neighbourhood attributes. The impact of the manifold sampling on the quality of the embedding obtained by manifold learning algorithms became known as the topological stability of the algorithm. Despite its importance, the question of topological stability for these algorithms is usually overlooked, and to the best of my knowledge, there is no major study in the machine learning literature that addresses this

question. However, as will be discussed in Chapter 7, the augmented space $\mathbb{X}$ introduced in Chapter 5 can be used to improve the topological stability of spectral manifold learning algorithms.

Chapter 4

Pareto Disciminant Analysis

In this chapter, I consider the problem of learning a low dimensional embedding for the data set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Y}$ when the *a priori* information in the form of labels y_i is available for learning. In terms of metric learning, this can be seen as learning an instance of the GQD $\|\cdot\|_{\mathbf{A}}$, where $\mathbf{A} \in \mathbb{R}^{p\times p}$, and $\mathbf{A} \succ 0$. However, to achieve linear dimensionality reduction together with metric learning, the matrix $\mathbf{A}$ is required to be low rank; that is $\mathbf{A} \succeq 0$, and $\mathbf{A} = \mathbf{B}\mathbf{B}^\top$, where $\mathbf{B} \in \mathbb{R}^{p\times p_0}$, and $p_0 \ll p$. Moreover, since labels are available for learning, it is required that the matrix $\mathbf{B}$ linearly projects the data onto a subspace that better discriminates between points from different classes.

Here, I rely on the framework for Fisher's linear discriminant analysis (LDA) in the multiclass setting for learning the matrix $\mathbf{B}$. I propose a new algorithm, namely Pareto discriminant analysis (PARDA) [28], for Fisher's LDA that is based on the machinery of multiobjective optimization [37, 38]. PARDA decomposes the multiclass problem to a set of pairwise objective functions representing the pairwise distance between different classes. Unlike existing extensions of Fisher's LDA to multiclass problems, that typically maximize the sum of pairwise distances between classes, PARDA simultaneously maximizes each pairwise distance, encouraging the case where all classes are equidistant from each other in the lower dimensional embedding space. Solving PARDA is a multiobjective optimization problem – simultaneously optimizing more than one, possibly conflicting, objective functions – and the resulting solution is known to be "Pareto Optimal". To the best of my knowledge, this is the first research to address the multiclass linear dimensionality reduction

problem as a multiobjective minimization problem.

4.1 Linear Discriminant Analysis (LDA)

Fisher Discriminant Analysis (FDA) originally developed by Fisher in 1936 [72] is a technique for dimensionality reduction that is optimal for classification under two assumptions: (1) the number of classes c is exactly two, and (2) the samples in each class are assumed to be generated from a multivariate Gaussian distribution with different means and equal covariance matrices (homoscedastic data) [73]. In this context, FDA is guaranteed to find a one dimensional subspace that will classify the samples with the optimal error rate, *Bayes error*, and the subspace is known to be *Bayes optimal* [73]. As an illustration, Figure 4.1 shows the difference between FDA and the well known principal component analysis (PCA) [63] technique for dimensionality reduction.

Rao [74] extended this approach to the multiclass homoscedastic case ($c > 2$), under the condition that the number of features $p \geq c$ (and assuming the number of samples $n > p$). The resultant $c - 1$ dimensional subspace is also guaranteed to be Bayes optimal, and the technique has become known as Linear Discriminant Analysis (LDA). Rao, however, noted that if the lower dimensional subspace has dimensionality $p_0 < c-1$, the resultant subspace will not be Bayes optimal. It is only recently that Hamsici and Martinez [75] pushed the homoscedastic case further and derived a Bayes optimal one dimensional subspace when $c > 2$.

When the covariance assumption does not hold for $c \geq 2$ (heteroscedastic data), Rao proposed to approximate the heteroscedastic problem with a homoscedastic setting and solve the approximated problem instead. His approximated problem considered that all classes have different means but share a common covariance matrix that is a weighted average of all the covariance matrices of the original problem. This approximation matrix became known as the pooled sample covariance matrix, or the average within–class scatters matrix $\mathbf{S}_w$. Rao's final solution became the well known formulation based on the Rayleigh quotient of the between–class scatter matrix $\mathbf{S}_b$ and $\mathbf{S}_w$. The obtained subspace, however, is not Bayes optimal for the original heteroscedastic problem.

Several researchers, backed by theoretical justifications, have scrutinized the limitations

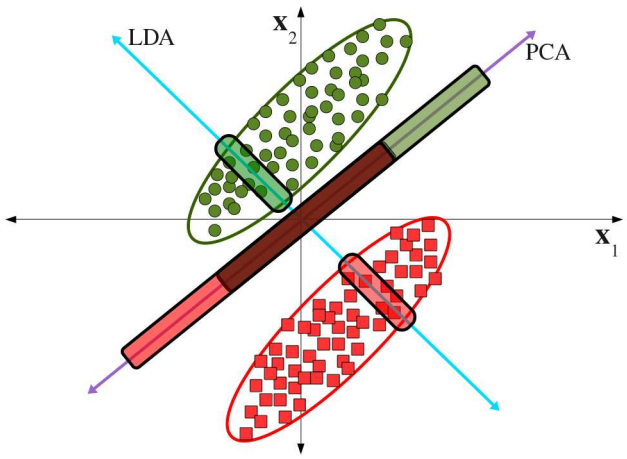


Fig. 4.1 The data points shown here are from two well separated Gaussian distributions (green and red) with different means and equal covariance matrices, and hence the two parallel ellipses. The one dimensional subspace defined by PCA (magenta line) is in the direction of the maximum variance of the total data distribution. Projecting on this subspace yields a strong overlap between the two classes. The one dimensional subspace defined by FDA (cyan line) is in the direction of maximal separation between the two classes. Projecting on this subspace yields optimal separation between the two classes, and hence minimal Bayes error, which is zero in this case.

and non–optimality of LDA when its strong assumptions do not hold, and proposed extensions derived from Gaussian assumptions [76, 77, 78, 79] and kernel methods [80, 81] to generalize LDA to the multiclass heteroscedastic case. The result was a plethora of algorithms that have been reported to perform well in a variety of application domains, most notably face recognition. A good review for these methods can be found in [82, 83, 84, 85, 75].

Of particular interest is the extension proposed by De La Torre and Kanade [86], namely multimodal oriented discriminant analysis (MODA), where it was shown that FDA's objective function is a special case of a more general objective that maximizes the symmetric Kullback–Leibler (KL) divergence between two Gaussian densities, when the two Gaussians share the same covariance matrix. Note that the symmetric KL divergence (SKLD) considers the difference in mean locations and the difference in covariance matrices. This is the rational for MODA, for which it searches for a low dimensional linear transformation

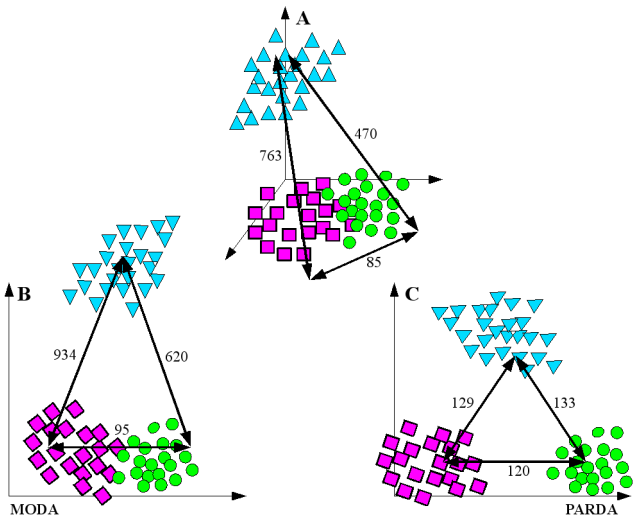


Fig. 4.2 (A) A Synthetic example of a 3–class problem with three dimensional data. The numbers shown on arrows indicate the symmetric Kullback–Leibler divergence (KLD). (B) Projection using MODA on a two–dimensional space. Observe that the two classes that are close in the input space proportionally increase the KL divergence less than the classes that are further in the input space. (C) Projections obtained by Pareto Discriminant Analysis (PARDA) encourages the classes to be equally spread from each other in the lower dimensional space.

that maximizes the SKLD between the two classes in the low dimensional subspace.

To account for the multiclass heteroscedastic case, MODA sums over all SKLDs between every pair of different classes and maximizes that sum in the lower dimensional subspace. This is very similar to LDA's objective function, which as shown by Loog *et al.* [82], maximizes the sum of pairwise FDAs between all pairs of different classes. Hence MODA is a consistent generalization of FDA/LDA to multimodal Gaussian distributions with different means and covariance matrices.

However, as noted by several researchers [82, 83, 87, 28], even if all the homoscedastic assumptions are satisfied, LDA and MODA suffer from the serious problem of merging classes that are close to each other in the original input space, *a.k.a* the class separation (or masking) problem. This is due to the fact that LDA and MODA shift the two–class problem to the multiclass setting by maximizing the sum of all SKLDs, which is a suitable

objective function when all classes are in proximity to each other in terms of KL divergence.

Figure 4.2 A depicts a synthetic example for a 3–class problem with three dimensional data. Traditional methods like LDA or MODA find projections that maximize the sum of pairwise Mahalanobis distances (LDA) or the SKLD (MODA) between pairwise classes. Note that the SKLD, and the Mahalanobis distance (a special case of SKLD) are positive quadratic distance functions. From the optimization of minimax functions [88], it is known that the sum of positive powered functions, $\sum_{j=1}^{m}[f_j]^p$, for $p > 1$, is a smooth approximation for $\max_{1\leq j\leq m}[f_j]^p$ as p increases to infinity, and hence $\sum_{j=1}^{m}[f_j]^p \approx [f_r]^p$ where $f_r > f_j \; \forall j \neq r$. Using this argument[1], it is possible to see that LDA and MODA are in fact maximizing a smooth approximation of the maximum of pairwise Mahalanobis distances and SKLDs respectively.

Hence, LDA and MODA intrinsically prefer solutions that encourage maximizing the largest distance in the input space to make it even larger in the lower dimensional subspace. In other words, LDA and MODA put needless effort to maximize already distant classes in the input space. This effect can be seen in Figure 4.2 B, where MODA's projection gives relatively better increase in terms of KL divergence to the classes that are farther away in the input space, while it only makes a slight effort to separate between classes that are very close to each other in the input space.

4.2 From LDA to Pareto Discriminant Analysis (PARDA)

We note that the multiclass problem for LDA and MODA defines an independent objective function for each pair of different classes that needs to be optimized, namely maximize the SKLD between every pair of different classes. Hence, the set of all pairs of different classes define an optimization problem with *multiple objective functions* that share one final solution, and they all need to be *simultaneously optimized*. Given this perspective, maximizing the sum over all pairwise SKLDs (or quadratic distances) does not consider each objective function independently, since as explained above, maximizing that sum approximates a max function that encourages maximizing the largest SKLD. This implies that the optimization procedure does not search for a solution that is in maximal agreement amongst all

[1]This will be explained in more detail in Section 4.6

independent and possibly conflicting objective functions. This shows that upgrading the problem of learning a discriminant subspace from the two–class setting to the multiclass setting by summing over all pairwise SKLDs as in LDA/MODA, is not the appropriate path to handle a multiobjective optimization problem [89, 37, 38], since by summing no maximal agreement is guaranteed between all pairwise KL divergences.

My contribution in this research direction stems from the above observation. In particular, I propose a set of new parametrized objective functions for multiclass HDA based on the theory of multiobjective optimization (MOP) [89, 37, 38]. Due to their parametrization, these objective functions can easily adapt to the class topology of the classification problem. While LDA and MODA's objectives pull apart the two classes with the largest SKLD, PARDA, or Pareto Discriminant Analysis, tries to equally spread all classes from each other.

PARDA concentrates its effort on overlapping classes while it safeguards well separated classes from overlapping in the lower dimensional subspace. In other words, PARDA puts more effort in maximizing the distance between classes that are closer in the projected space, and will relax the constraint between classes that are farther away. Figure 4.2 C shows the projection obtained by PARDA in a two dimensional space. Unlike MODA, the two-dimensional projection obtained by PARDA encourages the case where classes are equally spread from each other in the lower dimensional space.

4.3 Basic Formulation of Linear Discriminant Analysis

Here I review the basic and standard formulation of LDA. Given a data set $\mathcal{D} = \{(\mathbf{x}_i, \ell_i)_{i=1}^n \subseteq \mathbb{R}^p \times \mathcal{L}\}$ with labels $\ell_i \in \mathcal{L} = \{\mathcal{C}_1, \ldots, \mathcal{C}_c\}$, LDA's objective is to find a linear transformation $\mathbf{B} \in \mathbb{R}^{p \times p_0}$, with $p_0 \ll p$ such that the data of each class when projected in the low dimensional subspace is compact as much as possible, while all the classes are maximally separated from each other. A rather standard formulation to obtain the linear transforma-

tion matrix $\mathbf{B}^*$ is via the generalized Rayleigh quotient defined as follows:

$$\mathbf{B}^* = \arg\max_{\mathbf{B}} E_{\text{LDA}}(\mathbf{B}), \quad \text{where} \tag{4.1}$$
$$E_{\text{LDA}}(\mathbf{B}) = \text{tr}\{(\mathbf{B}^\top \mathbf{S}_1 \mathbf{B})^{-1}(\mathbf{B}^\top \mathbf{S}_2 \mathbf{B})\},$$

$\mathbf{S}_1 = \{\mathbf{S}_b, \mathbf{S}_b, \mathbf{S}_t\}$, $\mathbf{S}_2 = \{\mathbf{S}_w, \mathbf{S}_t, \mathbf{S}_w\}$, the columns of $\mathbf{B}^*$ are the generalized singular vectors of the generalized eigenvalue problem (GEP): $\mathbf{S}_1\mathbf{B} = \mathbf{\Lambda}\mathbf{S}_2\mathbf{B}$, and $\mathbf{\Lambda} = \mathbf{diag}\{\lambda_1, \ldots, \lambda_p\}$ is the generalized eigenvalue matrix. The matrices $\mathbf{S}_b$, $\mathbf{S}_w$ and $\mathbf{S}_t$ are known as the between–class scatter matrix, the within–class scatter matrix and the total–class scatter matrix respectively.

Formally, $\mathbf{S}_b$, $\mathbf{S}_w$ and $\mathbf{S}_t$ are defined as follows:

$$\mathbf{S}_w = \sum_{j=1}^{c} \Pr(\mathcal{C}_j)\hat{\mathbf{\Sigma}}_j, \tag{4.2}$$

$$\mathbf{S}_b = \sum_{j=1}^{c} \Pr(\mathcal{C}_j)(\hat{\boldsymbol{\mu}}_j - \hat{\boldsymbol{\mu}}_0)(\hat{\boldsymbol{\mu}}_j - \hat{\boldsymbol{\mu}}_0)^\top, \quad \text{and} \tag{4.3}$$

$$\mathbf{S}_t = \mathbf{S}_b + \mathbf{S}_w, \quad \text{where} \tag{4.4}$$
$$\hat{\mathbf{\Sigma}}_j = 1/(n_j - 1)\sum_{i \in \mathcal{C}_j}^{n_j} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_j)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_j)^\top,$$
$$\hat{\boldsymbol{\mu}}_j = 1/n_j \sum_{i \in \mathcal{C}_j}^{n_j} \mathbf{x}_i, \quad \hat{\boldsymbol{\mu}}_0 = \sum_{j=1}^{c} \Pr(\mathcal{C}_j)\hat{\boldsymbol{\mu}}_j,$$

with the prior probability of class $\mathcal{C}_j$ denoted by $\Pr(\mathcal{C}_j)$, and $\sum_{j=1}^{c} n_j = n$. The upper bound on the ranks of $\mathbf{S}_b$, $\mathbf{S}_w$ and $\mathbf{S}_t$ is $\min(c-1, p)$, $\min(n-c, p)$ and $\min(n-1, p)$ respectively.

Problem (4.1) with $\mathbf{S}_b$ and $\mathbf{S}_w$ replacing $\mathbf{S}_1$ and $\mathbf{S}_2$ respectively is considered the most popular LDA objective in the literature. The formulation, however, is restricted to the original homoscedastic setting when the samples in each class $\mathcal{C}_j$ are assumed to have a Gaussian distribution $\mathcal{G}(\ \cdot\ ; \boldsymbol{\mu}_j, \mathbf{\Sigma}_j)$ with $\mathbf{\Sigma}_1 = \cdots = \mathbf{\Sigma}_c = \mathbf{\Sigma}$. In practice, when the covariance assumption does not hold, or when the classes are not Gaussians, all the

Gaussian parameters are approximated by their sample estimates, and $\mathbf{\Sigma}$ is approximated by $\mathbf{S}_w$. Unfortunately, this approximation does not fully exploit the rich information in the heteroscedastic setting which is represented in the covariance matrix of each class.

4.3.1 A different formulation for multiclass heteroscedastic LDA

Various researchers have proposed different extensions from the homoscedastic case to the heteroscedastic one [76, 90, 78, 79]. Of particular interest, were the ideas proposed by Tou & Heyden in 1967 [91] on feature extraction, where they derived LDA's objective in Problem (4.1) from maximizing the symmetric KL divergence [92] between two Gaussian densities under the homoscedastic assumption. Independently, De La Torre and Kanade [86] paralleled Tou and Heyden's ideas in their much richer model MODA. Since MODA is a consistent generalization of LDA to multimodal Gaussian distributions with different means and covariance matrices, we will adopt MODA's formulation for our multiclass HDA framework. Let the symmetric KL divergence between two Gaussian densities $\mathcal{G}_i$ and $\mathcal{G}_j$ be defined as follows:

$$J(\mathcal{G}_i, \mathcal{G}_j) = (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^\top (\mathbf{\Sigma}_i^{-1} + \mathbf{\Sigma}_j^{-1})(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) + \operatorname{tr}\left(\mathbf{\Sigma}_i \mathbf{\Sigma}_j^{-1} + \mathbf{\Sigma}_i^{-1} \mathbf{\Sigma}_j - 2\mathbf{I}\right). \tag{4.5}$$

MODA seeks a linear transformation $\mathbf{B} \in \mathbb{R}^{p \times p_0}$ with $p_0 \ll p$ such that $J(\mathcal{G}_i, \mathcal{G}_j)$ in the lower dimensional subspace is maximized. Note that the linear transformation $\mathbf{B}$ can have any number of bases p_0 such that $1 \leq p_0 \leq p-1$. This is unlike FDA/LDA that can only define subspaces of dimensionality $p_0 \leq \min(c-1, p-1)$. In the lower dimensional subspace, classes ℓ_i and ℓ_j will be projected as $\mathcal{G}_i(\mathbf{B}^\top \boldsymbol{\mu}_i\ ,\ \mathbf{B}^\top \mathbf{\Sigma}_i \mathbf{B})$ and $\mathcal{G}_j(\mathbf{B}^\top \boldsymbol{\mu}_j\ ,\ \mathbf{B}^\top \mathbf{\Sigma}_j \mathbf{B})$ respectively. In turn, $J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B})$ in the lower dimensional subspace can be expressed as:

$$\begin{aligned} J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B}) = \mathbf{u}_{ij}^\top \mathbf{B} \left[(\mathbf{B}^\top \mathbf{\Sigma}_i \mathbf{B})^{-1} + (\mathbf{B}^\top \mathbf{\Sigma}_j \mathbf{B})^{-1}\right] \mathbf{B}^\top \mathbf{u}_{ij} + \\ \operatorname{tr}\{(\mathbf{B}^\top \mathbf{\Sigma}_i \mathbf{B})(\mathbf{B}^\top \mathbf{\Sigma}_j \mathbf{B})^{-1} + (\mathbf{B}^\top \mathbf{\Sigma}_i \mathbf{B})^{-1}(\mathbf{B}^\top \mathbf{\Sigma}_j \mathbf{B}) - 2\mathbf{B}^\top \mathbf{B}\}, \end{aligned} \tag{4.6}$$

where $\mathbf{u}_{ij} = (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)$. After some algebraic manipulation, Equation (4.6) can be simplified to:

$$J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B}) = \operatorname{tr}\left[(\mathbf{B}^\top \mathbf{\Sigma}_i \mathbf{B})^{-1} (\mathbf{B}^\top \mathbf{A}_{ij} \mathbf{B})\right] + \operatorname{tr}\left[(\mathbf{B}^\top \mathbf{\Sigma}_j \mathbf{B})^{-1} (\mathbf{B}^\top \mathbf{A}_{ji} \mathbf{B})\right], \tag{4.7}$$

where $\mathbf{A}_{ij} = \mathbf{u}_{ij}\mathbf{u}_{ij}^{\top} + \mathbf{\Sigma}_j$ and $\mathbf{A}_{ji} = \mathbf{u}_{ij}\mathbf{u}_{ij}^{\top} + \mathbf{\Sigma}_i$. Finally, for the c–class heteroscedastic setting, the linear transformation matrix $\mathbf{B}^*$ is obtained via MODA's objective function defined as:

$$\mathbf{B}^* = \arg\max_{\mathbf{B}} E_{\text{MODA}}(\mathbf{B}), \quad \text{where} \tag{4.8}$$
$$E_{\text{MODA}}(\mathbf{B}) = \sum_{i=1}^{c-1}\sum_{j=i+1}^{c} J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B}).$$

Note that maximizing Problem (4.8) under the assumption that $\mathbf{\Sigma}_i = \mathbf{\Sigma}_j = \mathbf{\Sigma}$ for $1 \leq i, j \leq c$, $i \neq j$, and with the constraint that the columns of $\mathbf{B}$ are orthogonal, will directly yield the standard LDA formulation for the multiclass problem. That is, MODA and LDA have the same general formulation and the difference between them is in the homoscedastic vs. heteroscedastic assumption.

4.4 Literature Review

The literature on discriminant analysis (DA) is immense and a thorough review will be beyond the scope of this thesis. The review presented here focuses on four different research directions of DA; heteroscedastic & multiclass extensions of LDA, the small sample size (SSS) problem, the class merging (separation) problem, and information theory based approaches for DA.

4.4.1 Heteroscedastic multiclass extensions of LDA

Campbell [76] was the first to develop a general formulation for LDA as a maximum likelihood estimation of the parameters of a Gaussian model. His model's structure relied on two assumptions: 1) all class means (or all discriminatory information between the classes) lie in a $(c-1)$–dimensional subspace of the original p–dimensional input (or feature) space; and 2) all classes have equal covariance matrices (homoscedastic setting). Hastie and Tibshirani [90] tried to work around the homoscedastic assumption of Campbell and proposed that each class can be modelled as mixtures of Gaussians while maintaining that all classes and sub–classes share a pooled covariance matrix. Based on this idea, Zhu and Martinez [84] proposed techniques for determining the optimal number of subclasses – or Gaussian

components – in each class.

Kumar and Andreou [78] extended Campbell's maximum likelihood model to the heteroscedastic setting and named it heterscedastic discriminant analysis (HDA). Their objective function is the log–likelihood of the Guassian models in the projected low dimensional subspace. By taking the gradient of this objective, they derive maximum likelihood estimators for the class means and covariances in the low dimensional subspace.

Saon *et al.* [93] defined an objective function based on the ratio of determinants instead of the trace of the generalized Rayleigh quotient in Equation (4.1) introduced earlier. That is, they maximize

$$E_{\text{SAON}}(\mathbf{B}) = \prod_{j=1}^{c} [(\mathbf{B}^\top \mathbf{\Sigma}_j \mathbf{B})^{-1} (\mathbf{B}^\top \mathbf{S}_B \mathbf{B})]^{n_j},$$

which is weighted product of each individual dimension (or direction) of the data. This objective function models the data orientation (or directionality) and has the property of being invariant to transformations of the range of the solution (eigenvectors). In addition, similar to LDA and HDA, it is invariant to linear transformations of the data in the input space. In the same spirit of Saon *et al.* , Zhu and Hastie [79] proposed the generalized feature extraction criterion, which generalizes the Fisher criterion when the covariance assumption does not hold, or even better, when the data of each class is not Gaussian.

4.4.2 The small sample size (SSS) problem

The SSS problem and its effect on the solutions obtained by DA is probably one of the very well studied problems in the literature of DA. The earliest formal treatment for the SSS problem is due to Friedman in [94] where he proposed a regularization framework for linear and quadratic discriminant analysis.

The pattern recognition and computer vision communities targeted this problem in the context of face recognition. Belhumeur *et al.* [95] proposed a two–stage LDA via their algorithm PCA+LDA. Later Chen *et al.* [96] showed that the null space of $\mathbf{S}_w$ contains the most discriminative information. Based on their work, Yu and Yang [97] proposed

direct–LDA which simultaneously diagonalizes $\mathbf{S}_w$ and $\mathbf{S}_b$ and discards the null space of $\mathbf{S}_b$. Howland and Park [98] proposed to solve the DA criterion via the generalized singular value decomposition (GSVD) algorithm. Alternatively, Ye and Li [99] proposed a two–stage LDA via a data transformation step based on the QR decomposition of the $\mathbf{S}_b$.

Recently, Zhang and Sim [85] established a neat understanding for the four main subspaces that define LDA using the Fukunaga–Koontz transform (FKT). Based on their analysis they showed that the FKT/LDA is equivalent to LDA/GSVD [98], and provided a unified framework for other subspace methods such as: Fisherface (PCA+LDA), PCA+NULL, LDA/QR and LDA/GSVD.

4.4.3 The class merging problem

To solve the class merging (or separation) problem, Lotlikar and Kothari [100] proposed fractional step DA (F–LDA) where the dimensionality is reduced in fractional steps; i.e. iteratively from p to $p-1$ (one dimension at a time) while applying proper weighting on the data in order to avoid the class merging problem. Lu *et al.* [83], in a two–stage algorithm, proposed a weighted variant of direct–LDA [97] combined with fractional step LDA [100]. For the between–class scatter matrix $\mathbf{S}_b$, they applied weights that are inversely proportional to the distance between class means. Alternatively, Loog *et al.* [82] suggested that the weights applied to $\mathbf{S}_b$ should link the distance between the class means to the amount of error they cause. Therefore, the weight between two classes is measured as $\frac{1}{2\delta_{ij}}\mathrm{erf}(\delta_{ij}/2\sqrt{2})$, where $\mathrm{erf}(\cdot)$ is the error function and δ_{ij} is the Euclidean distance between class means i and j in the whitened space.

Recently, there has been some interesting proposals for solving the class masking problem. These proposals can be found in the work of Zhang and Yeung [101], Yu *et al.* [102], and Bian and Tao [103]. The common objective in these solutions is the maximization of the smallest distance between the classes in the low dimensional subspace. This is unlike our objective function presented here, which focuses on maximizing every pairwise distance among all classes. In fact, our proposed framework presented here *encourages* solutions where all classes are well separated and equidistant from each others.

4.4.4 Information theoretic approaches

In the last decade, information theory based approaches for DA have gained more attention. The basic and common ingredient among these approaches is an objective function that maximizes a certain measure of information. These objective functions do not lead to GEPs, but rather, to optimization techniques based on gradient descent, quadratic optimization and their variants.

Based on advances in information theoretic learning using Renyi's entropy [104], Torkkola [105] used Renyi's entropy of order 2 [106] coupled with Parzen density estimators, and maximized the mutual information between the class labels and the data in the projected low dimensional space. This approach relaxes the Gaussian assumption of each class and naturally handles the heteroscedastic setting through non–parametric density estimators[2]. Inspired by Torkkola's model, Kaski and Pletonen [107] proposed another model with two different ingredients; 1) they use Shanon's entropy instead of Renyi's, and 2) they maximize the log likelihood of the data in the low dimensional subspace instead of maximizing the mutual information between the labels and the data.

MODA [86], as discussed earlier, is another instance of this category since it explicitly maximizes the symmetric KL divergence between different classes when each class is modelled as a mixture of Gaussian densities. In a similar vein, Tao *et al.* [87] proposed GADA, or general averaged divergence analysis, which is a further generalization of MODA. GADA replaces the symmetric KL divergence in MODA with the general Bregman divergence [108], and replaces the sum of all pairwise divergences by a general mean divergence function. Similar to MODA, GADA does not consider each pair of classes separately, and hence it puts needless effort on already distant classes.

4.5 Multiobjective Optimization

Multiobjective optimization (MOP), or vector optimization (VOP), is a branch of optimization science that is concerned with the simultaneous optimization of more than one objective function. In real world applications, it is often the case that the objectives are

[2]Please refer to the affiliated references in [105] for more details on these approaches.

contradictory in a way that optimizing one of the objectives entails a poor performance of another. In such cases, one would require a good compromise solution which is suboptimal but acceptable as much as possible to the individual objective functions. MOP, VOP or multicriteria optimization, is the science that can find this good compromise solution [37].

Let $\mathbf{f}(\boldsymbol{\theta}) = [f_1(\boldsymbol{\theta}) \dots f_\kappa(\boldsymbol{\theta})]^\top$ be the vector valued objective function to be optimized where $\mathbf{f}(\boldsymbol{\theta}) \in \mathbb{R}^\kappa$, $\boldsymbol{\theta} \in \mathcal{R} \subseteq \mathbb{R}^d$ is the parameter vector for the set of objective functions, $f_j(\boldsymbol{\theta}) \in \mathbb{R}$ is the jth objective function, $\mathcal{R}$ is the feasible set for the values of the parameter vector $\boldsymbol{\theta}$, and $\mathbb{R}^\kappa$ is the objective space. For the sake of a consistent discussion in this section, we will consider that our objective is to minimize[3] $\mathbf{f}(\boldsymbol{\theta})$. Accordingly, the goal of VOP is to find $\boldsymbol{\theta}^*$ that simultaneously minimizes all $f_j(.)$'s. In practice, the individual objective functions can be in contradiction to each other; i.e. an improvement with regard to one objective can cause the deterioration of at least another objective function.

Since minimization of any objective function presupposes that various objective function values can be compared with each other, an appropriate ordering concept that is suitable for VOP is needed on the objective space $\mathbb{R}^\kappa$. For reasons that will be shown later, it is difficult to have a total ordering that compares any two arbitrary elements in $\mathbb{R}^\kappa$, therefore a *weaker*, or a *partial ordering relation* denoted by "$\leq$" will be used instead.

Definition (*Order relation "$\leq$" in the objective space* $\mathbb{R}^\kappa$) Let $\mathbf{z}_1$ and $\mathbf{z}_2$ be two points in the objective space $\mathbb{R}^\kappa$. The order relation "$\leq$" is defined as $\mathbf{z}_1 \leq \mathbf{z}_2 \iff \mathbf{z}_2 - \mathbf{z}_1 \in \mathbb{R}^\kappa_+$, where $\mathbb{R}^\kappa_+ = \{\mathbf{z} \in \mathbb{R}^\kappa \mid z_i \geq 0, \quad \text{and} \quad 1 \leq i \leq \kappa\}$ is the nonnegative orthant of $\mathbb{R}^\kappa$ and, $\forall i \in \{1, \dots, \kappa\}$, $z_1^i \leq z_2^i$, $\exists j \in \{1, \dots, \kappa\}$ s.t. $z_1^j < z_2^j$.

Since $\mathbb{R}^\kappa_+$ is a special case of the convex cone, then "$\leq$" is guaranteed to be compatible with the linear structure of $\mathbb{R}^\kappa$; i.e. for $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3 \in \mathbb{R}^k$, and $\alpha \in \mathbb{R}$, $\alpha > 0$, then 1) if $\mathbf{z}_1 \leq \mathbf{z}_2 \Rightarrow \alpha\mathbf{z}_1 \leq \alpha\mathbf{z}_2$, and 2) if $\mathbf{z}_1 \leq \mathbf{z}_2 \Rightarrow \mathbf{z}_1 + \mathbf{z}_3 \leq \mathbf{z}_2 + \mathbf{z}_3$. In addition, properties such as reflexivity, transitivity, and antisymmetry are all satisfied for the relation "$\leq$" [37].

In optimization terms, if $\mathbf{z}_1 = \mathbf{f}(\boldsymbol{\theta}_1)$ and $\mathbf{z}_2 = \mathbf{f}(\boldsymbol{\theta}_2)$ represent two values of a vector valued objective function, then $\mathbf{z}_1 \leq \mathbf{z}_2$ implies that $\mathbf{z}_1$ is at least as small (as good) as $\mathbf{z}_2$

[3] Inverting the discussion on maximizing $\mathbf{f}(\boldsymbol{\theta})$ can be simply done by minimizing $-\mathbf{f}(\boldsymbol{\theta})$.

with regard to all objectives and it is strictly smaller (or better) with regard to at least one objective. In this case, $\boldsymbol{\theta}_1$ is said to *dominate* $\boldsymbol{\theta}_2$. There are vector pairs, however, for which neither $\mathbf{z}_1 \leq \mathbf{z}_2$ nor $\mathbf{z}_1 \geq \mathbf{z}_2$ are true; for instance the vectors $[2, 4]^\top$ and $[4, 2]^\top$. In such cases, the partial order relation reflects the fact that both objectives are of equal importance, and to select one solution, additional input is required from the domain expert, or the *decision maker*. To this end, it is very important to emphasize the main difference between scalar valued optimization and vector valued optimization. While the former possesses a total ordering relation induced by the real numbers, the latter possesses only a partial ordering relation according to the definition above. On the basis of this ordering concept, we can proceed with a formal definition for the task of VOP [89, 37].

Definition Let $\mathcal{Z} = \mathbf{f}(\mathcal{R}) \subseteq \mathbb{R}^\kappa$ be the image of the feasible set $\mathcal{R} \subseteq \mathbb{R}^d$ in the objective space. A point $\mathbf{z}^* \in \mathcal{Z}$ is called *Globally Efficient* with regards to the order relation "$\leq$" defined on $\mathbb{R}^\kappa$, if and only if there exists no other $\mathbf{z} \in \mathcal{Z}$ s.t. $\mathbf{z} \leq \mathbf{z}^*$ and $\mathbf{z} \neq \mathbf{z}^*$. A point $\boldsymbol{\theta}^* \in \mathcal{R}$ is called *Globally Pareto Optimal* if and only if $\mathbf{z}^* = \mathbf{f}(\boldsymbol{\theta}^*)$ is *globally efficient*. A point $\boldsymbol{\theta}_1 \in \mathcal{R}$ is said to *Dominate* another point $\boldsymbol{\theta}_2 \in \mathcal{R}$ if and only if $\mathbf{f}(\boldsymbol{\theta}_1) \leq \mathbf{f}(\boldsymbol{\theta}_2)$ and $\mathbf{f}(\boldsymbol{\theta}_1) \neq \mathbf{f}(\boldsymbol{\theta}_2)$.

Based on these definitions, VOP can be formally defined as finding *efficient points* $\mathbf{z}^* \in \mathcal{Z}$ with regard to the order relation "$\leq$" on $\mathbb{R}^\kappa$, along with their *Pareto optimal points* $\boldsymbol{\theta}^*$ pertaining to them [38]. It could be the case, however, that all objective functions are of equal importance. In this case, the best that VOP can do is to provide the decision maker a set of all *efficient points* along with their *Pareto optimal points* pertaining to them. The set of all *efficient points* and their pertaining *Pareto optimal points* are known as the *Efficient Set* and the *Pareto Set* respectively.

There are various techniques for solving VOP problems, and the interested reader can refer to [38] and [37] for a rigorous treatment of the subject. A class of these techniques form what are known as deterministic methods. These methods "*scalarize*" the vector optimization problem through a parametric formulation and then solve the new objective function using standard optimization techniques. From the deterministic class, we found that the weighted–sum method [38] and the weighted L_δ–Metric method [37] are very well studied scalarizing techniques with concrete theoretical results that guarantee *Pareto optimal solutions*. The L_δ–Metric method is also known as the compromise method, the

target method, or the approximation of the ideal point method [38] for reasons that will be explained in Subsection 4.5.2.

4.5.1 The weighted–sum method

The weighted–sum (WS) method was first introduced by Zadeh [109] and it is probably the most widely known vector optimization method. The WS method assigns a weight w_j to each objective function such that $w_j \geq 0$, $\forall\ j \in \{1, \ldots, \kappa\}$, and $\sum_{j=1}^{\kappa} w_j = 1$. That is, the WS method forms a convex combination of the objective functions. The final objective function to be minimized is:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta} \in \mathcal{R}} \ \mathbf{w}^\top \mathbf{f}(\boldsymbol{\theta}), \tag{4.9}$$

where $\mathbf{w} = [w_1 \ldots w_\kappa]^\top$. The weight w_j reflects the significance of the individual objective function $f_j(\cdot)$, and hence, it can reflect some *a priori* knowledge from the problem domain or, impose some bias on the final solution $\boldsymbol{\theta}^*$. By varying the weight vector $\mathbf{w}$, one can obtain a subset of the *Efficient Set* and its pertaining subset of *Pareto optimal solutions*. We state here Theorem 4.1 from [38] (*see Chapter 3 for a complete proof*) that guarantees a *Pareto optimal solution* for the WS method in Problem (4.9).

Theorem 4.5.1 *Let $\boldsymbol{\theta}^* \in \mathcal{R}$ be an optimal solution of (4.9), then the following statements hold:*

1. *If $\mathbf{w} \geq 0$, then $\boldsymbol{\theta}^*$ is* Pareto optimal.
2. *If $\mathbf{w} \geq 0$ and $\boldsymbol{\theta}^*$ is a unique optimal solution of (4.9), then $\boldsymbol{\theta}^*$ is* globally Pareto optimal.

Theorem 4.5.1 has more details in [38] than those presented here since it further discriminates between different *Pareto optimal solutions*. The most relevant detail to our discussion is that if $\mathbf{w} > 0$, i.e. all its components are strictly greater than zero, then the solution is known to be a *properly Pareto optimal* solution, while if $\mathbf{w} > 0$ and $\boldsymbol{\theta}^*$ is a unique solution, then $\boldsymbol{\theta}^*$ is known to be a *strong Pareto optimal* solution. We will rely on this property of the weight vector when this method will be presented in the context of LDA in Section 4.6.

The WS method however has an implicit assumption which can easily be a drawback in practice. The method requires that $\mathcal{Z} = \mathbf{f}(\mathcal{R})$ be a convex set. In practice the set $\mathcal{Z}$ might not necessarily be convex and as a side effect, there will be a set of efficient solutions $\mathbf{z}^*$ that can not be found using the WS method. In other words, the WS method might work poorly for non–convex $\mathcal{Z}$.

4.5.2 The L_δ–metric method

In an ideal situation, the objective of VOP is to achieve the optimal solution for each individual objective function $f_j(\cdot)$. Let $\mathbf{t}^* \in \mathbb{R}^\kappa$ be such an ideal target point in the objective space. Then, $\forall\ \mathbf{z} \in \mathcal{Z}$, $\mathbf{t}^* \leq \mathbf{z}$ and $\mathbf{t}^*$ might or might not be in $\mathcal{Z}$. Since in real world problems, the individual objectives might conflict with each other, achieving $\mathbf{t}^*$ is impossible, however it can serve as a reference point with the goal of seeking a solution as close as possible to $\mathbf{t}^*$ (see Figure 4.3). Formally, given a distance function $dist : \mathbb{R}^\kappa \times \mathbb{R}^\kappa \to \mathbb{R}_+$, the L_δ–Metric method is given by: $\min_{\boldsymbol{\theta}\in\mathcal{R}} dist(\mathbf{f}(\boldsymbol{\theta}) - \mathbf{t}^*)$. Since the objective space $\mathbb{R}^\kappa$ is endowed with a vector norm $\|\cdot\|$ then the induced weighted distance, or the L_δ–Metric method can be defined as follows:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\mathcal{R}} d(\boldsymbol{\theta}), \quad \text{where} \tag{4.10}$$
$$d(\boldsymbol{\theta}) = \left(\sum_{j=1}^{\kappa} w_j |f_j(\boldsymbol{\theta}) - t_j^*|^\delta \right)^{\frac{1}{\delta}},$$

$\delta \in [1, \infty]$, $w_j > 0$ is the weight for the j-th objective function, and $\sum_{j=1}^{k} w_j = 1$. Similar to the WS method, the weight w_j reflects the significance of the objective function $f_j(\cdot)$. Note that the WS method can be considered as a special case from the L_δ–Metric method. Also note that by definition of the L_δ–Metric method, the weight vector $\mathbf{w} > 0$, and according to Theorem 4.5.1, $\boldsymbol{\theta}^*$ will be at least a *properly Pareto optimal* solution. We now state Theorem 4.20 from [38] (see proof in pp. 112), that links the monotonicity of a norm to the solution obtained by Problem (4.10), in order to introduce our main result of this section in Corollary 4.5.3.

Theorem 4.5.2 (4.20 in [38]) *If $\|\cdot\|$ is a strictly monotonic norm and $\boldsymbol{\theta}^*$ is an optimal solution of Problem (4.10), then $\boldsymbol{\theta}^*$ is* Pareto optimal.

Corollary 4.5.3 *For the L_δ norm $\|\cdot\|_\delta$, if $1 \leq \delta < \infty$ and $\boldsymbol{\theta}^*$ is the optimal solution for Problem (4.10), then $\|\cdot\|_\delta$ is strictly monotonic, and $\boldsymbol{\theta}^*$ is* Pareto optimal.

The L_δ–Metric method has a nice interpretation in terms of level sets [38] $\{\mathbf{z} \in \mathbb{R}^\kappa \mid \|\mathbf{z} - \mathbf{t}^*\|_\delta \leq u\}$ where such sets contain all points of distance u or less to t^*. From that perspective, the goal of the L_δ–Metric method is to search for the smallest u such that the intersection of the corresponding level set with $\mathcal{Z} = \mathbf{f}(\mathcal{R})$ is nonempty. Figure 4.3 illustrates this concept for the L_2 norm.

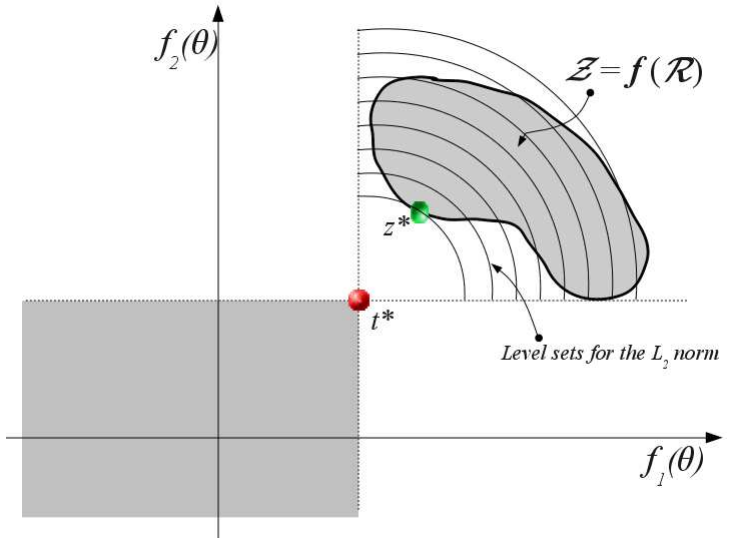


Fig. 4.3 The intersection of level sets for the L_δ–Metric method for $\delta = 2$ and with $\mathcal{Z} = \mathbf{f}(\mathcal{R})$ in the objective space. Note that the ideal point $\mathbf{t}^* \notin \mathbf{f}(\mathcal{R})$ and the efficient point $\mathbf{z}^*$ is the closest to it.

4.6 Pareto Discriminant Analysis

It is possible now to formulate the multiclass heteroscedastic discriminant analysis (HDA) model in the multiobjective optimization framework introduced in the previous section. Since the proposed framework is meant to handle the multiclass heteroscedastic setting while trying to avoid the class merging problem, a clear understanding for the class separation problem is needed first.

4.6.1 The class masking problem

Let us recall the original objective function optimized by FDA and MODA in the case of two–class problems:

$$J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B}) = \operatorname{tr}\left[(\mathbf{B}^\top \boldsymbol{\Sigma}_i \mathbf{B})^{-1}(\mathbf{B}^\top \mathbf{A}_{ij} \mathbf{B})\right] + \operatorname{tr}\left[(\mathbf{B}^\top \boldsymbol{\Sigma}_j \mathbf{B})^{-1}(\mathbf{B}^\top \mathbf{A}_{ji} \mathbf{B})\right] \ , \tag{4.11}$$

where $\mathbf{A}_{ij} = \mathbf{u}_{ij}\mathbf{u}_{ij}^\top + \boldsymbol{\Sigma}_j$, $\mathbf{A}_{ji} = \mathbf{u}_{ij}\mathbf{u}_{ij}^\top + \boldsymbol{\Sigma}_i$, and $\mathbf{u}_{ij} = (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)$. Maximizing Equation (4.11) with respect to $\mathbf{B}$ will find a projection into a lower dimensional subspace that maximizes the symmetric KL divergence between $\mathcal{G}_i$ and $\mathcal{G}_j$. The final solution $\mathbf{B}_{ij}^*$ will be optimal in terms of separation for classes $\mathcal{G}_i$ and $\mathcal{G}_j$. To account for the multiclass setting, LDA and MODA use the same objective function in which it sums over all pairwise SKLD and maximizes that sum:

$$E_{\text{MODA}}(\mathbf{B}) = \sum_{i=1}^{c-1} \sum_{j=i+1}^{c} J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B}). \tag{4.12}$$

Note that the original SKLD between two Gaussians $\mathcal{G}_i$ and $\mathcal{G}_j$,

$$J(\mathcal{G}_i, \mathcal{G}_j) = (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^\top (\boldsymbol{\Sigma}_i^{-1} + \boldsymbol{\Sigma}_j^{-1})(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) + \operatorname{tr}\left(\boldsymbol{\Sigma}_i \boldsymbol{\Sigma}_j^{-1} + \boldsymbol{\Sigma}_i^{-1} \boldsymbol{\Sigma}_j - 2\mathbf{I}\right), \tag{4.13}$$

has in fact two terms: the first term, which is a quadratic distance, measures the difference between the means $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$, and a second term which measures the difference, or the discrepancy between the two covariances $\boldsymbol{\Sigma}_i$ and $\boldsymbol{\Sigma}_j$ of $\mathcal{G}_i$ and $\mathcal{G}_j$ respectively (see Kullback [92] pp. 6–7 for this explanation). From the optimization of minimax functions [88], it is known that the sum of positive powered functions, $\sum_{j=1}^{m}[f_j]^p$, for $p > 1$, is a smooth approximation for $\max_{1\leq j\leq m}[f_j]^p$ as p is increasing, and hence $\sum_{j=1}^{m}[f_j]^p \approx [f_r]^p$ where $f_r > f_j \ \forall j \neq r$. Using this argument in the light of Equation (4.12), it is possible to see that LDA/MODA's objective function is in fact a smooth approximation of the following:

$$E_{\text{MODA}}(\mathbf{B}) \approx \max_{i,j} J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B}), \quad 1 \leq i, j \leq c \ , \ i \neq j \ , \tag{4.14}$$

due to the first term in Equation (4.13) which is the quadratic distance between the means of $\mathcal{G}_i$ and $\mathcal{G}_j$. Hence, LDA and MODA intrinsically prefer solutions that encourage maximizing the largest distance between $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$ in the input space to make it even larger in

the lower dimensional subspace. In other words, shifting the problem from the two–class setting to the multiclass setting using this scalarization technique (plain summing over all objectives) intrinsically yields the class separation problem. Consequently, LDA and MODA put needless effort to maximize already distant classes in the input space.

4.6.2 A multiobjective optimization framework for HDA

Here we propose a different scalarization for the multiclass heteroscedastic setting using the multiobjective optimization framework. In this framework, since each pair of classes, $\mathcal{G}_i$ and $\mathcal{G}_j$, $1 \leq i, j \leq c$, $i \neq j$, define their own individual objective function $J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B})$, then all $\kappa = c(c-1)/2$ pairs of classes result in κ objective functions that need be simultaneously optimized. Since it is expected that the objective functions can conflict with each other, using the VOP framework can guarantee that the obtained subspace will be in maximal agreement with all pairwise objectives, but suboptimal for each individual objective. Using the WS method or the L_δ–Metric method, and setting the appropriate weight vector for each model, the optimization effort will be distributed according to the class topology of the classification problem. The simultaneous optimization of the objective functions will put more effort on overlapping classes while safeguarding distant classes from overlapping in the lower dimensional subspace.

This is the major difference between MODA and LDA on one side and Pareto discriminant analysis (PARDA) on the other side. While MODA (and LDA in the homoscedastic case) sum over all $J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B})$'s, and search for a basis that maximizes that sum, PARDA plugs all the pairwise objective functions in a multiobjective optimization framework and searches for a basis that is in maximal agreement with all pairwise objective functions and simultaneously maximizes them.

Formally, using the scalarization of the WS method in Problem (4.9) (with Theorem 4.5.1), and the L_δ–Metric method in Equation (4.10) (with Corollary 4.5.3), Pareto dis-

criminant analysis (PARDA) can be defined using the following two optimization problems:

$$\mathbf{B}^*_{\text{WS}} = \arg\max_{\mathbf{B}\in\mathcal{R}} \; E_{\text{WS}}(\mathbf{B}), \quad \text{where} \tag{4.15}$$

$$E_{\text{WS}}(\mathbf{B}) = \sum_{i=1}^{c}\sum_{j=i+1}^{c} w_{ij} J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B}), \text{ s.t. } \sum_{i,j} w_{ij} = 1 \ , w_{ij} > 0 \ , \text{ and}$$

$$\mathbf{B}^*_{\text{L}\delta} = \arg\min_{\mathbf{B}\in\mathcal{R}} \; E_{\text{L}\delta}(\mathbf{B}), \quad \text{where} \tag{4.16}$$

$$E_{\text{L}\delta}(\mathbf{B}) = \sum_{i=1}^{c}\sum_{j=i+1}^{c} w_{ij} [J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B}) - t^*_{ij}]^2, \text{ s.t. } \sum_{i,j} w_{ij} = 1 \ , w_{ij} > 0 \ ,$$

where $\mathcal{R} \subseteq \mathbb{R}^{p\times p_0}$, and the sum was decomposed over all pairwise classes. For the L_δ–Metric method, δ was set to 2 to guarantee the strict monotonicity of the norm. According to Theorem 4.5.1 and Corollary 4.5.3, and given a proper target vector $\mathbf{t}^*$, the obtained solutions $\mathbf{B}^*_{\text{WS}}$ and $\mathbf{B}^*_{\text{L}\delta}$ from Problems (4.15) and (4.16) respectively, will be *Pareto Optimal*.

Note that MODA can be considered a special case of E_{WS} if $w_{ij} = 1 \ \forall \ i, j$ and, if the convex combination constraint on the weights is neglected. A similar remark follows for GADA [87] which is a generalized mean function over Bregman divergence measures. Also, note that since $\mathbf{B}^*_{\text{L}\delta}$ is found by minimizing Problem (4.16), then $t_{ij} > 0$, for if $t_{ij} = 0$, it will encourage reducing the divergence between $\mathcal{G}_i$ and $\mathcal{G}_j$ to zero and hence increase the overlap between them until they collapse over each other – which is obviously an undesirable solution.

4.6.3 Minimization of PARDA

Unfortunately, there is no closed form solution for the optimization of the objective functions in Problems (4.15) and (4.16), and an iterative algorithm based on gradient ascent (descent) is used instead. For E_{WS} ,

$$\mathbf{B}^{t+1} = \mathbf{B}^t + \eta_1 \frac{\partial E_{\text{WS}}(\mathbf{B})}{\partial \mathbf{B}}, \quad \text{where} \tag{4.17}$$

$$\frac{\partial E_{\text{WS}}(\mathbf{B})}{\partial \mathbf{B}} = \sum_{i=1}^{c}\sum_{j=i+1}^{c} w_{ij} \ \frac{\partial J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B})}{\partial \mathbf{B}} \quad , \tag{4.18}$$

while for $E_{\mathrm{L}\delta}$,

$$\mathbf{B}^{t+1} = \mathbf{B}^t - \eta_2 \frac{\partial E_{\mathrm{L}\delta}(\mathbf{B})}{\partial \mathbf{B}}, \text{ where} \tag{4.19}$$

$$\frac{\partial E_{\mathrm{L}\delta}(\mathbf{B})}{\partial \mathbf{B}} = \sum_{i=1}^{c} \sum_{j=i+1}^{c} 2w_{ij}(J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B}) - t^*_{ij}) \, \frac{\partial J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B})}{\partial \mathbf{B}} \, , \tag{4.20}$$

where η_1 and η_2 are the step lengths for the gradient ascent (descent) procedures in Equations (4.17) and (4.19) respectively. Fortunately, the gradient of $J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B})$ with respect to $\mathbf{B}$ has a closed form solution as follows:

$$\frac{\partial J(\mathcal{G}_i, \mathcal{G}_j; \mathbf{B})}{\partial \mathbf{B}} = \{2\mathbf{A}_{ij}\mathbf{B}\mathbf{\Phi}_i - 2\mathbf{\Sigma}_i\mathbf{B}\mathbf{\Phi}_i\mathbf{Q}_{ij}\mathbf{\Phi}_i\} + \{2\mathbf{A}_{ji}\mathbf{B}\mathbf{\Phi}_j - 2\mathbf{\Sigma}_j\mathbf{B}\mathbf{\Phi}_j\mathbf{Q}_{ji}\mathbf{\Phi}_j\}\,, \tag{4.21}$$

where

$$\begin{aligned} &\mathbf{\Phi}_i = (\mathbf{B}^\top\mathbf{\Sigma}_i\mathbf{B})^{-1} \, , \;\; \mathbf{\Phi}_j = (\mathbf{B}^\top\mathbf{\Sigma}_j\mathbf{B})^{-1} \, , \\ &\mathbf{Q}_{ij} = (\mathbf{B}^\top\mathbf{A}_{ij}\mathbf{B}) \, , \;\; \text{and} \;\; \mathbf{Q}_{ji} = (\mathbf{B}^\top\mathbf{A}_{ji}\mathbf{B}). \end{aligned} \tag{4.22}$$

The step length parameters, η_1 and η_2, are initially small (0.01 in all our experiments) and they are decreased by a factor of 38.2%[4] if the objectives, $E_{\mathrm{WS}}(\mathbf{B})$ and $E_{\mathrm{L}\delta}(\mathbf{B})$, decrease (instead of increase) or increase (instead of decrease) respectively. Other strategies such as explicit line search are possible but this simple method has provided very good results in all our experiments. Both objective functions in Problem (4.15) and (4.16) are non–convex problems and any gradient ascent (or descent) method can be trapped into local minima. Therefore, we typically start the algorithm with multiple initializations (10 times in all our experiments) and select the solution with the lowest training error. Alternatively, the best solution $\mathbf{B}^*$ can be chosen with cross-validation, or via minimizing the validation error. More explanation on these details can be found in Section 4.7.

4.6.4 Initial basis $\mathbf{B}_0$

In order for the gradient ascent (descent) procedure to find a better and stable solution $\mathbf{B}^*$ away from local minima, a good initial guess $\mathbf{B}_0$ is needed to start the procedures. For this, we use one of the well known objectives of LDA, max $\mathrm{tr}\{(\mathbf{B}^\top\mathbf{S}_w\mathbf{B})^{-1}(\mathbf{B}^\top\mathbf{S}_t\mathbf{B})\}$, to

[4]This is in fact the golden ratio factor (1.618034) used in many single variable optimization procedures.

give an initial basis $\mathbf{B}_0$. Based on the results of Zhang & Sim [85] and Ding & Li [110], the null space of $\mathbf{S}_t$ contains no useful discriminatory information in the data, and hence we first apply whitening transformation on the data and discard the dimensions with zero eigenvalue. Next, in the whitened space, the eigenvectors of $\mathbf{S}_w$ are computed and used as an initial guess for $\mathbf{B}_0$. Note that the maximum rank for $\mathbf{S}_w$ is $\min(n-c, p)$.

4.6.5 The weight w_{ij}

The weight vector plays the crucial role in the WS and the L_δ–Metric methods since it drives the optimization procedure to concentrate its effort on more important objectives in favour of other less important ones. For discriminant analysis, one would desire to bias the solution towards classes that will overlap in the lower dimensional space. In our previous work [28], the weights relied on the target vector of the L_δ–Metric method.

Here I propose a more general approach for deciding the weights w_{ij} that does not depend on the target vector and can be used with the WS method and the L_δ–Metric method as well. In the first step, the symmetric KL divergence $J(\mathcal{G}_i, \mathcal{G}_j)$ is computed between every pair of classes in the lower dimensional subspace obtained by the initial basis $\mathbf{B}_0$. Then $w_{ij} = \bar{w}_{ij} / \sum_{i,j} \bar{w}_{ij}$, where $\bar{w}_{ij} = [2.J(\mathcal{G}_i, \mathcal{G}_j)]^{-2}$. This weighting scheme gives very small weights to distant classes (in terms of KL divergence) relative to the overall divergences between all classes, while it assigns large weights to close classes relative to the overall divergences between classes. In this way, the optimization procedure will focus more on finding linear transformations that separate nearby classes, while safeguard distant classes from overlapping.

4.6.6 Adaptive weights and the Pareto set of optimal solutions $\mathbf{B}^*$

The weight vector $\mathbf{w}$ assigned to the multiobjective function (WS or L_δ–Metric) at the beginning of the optimization procedure is fixed and does not change during the iterative optimization. The reason for that has to do with convergence[5]. If the weights are changed in each iteration, each set of weights will define a new optimization problem in each gradient ascent (descent) step. Consequently, the sequence of gradient vectors will point to different directions in the objectives space and will not converge to a solution. Having adaptive weights at each step of the gradient ascent (descent) procedure is possible assuming that

[5]Personal communication with Prof. M. Ehrgott [38].

the weights updating rule will result in consistent gradient directions towards the optimal solution.

A common practice in the MOP literature is to assign different weight vectors to the multiobjective function based on some *a priori* knowledge from the problem domain. For each weight vector, a Pareto optimal solution is obtained and the decision maker selects which Pareto solution will better fit the multiobjective functions involved. A more principled approach is to explore the manifold of Pareto solutions and how it reflects on the weight of each objective function. This gives the decision maker an informative interpretation of each Pareto solution. The set of Pareto optimal solutions lying on this differentiable manifold [37] is known as the *Pareto set*. Recently, there are two approaches to explore this manifold; the stochastic based approach which is due to Schäffler [111], and the deterministic based approach which is due to Hillermeier [37]. Investigating an adaptive weight approach for MOP, and the use of Schäffler or Hillermeier approaches to obtain a Pareto set in the context of discriminant analysis are worthwhile research directions that we leave for future research work.

4.6.7 The target vector $\mathbf{t}^*$

The target vector $\mathbf{t}^*$ plays an important role together with the weights w_{ij} for the L_δ–Metric method. To see this, note that the WS method, similar to LDA and MODA, does not impose any constraints on the minimum divergence between classes. This is unlike the L_δ–Metric method which uses the target vector that can act as a constraint on the minimum divergence between the classes. In the context of discriminant analysis, an ideal setting would be to find a subspace in which all classes are equally spread, or equidistant from each other. Given a set of properly selected weights, the target vector in the L_δ–Metric method will encourage the optimization procedure to favour solutions in which all classes are equidistant from each other. In other words, it will encourage the multiobjective optimization to focus its effort on separating overlapping classes, while safeguards well separated classes from overlapping in the lower dimensional subspace. This is the rational for our approach described here to select the target values for the L_δ–Metric method.

In our previous work [28], we proposed a method inspired by the *"ideal point"* concept

introduced in the previous section. Although this method helped to validate the theoretical framework in the preliminary experiments stage, in practice, and for large data sets, this method is not efficient since it requires solving each individual objective J using MODA first. Hence, computationally, it is not feasible for high dimensional data sets with large number of classes.

Here, I rely again on the initial basis $\mathbf{B}_0$ and all pairwise SKLD measures $J(\mathcal{G}_i, \mathcal{G}_j)$ in the low dimensional subspace obtained by $\mathbf{B}_0$. To achieve the ideal setting of equally spreading all the classes in the low dimensional subspace, I set the target values to be very large and equal for all the objective functions. That is, all t_{ij} values are set equal to one large value t^* that is obtained as follows:

$$t^* = [2.\max_{i,j} J(\mathcal{G}_i, \mathcal{G}_j)]^2 \ , \quad 1 \leq i, j \leq c \ , \quad \text{and} \quad i \neq j \ . \tag{4.23}$$

4.6.8 Handling large number of classes

The above PARDA formulation imposes a complexity constraint for high dimensional data sets with large number of classes. For a c–class problem, PARDA will construct a multiobjective optimization problem with $\kappa = c(c-1)/2$ objective functions. For an experimental small data set such as the CMU-PIE face data set [112][6] with only 68 classes and $32 \times 32 = 1024$ pixels (no. of fearures), the number of objective functions is 2278. Although the MOP theory is invariant to the number of objective functions, the capacity of computational resources will render this formulation non feasible. In the following we propose a different configuration that reduces the number of objective functions from $c(c-1)/2$ to only c objective functions.

The previous formulation considers one objective function for each pair of classes. This can be considered as a one–vs–one configuration which in turn drives the large number of objective functions. An alternative configuration is to consider a one–vs–all strategy where each class $\mathcal{G}_i$ is encouraged to pull itself away from all other classes (combined), while all other classes (combined) are also encouraged to be pulled far away from class $\mathcal{G}_i$.

[6] http://www.zjucadcg.cn/dengcai/Data/PIE/PIE_32x32.mat

Recall the objective function in Equation (4.11) between classes $\mathcal{G}_i$ and $\mathcal{G}_j$. For a c–class problem, $\mathcal{G}_i$ is the Gaussian distribution whose parameters $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ are estimated from the points in class ℓ_i . For all other data points that are not in class ℓ_i , let $\mathcal{G}'_i$ be the Gaussian distribution that models these data points with sample mean $\boldsymbol{\mu}'_i$ and a sample covariance matrix $\boldsymbol{\Sigma}'_i$. Plugging $\mathcal{G}_i$ and $\mathcal{G}'_i$ in Equation (4.11) will yield:

$$J(\mathcal{G}_i, \mathcal{G}'_i; \mathbf{B}) = \operatorname{tr}\left[(\mathbf{B}^\top \boldsymbol{\Sigma}_i \mathbf{B})^{-1}(\mathbf{B}^\top \mathbf{R}'_i \mathbf{B})\right] + \operatorname{tr}\left[(\mathbf{B}^\top \boldsymbol{\Sigma}'_i \mathbf{B})^{-1}(\mathbf{B}^\top \mathbf{R}_i \mathbf{B})\right] \ , \tag{4.24}$$

where $\mathbf{R}'_i = (\boldsymbol{\mu}_i - \boldsymbol{\mu}'_i)(\boldsymbol{\mu}_i - \boldsymbol{\mu}'_i)^\top + \boldsymbol{\Sigma}'_i$, and $\mathbf{R}_i = (\boldsymbol{\mu}_i - \boldsymbol{\mu}'_i)(\boldsymbol{\mu}_i - \boldsymbol{\mu}'_i)^\top + \boldsymbol{\Sigma}_i$.

Maximizing Equation (4.24) with respect to $\mathbf{B}$ will yield a subspace $\mathbf{B}^*_i$ that is optimal in terms of separation between $\mathcal{G}_i$ and all other classes combined. For a c–class problem, there will be c such objective functions that need to share one optimal solution $\mathbf{B}^*$. Again, this is a multiobjective optimization setting which can be encapsulated in a PARDA framework. Using the WS and the L_δ–Metric methods, the optimal linear transformation $\mathbf{B}^*$ is obtained using the new objective functions as follows:

$$\mathbf{B}^*_{\text{WS}} = \arg\max_{\mathbf{B}\in\mathcal{R}} \ E_{\text{WS}}(\mathbf{B}), \quad \text{where} \tag{4.25}$$

$$E_{\text{WS}}(\mathbf{B}) = \sum_{i=1}^{c} w_i J(\mathcal{G}_i, \mathcal{G}'_i; \mathbf{B}), \ \text{s.t.} \ \sum_i w_i = 1 \ , w_i > 0 \ , \ \text{and}$$

$$\mathbf{B}^*_{\text{L}\delta} = \arg\min_{\mathbf{B}\in\mathcal{R}} \ E_{\text{L}\delta}(\mathbf{B}), \quad \text{where} \tag{4.26}$$

$$E_{\text{L}\delta}(\mathbf{B}) = \sum_{i=1}^{c} w_i [J(\mathcal{G}_i, \mathcal{G}'_i; \mathbf{B}) - t^*_i]^2, \ \text{s.t.} \ \sum_i w_i = 1 \ , w_i > 0 \ .$$

Obviously, the new one–vs–all (OVA) configuration for the multiclass setting allows PARDA to require less memory resources than the one–vs–one (OVO) configuration. Similarly, if a parallel implementation is considered on today's multicore architectures, then PARDA with an OVA configuration will be much faster than PARDA with an OVO configuration. The weight values and the target values are not differently set in the new configuration, and they are exactly as described in the previous sections. The optimization using gradient ascent (descent) is slightly modified to accommodate the new configuration,

where the gradient equations in (4.18) and (4.20) are now changed to be:

$$\frac{\partial E_{\text{WS}}(\mathbf{B})}{\partial \mathbf{B}} = \sum_{i=1}^{c} w_i \, \frac{\partial J(\mathcal{G}_i, \mathcal{G}'_i; \mathbf{B})}{\partial \mathbf{B}} \ , \text{ and} \tag{4.27}$$

$$\frac{\partial E_{\text{L}\delta}(\mathbf{B})}{\partial \mathbf{B}} = \sum_{i=1}^{c} 2w_i \, \left(J(\mathcal{G}_i, \mathcal{G}'_i; \mathbf{B}) - t^*_{ij}\right) \frac{\partial J(\mathcal{G}_i, \mathcal{G}'_i; \mathbf{B})}{\partial \mathbf{B}} \ \text{ respectively.} \tag{4.28}$$

4.6.9 A note on computational complexity

The computational bottleneck for the OVA configuration of PARDA lies in evaluating the objective function in Equations (4.24), (4.25), and (4.26), and in evaluating the gradients in Equations (4.21), (4.27), and (4.28). The computational complexity for the objective function in Equation (4.24) is roughly $O\left(\kappa(p_0^3 + p_0 p^2)\right)$ in the worst case, where κ is constant factor. That is for a c class problem, the complexity is roughly $O\left(c\kappa(p_0^3 + p_0 p^2)\right)$ in the worst case. The computational complexity for the gradient in Equation (4.21) is roughly $O\left(\pi(p_0^3 + p^2 p_0^2)\right)$, where π is another constant factor, and for a c class problem it is $O\left(c\pi(p_0^3 + p^2 p_0^2)\right)$ in the worst case. Note that cubic terms are due to matrix inversions of the covariance matrices which luckily occur only in the lower dimensional space. Since minimizing PARDA is an iterative process based on a gradient descent algorithm, the above computational complexity is multiplied by the number of steps taken by the algorithm.

4.7 Experiments

We conducted extensive experiments on a diversity of real data sets to evaluate PARDA's performance. Four different PARDA objective functions were used in these experiments :

1. The original PARDA formulation using the one–vs–one configuration combined with both scalarization methods, L_δ–Metric (OVO-L_δ) and WS (OVO-WS).

2. The alternative PARDA formulation that uses the one–vs–all configuration combined with both scalarization methods, L_δ–Metric (OVA-L_δ) and WS (OVA-WS).

Six different algorithms are used for comparisons with PARDA's four algorithms: direct LDA (dLDA) [97] based on the implementation in [83], White+LDA (WLDA) [95],

Table 4.1 Specifications of the eighteen data sets used in our experiments.

Dataset	Size (n)	No. of features (p)	No. classes (c)	Source
glass	214	9	6	UCI [1]
iris	150	4	3	"
isolet	7797	617	26	"
letter	20000	16	26	"
lymphography	148	18	4	"
new thyroid	215	5	3	"
page blocks	5473	10	5	"
satimages	6435	36	6	"
segment	2310	18	7	"
shuttle	58000	9	7	"
vowel	990	11	11	"
yeast	1484	6	10	"
MNIST	10000	24×24	10	[113]
USPS	9298	16×16	10	[114]
CMU-PIE	11554	32×32	68	[112]
Yale--B	2414	32×32	38	[115]
ETH80	3280	64×64	8	[116]
OSP	2500	1080	10	[117]

aPAC [82], principle component analysis (PCA), relevant component analysis (RCA) [13], and MODA [86]. That is, including the four PARDA algorithms, ten algorithms in total will be used in this comparative study. Note that DLDA, WLDA and aPAC can find low dimensional subspaces of dimensionality p_0, where $1 \leq p_0 \leq \min(c-1, p-1)$, and their optimal discriminating subspace is achieved when $p_0 = \min(c-1, p-1)$.

4.7.1 Data sets

Our data set pool consists of eighteen data sets from various domains :

- Twelve data sets from the UCI machine learning repository [1] – these are `glass`, `iris`, `isolet`, `letter`, `lymphography`, `new thyroid`, `page blocks`, `satimages`,

`segment`, `shuttle`, `vowel` and `yeast`.

- Two handwritten digits data sets – MNIST [113] and USPS [114].
- Two data sets for face recognition – CMU-PIE [112] and the extended Yale–B [115] [7].
- One object recognition data set – ETH80 [116].
- The Ohio sitting posture data set (OSP) [117].

Note that all these data sets corresponds to multiclass problems on purpose. The size, number of features and the number of classes for these data sets are shown in Table 4.1.

Due to the large size of MNIST, we used only the first 1000 images from each class in the training set, which makes the total size of this data set 10000 samples. Unlike all other data sets, the original format for isolet, satimages, and USPS has explicit training and test sets. In order to have a homogenous set of experiments and a common ground for performance comparisons using cross validation, each test set was concatenated to its corresponding training set to form a unified data set like all other sets used in the experiments.

For ETH80, we used the cropped–close128 set, in which all images are RGB, cropped to 128×128 pixels, and the object is centred in the image. These images were transformed to intensity (grey scale) images and reduced in size to 64×64 pixels using a Gaussian pyramid of one level. Except for ETH80, no other preprocessing was applied to any data set.

4.7.2 Visual comparison of low dimensional projections

We first compare the different projections obtained from each algorithm when applied on real data sets. For the purpose of demonstration, we use two data sets from the UCI machine learning repository, iris and new thyroid, where each data set has three classes. Since $c = 3$, the linear transformation $\mathbf{B}^{p \times p_0}$ obtained from DLDA, WLDA and aPAC will be optimal when $p_0 = 2$.

[7] http://www.zjucadcg.cn/dengcai/Data/FaceData.html. CMU-PIE is in the random.mat file, and Yale–B is in the YaleB_32x32.mat file.

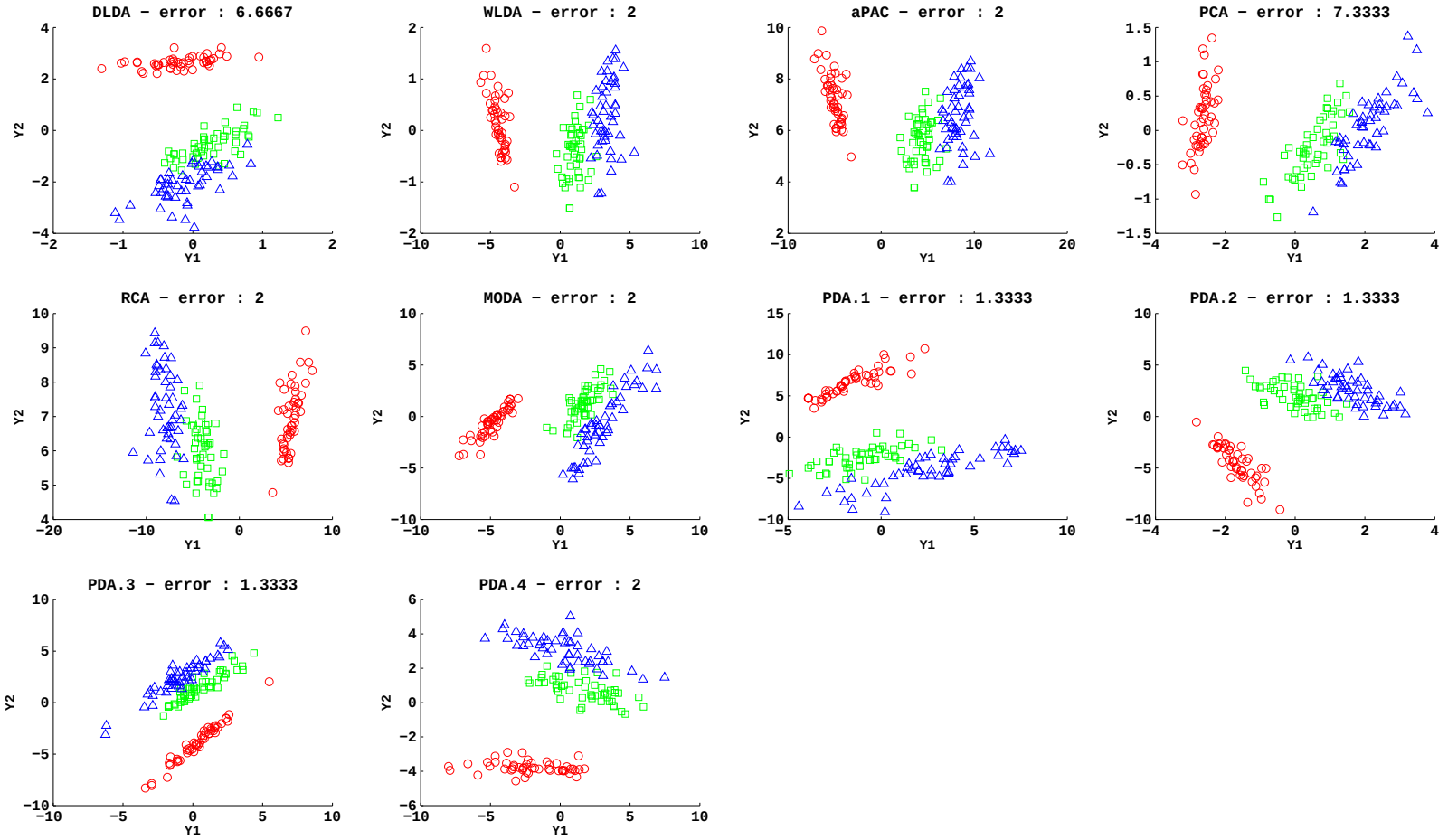


Fig. 4.4 Projections obtained from the ten algorithms used in this study on the `iris` data set ($c = 3$, $p = 4$, $n = 150$).

Table 4.2 Comparing the empirical error (%), with standard deviation, for DLDA, WLDA, aPAC, PCA, RCA and MODA with that of OVO-L_δ and OVO-WS for $p_0 = c - 1$.

Dataset	DLDA	WLDA	aPAC	PCA	RCA	MODA	OVO-L_δ	OVO-WS
glass	45.7 (7.8)	44.7 (9.6)	47.8 (9.0)	44.2 (11.1)	47.8 (9.0)	45.7 (10.5)	**41.0 (10.7)**	**43.6 (13.1)**
iris	4.6 (4.5)	**2.0 (4.5)**	**2.0 (4.5)**	2.6 (3.4)	**2.0 (4.5)**	**2.0 (4.5)**	**2.0 (4.5)**	5.3 (7.5)
isolet	6.9 (2.3)	6.3 (1.8)	6.2 (1.8)	9.2 (2.7)	6.2 (1.8)	11.5 (1.7)	**5.9 (1.0)**	**5.3 (1.2)**
letter	13.3 (0.9)	12.9 (1.1)	12.9 (1.1)	13.1 (0.9)	12.9 (1.0)	**12.7 (0.9)**	**12.7 (1.1)**	16.9 (1.5)
lymphography	32.5 (11.7)	28.7 (7.9)	28.7 (8.9)	38.1 (11.1)	30.0 (8.7)	42.5 (13.4)	**25.0 (9.7)**	**21.2 (11.8)**
new thyroid	24.7 (12.8)	10.4 (8.0)	9.0 (5.7)	17.1 (11.0)	9.0 (5.7)	4.7 (5.0)	6.6 (6.4)	**3.8 (7.0)**
page blocks	57.0 (15.9)	25.6 (13.0)	43.5 (15.2)	67.9 (12.9)	43.9 (15.1)	28.3 (19.2)	**17.4 (10.2)**	**15.3 (9.0)**
satimages	16.1 (4.8)	16.9 (4.7)	17.3 (4.8)	17.5 (5.1)	17.3 (4.8)	19.7 (5.3)	**15.9 (5.1)**	19.0 (3.7)
segment	22.6 (1.8)	7.4 (1.3)	7.6 (1.8)	22.1 (1.7)	14.4 (6.1)	7.7 (1.6)	**6.4 (1.5)**	7.7 (1.1)
shuttle	4.2 (0.2)	18.0 (4.7)	6.1 (0.4)	4.2 (0.2)	6.1 (0.4)	4.4 (0.3)	**3.7 (0.7)**	11.4 (2.5)
vowel	36.8 (9.6)	42.3 (8.8)	42.3 (8.8)	35.8 (8.3)	42.2 (8.9)	35.6 (8.5)	**32.5 (6.5)**	41.8 (8.9)
yeast	54.1 (3.0)	53.2 (3.1)	54.1 (2.4)	53.8 (3.0)	54.5 (2.5)	56.7 (2.9)	53.9 (3.4)	**52.8 (4.1)**
MNIST	13.1 (1.4)	13.0 (1.3)	15.6 (1.6)	13.9 (1.5)	13.6 (1.4)	18.1 (2.0)	**11.8 (1.0)**	**12.1 (1.2)**
USPS	9.7 (1.0)	6.8 (1.3)	7.1 (1.4)	11.2 (1.2)	7.1 (1.4)	8.6 (1.9)	**5.8 (1.2)**	**5.9 (5.9)**
CMU-PIE	7.6 (17.2)	11.4 (18.6)	15.7 (20.3)	7.7 (17.0)	15.7 (20.3)	**6.3 (16.0)**	N.A.	N.A.
Yale--B	6.9 (10.8)	12.6 (15.5)	14.2 (16.3)	6.5 (10.1)	14.2 (16.3)	**5.1 (8.4)**	8.5 (11.5)	27.9 (18.4)
ETH80	38.6 (10.2)	**28.9 (10.1)**	67.7 (5.1)	44.5 (7.6)	86.0 (1.5)	50.7 (6.3)	37.3 (9.1)	40.0 (7.7)
OSP	31.7 (7.2)	31.4 (7.1)	46.6 (8.5)	31.5 (8.7)	68.3 (5.7)	32.0 (7.2)	**30.0 (8.4)**	31.4 (6.4)

Figures 4.4 and (4.5) show the projections, with classification error, for the ten different algorithms on the iris and new thyroid data sets respectively. The reported error on the data sets is based on a quadratic classifier that is explained in the next subsection. It can be clearly seen that PARDA algorithms achieve the lowest errors amongst other algorithms, and their projections are very comparable to the best projections obtained from all other algorithms. On these two data sets, the WS method with both configurations, one–vs–one (PDA.2) and one–vs–all (PDA.4), achieve the lowest error rates.

4.7.3 Comparing classification error of low dimensional projections

The performance of all algorithms was measured using the classification error rate (%) in the low dimensional subspace defined by the linear transformation matrix $\mathbf{B}$ obtained from the different algorithms. The classification error reported here is the empirical error of a quadratic classifier (with standard deviation) using a 10 folds cross validation scheme. The quadratic classifier is used in the low dimensional subspace, where a new sample $\mathbf{x} \in \mathbb{R}^d$ is first projected into the lower dimensional subspace as $\mathbf{y} = \mathbf{B}^\top \mathbf{x}$, and then $\mathbf{y}$ is assigned the label of the nearest class. The nearest class is decided based on the minimum Mahalanobis distance $\sqrt{(\mathbf{y} - \hat{\boldsymbol{\mu}}_i)^\top \hat{\boldsymbol{\Sigma}}_i (\mathbf{y} - \hat{\boldsymbol{\mu}}_i)}$, where $\hat{\boldsymbol{\mu}}_i$ and $\hat{\boldsymbol{\Sigma}}_i$ are the sample mean and the sample covariance matrix for class $\mathcal{G}_i$ in the low dimensional subspace.

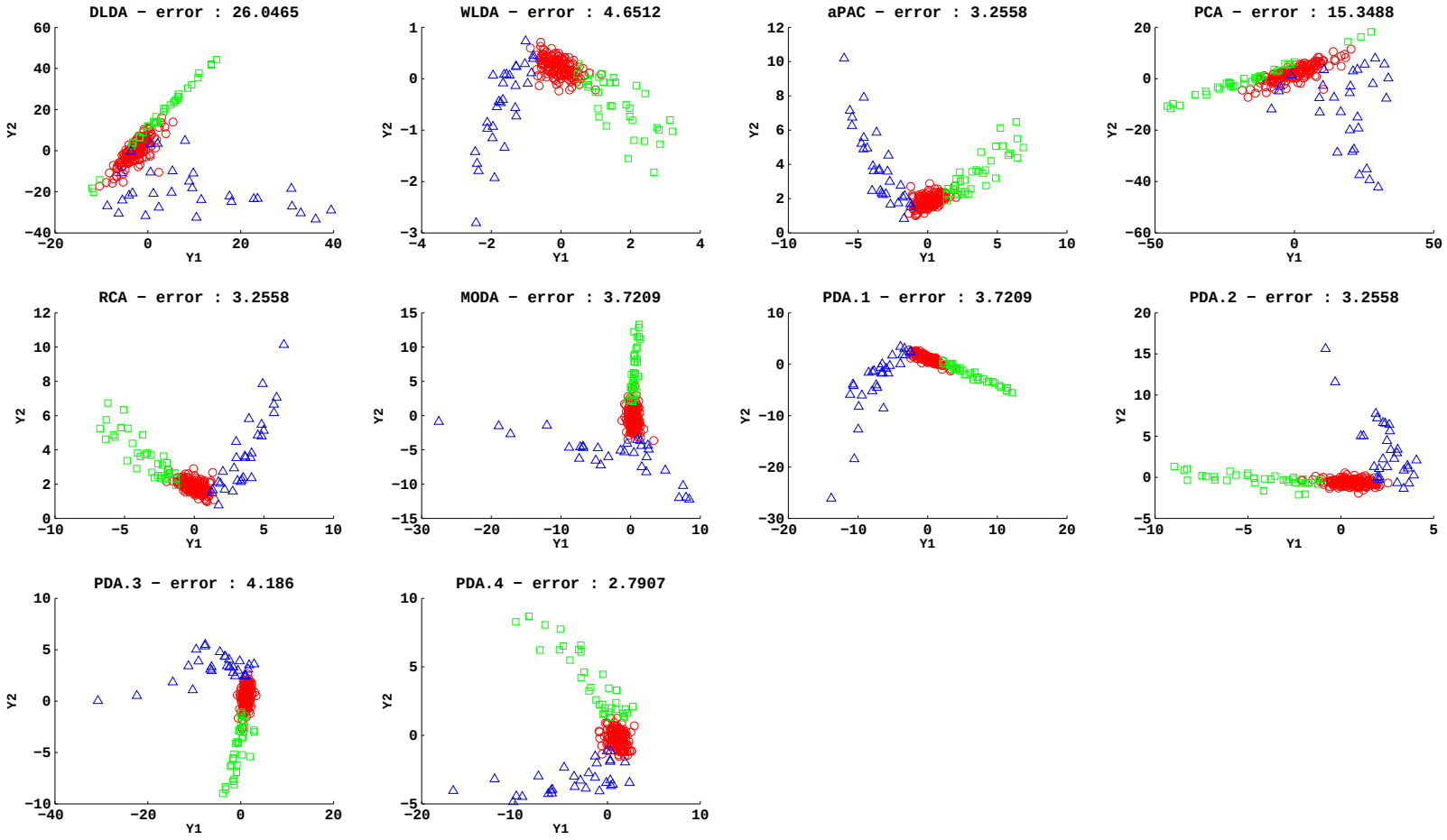


Fig. 4.5 Projections obtained from the ten algorithms used in this study on the `newthyroid` data set ($c = 3$, $p = 5$, $n = 215$).

Table 4.3 Comparing the empirical error (%), with standard deviation, for DLDA, WLDA, aPAC, PCA, RCA and MODA with that of OVA-L_δ and OVA-WS for $p_0 = c - 1$.

Dataset	DLDA	WLDA	aPAC	PCA	RCA	MODA	OVA-L_δ	OVA-WS
glass	45.7 (7.8)	44.7 (9.6)	47.8 (9.0)	**44.2 (11.1)**	47.8 (9.0)	45.7 (10.5)	51.0 (11.9)	52.6 (7.4)
iris	4.6 (4.5)	**2.0 (4.5)**	**2.0 (4.5)**	2.6 (3.4)	**2.0 (4.5)**	**2.0 (4.5)**	**2.0 (3.2)**	2.6 (4.6)
isolet	6.9 (2.3)	6.3 (1.8)	**6.2 (1.8)**	9.2 (2.7)	**6.2 (1.8)**	11.5 (1.7)	6.7 (1.4)	6.7 (6.1)
letter	13.3 (0.9)	12.9 (1.1)	12.9 (1.1)	13.1 (0.9)	12.9 (1.0)	**12.7 (0.9)**	**12.7 (1.0)**	17.1 (1.5)
lymphography	32.5 (11.7)	28.7 (7.9)	28.7 (8.9)	38.1 (11.1)	30.0 (8.7)	42.5 (13.4)	**26.8 (8.8)**	**18.1 (9.9)**
new thyroid	24.7 (12.8)	10.4 (8.0)	9.0 (5.7)	17.1 (11.0)	9.0 (5.7)	**4.7 (5.0)**	5.7 (6.2)	**4.7 (6.3)**
page blocks	57.0 (15.9)	25.6 (13.0)	43.5 (15.2)	67.9 (12.9)	43.9 (15.1)	28.3 (19.2)	**17.1 (7.9)**	**15.8 (8.9)**
satimages	16.1 (4.8)	16.9 (4.7)	17.3 (4.8)	17.5 (5.1)	17.3 (4.8)	19.7 (5.3)	**15.6 (5.5)**	18.9 (3.8)
segment	22.6 (1.8)	7.4 (1.3)	7.6 (1.8)	22.1 (1.7)	14.4 (6.1)	7.7 (1.6)	**6.3 (1.8)**	8.2 (1.3)
shuttle	4.2 (0.2)	18.0 (4.7)	6.1 (0.4)	4.2 (0.2)	6.1 (0.4)	4.4 (0.3)	**3.8 (0.2)**	12.2 (2.4)
vowel	36.8 (9.6)	42.3 (8.8)	42.3 (8.8)	35.8 (8.3)	42.2 (8.9)	35.6 (8.5)	**34.9 (7.5)**	41.2 (9.7)
yeast	54.1 (3.0)	**53.2 (3.1)**	54.1 (2.4)	53.8 (3.0)	54.5 (2.5)	56.7 (2.9)	**53.2 (4.0)**	54.2 (1.7)
MNIST	13.1 (1.4)	13.0 (1.3)	15.6 (1.6)	13.9 (1.5)	13.6 (1.4)	18.1 (2.0)	**12.4 (1.4)**	13.2 (1.1)
USPS	9.7 (1.0)	6.8 (1.3)	7.1 (1.4)	11.2 (1.2)	7.1 (1.4)	8.6 (1.9)	**6.3 (1.1)**	7.2 (0.9)
CMU-PIE	7.6 (17.2)	11.4 (18.6)	15.7 (20.3)	7.7 (17.0)	15.7 (20.3)	**6.3 (16.0)**	10.1 (20.4)	16.6 (22.2)
Yale--B	6.9 (10.8)	12.6 (15.5)	14.2 (16.3)	6.5 (10.1)	14.2 (16.3)	**5.1 (8.4)**	8.8 (12.4)	29.4 (18.2)
ETH80	38.6 (10.2)	**28.9 (10.1)**	67.7 (5.1)	44.5 (7.6)	86.0 (1.5)	50.7 (6.3)	39.6 (9.7)	40.9 (7.4)
OSP	31.7 (7.2)	31.4 (7.1)	46.6 (8.5)	31.5 (8.7)	68.3 (5.7)	32.0 (7.2)	**31.2 (7.6)**	**30.7 (6.6)**

Since the four PARDA algorithms studied here rely on MODA's formulation, and MODA is included in our comparative study, the same set of initialization parameters used with PARDA were applied to MODA. That is, the number of different initializations for MODA was 10, and the initial step length for its gradient ascent procedure was 0.01. In addition, similar to the four PARDA algorithms, MODA is applied on the whitened space of the data and not on the data's original input space.

4.7.4 Analysis of the results

Table 4.2 compares the empirical error for the six algorithms (DLDA, WLDA, aPAC, PCA, RCA and MODA) with PARDA's original configuration, one–vs–one, using the L_δ–Metric method (PDA.1) and the WS method (PDA.2), and for $p_0 = c - 1$. For most of the data sets, fifteen out of eighteen, it can be seen that PDA.1 and PDA.2 achieve the lowest error amongst other algorithms, with a slight edge for PDA.1 over PDA.2. Due to the high dimensionality (1024) and the large number of classes (68) for CMU-PIE, neither the WS method nor the L_δ–Metric method could be solved on our computational servers due to the

excessive amount of memory required by this configuration. This was simply solved using the one–vs–all configuration as shown in Table 4.3.

For CMU-PIE, Yale–B and ETH80, we note that MODA and WLDA had the lowest error on these data sets. There are two equally important reasons for that. First, our experience with PARDA algorithms tells that for data sets with a large number of classes and high dimensionality, the gradient ascent (descent) procedure can easily get stuck into local minima. An easy remedy for this problem is to increase the number of different initializations, albeit this would increase the total training time. A more principled approach is to investigate other optimization techniques specifically for multiobjective optimization in the context of discriminant analysis.

The second reason for PARDA's decreased performance is the weight vector. Since MODA is a special case of the WS method – i.e. no weights on the pairwise KL divergences – the improved performance of MODA (for CMU-PIE and Yale–B) over all other algorithms implies that all the classes for these problems are in proximity to each other in terms of KL divergence. This implies that the weights for PDA.1 and PDA.2 should be more uniform in these particular cases. Unfortunately, the weighting scheme described in Subsection 4.6.5 does not encourage such a uniform distribution for the weights.

It is important to note that the target vector for the L_δ–Metric method does not contribute to this decrease in performance. As explained in the previous section, this is due to our rationale for selecting target values that encourage the optimization procedure to prefer solutions in which all classes are equidistant from each other.

Table (4.3) compares the empirical error for the six algorithms with PARDA's alternative configuration, one–vs–all, using the L_δ–Metric method (PDA.3) and the WS method (PDA.4), for $p_0 = c-1$ as well. Here we note a slight performance decrease for PARDA algorithms in favour of WLDA and MODA. Nevertheless, for thirteen data sets out of eighteen, either PDA.3 or PDA.4 achieve the lowest error amongst all other algorithms. For other data sets, PDA.3 and PDA.4 maintain a competitive performance with all other algorithms.

For CMU-PIE, the error reported using the new one–vs–all PARDA configuration shows

Table 4.4 Comparing the lowest empirical error (%) of DLDA, WLDA, aPAC, PCA and MODA with the empirical error of OVO-L_δ, OVO-WS, OVA-L_δ, and OVA-WS for $p_0 = c - 1$.

Dataset	Lowest Error	OVO-L_δ	OVO-WS	OVA-L_δ	OVA-WS
glass	PCA 44.2 (11.1)	**41.0 (10.7)**	43.6 (13.1)	51.0 (11.9)	52.6 (7.4)
iris	**WLDA, aPAC, RCA, MODA 2.0 (4.5)**	**2.0 (4.5)**	5.3 (7.5)	2.0 (3.2)	2.6 (4.6)
isolet	aPAC, RCA 6.2 (1.8)	5.9 (1.0)	**5.3 (1.2)**	6.7 (1.4)	6.7 (6.1)
letter	**MODA 12.7 (0.9)**	**12.7 (1.1)**	16.9 (1.5)	**12.7 (1.0)**	17.1 (1.5)
lymphography	WLDA 28.7 (7.9)	25.0 (9.7)	21.2 (11.8)	26.8 (8.8)	**18.1 (9.9)**
new thyroid	MODA 4.7 (5.0)	6.6 (6.4)	**3.8 (7.0)**	5.7 (6.2)	4.7 (6.3)
page blocks	WLDA 25.6 (13.0)	17.4 (10.2)	**15.3 (9.0)**	17.1 (7.9)	15.8 (8.9)
satimages	DLDA 16.1 (4.8)	15.9 (5.1)	19.0 (3.7)	**15.6 (5.5)**	18.9 (3.8)
segment	WLDA 7.4 (1.3)	6.4 (1.5)	7.7 (1.1)	**6.3 (1.8)**	8.2 (1.3)
shuttle	DLDA, PCA 4.2 (0.2)	**3.7 (0.7)**	11.4 (2.5)	3.8 (0.2)	10.1 (2.1)
vowel	MODA 35.6 (8.5)	**32.5 (6.5)**	41.8 (8.9)	34.9 (7.5)	41.2 (9.7)
yeast	WLDA 53.2 (3.1)	53.9 (3.4)	**52.8 (4.1)**	53.2 (4.0)	54.2 (1.7)
MNIST	WLDA 13.0 (1.3)	**11.8 (1.0)**	12.1 (1.2)	12.4 (1.4)	13.2 (1.1)
USPS	WLDA 6.8 (1.3)	**5.8 (1.2)**	5.9 (5.9)	6.3 (1.1)	7.2 (0.9)
CMU-PIE	MODA **6.3 (16.0)**	N.A.	N.A.	10.1 (20.4)	18.3 (22.7)
Yale--B	MODA **5.1 (8.4)**	8.5 (11.5)	27.9 (18.4)	8.8 (12.4)	29.4 (18.2)
ETH80	WLDA **28.9 (10.1)**	37.3 (9.1)	40.0 (7.7)	39.6 (9.7)	40.9 (7.4)
OSP	WLDA 31.4 (7.1)	**30.0 (8.4)**	31.4 (6.4)	31.2 (7.6)	30.7 (6.6)

that this configuration can accommodate data sets with large number of classes and large number of input features, while it can maintain a comparable performance with other algorithms. However, as explained earlier, the decreased performance for CMU-PIE, Yale–B and ETH80 is due local minima and the initial weights on the pairwise KL divergences.

Since the one–vs–all PARDA configuration is an approximation to the original one–vs–one PARDA configuration, it is important to compare these two configurations using the two proposed scalarization techniques, WS and L_δ–Metric. Table 4.4 compares the four different PARDA algorithms with the lowest error achieved by any of the other six different algorithms.

First, by comparing the L_δ–Metric scalarization method using both configurations, one–vs–one (PDA.1) and one–vs–all (PDA.3), it can be seen that both models maintain very similar results with a slight improvement for PDA.1 over PDA.3. This is expected since the the one–vs–one configuration is more faithful to the class topology and assigns an objective function for every pair of different classes. A similar remark can be made with regard

to the WS scalarization method under both configurations. That is, in general there is no significant increase in error when considering the one–vs–all instead of the one–vs-one configuration.

This is an encouraging result since it permits, without sacrificing performance, to replace the one–vs–one configuration that scales quadratically with the number of classes, with an efficient configuration that scales linearly with the number of classes. This downscale of model complexity has a direct impact on the time and space complexity of the PARDA framework.

Second, by comparing both scalarization methods, L_δ–Metric vs. WS, under both configurations, it can be seen that the L_δ–Metric method is usually superior to the WS method. Note that both methods use the same approach for setting the weights on the pairwise SKLDs as explained in Subsection 4.6.5, and hence the inferior performance of WS in general is not due the weights selection. However, there are two reasons for the superior performance of the L_δ–Metric method. 1) As discussed in Subsection 4.5.1, the WS method assumes that the set of efficient solutions in the objective space is a convex set. Since in practice this might not be true, there will be a set of efficient solutions that can not be found by the WS method. 2) Similar to MODA, the WS method does not impose any constraints on the minimum distance between the classes. This is unlike the L_δ–Metric method in which the target vector encourages the optimization procedure to equally spread the classes in the low dimensional subspace. Finally, in terms of running time, it was noticed that the L_δ–Metric method, in both configurations, is much faster than the WS method.

In summary, the above analysis strongly suggests that PARDA using the L_δ–Metric scalarization method and a one–vs–all configuration, form an efficient algorithm for multiclass heteroscedastic discriminant analysis.

4.8 Discussion and Concluding Remarks

In this chapter I have presented a supervised subspace learning algorithm based on Fisher discriminant analysis (FDA). The algorithm is based on a fundamentally new perspective

for the multiclass linear dimensionality reduction problem. Here, the multiclass problem is perceived as a set of pairwise, possibly conflicting, objective functions representing the pairwise distance between different classes. This perspective raised the need for the machinery of multiobjective optimization for which its optimal solution is known to exist.

The optimal solution for this multiobjective problem, known as Pareto optimal, is suboptimal for each individual objective function, but is in maximal agreement between all the possibly conflicting objective functions. It is this nature of the Pareto solution that allowed us to use it efficiently in subspace learning. In terms of discriminating subspaces, the Pareto subspace separates between classes that overlap in the input space, while safeguards already distant classes from overlapping in the embedding space. In fact, the proposed algorithm encourages Pareto subspaces in which all classes are equidistant from each other.

The Pareto framework presented here for discriminant analysis imposes different questions in various research directions. The first question is on the initial weight for each objective function. How to choose these weights, should they be updated or not, and if so, what is the update rule for these weights to guarantee convergence to the right Pareto solution? A second question is about the objective functions in Equations (4.25) and (4.26). These objective functions do not have a regularization term on the variable $\mathbf{B}$, nor do they impose an orthogonality constraint on $\mathbf{B}$ in the embedding space. A third question is on the gradient descent procedure used for optimization. Since the objective functions in Equations (4.25) and (4.26) are weighted summations of convex functions (the symmetric KL divergence) with positive weights, is there any potential for using more sophisticated algorithms such as the Newton method? Another question is whether the stochastic approach of Schäffler [111], or the deterministic algorithm of Hillermeier [37] for exploring all the Pareto set of solutions are suitable for discriminant analysis, since the variable of interest here is the matrix $\mathbf{B}$ which has a particular structure.

Taking few steps backward from the discriminant analysis context and terminology, the matrix $\mathbf{B}^*$ defines a low rank, square symmetric matrix $\mathbf{A}^* = \mathbf{B}^*\mathbf{B}^{*\top}$, which is the main element for a GQD-type semi-metric. That is, since $\mathbf{A}^*$ is low rank, then $\mathbf{A}^*$ is PSD, and hence it defines a semi-metric space $(\mathbb{R}^p, \|\cdot\|_{\mathbf{A}^*})$, which is suitable for classification purposes. While semi-metric spaces can be useful for classification purposes, they are

risky for clustering and unsupervised learning. As it will be shown in the next chapter, unsupervised embedding in semi-metric spaces can yield unstable embeddings, and can easily collapse all points in the input space into one point in the embedding space. These are undesirable outcomes for unsupervised algorithms for metric learning and dimensionality reduction, since there are no labels to validate the new configuration of the points. In the following chapter, I will consider this more difficult setting for subspace learning in which there are no class or group labels on the data, and it is required to learn an embedding into a low dimensional subspace that reveals the natural structure and groupings in the data.

Chapter 5

Unsupervised Metric Learning

In this chapter, I consider the problem of learning a metric space for the data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$ when no *a priori* information in the form of labels or side-information are available for learning. Here I propose an algorithmic framework for learning a metric space based on spectral embedding methods. The algorithm has the following properties. (i) The algorithm is totally unsupervised, and hence it does not require partial labels nor partial side-information. (ii) The algorithm, simultaneously, overcomes the global Gaussian assumption and defines a metric that varies according to the sample density in the input space. One one hand, the algorithm can better accommodate the characteristics of real world data sets, and overcome the uneven sample distribution in the input space. On the other hand, the metric space obtained by the algorithm can reveal more about the non-spherical and non-compact clusters in the data, which finally improves the efficiency of clustering algorithms when applied to data embedded in this new metric space.

5.1 Motivation

The traditional approach for learning a metric over the data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X} \subset \mathbb{R}^p$ learns an instance of the generalized quadratic distance (GQD) :

$$d(\mathbf{x}_i, \mathbf{x}_j; \mathbf{A}) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{A} (\mathbf{x}_i - \mathbf{x}_j)}, \quad \mathbf{A} \succ 0,$$

in a supervised or a semi-supervised manner. A major limitation for this type of metric learning is that learning the matrix $\mathbf{A}$ requires the existence of at least partial labels or

partial side-information. Hence, these algorithms can not be applied in a fully unsupervised setting. A second limitation of these algorithms is that the GQD enforces a global Gaussian assumption over $\mathcal{D}$, which is unjustified given the characteristics of real world data sets discussed in Section 2.1.2. A third limitation, again due to the GQD, is that the metric is constant over the entire space, and hence, it does not take into consideration the uneven sample distribution in the input space. In this chapter, I propose a metric learning algorithm, based on spectral methods, that is totally unsupervised and can overcome the previously mentioned limitation of traditional metric learning algorithms.

Spectral embedding methods [18, 19, 20, 25, 26, 27] are a group of unsupervised, non-parametric learning algorithms that share the use of an eigendecomposition step for obtaining a lower dimensional embedding of the data set $\mathcal{D}$. The shared eigendecomposition step characterizes the nonlinear manifold $\mathcal{M}^{p_0}$ on the hyperplane $\mathbb{R}^{p_0} \subset \mathbb{R}^p$ on which the data set $\mathcal{D}$ would lie [22]. During this characterization, spectral methods perform two simultaneous tasks; dimensionality reduction, and the characterization of non-spherical, non-compact clusters which are intimately related to nonlinear manifolds (both are regions of high densities). Hence spectral methods, finally, obtain an embedding for the set $\mathcal{D}$ in the low dimensional space, where the structure and grouping in the data are manifested by the Euclidean distance.

Despite the interesting properties of spectral embedding methods, there are few issues that require careful consideration with this type of algorithm:

1. The limitations mentioned above for the GQD can still echo in spectral embedding methods through the Euclidean distance.

2. The embedding process itself can result in unstable, counter intuitive embeddings, or can easily collapse some (or all) points in $\mathcal{D}$ onto one or multiple points in the final embedding space.

3. The uneven sample distribution; most of spectral manifold learning algorithms try to preserve a certain notion of geometry, either globally and/or locally. If the data do not densely sample the manifold, the local and/or global structure of the manifold becomes difficult to estimate, and these algorithms recover low dimensional points that do not exhibit the desired attributes [71].

These are undesirable states for any unsupervised learning algorithm that will be deployed in the new embedding space. An unstable embedding can make the clustering problem harder, while collapsing the points can easily mislead the clustering algorithm.

Both issues of spectral methods are related to the input of the eigendecomposition step of spectral embedding methods. That is, the eigendecomposition step takes as input a symmetric PSD matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ that is extracted from the adjacency matrix for the graph $G(\mathcal{D}, \mathcal{E})$ defined over the data set $\mathcal{D}$, where $\mathcal{D}$ now acts as the set of vertices, and $\mathcal{E}$ is the set of edges. For instance, KPCA defines a fully connected graph over $\mathcal{D}$, while LLE, Isomap, Laplacian eigenmaps, and spectral clustering define fully connected or neighbourhood graphs using ϵ-balls or the k nearest neighbours (NN) of each $\mathbf{x}_i \in \mathcal{D}$.

The first issue of spectral methods is related to the similarity on the edge e_{ij} which, in many cases, is directly related to the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$. For instance, the similarity in KPCA, Laplacian eigenmaps, and spectral clustering, can be measured using kernels such as the linear dot product kernel, the exponential kernel $K_E = \frac{\|\mathbf{x}-\mathbf{y}\|_2}{\sigma}$, and the Gaussian kernel $K_G = \frac{\|\mathbf{x}-\mathbf{y}\|_2^2}{\sigma}$, where $\sigma > 0$ (see [22] for more details on LLE and Isomap). Since the Euclidean distance is a more restricted version of the GQD – replacing $\mathbf{A}$ with the identity matrix $\mathbf{I}$ – then the same limitation for the GQD applies to the Euclidean distance. What slightly leverages these limitations is the existence of affinity control parameters, as σ in K_E and K_G, that can be optimized according to the task under consideration.

The second issue of spectral methods is due to the metric properties of the similarity measure on the edge e_{ij}, even if it is not related to the Euclidean distance. If the similarity measure relies on a distance metric, where all metric axioms are satisfied, then the embedding process will respect these metric properties while characterizing the non-spherical and non-compact clusters in the data, yielding a meaningful embedding in the metric space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$. However, if the similarity measure relies on a semi-metric distance, then Axioms (3) and (4) are not required to hold[1]. That is, the triangle inequality might not be satisfied, and the distance between any two points a and b can be zero for any $a \neq b$. Then, during the embedding, these metric properties will be violated while characterizing the clusters, which might easily result in unstable embeddings, or collapse all the points onto one point

[1]See Appendix

in the embedding space. For instance, the difference between these two types of similarity measures can be noticed in K_E and K_G, where the former relies on a metric distance, while the latter relies on a semi-metric one. This issue will be discussed in detail with examples in Section 5.4.2, where the discussion is completely based on the results of I. Schoebërg in [43].

The above limitations of metric learning algorithms, and the special considerations of spectral embedding methods suggest that an unsupervised metric learning algorithm should meet the following requirements :

(i) The algorithm should not rely on partial labels nor partial side-information on the data.

(ii) The algorithm should further leverage the limitations of the GQD (and the Euclidean distance), which implies overcoming the global Gaussian assumption and the fact that the GQD is constant over the entire input space.

(iii) The algorithm should be careful about the metric properties of similarity measures used on the graph $G(\mathcal{D}, \mathcal{E})$, in order to obtain a proper embedding in the metric space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$, such that the structure and grouping in the data are better manifested by the Euclidean distance.

5.1.1 Requirements analysis

The first requirement above is to avoid any reliance on labels or side information for learning a metric space. To this end, I will rely on spectral embedding algorithms due to their unsupervised and non-parametric nature, and their flexibility that can encompass the second and third requirements above (as will be shown shortly). In addition, similar to PCA [63] and classical multi-dimensional scaling (MDS) [39, 41, 42], spectral methods are techniques that rely on the machinery of eigensolvers. Hence, their optimization algorithms do not suffer from local minima and can scale well with large and high dimensional data sets thanks to state-of-the-art eigensolvers.

There are two reasons for which spectral embedding methods can be considered as metric space learning algorithms:

- The fact that the eigendecomposition step characterizes the nonlinear manifold $\mathcal{M}^{p_0}$ on the hyperplane $\mathbb{R}^{p_0} \subset \mathbb{R}^p$. If $\mathcal{M}^{p_0}$ is not defined on $\mathbb{R}^{p_0}$, then learning a hypothesis directly on $\mathcal{M}^{p_0}$ is very hard since it requires computing the geodesics between points instead of distances, and requires estimating the dimensionality p_0 for $\mathcal{M}$. Since there is no *a priori* knowledge on $\mathcal{D}$, these two problems become notoriously hard to solve.

- The fact that all spectral embedding methods are all KPCA algorithms with different kernels (and normalizations) that are learned from the data, for which generalization to out-of-sample examples is accurately obtained using the Nyström formula [22]. If there is no generalization to out-of-sample examples, then the embedding is restricted to the training samples only, and can not be defined as a mapping nor as a transformation, which defies the notion of generalization for learning algorithms.

Requirement (ii) addresses the incompatibility between the GQD on one hand, and the characteristics of real world data sets and the uneven sample distribution on the other other hand. That is, the global Gaussian assumption, and the fact that the GQD is constant over the entire space, can not accommodate the characteristics of real world data sets, nor the uneven data distribution in the input space. This incompatibility, however, suggests that the global Gaussian assumption should be abandoned. Also, it suggests that the distance or the similarity measure between the vertices of the graph $G(\mathcal{D}, \mathcal{E})$ should vary in according to underlying sample distribution in the input space. Fortunately, these two suggestions meet nicely with the principal of local learning algorithms proposed by Bottou and Vapnik [47]. More specifically, theoretical analysis [118] supported by empirical results [47], suggest that a learning algorithm that adopts a local adjustment by means of local parameters, whose impact is limited to small neighbourhoods in the input space, can accommodate the characteristics of real world data sets and the uneven sample distribution, yielding a significant improvement of the overall performance of the learning algorithm. Therefore, the proposed unsupervised metric learning algorithm will rely on the principle of local learning to compute the distance or the similarity between the vertices of the graph $G(\mathcal{D}, \mathcal{E})$.

Requirement (iii) stresses that this distance or similarity measure that will vary over the input space according to the underlying sample density should be induced from a proper metric in order to guarantee that the embedding space is the metric space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$.

5.2 Skeleton of the Proposed Algorithm

The proposed algorithm for unsupervised metric learning is comprised of two steps. In the first step, as depicted in Figure 5.1, the global Gaussian assumption imposed by the GQD is relaxed and allowed to hold only in a local neighbourhood around each sample $\mathbf{x}_i \in \mathcal{D}$. Note that the local Gaussian assumption does not impose any constraints nor assumptions on the global data distribution. The local Gaussian assumption, however, associates with each $\mathbf{x}_i$ a symmetric PD matrix $\mathbf{A}_i \in \mathbb{S}_{++}^{p\times p}$, which is the covariance matrix of the local Gaussian distribution centred at $\mathbf{x}_i$. This matrix $\mathbf{A}_i$ naturally emerges as the necessary local parameter needed to leverage the uneven sample distribution effect. However, introducing the matrices $\mathbf{A}_i$ changes the structure of the data from the simple set of vectors $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subseteq \mathcal{X}$, to a new *augmented data set* $\mathcal{D}_A = \{(\mathbf{x}_i, \mathbf{A}_i)\}_{i=1}^n \subseteq \mathbb{X}$ of the 2-tuples $(\mathbf{x}_i, \mathbf{A}_i)$, where $\mathbb{X}$ is defined as the new augmented space for the data. Note that $\mathbb{X}$ carries all the information on the varying data density in the input space. Note also that the augmented space $\mathbb{X}$ can be obtained in a supervised or unsupervised manner, however here I consider the unsupervised case since the objective is unsupervised metric space learning.

In the second step, spectral embedding algorithms are used to embed the augmented data set $\mathcal{D}_A \subset \mathbb{X}$ into a low dimensional Euclidean space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$. That is, unlike the traditional setting where spectral algorithms are directly applied on $\mathcal{D}$, here spectral methods are applied on the augmented data set $\mathcal{D}_A \subset \mathbb{X}$ that carries all the information on the varying density in the input space $\mathcal{X}$.

To apply spectral methods on $\mathcal{D}_A$, a similarity or a distance measure needs to be defined over the tuples $(\mathbf{x}_i, \mathbf{A}_i)$ and $(\mathbf{x}_j, \mathbf{A}_j)$ in order to define the similarity matrix of the new graph $G(\mathcal{D}_A, \mathcal{E})$. Due to the particular structure of $\mathbb{X}$, convolution kernels [119] become the intuitive similarity functions for this new augmented space. In particular, based on the structure of the exponential kernel K_E and the Gaussian kernel K_G, I introduce the relaxed exponential kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ for the augmented space $\mathbb{X}$. On one hand, the new relaxed kernels allow all kernel based learning algorithm to be directly applied on the augmented space $\mathbb{X}$. On the other hand, they naturally induce two new distance metrics[2], the Jeffreys-Riemann metric $d_{J\mathcal{R}}$, and the Bhattacharyya-Riemann metric $d_{B\mathcal{R}}$

[2]According to the definition of a metric in Chapter 2

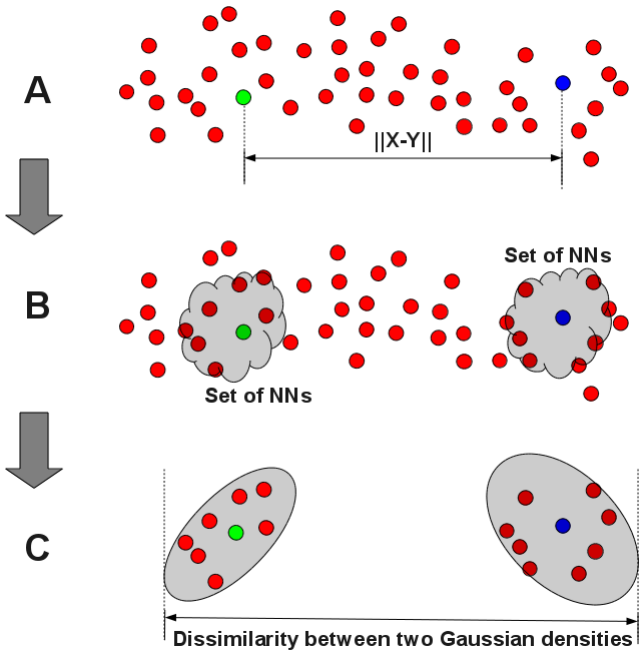


Fig. 5.1 (A) In the traditional setting, spectral methods rely on the Euclidean distance between X (green) and Y (blue), either explicitly as in classical MDS, or implicitly via the exponential kernel K_E or the Gaussian kernel K_G as in spectral clustering. (B) The local Gaussian assumption proposed here, considers the few nearest neighbours (NNs) around X and Y, and then each set of NNs is modelled as a Gaussian distribution as in (C). The spectral methods proposed here will rely on the dissimilarity (or difference) between the two Gaussian distributions instead of the Euclidean distance between X and Y.

which introduce the new metric spaces $(\mathbb{X}, d_{J\mathcal{R}})$ and $(\mathbb{X}, d_{B\mathcal{R}})$ respectively.

Based on the results of I. Schoënberg [43], I show that $\mathbb{X}$ can be embedded in the low dimensional metric space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$ using $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$. Similarly, using the results of Young & Householder [39], and Gower & Legendre [40], I show that $\mathbb{X}$ can be embedded in $(\mathbb{R}^{p_0}, \|\cdot\|_2)$ using $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$. While the first embedding is realized via Laplacian embedding algorithms [25, 26, 27, 20], the second embedding is realized via classical MDS algorithm, or Euclidean embedding [39, 41, 42], using the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$.

Although exponential kernels based on semi-metrics have been explored in different incarnations [44, 45], the issue of adhering to metric properties as proposed here has been

overlooked. We also note that the issue of adhering to metric properties is stressed in the theorems of Young & Householder [39] and Gower & Legendre [40] for Euclidean embedding. As mentioned earlier, semi-metrics violate the properties of metric spaces, and in the context of embedding they result in unstable embeddings, and can collapse all the point into one point in the embedding space. In this chapter, I will also discuss the drawbacks that result from such violations.

5.3 The Augmented Space $\mathbb{X}$

The proposal for relaxing the constraint which enforces the matrix $\mathbf{A}$ in the GQD to be globally defined for the entire input space, is equivalent to relaxing the global Gaussian assumption on the data to be only valid in a small neighbourhood around each sample $\mathbf{x}_i \in \mathcal{D}$. Note that this mild assumption on the local distribution around each $\mathbf{x}_i$ does not impose any constraints nor assumptions on the global data distribution. To realize the local Gaussian assumption, each $\mathbf{x}_i$ is associated with a symmetric matrix $\mathbf{A}_i \succ 0$ defined as:

$$\mathbf{A}_i = \frac{1}{m-1} \sum_{\mathbf{x}^j \in \mathcal{N}_i}^{m} (\mathbf{x}^j - \mathbf{x}_i)(\mathbf{x}^j - \mathbf{x}_i)^\top + \gamma \mathbf{I} \ , \tag{5.1}$$

where $\mathbf{x}^j \in \mathcal{X}$, $\mathcal{N}_i = \{\mathbf{x}^j\}_{j=1}^m$ is the set of m nearest neighbours (NNs) to $\mathbf{x}_i$, and $0 < \gamma \in \mathbb{R}$ is a regularization parameter. The regularization here is necessary to avoid the expected rank deficiencies in $\mathbf{A}_i$'s, which are due to the small number of NNs considered around $\mathbf{x}_i$, together with the high dimensionality of the data[3], and hence, this helps avoid over-fitting and outlier reliance. The definition of $\mathbf{A}_i$ in (5.1) is simply the average variance–covariance matrix between $\mathbf{x}_i$ and its m NNs. In the context of local learning, $\mathbf{A}_i$ is the local parameter that introduces the necessary local adjustment for the learning algorithm to leverage the uneven sample distribution effect.

The local Gaussian assumption, as depicted in Figure 5.1, can be seen as anchoring a Gaussian density $\mathcal{G}_i(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ at point $\mathbf{x}_i$, where its mean $\boldsymbol{\mu}_i \equiv \mathbf{x}_i$ and its covariance matrix $\boldsymbol{\Sigma}_i \equiv \mathbf{A}_i$. This assumption can be extended in the spirit of [120, 121] by letting $\mathbf{x}_i \in \mathcal{N}_i$

[3]Note that γ is unique for all $\mathbf{A}_i$'s.

which yields that:

$$\boldsymbol{\mu}_i \equiv \hat{\boldsymbol{\mu}}_i = \frac{1}{m+1} \sum_{\mathbf{x}^j \in \mathcal{N}_i} \mathbf{x}^j, \quad \text{and}$$
$$\boldsymbol{\Sigma}_i \equiv \hat{\boldsymbol{\Sigma}}_i = \frac{1}{m} \sum_{\mathbf{x}^j \in \mathcal{N}_i} (\mathbf{x}^j - \hat{\boldsymbol{\mu}}_i)(\mathbf{x}^j - \hat{\boldsymbol{\mu}}_i)^\top + \gamma \mathbf{I} \; . \tag{5.2}$$

This can be seen as a local smoothing of the data, combined with local feature extraction by means of a generative model, where the features are the parameters $\hat{\boldsymbol{\mu}}_i$ and $\hat{\boldsymbol{\Sigma}}_i$ for each $\mathbf{x}_i \in \mathcal{X}$. Note that $\mathbf{A}_i$ and $\hat{\boldsymbol{\Sigma}}_i$ are defined in an unsupervised manner. However, if auxiliary information is available in the form of labels or side information, then the proposed approach here can naturally be extended to supervised and semi-supervised learning.

Here, the set of NNs for each $\mathbf{x}_i$ is selected using the Euclidean distance. Although this is might not be suitable for very high dimensional data, in a small local neighbourhood the Euclidean distance is more reliable than it is for far away points. This is due to our initial assumption in Section 2.2 that the input space $\mathcal{X}$ is smooth and locally Euclidean. Also, selecting the very few NNs of a point $\mathbf{x}_i$ does not violate the notion of locality in high dimensional spaces. Nevertheless, it is essential to use more sophisticated techniques for finding the NNs of each $\mathbf{x}_i$ [122] in high dimensional spaces.

The result of the local Gaussian assumption introduces a new component $\mathbf{A}_i$ for each $\mathbf{x}_i \in \mathcal{X}$ which changes the structure of the input data from the set of vectors $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n$ to an augmented data set $\mathcal{D}_A = \{(\mathbf{x}_i, \mathbf{A}_i)\}_{i=1}^n \subseteq \mathbb{X}$ of 2-tuples $(\mathbf{x}_i, \mathbf{A}_i)$, where $\mathbb{X}$ is the desired augmented space. This change in the data structure, in turn, requires a new measure for the (dis)similarity between $(\mathbf{x}_i, \mathbf{A}_i)$ and $(\mathbf{x}_j, \mathbf{A}_j)$, since the Euclidean distance can only operate on the first element of the 2-tuples $(\mathbf{x}_i, \mathbf{A}_i)$ – i.e. elements in $\mathbb{R}^p$ – and not the symmetric matrix $\mathbf{A}_i \succ 0$.

Note that the augmented space $\mathbb{X}$ implicitly represents the parameters for the set of local Gaussians $\mathscr{G} = \{\mathcal{G}_i(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}_{i=1}^n$, which will be referred to as *the dual perspective* for $\mathbb{X}$. In order to avoid any future confusion in the notation, this will be the default definition for $\mathbb{X}$, where implicitly, $(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \equiv (\mathbf{x}_i, \mathbf{A}_i)$, or $(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \equiv (\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i)$. In the following sections, I

formally define suitable kernels and distance measures for the augmented space $\mathbb{X}$.

5.4 A Convolution Kernel for The Augmented Space $\mathbb{X}$

The framework of convolution kernels suggests that a possible kernel for the augmented space $\mathbb{X}$ can have the following structure [119]:

$$K_{\mathbb{X}}\{(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), (\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)\} = K_{\mu}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j) K_{\Sigma}(\boldsymbol{\Sigma}_i, \boldsymbol{\Sigma}_j),$$

where K_{μ} and K_{Σ} are symmetric PSD kernels, which yields that $K_{\mathbb{X}}$ is symmetric and PSD as well. Our approach for defining K_{μ} and K_{Σ} is based on the definition of K_E, which is an exponential function of the Euclidean distance between its two inputs. Due to the PSD and symmetry properties of metrics and semi-metrics, Axioms (1), (2), & (4), it follows that K_E is symmetric and PSD. This result is due to Theorem 4 in [43] which states that:

Theorem 5.4.1 *The most general positive function $f(x)$ which is bounded away from zero and whose positive powers $[f(x)]^{\alpha}$, $\alpha > 0$, are PSD is of the form: $f(x) = \exp\{c + \psi(x)\}$, where $\psi(x)$ is PSD and $c \in \mathbb{R}$.*

If $\psi(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) = \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|$, $\sigma > 0$, and for any real c then K_E is PSD. Note that it is only due to the axiomatic definition of metrics and semi-metrics that we are allowed to state that metrics and semi-metrics are PSD. Also, it is important to emphasize that while metrics and semi-metrics are PSD by their axiomatic definition, K_E is PSD in the sense of PSD functions [43] and Mercer kernels [123] introduced in the Appendix.

The above discussion suggests that if $d_{\mu}(\cdot,\cdot)$ and $d_{\Sigma}(\cdot,\cdot)$ are metrics (or semi-metrics) for $\{\boldsymbol{\mu}_i\}_{i=1}^n$ and $\{\boldsymbol{\Sigma}_i\}_{i=1}^n$ respectively, then K_{μ} and K_{Σ} can be defined as:

$$\begin{aligned} K_{\mu}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j) &= \exp\left\{-\tfrac{1}{\sigma} d_{\mu}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j)\right\}, \\ K_{\Sigma}(\boldsymbol{\Sigma}_i, \boldsymbol{\Sigma}_j) &= \exp\left\{-\tfrac{1}{\sigma} d_{\Sigma}(\boldsymbol{\Sigma}_i, \boldsymbol{\Sigma}_j)\right\}, \text{ and hence} \\ K_{\mathbb{X}} &= \exp\left\{-\tfrac{1}{\sigma}[d_{\mu} + d_{\Sigma}]\right\}, \end{aligned} \tag{5.3}$$

where $\sigma > 0$, and $[d_{\mu} + d_{\Sigma}]$ is a (semi-)metric for the augmented space $\mathbb{X}$. In Section 5.5.2, it will be shown that, in general, d_{μ} is the GQD between $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$, while d_{Σ} is a (semi-)metric for symmetric PD covariance matrices.

5.4.1 Isometric embedding in a Hilbert space $\mathcal{H}$

An interesting property of the exponential function in K_E and K_G is its ability to perform an isometric embedding for $(\mathbb{R}^p, \|\cdot\|_2)$ and $(\mathbb{R}^p, \|\cdot\|_2^2)$ into a Hilbert space $\mathcal{H}$. This result is due to Theorem 1 in [43] which states that:

Theorem 5.4.2 *A necessary and sufficient condition that a separable space $\mathcal{S}$ with a semi-metric distance d, be isometrically embeddable in $\mathcal{H}$, is that the function $\exp\{-\alpha d^2\}$, $\alpha > 0$, be PSD in $\mathcal{S}$. Moreover, if d is a metric, then the triangle inequality is preserved through the embedding, and the new space becomes a metric space*[4].

Therefore, if d_μ and d_Σ are (semi-)metrics for $\{\boldsymbol{\mu}_i\}_{i=1}^n$ and $\{\boldsymbol{\Sigma}_i\}_{i=1}^n$ respectively, then by Theorem 5.4.1, $K_\mu \succeq 0$ and $K_\Sigma \succeq 0$, and by Theorem 5.4.2, $(\{\boldsymbol{\mu}_i\}_{i=1}^n, d_\mu)$, $(\{\boldsymbol{\Sigma}_i\}_{i=1}^n, d_\Sigma)$ and $(\mathbb{X}, [d_\mu + d_\Sigma])$ are isometrically embeddable in $\mathcal{H}$.

Another result of Theorem 5.4.2 is that it clarifies the difference between embeddings obtained via semi-metrics, and those obtained via metrics. While the former will result in a semi-metric space, the latter will yield a metric space. This will be clarified with a real example shortly.

Theorem 2 in [43] is similar to Theorem 5.4.2; however it addresses the particular case of spaces with m real numbers, $\mathcal{S}_m$, and equipped with a norm function $\varphi(\mathbf{x})$, $\mathbf{x} \in \mathcal{S}_m$, and a distance function $\varphi(\mathbf{x} - \mathbf{x}')^{\frac{1}{2}}$. This theorem will be used instead of Theorem 5.4.2, when the Riemannian metric for symmetric PD matrices is introduced.

5.4.2 Metrics vs. semi-metrics for isometric embedding

The crucial difference between metrics and semi-metrics in the context of embedding can be explained in the light of Theorem 5.4.2. Semi-metrics are relaxed versions of metric measures in which Axiom (3), $d(a, b) = 0$ iff $a = b$, and the triangle inequality are not required to hold. A result of this relaxation is that semi-metrics can mislead an algorithm that relies on distance metrics since $d(a, b)$ can be zero for any pairs a,b and $a \neq b$. Moreover, violating the triangle inequality results in violating the relative distance between the points. As a net result, semi-metrics have tendency to collapse all the points into a single

[4] See footnote in [43, p. 525].

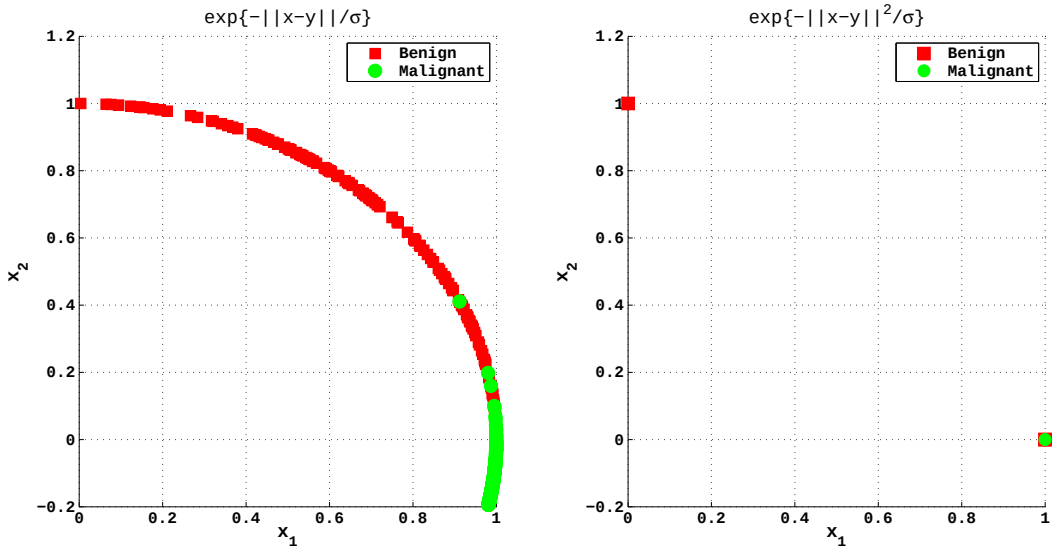


Fig. 5.2 Embedding of the Wisconsin database for breast cancer (WDBC) [1] obtained by SC using two different kernels, K_E *(left)* and K_G *(right)*. The data set has two classes, 569 samples, and 30 features.

point in the embedding space.

To see this, consider the real example depicted in Figure 5.2 which shows the embedding obtained by Laplacian eigenmaps using two different kernels to fill in the affinity matrix; K_E *(left)* which uses a metric, and K_G *(right)* which uses a semi-metric. It can be seen that the semi-metric space obtained by K_G led to catastrophic results since it collapsed all the points from the two classes into a single point at $(1, 0)$. This scenario is guaranteed not to happen in metric spaces due to Axioms (3) and (5), which explains why metrics can be favoured over semi-metrics in the context of embedding. Based on this insight, in the following section, I will define some metrics and semi-metrics that will characterize the kernel $K_{\mathbb{X}}$.

5.5 Kernels for Probability Distributions

To derive d_μ and d_Σ, our discussion begins from the dual perspective for $\mathbb{X}$, or the set $\mathscr{G} = \{\mathcal{G}_i(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}_{i=1}^n$, and the definition of K_E as an exponential function of the Euclidean distance between its input vectors. The fundamental difference here is that the elements of interest are not the vectors $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^p$, but rather the two Gaussian distributions

$\mathcal{G}_i, \mathcal{G}_j \in \mathbb{G}_p$, with $\boldsymbol{\mu}_i \neq \boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_i \neq \boldsymbol{\Sigma}_j$. It follows that the Euclidean distance describing the difference between $\mathbf{x}_i$ and $\mathbf{x}_j$ needs to be replaced with a dissimilarity measure for probability distributions, and this measure should be at least a semi-metric in order to guarantee that the resulting kernel is PSD, according to Theorem 5.4.2.

A natural measure for the dissimilarity between probability distributions is the divergence, which by definition according to Ali & Silvey [124], and Csiszar [125], is not a metric. To see this, let $\mathcal{P}$ be a family of probability distributions, and let $P_1, P_2 \in \mathcal{P}$ be defined over the same domain of events $\mathcal{E}$, then the divergence of P_2 from P_1 is:

$$\text{div}(P_1, P_2) = \mathbb{E}_{p_1}\{C(\phi)\} = \int_{\mathcal{E}} p_1(x) C(\phi(x)) dx, \tag{5.4}$$

where $\text{div}(P_1, P_2) \in [0, \infty)$, p_1, p_2 are the probability density functions of P_1 and P_2 respectively, $\phi(x) = p_1(x)/p_2(x)$ is the likelihood ratio, and C is a continuous convex function on $(0, \infty)$.

Note that by definition, $\text{div}(P_1, P_2) \geq 0$, and equality only holds when $P_1 = P_2$ [124]. This is equivalent to Axioms (1), (2) & (3) of a metric, and hence $\text{div}(P_1, P_2)$ is PSD (by the Axioms of metric definition). The divergence as defined in Equation (5.4) is not symmetric[5] since $\text{div}(P_1, P_2) \neq \text{div}(P_2, P_1)$. A possible symmetrization for the divergence can be : $\text{sdiv}(P_1, P_2) = \text{div}(P_1, P_2) + \text{div}(P_2, P_1)$, where sdiv preserves all the properties of a divergence as postulated by Ali–Silvey and Csiszar [92]. Hence, sdiv is symmetric and PSD and a possible kernel for P_1 and P_2 can be:

$$K_{\mathcal{P}}(P_1, P_2) = \exp\{-\tfrac{1}{\sigma}\text{sdiv}(P_1, P_2)\}, \ \sigma > 0. \tag{5.5}$$

Using Theorems 5.4.1 and 5.4.2, $K_{\mathcal{P}}$ is symmetric and PSD, and $(\mathcal{P}, \text{sdiv})$ is isometrically embeddable in $\mathcal{H}$. Note that $K_{\mathcal{P}}$ is in the same spirit of the exponential kernel K_E as explained above. In addition, $K_{\mathcal{P}}$ is valid for any symmetric divergence measure from the class of Ali–Silvey or f–divergence [125], and hence it is valid for any probability distribution. Note that the kernel $K_{\mathcal{P}}$ is not the only kernel for probability distributions,

[5]Depending on the choice of $C(\cdot)$ in (5.4) and its parametrization, one can derive symmetric divergence measures, see [124] for examples.

and other kernels have been proposed in [126, 127, 128].

5.5.1 The case of Gaussian densities

We now consider the particular case of Gaussian densities under some classical symmetric divergence measures such as the symmetric KL divergence, or Jeffreys divergence d_J, the Bhattacharyya divergence d_B, and the Hellinger distance d_H. For $\mathcal{G}_1, \mathcal{G}_2 \in \mathbb{G}_p$, Jeffreys divergence d_J can be expressed as:

$$d_J(\mathcal{G}_1, \mathcal{G}_2) = \tfrac{1}{2}\mathbf{u}^\top \boldsymbol{\Psi}\mathbf{u} + \tfrac{1}{2}\mathrm{tr}\{\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_2 + \boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1\} - p, \tag{5.6}$$

where $\boldsymbol{\Psi} = (\boldsymbol{\Sigma}_1^{-1} + \boldsymbol{\Sigma}_2^{-1})$, and $\mathbf{u} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$. The Bhattacharyya divergence d_B and the Hellinger distance d_H are both derived from the Bhattacharyya coefficient ρ, which is a measure of similarity between probability distributions:

$$\rho(\mathcal{G}_1, \mathcal{G}_2) = |\boldsymbol{\Gamma}|^{-\frac{1}{2}}|\boldsymbol{\Sigma}_1|^{\frac{1}{4}}|\boldsymbol{\Sigma}_2|^{\frac{1}{4}}\exp\{-\tfrac{1}{8}\mathbf{u}^\top\boldsymbol{\Gamma}^{-1}\mathbf{u}\},$$

where $\boldsymbol{\Gamma} = (\frac{1}{2}\boldsymbol{\Sigma}_1 + \frac{1}{2}\boldsymbol{\Sigma}_2)$. The Hellinger distance d_H can be derived from ρ as $\sqrt{2[1-\rho(\mathcal{G}_1, \mathcal{G}_2)]}$, while d_B is defined as $-\log[\rho(\mathcal{G}_1, \mathcal{G}_2)]$:

$$d_B(\mathcal{G}_1, \mathcal{G}_2) = \tfrac{1}{8}\mathbf{u}^\top\boldsymbol{\Gamma}^{-1}\mathbf{u} + \tfrac{1}{2}\ln\left\{|\boldsymbol{\Sigma}_1|^{-\frac{1}{2}}|\boldsymbol{\Sigma}_2|^{-\frac{1}{2}}|\boldsymbol{\Gamma}|\right\}. \tag{5.7}$$

Kullback [92] and Kailath [129] note that d_J and d_B are positive and symmetric but violate the triangle inequality, while d_H meets all metric axioms. Using the kernel definition in (5.5), it is straight forward to define the following kernels:

$$K_J(\mathcal{G}_1, \mathcal{G}_2) = \exp\{-\tfrac{1}{\sigma}d_J(\mathcal{G}_1, \mathcal{G}_2)\},\ \sigma > 0, \tag{5.8}$$

$$K_H(\mathcal{G}_1, \mathcal{G}_2) = \exp\{-\tfrac{1}{\sigma}d_H(\mathcal{G}_1, \mathcal{G}_2)\},\ \sigma > 0,\ \text{and} \tag{5.9}$$

$$K_B(\mathcal{G}_1, \mathcal{G}_2) = \exp\{-d_B(\mathcal{G}_1, \mathcal{G}_2)\} = \rho(\mathcal{G}_1, \mathcal{G}_2). \tag{5.10}$$

We note that [44] have proposed the Bhatacharyya kernel $\rho(\mathcal{G}_1, \mathcal{G}_2)$ and confirm that it is PSD through the product probability kernel (PPK). In contradiction, [45] have proposed the KL kernel $K_J(\mathcal{G}_1, \mathcal{G}_2)$ and claim, without justification, that it is not PSD. Since d_J and d_B are semi-metrics, and d_H is a metric, then using Theorems 5.4.1 and 5.4.2, K_J, K_H and

K_B are symmetric and PSD kernels, and $(\mathbb{X}, d_J)$, $(\mathbb{X}, d_B)$, and $(\mathbb{X}, d_H)$ are isometrically embeddable in $\mathcal{H}$.

5.5.2 A close look at d_J and d_B

Kullback [92, pp. 6,7] describes $d_J(\mathcal{G}_1, \mathcal{G}_2)$ as a sum of two components, one due to the difference in means weighted by the covariance matrices (the first term), and the other due to the difference in variances and covariances (the second term). Note that this explanation is also valid for $d_B(\mathcal{G}_1, \mathcal{G}_2)$. Recalling $K_{\mathbb{X}}$ from Equation (5.3), then d_μ and d_Σ can be characterized as follows.

The first term in Equations (5.6) and (5.7) is equivalent to the GQD, up to a constant and a square root – i.e. semi-metrics. If $\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \mathbf{\Sigma}$, then:

$$\left.\begin{aligned} d_J(\mathcal{G}_1, \mathcal{G}_2) &= \mathbf{u}^\top \mathbf{\Psi} \mathbf{u}, \\ d_B(\mathcal{G}_1, \mathcal{G}_2) &= \mathbf{u}^\top \mathbf{\Gamma}^{-1} \mathbf{u}. \end{aligned}\right\} d_\mu \tag{5.11}$$

The second term in Equations (5.6) and (5.7) is a discrepancy measure between two covariance matrices that is independent from $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$. If $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \boldsymbol{\mu}$ then:

$$\left.\begin{aligned} d_J(\mathcal{G}_1, \mathcal{G}_2) &= \operatorname{tr}\{\mathbf{\Sigma}_1^{-1}\mathbf{\Sigma}_2 + \mathbf{\Sigma}_2^{-1}\mathbf{\Sigma}_1\} - p, \\ d_B(\mathcal{G}_1, \mathcal{G}_2) &= \ln\left\{|\mathbf{\Gamma}||\mathbf{\Sigma}_1|^{-\frac{1}{2}}|\mathbf{\Sigma}_2|^{-\frac{1}{2}}\right\}, \end{aligned}\right\} d_\Sigma \tag{5.12}$$

which define two semi-metrics between $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$. Although the quadratic terms in Equation (5.11) can be transformed into metrics by taking the square root of each term, it is not clear how the semi-metrics in Equation (5.12) can be transformed into metrics. This is investigated in the following subsection.

5.5.3 A metric for symmetric PD matrices

The drawback of the distance measures in Equation (5.12) is that they are semi-metrics, and hence they violate the geometry of $\mathbb{S}_{++}^{p\times p}$ which is a metric space. The factorizable nature of $K_{\mathbb{X}}$, and the decomposition of $d_J(\mathcal{G}_1, \mathcal{G}_2)$ and $d_B(\mathcal{G}_1, \mathcal{G}_2)$ into two different components, where the second term is independent from $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, allows us to introduce a metric

for symmetric PD matrices that can be used instead of the semi-metrics in Equation (5.12).

A symmetric PD matrix is a geometric object and the manifold $\mathbb{S}_{++}^{p\times p}$ has a specific structure with defined geometric properties. This is the subject of Riemannian geometry, and fortunately, $\mathbb{S}_{++}^{p\times p}$ is equipped with an inner product that induces a natural distance metric, or a Riemannian metric, between all its elements. The Riemannian metric respects the geometry of $\mathbb{S}_{++}^{p\times p}$, which is unlike the semi-metrics in (5.12) that are just derived from $d_J(\mathcal{G}_1, \mathcal{G}_2)$ and $d_B(\mathcal{G}_1, \mathcal{G}_2)$, and unaware of the geometry of $\mathbb{S}_{++}^{p\times p}$.

If $d_{\mathcal{R}}$ is the Riemannian metric for $\mathbb{S}_{++}^{p\times p}$, then d_Σ in Equation (5.3) can be replaced with $d_{\mathcal{R}}$, and hence $K_{\mathbb{X}}$ can be redefined as follows:

$$\begin{aligned} K_{\mathbb{X}} &= K_\mu(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2) K_{\mathcal{R}}(\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2), && (5.13)\\ &= \exp\{-\tfrac{1}{\sigma} d_\mu\} \exp\{-\tfrac{1}{\sigma} d_{\mathcal{R}}\}, \\ &= \exp\{-\tfrac{1}{\sigma}[d_\mu + d_{\mathcal{R}}]\},\ \sigma > 0, && (5.14) \end{aligned}$$

where $d_{\mathcal{R}}$ is formally introduced in the next subsection.

5.5.4 The Riemannian metric for $\mathbb{S}_{++}^{p\times p}$

The manifold of symmetric PD matrices $\mathbb{S}_{++}^{p\times p}$ is a differentiable manifold in which each point $\mathbf{A} \in \mathbb{S}_{++}^{p\times p}$ has a tangent space $\mathcal{T}_{\mathbf{A}}(\mathbb{S}_{++}^{p\times p})$ that is endowed with an inner product, or a Riemmanian metric $\langle \cdot, \cdot \rangle_{\mathbf{A}}$, on the elements of the tangent space. The dimensionality of $\mathbb{S}_{++}^{p\times p}$ and its tangent space is $p(p+1)/2$. The inner product induces a norm on the elements of the tangent space such that for $\mathbf{B}' \in \mathcal{T}_{\mathbf{A}}(\mathbb{S}_{++}^{p\times p})$, the norm is defined as: $\|\mathbf{B}'\|_{\mathbf{A}}^2 = \langle \mathbf{B}', \mathbf{B}' \rangle_{\mathbf{A}}$. The Riemannian metric by definition of an inner product, is bilinear, symmetric, PSD, and C^∞ in $\mathbf{A}$, $\forall \mathbf{A} \in \mathbb{S}_{++}^{p\times p}$ [130]. Due to the inner product $\langle \cdot, \cdot \rangle_{\mathbf{A}}$, the tangent space in the case of Riemannian manifolds is a finite dimensional Euclidean space. Being C^∞, the inner product and the induced norm vary smoothly from point to point on the manifold.

The metric $d_{\mathcal{R}}$ was first derived by C. Rao [131], and thoroughly studied by Atkinson and Mitchel in [132] (see also their affiliated references for a comprehensive review). Independently, in geodesic sciences, Förstner and Moonen [133] derived the same metric

ten years latter. The metric was initially introduced to the computer vision community through the work of X. Pennec [134] in the context of diffusion tensor imaging (DTI) based on the work of Atkinson and Mitchel. The skeleton of the derivation below is based on the concise derivation of the metric presented in the work of Tuzel *et al.* [135].

The distance between two points $\mathbf{A}$ and $\mathbf{B}$ on $\mathbb{S}_{++}^{p\times p}$ is equal to the minimal length curve connecting the two points, or the *geodesic*. The geodesic between two points is unique, and it starts from the tangent space of $\mathbf{A}$ until it reaches point $\mathbf{B}$ on the manifold. Given point $\mathbf{B}$, the inverse mapping, or logarithmic map $\log_{\mathbf{A}}$, finds the point $\mathbf{B}' \in \mathcal{T}_{\mathbf{A}}(\mathbb{S}_{++}^{p\times p})$ that starts the geodesic connecting $\mathbf{A}$ to $\mathbf{B}$. That is, $\log_{\mathbf{A}} : \mathbb{S}_{++}^{p\times p} \longmapsto \mathcal{T}_{\mathbf{A}}(\mathbb{S}_{++}^{p\times p})$. The inverse mapping is uniquely defined across all the manifold for symmetric PD matrices. It turns that the length of the geodesic connecting $\mathbf{A}$ to $\mathbf{B}$ is the norm of $\mathbf{B}' \in \mathcal{T}_{\mathbf{A}}(\mathbb{S}_{++}^{p\times p})$ [136]:

$$d^2(\mathbf{A},\mathbf{B}) = \|\mathbf{B}'\|_{\mathbf{A}}^2 = \langle \log_{\mathbf{A}}(\mathbf{B}), \log_{\mathbf{A}}(\mathbf{B})\rangle_{\mathbf{A}}. \tag{5.15}$$

To obtain the explicit form of the metric, it remains to define the inner product $\langle\cdot,\cdot\rangle_{\mathbf{A}}$ on $\mathcal{T}(\mathbb{S}_{++}^{p\times p})$ and the inverse mapping $\log_{\mathbf{A}}(\mathbf{B})$ for $\mathbb{S}_{++}^{p\times p}$ [136]:

$$\langle \mathbf{B}'_1, \mathbf{B}'_2\rangle_{\mathbf{A}} = \mathrm{tr}\{\mathbf{A}^{-\frac{1}{2}}\mathbf{B}'_1\mathbf{A}^{-1}\mathbf{B}'_2\mathbf{A}^{-\frac{1}{2}}\}, \text{ and} \tag{5.16}$$

$$\log_{\mathbf{A}}(\mathbf{B}) = \mathbf{A}^{\frac{1}{2}}\log(\mathbf{A}^{-\frac{1}{2}}\mathbf{B}\mathbf{A}^{-\frac{1}{2}})\mathbf{A}^{\frac{1}{2}} \equiv \mathbf{B}'_3, \tag{5.17}$$

where $\mathbf{B}'_1,\ \mathbf{B}'_2,\ \mathbf{B}'_3 \in \mathcal{T}_{\mathbf{A}}(\mathbb{S}_{++}^{p\times p})$. Note that the manifold logarithmic operator $\log_{\mathbf{A}}$ which is manifold and point specific, should not be confused with the ordinary matrix logarithmic operator. By plugging (5.17) into (5.15) and expressing the inner product as in (5.16) we obtain the Riemannian metric for $\mathbb{S}_{++}^{p\times p}$:

$$d_{\mathcal{R}}(\mathbf{A},\mathbf{B}) = \mathrm{tr}\{\log^2 \mathbf{\Lambda}(\mathbf{A},\mathbf{B})\}^{\frac{1}{2}}, \tag{5.18}$$

where $\mathbf{\Lambda}(\mathbf{A},\mathbf{B}) = \mathrm{diag}(\lambda_1,\ldots,\lambda_p)$ is the generalized eigenvalue matrix for the generalized eigenvalue problem: $\mathbf{A}\mathbf{\Phi} = \mathbf{\Lambda}\mathbf{B}\mathbf{\Phi}$, and $\mathbf{\Phi}$ is the column matrix of its generalized eigenvectors. Note that $d_{\mathcal{R}}$ is invariant to inversion and to affine transformations of the coordinate system [133]. Since $d_{\mathcal{R}}$ is induced by a norm on $\mathcal{T}(\mathbb{S}_{++}^{p\times p})$, then using Theorems 5.4.1 and 5.4.2, $K_{\mathcal{R}}$ is PSD, and $(\mathcal{T}_{\mathbf{A}}(\mathbb{S}_{++}^{p\times p}), d_{\mathcal{R}})$ is isometrically embeddable in $\mathcal{H}$, for all $\mathbf{A} \in \mathbb{S}_{++}^{p\times p}$.

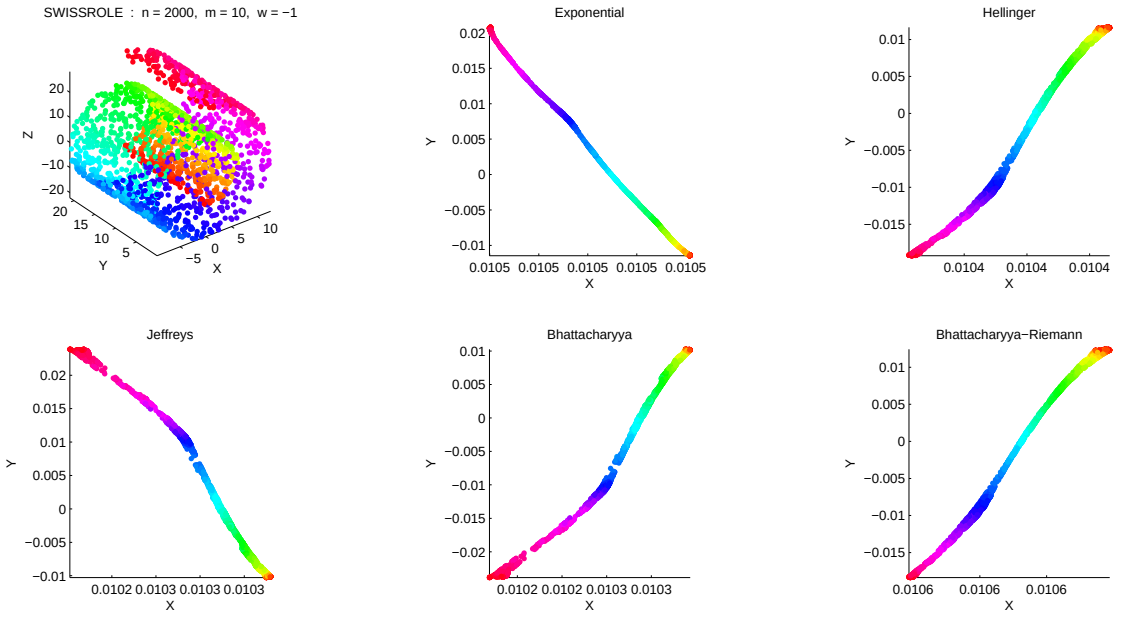


Fig. 5.3 Embeddings obtained by Laplacian embedding or spectral clustering using K_E, K_H, K_J, K_B, and $K_{B\mathcal{R}}$ on the `swiss role` data set. Note the discontinuities in the embedding obtained by K_J and K_B.

5.6 Relaxed Kernels for The Augmented Space $\mathbb{X}$

Besides Jeffrey's kernel K_J, Hellinger's kernel K_H, and Bhattacharyya's kernel K_B in Equations (5.8), (5.9) and (5.10) respectively, we define two new kernels for the augmented space $\mathbb{X}$ based on the metric $d_{\mathcal{R}}$:

$$K_{J\mathcal{R}}(\mathcal{G}_1, \mathcal{G}_2) = \exp\{-\tfrac{1}{\sigma} d_{J\mathcal{R}}(\mathcal{G}_1, \mathcal{G}_2)\}, \quad \text{and} \tag{5.19}$$

$$K_{B\mathcal{R}}(\mathcal{G}_1, \mathcal{G}_2) = \exp\{-\tfrac{1}{\sigma} d_{B\mathcal{R}}(\mathcal{G}_1, \mathcal{G}_2)\}, \quad \text{where} \tag{5.20}$$

$$d_{J\mathcal{R}}(\mathcal{G}_1, \mathcal{G}_2) = (\mathbf{u}^\top \boldsymbol{\Psi} \mathbf{u})^{\frac{1}{2}} + d_{\mathcal{R}}(\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2),$$

$$d_{B\mathcal{R}}(\mathcal{G}_1, \mathcal{G}_2) = (\mathbf{u}^\top \boldsymbol{\Gamma}^{-1} \mathbf{u})^{\frac{1}{2}} + d_{\mathcal{R}}(\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2),$$

$$\boldsymbol{\Psi} \succ 0, \quad \boldsymbol{\Gamma}^{-1} \succ 0, \quad \text{and} \quad \sigma > 0.$$

The relaxed kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ introduce two new metrics, or corrected divergence measures, for the augmented space $\mathbb{X}$; the Jeffreys-Riemann metric $d_{J\mathcal{R}}$, and the Bhattacharyya-Riemann metric $d_{B\mathcal{R}}$. The positive definiteness of $\boldsymbol{\Psi}$ and $\boldsymbol{\Gamma}^{-1}$, and the square root on the quadratic terms of $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$, assure that the quadratic terms are metrics. If

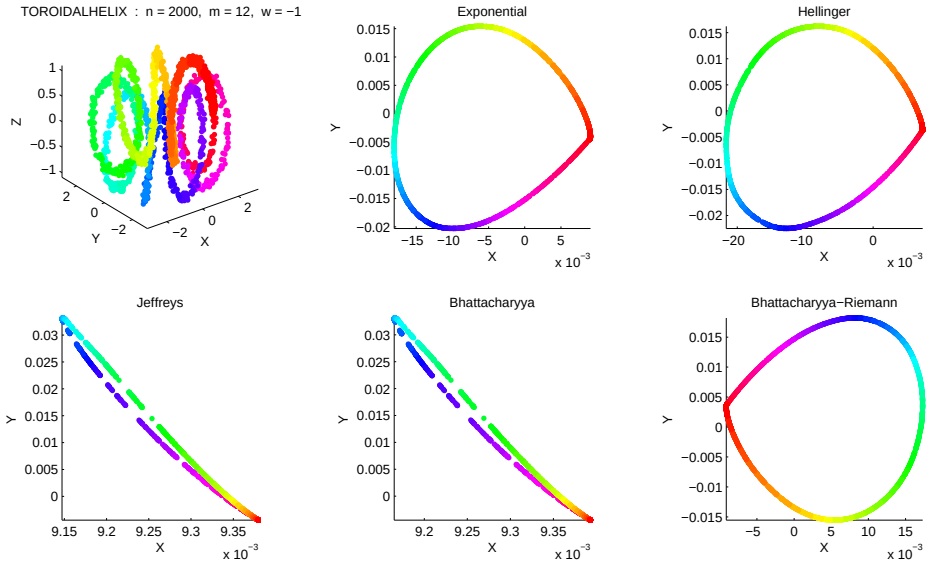


Fig. 5.4 Embeddings obtained by Laplacian embedding or spectral clustering using K_E, K_H, K_J, K_B, and $K_{B\mathcal{R}}$ on the `toroidal hellix` data set. Note how K_J and K_B yield different embeddings with discontinuities. Note also the tendency of d_J and d_B to overlap points over each other.

$\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \boldsymbol{\mu}$, then $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ will yield the Riemannian metric $d_{\mathcal{R}}$, and hence, $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ will be equal to $K_{\mathcal{R}}$. If $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$, then $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ will yield the GQD. If $\boldsymbol{\Sigma} = \mathbf{I}$, the GQD will be equal to the Euclidean distance, and $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ will yield the original exponential kernel K_E.

Similar to K_E and K_G, the relaxed kernels K_J, K_H, K_B, $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ rely on the distance between the 2-tuples $(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$. Moreover, they all provide an isometric embedding for the space $\mathbb{X}$, and the difference between these embeddings is due the metric or semi-metric defining each kernel. While d_J and d_B are semi-metrics, d_H, $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ are metrics. Since Axioms (3) & (5) do not hold for semi-metrics, it follows that d_J and d_B will not preserve the relative geometry between the elements in $\mathbb{R}^p$, and that between the elements in $\mathbb{S}_{++}^{p\times p}$. Although d_H is a metric, it relies on a semi-metric for covariances matrices, which is not the case for $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$.

This crucial difference between d_J and d_B on one hand, and d_H, $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ on the other hand, together with the augmented space $\mathbb{X}$ is illustrated in the embeddings obtained

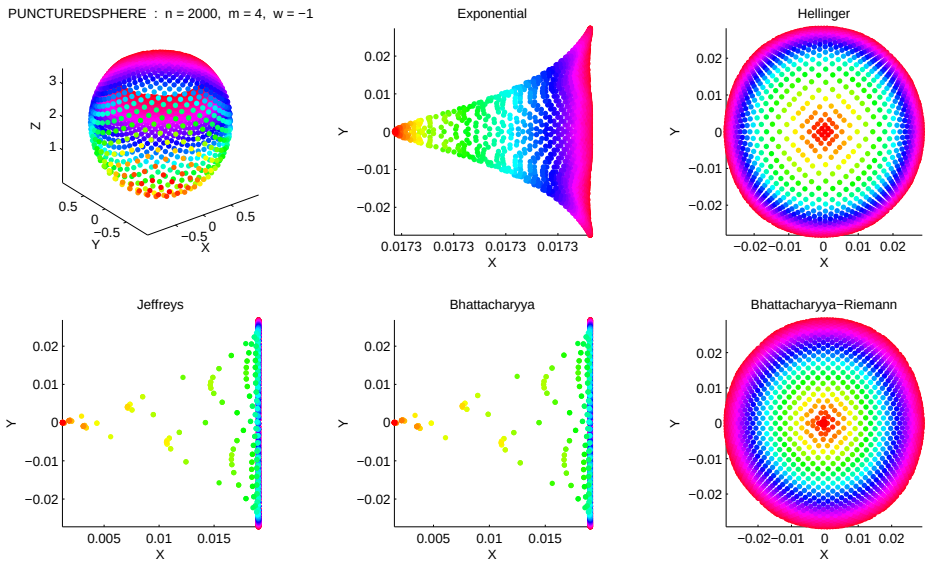


Fig. 5.5 Embeddings obtained by Laplacian embedding or spectral clustering using K_E, K_H, K_J, K_B, and $K_{B\mathcal{R}}$ on the `punctured sphere` data set. Note how for K_H and $K_{B\mathcal{R}}$ the local neighbourhood modelling together with metric properties yield the expected embedding of the data set, which is a disc. Note also how K_J and K_B yield an embedding which roughly has the same shape as that of K_E, while trying to collapse all the points along a vertical line.

on three toy data sets with known embeddings. These figures, better seen in colour (or on a coloured display), aim to give a qualitative measure on the stability of the embedding. Figure 5.3, shows five different embeddings for the `swiss role` data set. The expected embedding is a continuous rectangular sheet or a continuous straight line in two dimensions with the same colour ordering of the original three dimensional data set. It can be seen that all the kernels yield almost the same embedding except for the discontinuities encountered by K_J and K_B.

Figure 5.4, shows five different embeddings for the `toroidal helix` data set. The expected embedding for this data set is continuous circle in two dimensions with the same colour ordering of the original data set. Here the embeddings via the augmented space with K_J or K_B do not yield such results, and in fact, the embeddings try to collapse all the points onto a line. Note also the discontinuities in these embeddings. The reader should note that in these two cases, `swiss-role` and `toroidal helix`, the exponential kernel K_E

gave the expected embeddings, and the same follows for the embeddings via the augmented space with K_H and $K_{B\mathcal{R}}$.

Figure 5.5, shows five different embeddings for the `punctured sphere` data set. Here the expected embedding is a disk in two dimensions with the same color ordering of the original sphere in three dimensions. The Laplacian embedding via K_E failed to deliver the expected embedding which shows the incapability of K_E to preserve the original structure in the data. The same follows for the embeddings obtained via the augmented space with K_J and K_B, with the additional side effect of trying to collapse all the points onto a line. Here the power of the augmented space, together with K_H and $K_{B\mathcal{R}}$ is more obvious. That is, the augmented space with the relaxed kernels that preserve all the metric properties was able to deliver the expected embedding in this case and in all the previous cases as well. This is unlike the embeddings obtained by the augmented space with K_J and K_B that lacked the triangle inequality property in d_J and d_B respectively.

The relaxed kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ and the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ are now ready to embed $\mathcal{D}_A \subset \mathbb{X}$ in the metric space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$. In the following, I will present two different types of spectral algorithms to obtain such an embedding. The first algorithm is Laplacian eigenmaps, and the second is classical MDS or Euclidean embedding. In order to simplify the notation cumbersomeness, only $K_{B\mathcal{R}}$ and $d_{B\mathcal{R}}$ will be used in the discussions and examples presented in the following sections.

5.7 Laplacian Embedding for $\mathbb{X}$

Given the augmented data set $\mathcal{D}_A = \{(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}_{i=1}^n \equiv \{\mathcal{G}_i\}_{i=1}^n$, the relaxed kernel $K_{B\mathcal{R}}$, and the dimensionality p_0 of the embedding space, an embedding for $(\mathcal{D}_A \subset \mathbb{X}, d_{B\mathcal{R}})$ into $(\mathbb{R}^{p_0}, \|\cdot\|_2)$ can be obtained using Laplacian eigenmaps as follows:

1. Construct the data graph $G(\mathcal{D}_A, \mathcal{E})$, where G can be a fully connected graph, an ϵ-ball graph, or a k-NN graph, and $\mathcal{E}$ is the set of edges for G.

2. Construct the affinity (or similarity) matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ such that:

$$[k_{ij}] = \begin{cases} K_{B\mathcal{R}}(\mathcal{G}_i, \mathcal{G}_j) & \text{if } e_{ij} = 1, \\ 0 & \text{else,} \end{cases}$$

   where $e_{ij} \in \mathcal{E}$.

3. Compute the diagonal matrix $\mathbf{D}$, where $[d_{ii}] = \sum_{j=1}^{n} \mathbf{K}_{ij}$, and $1 \leq i \leq n$.

4. Compute the normalized Laplacian matrix $\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{K}\mathbf{D}^{-\frac{1}{2}}$.

5. Perform an eigendecomposition for the Laplacian matrix $\mathbf{L}$ to $\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\top}$, and select the first p_0 eigenvectors $[\mathbf{u}_1, \ldots, \mathbf{u}_{p_0}]$ and their corresponding eigenvalues $(\lambda_1, \ldots, \lambda_{p_0})$, where $\lambda_1 > \lambda_2 > \cdots > \lambda_{p_0}$.

6. Form the matrix $\mathbf{Y} = \mathbf{U}_{1:p_0} = [\mathbf{u}_1 \ldots \mathbf{u}_{p_0}]$, where $\mathbf{Y} \in \mathbb{R}^{n \times p_0}$.

Now each row $\mathbf{y}_i$ of $\mathbf{Y}$ is the embedding of the 2-tuple $(\boldsymbol{\mu}_i, \mathbf{\Sigma}_i) \equiv \mathcal{G}_i$ which represents the original data point $\mathbf{x}_i$ with its neighbourhood $\mathcal{N}_i$. Note that in the context of clustering, p_0 is usually equal to the number of clusters in the data. When this is not the case, and the number of clusters is not known, p_0 can be considered as the intrinsic dimensionality of the data which can be estimated by an algorithm such as [137], where usually $p_0 \ll \min(p, n)$. In general, selecting the dimensionality p_0 or the number of clusters is a fundamental question of model selection that is not addressed here. The eigenvalues $(\lambda_1, \ldots, \lambda_{p_0})$ computed above will play a crucial role in generalizing the embedding obtained here to out-of-sample examples as described in Section 5.9.

It can be seen that the algorithm above does not rely on labels nor side-information, however when such additional information is available, they can be easily incorporated in the algorithm. Note that the spectral clustering algorithm in [27] has the same steps above, followed by normalizing the rows of $\mathbf{Y}$ to have unit length and finally apply k-Means clustering on the normalized vectors.

5.7.1 Discussion

The steps of the above algorithm are originally due to the spectral graph theory described in the work of F. Chung [69]. This algorithm first appeared in the work of Shi & Malik

[25] on image segmentation, and further analysed by Y. Weiss in [26]. Independently, in the context of kernel methods, Cristianini and Shawe-Taylor [138] introduced the Laplacian to obtain a clustering from the gram (or similarity) matrix of the data. This is was the first link between kernel methods and spectral graph theory methods for unsupervised learning. In NIPS 2002, the normalized Laplacian was introduced in a very different way as a manifold learning algorithm under the title *Laplacian eigenmaps* by Belking & Nyiogi [20]. Surprisingly, one year latter (NIPS 2003), He & Nyiogi reintroduced the same steps of Laplacian eigenmaps with different motivations under the title of *locality preserving projection* (LPP) [139]. This reinvention of the algorithm under different motivations was also noted in [22]. Ng *et al.* [27], based on careful theoretical analysis and justifications, added two further steps to the above algorithm; (1) the normalization step for the rows of $\mathbf{Y}$, and (2) the k-Means clustering step in the embedding space. Ng *et al.* also noted the intriguing link between kernel methods and spectral graph theory in the work of Cristianini and Shawe-Taylor.

To the best of my knowledge, the research work presented here has the following contributions over the work in [27, 20]:

1. This research work is the first to consider the metric properties of the embedding through the exponential function based on the results of I. Schoenberg [43].

2. This research work is the first to consider the limitations of the GQD (including the Euclidean distance) and propose that the initial distance or similarity measure on the data graph for spectral methods should vary according to the sample density in the input space. Further, this varying distance measure should preserve all the metric properties to guarantee a proper embedding in a low dimensional Euclidean space.

3. These two previous issues lead to the introduction of the augmented space $\mathbb{X}$, studying the metric properties of divergence measures, and in particular the case of Gaussian densities, and finally the introduction of the corrected divergence measures $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ [29], and the relaxed kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ [30].

In the following section, I show that the augmented metric space $(\mathbb{X}, d_{B\mathcal{R}})$ can also be embedded in $(\mathbb{R}^{p_0}, \|\cdot\|_2)$ using the classical MDS or Euclidean embedding algorithm, thanks to the metric properties of $d_{B\mathcal{R}}$.

5.8 Euclidean Embedding for $\mathbb{X}$

Using the metric $d_{B\mathcal{R}}$, it is straightforward to define an Euclidean embedding for the metric spaces $(\mathbb{X}, d_{B\mathcal{R}})$, using the Theorems of Young & Househlder [39] and Gower & Legendre [40]. Before proceeding to the embedding steps, it is important to define metric matrices for a general set of points in $\mathbb{R}^p$, their PSD properties, and their low dimensional Euclidean embedding.

For a set of n unknown points, assume the matrix $[d_{ij}] = \mathbf{D} \in \mathbb{R}^{n \times n}$ is given with all the mutual distances (or dissimilarities) between the n points, such that $d_{ij} = d_{ji}$, $d_{ii} = 0$, and $d_{ij} \geq 0$, $\forall\ i, j$. Note here that the points and the distance function are not specified. Gower & Legendre [40] define a metric matrix as follows:

Definition $\mathbf{D}$ is said to be a **distance metric matrix** (DMM) if the *metric (triangle) inequality* $d_{ij} + d_{ik} \geq d_{jk}$ holds for all triples (i, j, k).

Note that the metric d of any metric space $(\mathcal{M}, d)$, where $\mathcal{M}$ is any non-empty set of objects, can define a DMM, while semi-metrics can not define DMMs since Axiom (3) and the triangle inequality of metrics are not required to hold. Euclidean distance matrices (EDMs), for example, share the same definition above since the Euclidean distance is a metric. However, an EDM has a more specific definition, which is Definition (2) in [40]:

Definition $\mathbf{D}$ is said to be an **Euclidean distance matrix** (EDM) if the n points can be embedded in an Euclidean space as $\{\mathbf{p}_i\}_{i=1}^n$, such that the Euclidean distance between $\mathbf{p}_i$ and $\mathbf{p}_j$ is d_{ij}, $\forall\ i, j$.

The definition, alone, does not state how to formally validate whether $\mathbf{D}$ is an EDM or not. The necessary and sufficient condition for $\mathbf{D}$ to be an EDM is in Theorem III in [39], and Theorem 4 in [40] which is stated after the following definitions.

Let $\mathbf{D}$ be defined as above, and let $[-\frac{1}{2}d_{ij}^2] = \mathbf{S} \in \mathbb{R}^{n \times n}$, $\forall\ i, j$. Define the centering matrix $\mathbf{H} \equiv \mathbf{H}_{n \times n} = \mathbf{I}_{n \times n} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, where $\mathbf{I}$ is the identity matrix, and $\mathbf{1}$ is a vector of ones.

Theorem 5.8.1 $\mathbf{D}$ *is an EDM if and only if the matrix* $\mathbf{K} = \mathbf{HSH}$ *is PSD.*

Young & Householder [39] further discuss the reverse direction of the theorem. That is, if $\mathbf{K}$ is symmetric and PSD, then there exist a set of n real points in an Euclidean space with mutual distance $d_{ij} = d_{ji}$, and these points can be obtained as follows. Since $\mathbf{K}$ is symmetric and PSD, by eigendecomposition of $\mathbf{K}$ to $\mathbf{VLV}^\top$, where the columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{K}$, $\mathbf{L} = \text{diag}\{\ell_1, \ldots, \ell_{p_0}, 0, \ldots, 0\}$ is its eigenvalue matrix, and $\ell_1 > \ell_2 > \cdots > \ell_{p_0}$, then the coordinates of these n points are the rows of the matrix $\mathbf{Y} = \mathbf{VL}^{\frac{1}{2}}$, where $\mathbf{Y} \in \mathbb{R}^{n \times p_0}$.

The key observation here is that from Theorem 5.8.1 and the previous definitions, it follows directly that if $\mathbf{K}$ is symmetric and PSD, then $\mathbf{D}$ is also a DMM. Hence, given only a DMM, and not necessarily an EDM, one can easily obtain its representing set of n real points in an Euclidean space $\mathbb{R}^{p_0}$, with $p_0 \ll n$. Recalling the definition of a metric space $(\mathcal{M}, d)$, a DMM can represent the mutual distances between all the elements of the non-empty set $\mathcal{M}$ since d is a metric by definition. Therefore, for any metric space $(\mathcal{M}, d)$ it is possible to obtain an Euclidean embedding for this set as long as d is a metric. Note that matrix $\mathbf{K}$ is in fact a centralized dot product matrix, or a centralized gram matrix, which describes the similarity between the original input points. If d is a semi-metric, the similarity matrix $\mathbf{K}$ is not guaranteed to be PSD, and hence the resulting low dimensional subspace will be a semi-metric space where metric properties and relatives distances between points can be violated.

In a similar fashion, and using Theorem 5.8.1 with the previous definitions, the metric space $(\mathcal{D}_A \subset \mathbb{X}, d_{B\mathcal{R}})$ can be embedded in the low dimensional Euclidean space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$ using the following procedure.

1. Define the DMM $\mathbf{D} \in \mathbb{R}^{n \times n}$ such that $[d_{ij}] = d_{B\mathcal{R}}(\mathcal{G}_i, \mathcal{G}_j)$, $\forall\ i, j$.

2. Compute the similarity matrix $\mathbf{K} = \mathbf{HSH}$, where $\mathbf{S} = [-\frac{1}{2}d_{ij}^2]$, and $\mathbf{H}$ is the centering matrix as defined earlier. Since $d_{B\mathcal{R}}$ is a metric, then according to Theorem 5.8.1 $\mathbf{K}$ is PSD.

3. Perform an eigendecomposition for $\mathbf{K}$ to $\mathbf{VLV}^\top$, and construct the matrix $\mathbf{Y} = \mathbf{V}_{1:p_0}\mathbf{L}_{1:p_0}^{\frac{1}{2}}$, where $\mathbf{Y} \in \mathbb{R}^{n \times p_0}$.

Again, each row $\mathbf{y}_i$ of $\mathbf{Y}$ is the embedding of the 2-tuple $(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \equiv \mathcal{G}_i$ which represents

the original data point $\mathbf{x}_i$ with its neighbourhood $\mathcal{N}_i$. Similar to the Laplacian embedding, this procedure is totally unsupervised and does not require any labels nor side-information. Finally, it is worth noticing that any hypothesis learning algorithm can be directly applied on the set $\{\mathbf{y}_i\}_{i=1}^n$ instead of the original data set $\{\mathbf{x}_i\}_{i=1}^n$.

5.9 Generalization to Out-of-Sample Examples

The procedures above for Laplacian embedding and Euclidean embedding describe the training phase for embedding $(\mathbb{X}, d_{B\mathcal{R}})$ into a low dimensional Euclidean space $\mathbb{R}^{p_0}$. For Laplacian embedding, the parameters learned during the training phase are the matrices $\mathbf{U}_{1:p_0} = [\mathbf{u}_1 \dots \mathbf{u}_{p_0}]$ and $\mathbf{\Lambda}_{1:p_0} = \text{diag}(\lambda_1, \dots, \lambda_{p_0})$. For Euclidean embedding, these are the matrices $\mathbf{V}_{1:p_0} = [\mathbf{v}_1 \dots \mathbf{v}_{p_0}]$ and $\mathbf{L}_{1:p_0} = \text{diag}(\ell_1, \dots, \ell_{p_0})$.

Suppose we are given m new 2-tuples $\mathcal{D}_A^* = \{(\boldsymbol{\mu}_1^*, \mathbf{\Sigma}_1^*), \dots, (\boldsymbol{\mu}_m^*, \mathbf{\Sigma}_m^*)\} \equiv \{\mathcal{G}_j^*\}_{j=1}^m$ that were not included during the training phase, and it is desired to compute their low dimensional embeddings in $\mathbb{R}^{p_0}$ using the parameters estimated above. Note that the neighbourhoods $\mathcal{N}_j^*$, $1 \leq j \leq m$ are constructed from the original training set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n$. This is the problem of generalizing Laplacian embedding and Euclidean embedding to out-of-sample examples which was thoroughly studied in [22] for most spectral learning algorithms such as classical MDS, LLE, Isomap, Laplacian eigenmaps, and spectral clustering methods. Since all these algorithms share a spectral embedding step, it was shown that all these methods are learning eigenfunctions of similarity between input points, and for which the Nyström formula [140] provides a method for generalizing these algorithms to out-of-sample examples.

To obtain the the Nyström formula for Laplacian and Euclidean embeddings, a data dependent symmetric PSD kernel $K_n(\mathbf{x}_i, \mathbf{x}_j)$ needs to be defined over the training data. From $K_n(\mathbf{x}_i, \mathbf{x}_j)$, the Nyström formula can be defined in a closed matrix form. Note that $K_n(\mathbf{x}_i, \mathbf{x}_j)$ may not only depend on $\mathbf{x}_i, \mathbf{x}_j$, but on all the n samples in the training set, and hence the notation K_n. In general, it can be shown that the similarity matrix $\mathbf{K}$ for all spectral methods, can be constructed from such data dependent kernel $K_n(\mathbf{x}_i, \mathbf{x}_j)$.

Let v_{ik} be the i-th coordinate of the k-th eigenvector of $\mathbf{K}$, where the eigenvector $\mathbf{v}_k$ is

associated with eigenvalue ℓ_k. Then the Nyström formula for the k-th component of the out-of-sample point $\mathbf{x}^* = [x_1^* \ \dots \ x_k^* \ \dots \ x_p^*]^\top$ can be written as:

$$y_k^* \equiv f_{k,n}(x_k^*) = \frac{\sqrt{n}}{\ell_k} \sum_{i=1}^{n} v_{ik} K_n(\mathbf{x}^*, \mathbf{x}_i) \tag{5.21}$$

where $f_{k,n}$ is the k-th Nyström estimator with n samples. That is, the final embedding for the out-of-sample point $\mathbf{x}^*$ is the vector $\mathbf{y}^* = [y_1^* \ \dots \ y_k^* \ \dots \ y_{p_0}^*]^\top$, where p_0 is the dimensionality of the embedding space. In the following, I will use two data dependent kernels, one for Laplacian embedding and the other for Euclidean embedding, for which generalization to out-of-sample examples is straight forward using the Nyström formula.

5.9.1 Generalization of Laplacian eigenmaps

The data dependent kernel for Laplacian eigenmaps can be defined as follows [22]:

$$K_n^{Laplacian}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{n} \frac{\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)}{[\mathbb{E}_{\mathbf{x}_j}\{\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)\}]^{\frac{1}{2}} \ [\mathbb{E}_{\mathbf{x}_i}\{\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)\}]^{\frac{1}{2}}}, \tag{5.22}$$

where:

$$\begin{aligned} \mathbb{E}_{\mathbf{x}_j}\{\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)\} &= \tfrac{1}{m} \textstyle\sum_{j=1}^{m} \tilde{K}(\mathbf{x}_i, \mathbf{x}_j), \\ \mathbb{E}_{\mathbf{x}_i}\{\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)\} &= \tfrac{1}{n} \textstyle\sum_{i=1}^{n} \tilde{K}(\mathbf{x}_i, \mathbf{x}_j), \end{aligned}$$

and $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)$ is any off-the-shelf kernel such as the linear dot product kernel, the Gaussian kernel, etc. However in this particular case, $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = K_{B\mathcal{R}}(\mathcal{G}_i, \mathcal{G}_j)$. Note that all the expectations in Equation (5.22) are defined over the training set. The first appearance of this normalized and centralized data dependent kernel was in the context of kernel target alignment [138], which was the first link between kernel methods and clustering by means of the Laplacian operator.

Using the general form of the Nyström estimator, the kernel $K_n^{Laplacian}$, the training data set $\mathcal{D}_A$, the estimated parameters $\mathbf{U}_{1:p_0} = [\mathbf{u}_1 \dots \mathbf{u}_{p_0}]$ and $\mathbf{\Lambda}_{1:p_0} = \text{diag}(\lambda_1, \dots, \lambda_{p_0})$, an embedding for the test set $\mathcal{D}_A^*$ in $\mathbb{R}^{p_0}$ can be obtained as follows:

1. Define the similarity (or gram) matrix $\tilde{\mathbf{K}} \in \mathbf{R}^{n \times m}$, where $[\tilde{k}_{ij}] = K_{B\mathcal{R}}(\mathcal{G}_i, \mathcal{G}_j^*)$, where

$1 \leq i \leq n$, and $1 \leq j \leq m$.

2. Compute the diagonal matrix $\mathbf{D}_{\text{left}} \in \mathbb{R}^{n \times n}$ such that:

$$\mathbf{D}_{\text{left}} = \text{diag}\left(\sqrt{\tfrac{1}{m}\sum_{j=1}^{m}\tilde{\mathbf{K}}_{(1,j)}} \ \cdots \ \sqrt{\tfrac{1}{m}\sum_{j=1}^{m}\tilde{\mathbf{K}}_{(n,j)}}\right).$$

3. Compute the diagonal matrix $\mathbf{D}_{\text{right}} \in \mathbb{R}^{m \times m}$ such that:

$$\mathbf{D}_{\text{right}} = \text{diag}\left(\sqrt{\tfrac{1}{n}\sum_{i=1}^{n}\tilde{\mathbf{K}}_{(i,1)}} \ \cdots \ \sqrt{\tfrac{1}{n}\sum_{i=1}^{n}\tilde{\mathbf{K}}_{(i,m)}}\right).$$

4. Compute the normalized similarity (gram) matrix $\mathbf{K}^*$:

$$\mathbf{K}^* = \tfrac{1}{n}\ \mathbf{1}_n\ \mathbf{1}_m^\top\ \mathbf{D}_{\text{left}}^{-1}\ \tilde{\mathbf{K}}\ \mathbf{D}_{\text{right}}^{-1}.$$

5. Apply the Nyström formula on $\mathbf{K}^*$ to obtain the embedding for the out-of-sample 2-tuples $\mathcal{D}_A^* = \{(\boldsymbol{\mu}_1^*, \boldsymbol{\Sigma}_1^*), \ldots, (\boldsymbol{\mu}_m^*, \boldsymbol{\Sigma}_m^*)\}$:

$$\mathbf{Y}^* = \sqrt{n}\ \mathbf{K}^{*\top}\ \mathbf{U}_{1:p_0}\ \boldsymbol{\Lambda}_{1:p_0}^{-1}, \tag{5.23}$$

where $\mathbf{Y}^* \in \mathbb{R}^{m \times p_0}$.

Now each row $\mathbf{y}_j^*$ of $\mathbf{Y}^*$ is the embedding of the out-of-sample 2-tuple $(\boldsymbol{\mu}_j^*, \boldsymbol{\Sigma}_j^*) \equiv \mathcal{G}_j^*$ which represents the original data point $\mathbf{x}_j^*$ with its neighbourhood $\mathcal{N}_j^*$.

5.9.2 Generalization of Euclidean embedding

Similar to Laplacian embedding, classical MDS or Euclidean embedding can be defined in terms of a data dependent kernel, from which, the generalization to out-of-sample examples is straight forward via the Nyström formula. The data dependent kernel for classical MDS can be defined as follows [22]:

$$K_n^{cMDS}(\mathbf{x}_i, \mathbf{x}_j) = -\frac{1}{2}\left(d^2(\mathbf{x}_i, \mathbf{x}_j) - \mathbb{E}_{\mathbf{x}_i}\{d^2(\mathbf{x}_i, \mathbf{x}_j)\} - \mathbb{E}_{\mathbf{x}_j}\{d^2(\mathbf{x}_i, \mathbf{x}_j)\} + \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j}\{d^2(\mathbf{x}_i, \mathbf{x}_j)\}\right), \tag{5.24}$$

where:

$$\begin{aligned}\mathbb{E}_{\mathbf{x}_i}\{d^2(\mathbf{x}_i,\mathbf{x}_j)\} &= \tfrac{1}{n}\sum_{i=1}^{n} d^2(\mathbf{x}_i,\mathbf{x}_j),\\ \mathbb{E}_{\mathbf{x}_j}\{d^2(\mathbf{x}_i,\mathbf{x}_j)\} &= \tfrac{1}{m}\sum_{j=1}^{m} d^2(\mathbf{x}_i,\mathbf{x}_j),\\ \mathbb{E}_{\mathbf{x}_i,\mathbf{x}_j}\{d^2(\mathbf{x}_i,\mathbf{x}_j)\} &= \tfrac{1}{nm}\sum_{i,j=1}^{n,m} d^2(\mathbf{x}_i,\mathbf{x}_j),\end{aligned}$$

and d is a distance metric between the elements $\mathbf{x}_i$ and $\mathbf{x}_j$. In this particular context, $d(\mathbf{x}_i,\mathbf{x}_j) = d_{B\mathcal{R}}(\mathcal{G}_i,\mathcal{G}_j)$. Again, all the expectations in Equation (5.24) are taken over the training set.

Using the general form of the Nyström estimator, the kernel K_n^{cMDS}, the training data set $\mathcal{D}_A$, the estimated parameters $\mathbf{V}_{1:p_0} = [\mathbf{v}_1 \dots \mathbf{v}_{p_0}]$ and $\mathbf{L}_{1:p_0} = \text{diag}(\ell_1,\dots,\ell_{p_0})$, an embedding for the test set $\mathcal{D}_A^*$ in $\mathbb{R}^{p_0}$ can be obtained as follows:

1. Define the DMM $\mathbf{D}^* \in \mathbb{R}^{m\times n}$ such that $[d_{ji}^*] = d_{B\mathcal{R}}(\mathcal{G}_j^*,\mathcal{G}_i)$, for $1 \le j \le m$, and $1 \le i \le n$.

2. Compute the similarity matrix $\mathbf{K}^*$ such that:

$$\mathbf{K}^* = -\tfrac{1}{2}\left[\mathbf{D}^*\ \mathbf{H}_{n\times n} - \tfrac{1}{n}\ \mathbf{1}_m\mathbf{1}_n^\top\ \mathbf{D}\ \mathbf{H}_{n\times n}\right], \tag{5.25}$$

   where $\mathbf{H}$ is the centering matrix defined earlier, and $\mathbf{D}$ is the DMM for the training set defined in step (1) in the training phase of Euclidean embedding.

3. Apply the Nyström formula on $\mathbf{K}^*$ to obtain the embedding for the out-of-sample 2-tuples $\mathcal{D}_A^* = \{(\boldsymbol{\mu}_1^*,\boldsymbol{\Sigma}_1^*),\dots,(\boldsymbol{\mu}_m^*,\boldsymbol{\Sigma}_m^*)\}$:

$$\mathbf{Y}^* = \mathbf{K}^*\ \mathbf{V}_{1:p_0}\ \mathbf{L}_{1:p_0}^{-\frac{1}{2}}, \tag{5.26}$$

   where $\mathbf{Y}^* \in \mathbb{R}^{m\times p_0}$.

Again, the row $\mathbf{y}_j^*$ of $\mathbf{Y}^*$ is the embedding for the out-of-sample 2-tuple $(\boldsymbol{\mu}_j^*,\boldsymbol{\Sigma}_j^*) \equiv \mathcal{G}_j^*$ which represents the original data point $\mathbf{x}_j^*$ with its neighbourhood $\mathcal{N}_j^*$. Note that Equations (5.25) and (5.26) are also due to [141] which are based on KPCA formulation.

5.9.3 Discussion

From the generalization via the Nyström formula presented above, one can consider another advantage for adhering to metric properties via measures such as $d_{J\mathcal{R}}$, $d_{B\mathcal{R}}$ and d_H in the context of Euclidean embedding. The benefits of $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ over d_H, however, will be discussed in the next chapter. Euclidean embedding via semi-metrics instead of metrics will result in the following consequences: *First*, a DMM can not be defined since Axiom (3) and the triangle inequality of a metric may not hold, and *Second*, it follows that the resulting similarity matrix $\mathbf{K}$ will be indefinite.

A first option to overcome this situation is via non-metric MDS [65, 66], which defines a transformation by minimizing a stress (or error) function. Unfortunately, this transformation does not provide an embedding nor it can be considered a mapping, and hence, generalization to out-of-sample examples can not be obtained [142]. Another solution is to approximate the matrix $\mathbf{K}$ to a nearby PSD matrix by truncating the negative eigenvalues of $\mathbf{L}$, or using minimum shift embedding [141] which adds the smallest constant to $\mathbf{L}$ such that it transforms $\mathbf{K}$ to a PSD matrix. Although generalization via the Nyström formula can be obtained for the approximated matrix, relying on proper distance metrics suppresses any need for such approximations.

Similar remarks can be made for Laplacian embedding methods (spectral clustering and Laplacian eigenmaps), however due to their reliance on parametrized kernels, indefinite similarity matrices $\mathbf{K}$ can *sometimes* be overcome by tuning the kernel parameters. The scaling parameter σ that appears in the Gaussian kernel, is an example of such a tuning parameter. Nevertheless, for Laplacian embedding, even if the similarity matrix is based on a semi-metric and $\mathbf{K}$ is PSD, semi-metrics can still have a negative impact on the embedding as demonstrated in Section 5.4.2.

5.9.4 A note on computational complexity

The computational bottleneck for the augmented space $\mathbb{X}$ is in finding the nearest neighbours for each point, forming the covariance matrices, and more importantly filling the affinity matrix $\mathbf{K}$ using the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$, or their exponentiated versions. In almost all of our experiments, both metrics gave very similar results, however, $d_{J\mathcal{R}}$ is com-

putationally less expensive than $d_{B\mathcal{R}}$ (when taking into consideration a large data set). This is due to the first term of $d_{B\mathcal{R}}$ which requires computing the inverse of the sum for two covariance matrices (one for each pair of points). This is however, unlike the first term in $d_{J\mathcal{R}}$ which is the sum of the inverse of the two covariance matrices (one for each point), which can be done once for every point before filling the matrix $\mathbf{K}$.

The real complexity occurs in computing the Riemmanian metric for two covariance matrices which involves solving a generalized eigenvalue problem for each pair of points. The complexity of such a problem is usually $O(p^3)$ in the worst case, and if the matrix is of rank r, and one seeks only the first r eigenvectors, then this reduces to $O(r^3)$ in the worst case. However, what increases the complexity is filling the affinity matrix $\mathbf{K}$ which requires exactly $O(\frac{n(n-1)}{2}r^3)$.

5.10 Related Work to $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$

So far I have presented the main idea of the unsupervised metric space learning algorithm using spectral methods and the augmented space $\mathbb{X}$ extracted from the input data set $\mathcal{D}$, which naturally led to the Jeffreys-Riemann metric $d_{J\mathcal{R}}$ and the Bhattacharyya-Riemann metric $d_{B\mathcal{R}}$. The metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ can also be seen as metrics over the neighboorhoods $\mathcal{N}_i$, $1 \leq i \leq n$, when modelled as Gaussian distributions, and hence they can be seen as an alternative to the Euclidean distance between any two points in $\mathcal{D}$. This perspective parallels a stream of ideas that considers distances (or similarities) between two subspaces, tangent spaces, or sets of vectors (SOVs), instead of the direct distance (or similarity) between points.

In the context of object recognition, Simard *et al.* [143] represent each image as set of modified images obtained by different linear transformations. Based on the new representation, the distance between two sets of images is the distance between the two tangent planes passing through each set of images. In a similar vein, Ghodsi & Schuurmans [144] use a similar approach of linear transformations to images in the context of manifold learning and nonlinear dimensionality reduction. Due to the nature of linear transformations that are specific to images, this approach can not be easily applicable to general data sets. Vincent & Bengio [145] propose a different idea in the context of k-NN classification. In

Table 5.1 The seventeen (17) UCI data sets used in the experiments.

Data set	Classes	Size	Attributes	Data set	Classes	Size	Attributes
Balance	3	625	4	Monks–2	2	601	6
Bupa	2	345	6	Monks–3	2	554	6
German	2	1024	2	NewThyroid	3	215	5
Glass	6	214	9	Segment	7	2310	16
HouseVotes	2	435	16	Sonar	2	208	60
Ionosphere	2	351	33	WDBC	2	569	30
Iris	3	150	4	Wine	3	178	13
Lymphography	4	148	18	Yeast	10	1484	6
Monks–1	2	556	6				

their algorithm, each point in the training set, together with its m NNs of the same class ($m \neq k$), define a subspace. The distance between a query point and all other points in the data set, reduces to the length of the orthogonal projection from the query point to each subspace, and hence the name of their algorithm k-local hyperplanes.

In the context of learning over SOVs, Wolf & Shashua [146] propose a general learning approach within the kernel framework. For two SOVs, their kernel is based on the principal angles between two subspaces, each spanned by one of the two SOVs. Kondor & Jebara [44] represent each image as a bag of pixels, which is also a SOV. Each SOV is modelled as a Gaussian distribution, and the Bhattacharyya kernel K_B is used with SVMs to classify the images. Similarly, Moreno & Vasconcelos [45] represent each multimedia object (an image or an audio signal) as a bag of features (or one SOV), and then model each SOV as Gaussian distribution. However, instead of K_B, they use the KL kernel K_J with SVMs to classify the multimedia objects. As an application in Chapter 6, $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ will be used as metrics for SOVs in the context of classification and clustering of human actions extracted from video data

5.11 Experimental Results

The unsupervised metric space learning algorithm presented in this chapter is validated in the context of unsupervised learning via clustering algorithms. The experimental setting for showing the efficacy of the augmented space $\mathbb{X}$ and the proposed metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ is

based on measuring the performance of k-Means clustering in different embedding spaces. More specifically, for a data set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$, k-Means is run on the input space $\mathcal{X}$ as a baseline performance measure. This baseline k-Means performance is compared to: 1) spectral clustering (SC) according to the version of Ng *et al.* [27] using the exponential kernel K_E, and 2) SC over the augmented space $\mathbb{X}$ using four (4) different kernels: the KL kernel K_J [45], the Bhattacharyya kernel K_B [44], the Hellinger kernel K_H, and the proposed kernel $K_{B\mathcal{R}}$. Although the experiments included $K_{J\mathcal{R}}$, it was found that the results of $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ are very close to each other, and hence we show only the results for $K_{B\mathcal{R}}$. This shows that the main difference between $d_J(\mathcal{G}_1, \mathcal{G}_2)$ and $d_B(\mathcal{G}_1, \mathcal{G}_2)$ are the semi-metrics for covariance matrices in Equation (5.12).

It is important to recall that the original SC algorithm of Ng *et al.* , is in fact, a Laplacian embedding for $\mathcal{X}$ into a metric space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$ or a semi-metric space $(\mathbb{R}^{p_0}, \|\cdot\|_2^2)$ (depending on the kernel), followed by k-Means clustering in the embedding space. Therefore SC over $\mathbb{X}$, implies Laplacian embedding for $\mathcal{D}_A \subset \mathbb{X}$ using the different kernels K_J, K_B, K_H, and $K_{B\mathcal{R}}$, and then applying k-Means clustering on the data in the different embedding spaces. Therefore, comparing with SC is equivalent to comparing with Laplacian eigenmaps [20].

The objective of the experiments presented here is twofold:

1. Validate the efficacy of the augmented space $\mathbb{X}$ over the input space $\mathcal{X}$ for unsupervised learning. This implictly includes validating that $\mathbb{X}$ can accommodate the characteristics of real world data sets, and the uneven sample distribution in the input space $\mathcal{X}$.

2. Show the efficacy of the proposed relaxed kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$, and more specifically the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$, over other divergence measures that do not adhere to metric properties. This also includes the metric d_H for the case of Gaussian densities, which despite being a metric, does not respect the geometry of $\mathbb{S}_{++}^{p\times p}$.

5.11.1 Experimental setting

All algorithms were run on 17 data sets from the UCI machine learning repository [1], shown in Table 5.1. Clustering accuracy was measured using the Hungarian score used in

Table 5.2 Clustering accuracy (%), with standard deviation, for k-Means, SC with K_E, and SC over $D_A = \{(\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i)\}_{i=1}^n$ with K_J, K_B, K_H, and $K_{B\mathcal{R}}$.

Data set	k-Means	K_E	K_J	K_B	K_H	$K_{B\mathcal{R}}$
Balance	51.1 (3.2)	59.3 (2.6)	54.5 (2.7)	58.7 (1.2)	57.5 (2.41)	**63.4** (0.5)
Bupa	55.1 (0.1)	56.8 (0.1)	57.6 (0.01)	57.3 (0.01)	58.2 (0.01)	**62.3**(0.2)
German	67.6 (0.1)	70.0 (0.1)	**71.5** (0.02)	**71.4** (0.02)	62.7 (0.1)	70.0 (0.05)
Glass	49.7 (3.7)	49.7 (3.8)	52.8 (2.1)	**53.2** (1.5)	51.6 (1.1)	**53.8** (0.8)
HouseVotes	**87.8** (0.1)	**87.8** (0.2)	82.1 (0.02)	81.2 (0.2)	83.2 (0.01)	**87.5** (0.1)
Ionosphere	70.9 (1.2)	70.3 (0.1)	84.9 (0.1)	**85.1** (0.03)	75.7 (0.02)	75.1 (0.4)
Iris	79.8 (15.7)	88.7 (6.6)	90.0 (0.1)	90.0 (0.01)	90.6 (0.01)	**96.6** (0.1)
Lymphography	47.0 (6.5)	42.9 (4.8)	59.9 (1.1)	**60.8** (1.1)	53.1 (3.5)	55.5 (0.7)
Monks–1	62.6 (6.3)	66.5 (0.02)	68.7 (0.06)	68.7 (0.09)	**69.6** (0.04)	68.3 (0.01)
Monks–2	51.4 (1.7)	50.7 (0.07)	57.1 (0.08)	57.0 (0.02)	57.4 (0.1)	**63.2** (1.4)
Monks–3	63.9 (6.4)	65.1 (0.1)	69.4 (0.2)	69.8 (0.32)	69.9 (0.09)	**80.1** (0.1)
NewThyroid	76.9 (9.4)	75.2 (0.4)	87.5 (5.5)	93.0 (0.01)	92.4 (0.2)	**94.4** (0.05)
Segment	50.7 (8.3)	63.3 (4.1)	22.1 (1.1)	43.1 (2.9)	49.9 (3.0)	**65.5** (3.3)
Sonar	54.7 (0.8)	55.7 (0.1)	57.2 (0.02)	57.0 (0.2)	58.6 (0.04)	**61.9** (0.2)
WDBC	85.4 (0.1)	**90.8** (0.1)	65.5 (0.1)	71.7 (0.05)	75.2 (0.01)	**89.4** (0.07)
Wine	66.4 (5.9)	70.2 (0.2)	90.4 (0.07)	90.4 (0.03)	91.5 (0.09)	**95.2** (0.2)
Yeast	34.2 (1.8)	32.2 (0.9)	32.4 (1.6)	32.7 (1.6)	32.7 (0.97)	**37.0** (1.1)

[147, 148], and the performance of each algorithm was averaged over 30 runs of k–Means with different initializations. Since the number of classes of the UCI data sets is given, the number of clusters is assumed to be known.

The parameter σ for K_E, K_J, K_B, K_H and $K_{B\mathcal{R}}$ was selected using a simple quantile based approach[6]. In all the experiments, the number of NNs for the local modelling was allowed to range between 5 and 16, and the regularization parameter γ in Equations (5.1) and (5.2) was set to $\gamma = 1$. It is important to note that selecting the best parameter values for σ, γ, the number of NNs for local modelling, and the number of clusters, is a fundamental question of model selection, and hence, it should not be confounded with verification of the efficacy of the augmented space $\mathbb{X}$ and the the proposed metric $d_{B\mathcal{R}}$ reflected by its use in $K_{B\mathcal{R}}$. Further, even when the *best* γ value is not selected, the results nevertheless show that, under this fixed γ setting, clustering after embedding the augmented space $\mathbb{X}$ using $K_{B\mathcal{R}}$ into $\mathbb{R}^{p_0}$ typically shows significantly better results.

[6]The approach was suggested in Alex Smola's blog: `http://blog.smola.org/page/2`

Table 5.3 Clustering accuracy (%), with standard deviation, for k-Means, SC with K_E, and SC over $D_A = \{(\mathbf{x}_i, \mathbf{A}_i)\}_{i=1}^n$ with K_J, K_B, K_H, and $K_{B\mathcal{R}}$.

Data set	k–Means	K_E	K_J	K_B	K_H	$K_{B\mathcal{R}}$
Balance	51.1 (3.2)	59.3 (2.6)	60.7 (0.5)	60.5 (1.9)	60.2 (1.3)	**64.3** (4.9)
Bupa	55.1 (0.1)	56.8 (0.1)	**57.6** (0.05)	56.2 (0.1)	**57.5** (0.1)	**57.3** (0.07)
German	67.6 (0.1)	70.0 (0.1)	**71.4** (0.01)	57.7 (0.1)	59.2 (0.1)	70.00 (0.04)
Glass	49.7 (3.7)	49.7 (3.8)	**53.3** (1.7)	51.5 (1.8)	48.5 (4.9)	49.7 (3.9)
HouseVotes	87.8 (0.1)	87.8 (0.2)	82.0 (0.02)	82.9 (0.3)	85.0 (1.3)	**88.0** (0.01)
Ionosphere	70.9 (1.2)	70.3 (0.1)	**83.4** (0.02)	71.5 (0.05)	71.2 (0.05)	72.0 (0.01)
Iris	79.8 (15.7)	**88.7** (6.6)	82.0 (0.05)	82.3 (0.3)	82.6 (0.02)	82.6 (0.06)
Lymphography	47.0 (6.5)	42.9 (4.8)	59.1 (1.1)	52.7 (0.05)	51.5 (0.8)	**60.5** (4.2)
Monks–1	62.6 (6.3)	66.5 (0.02)	66.5 (0.01)	66.5 (0.05)	66.5 (0.01)	66.5 (0.01)
Monks–2	51.4 (1.7)	50.7 (0.07)	55.0 (0.07)	57.4 (0.2)	55.2 (2.4)	**65.2** (0.4)
Monks–3	63.9 (6.4)	65.1 (0.1)	**69.3** (0.03)	64.2 (1.3)	63.5 (0.02)	**69.6** (0.03)
NewThyroid	76.9 (9.4)	75.2 (0.4)	78.1 (2.1)	87.1 (4.4)	55.0 (0.2)	**88.3** (0.02)
Segment	50.7 (8.3)	63.3 (4.1)	24.0 (1.4)	56.0 (3.5)	59.5 (2.3)	**66.9** (7.5)
Sonar	54.7 (0.8)	55.7 (0.1)	58.6 (0.02)	57.5 (0.7)	57.2 (0.02)	**61.0** (0.02)
WDBC	85.4 (0.1)	**90.8** (0.1)	63.7 (0.03)	82.2 (8.5)	73.6 (0.02)	87.3 (0.02)
Wine	66.4 (5.9)	70.2 (0.2)	91.0 (0.01)	89.3 (0.03)	89.3 (0.04)	**95.5** (0.04)
Yeast	34.2 (1.8)	32.2 (0.9)	33.9 (1.5)	34.0 (1.4)	33.5 (1.5)	**36.0** (1.4)

5.11.2 Analysis of the results

Column 2 in Tables 5.2 and 5.3 show the results of k-Means on the original data input space $\mathcal{X}$. Column 3 in Tables 5.2 and 5.3 show the results of k-Means after the Laplacian embedding for $\mathcal{X}$ via K_E, which is the SC algorithm. Columns 4 to 7 in Tables 5.2 and 5.3 show the results of k-Means after the Laplacian embedding for the augmented data sets $D_A = \{(\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i)\}_{i=1}^n$, and $D_A = \{(\mathbf{x}_i, \mathbf{A}_i)\}_{i=1}^n$ respectively, using K_J, K_B, K_H, and $K_{B\mathcal{R}}$ into $\mathbb{R}^{p_0}$. It can be seen that the results in these Tables validate the key ideas in this chapter; 1) the efficiency and the efficacy of the augmented space $\mathbb{X}$, and 2) the efficacy of the proposed metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ over the semi-metrics d_J and d_B.

First, it can be seen that for most of the cases, Laplacian embedding for $\mathcal{X}$ via K_E yields better clustering accuracy over the base line k-Means algorithm, which is the expected performance for SC. Second, in general, the accuracy of k-Means after embedding the augmented space $\mathbb{X}$ via the different relaxed kernels, is higher than it is for the standard SC algorithm. This general trend reflects the efficacy of the augmented space $\mathbb{X}$ and its underlying motivation. In addition, it can be noticed that the accuracy of the relaxed kernels

over $D_A = \{(\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i)\}_{i=1}^n$ is higher than it is over $D_A = \{(\mathbf{x}_i, \mathbf{A}_i)\}_{i=1}^n$, which is probably due to the smoothing included in defining the 2-tuple $(\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i)$. In the cases when the accuracy using the relaxed kernels is very close, or slightly inferior, to the accuracy of k–Means on $\mathcal{X}$ and SC via K_E, then this is due to the unified γ parameter, which shows a nice opportunity for improvement under an optimized γ.

In terms of kernels performance, $K_{B\mathcal{R}}$ is usually better or as good as all other kernels, which reflects the combined effect of the augmented space $\mathbb{X}$ and adhering to the metric properties of $\mathbb{R}^p$ and $\mathbb{S}_{++}^{p\times p}$. While K_H and $K_{B\mathcal{R}}$ are both relaxed kernels over $\mathbb{X}$, with satisifed metric properties via d_H and $d_{B\mathcal{R}}$ respectively, $d_{B\mathcal{R}}$ seems to be a better formulation than d_H and all other measures. As mentioned earlier, d_H uses a semi-metric for symmetric PD matrices that is unaware of the geometry of $\mathbb{S}_{++}^{p\times p}$, which is not the case for $d_{B\mathcal{R}}$.

Finally, the particular cases of `housevotes`, `segment`, and `WDBC` are another example on how semi-metrics can yield unreliable embeddings. While K_J, K_B and K_H are quite inferior to the base line k–Means and to SC via K_E, $K_{B\mathcal{R}}$ maintains a consistent performance equal to or higher than these raw algorithms.

5.12 Discussion and Concluding Remarks

In this chapter I have introduced an algorithmic framework for unsupervised metric learning that is based on spectral methods and not on learning an instance of the GQD. For a data set $\mathcal{D} \subset \mathcal{X} \subset \mathbb{R}^p$, and without any labels nor side-information, the algorithm extracts local density information from each point $\mathbf{x}_i \in \mathcal{D}$, and forms the augmented space $\mathbb{X}$. The motivation for $\mathbb{X}$ is to accommodate the characteristics of real world data sets and the uneven sample distribution in the input space, where both factors negatively affect GQD-type measures. This unsupervised gathering of local information in the form of regularized local covariance matrices can be seen as a *self supervised* learning of context from the data when no *a priori* information is available. This bootstrap learning of context indeed has a computational overhead to obtain the augmented data set $\mathcal{D}_A$, however this is needed to compensate the absence of *a priori* information.

The augmented space $\mathbb{X}$ naturally led to the relaxed kernels over Gaussian densities via convolution kernels. The factorizable nature of convolution kernels allowed introducing the Riemannian metric for covariance matrices, which finally led to the new kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$, and consequently the Jeffreys-Riemann metric $d_{J\mathcal{R}}$, and the Bhattacharyya-Riemann metirc $d_{B\mathcal{R}}$. Note that the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ were originally introduced in [29], and based on their axiomatic metric properties, the kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ were built on top of them. However, here and in [30][7], convolution kernels were the entry point to $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$, which finally led to the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$.

The augmented metric space $\mathbb{X}$ is richer in information than $\mathcal{X}$ about the local structure in the data since it considers the local density around each point $\mathbf{x}_i \in \mathcal{D}$. Hence the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ take this varying density into consideration, and code it as a distance measure that respects all metric properties of its constituting arguments. In other words, the metric spaces $(\mathbb{X}, d_{J\mathcal{R}})$ and $(\mathbb{X}, d_{B\mathcal{R}})$ reorganize the proximity between the points in $\mathcal{D}$ based on $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ respectively, which take the varying local density of the input space into consideration, and respect the geometry of $\mathbb{R}^p$ and $\mathbb{S}_{++}^{p\times p}$. This is unlike the GQD type measures that are constant over the entire input space and do not take this varying density into consideration. This makes the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ more suitable for the characteristics of real world data sets, and the uneven sample distribution in the input space.

An interesting feature for the metric spaces $(\mathcal{D}_A, d_{J\mathcal{R}})$ and $(\mathcal{D}_A, d_{B\mathcal{R}})$, is that they can be encapsulated with any spectral or manifold learning algorithm. Based on the results in previous section, it is expected that this combination significantly improves the performance of hypothesis learning in the low dimensional space. In principle, the metric spaces $(\mathbb{X}, d_{J\mathcal{R}})$ and $(\mathbb{X}, d_{B\mathcal{R}})$ can be used with any hypothesis learning algorithm. However, since learning directly over these metric spaces is computationally expensive, embedding via spectral methods offers a great reduction in terms of computational and space complexities, while preserving all the information on proximities based on $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$, local densities, and the correlations among variables during the embedding. In fact, using the analysis of diffusion maps [149], it can be shown that the Euclidean distance in the embedding space approximates the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ in the augmented space $\mathbb{X}$, upto a scaling factor. This, however, is left as a future research work.

[7] Thanks to Mohak Shah.

Due to the information captured in $\mathbb{X}$, this augmented space can be called an *information layer* over the input space $\mathcal{X}$. This information layer is adaptive since it is controlled by the regularization parameter γ in Equations (5.1) and (5.2), and the size of the neighbourhood for local modelling. These parameters can be jointly optimized with the parameters of the learning algorithm in the embedding space to improve the overall generalization of the learning algorithm, however this perspective for $\mathbb{X}$ needs further investigation on its own.

Although all the ideas presented here are under an unsupervised learning setting, they can be easily extended to the semi-supervised and supervised settings by incorporating the additional information in the neighbourhood forming and modelling stage, and in computing the similarity matrices $\mathbf{K}$ in the Laplacian and Euclidean embedding stages. This additional information will not only improve the quality of the embedding for the training data, but will also improve generalization via the Nyström formula. Again, considering $\mathbb{X}$ (with $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$) as an adaptive information layer, and extending it in supervised, and semi-supervised learning settings remains to be explored in an independent future work.

In the following chapter, I consider the problem of learning a hypothesis (classification and clustering) over sets of vectors (SOVs), a.k.a bags of features, that appeared in the work of Kondor & Jebara [44], and Moreno *et al.* [45]. Surprisingly, the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$, and the relaxed kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$ naturally fit in this setting and will be used as distance and similarity measures for this type of data. It will be shown that these measures together with Laplacian and Euclidean embeddings, can be used in classification and clustering of SOVs, and they usually lead to better results than the measures proposed in [44] and [45]. This will be demonstrated using preliminary experiments for classification of human actions and clustering of human motion in video sequences.

Chapter 6

A Framework for Hypothesis Learning Over Sets of Vectors

In this chapter, I extend the unsupervised metric space learning algorithms developed in Chapter 5 to the problem of hypothesis learning (classification, clustering, etc.) over *sets of vectors* (SOVs) [31], a.k.a. *bags of features*. The term *"sets of vectors"* is due to Kondor & Jebara [44], and I use it here to generalize the term *"bags of features"* since the framework proposed here can be applied to any vectorial data which has this particular structure. The proposed framework for hypothesis learning over SOVs fully relies on spectral embedding via the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$, and the relaxed exponential kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$. The basic idea of the framework is to represent, or embed, each SOV into a single vector in a low dimensional Euclidean space. This embedding is not independent for each SOV, rather it is a collective embedding, for all SOVs, that depends on the similarities among all SOVs. Hence, classification and clustering can be achieved in the lower dimensional space on this simpler yet unified data representation, instead of the original structure as sets of vectors. The proposed framework is validated in two different learning contexts from video data; 1) supervised learning for human action recognition, and 2) unsupervised learning for clustering human motion.

6.1 Motivation

Sets of vectors are a common data representation in various domains such as computer vision in which an image is represented as a bag of features [150], motion analysis in video

in which a short video segment is represented as set of spatio–temporal gradient vectors [151, 152, 153, 154], and in speech recognition in which an utterance is represented as a set of MFCC vectors [155, 45], to mention a few. Despite their flexibility and richness as a representation, a major obstacle for directly learning a hypothesis (classification, clustering, etc.) over sets of vectors is their special structure, in which each object D_i in a data set of objects $\mathscr{D}$ is represented by a different number of vectors of fixed dimensionality, forming that one *set of vectors* (SOV). This nonuniform format of the input data requires the learning algorithm, and consequently the algorithm designer, to implicitly handle this non-regular type of input, either by unifying the format of the input, or by extracting the necessary information out of it, such as the (dis)similarity between two SOVs.

In this chapter I propose a principled, application independent framework that unifies the representation of SOVs in order to ease hypothesis learning over this type of data. In particular, as depicted in Figure 6.1, I propose an unsupervised learning approach that maps each SOV, or bag of features, to a single vector in a low dimensional Euclidean space. The proposed framework has slight overlap with some ideas introduced in [156, 44, 45, 150, 153] which shall be briefly reviewed in the following section. Although the speech recognition community [155] has pioneered learning over time-series or sequential data, which are special cases of SOVs, the present work is concerned with geenralized SOVs including sequential and time-series data.

The advantages of the proposed framework are as follows:

- The framework allows any learning algorithm to be transparently applied on SOVs through their images residing in the low dimensional subspace, and hence it frees the learning algorithm from the overhead of accommodating their special structure.
- The framework offers a reduction, by orders of magnitude, in the data's space complexity, which correlates directly with the computational complexity of the learning algorithm, resulting in significantly faster hypothesis learning.
- The framework is unsupervised, and hence it does not require labels nor side-information. However, if labels or side-information are available, they can be naturally integrated into the framework.

- The spectral embedding algorithm in the framework reveals the natural clusters in $\mathscr{D}$; i.e. as a by-product, the framework performs implicit clustering for SOVs which is reflected on the images in the low dimensional subspace.
- The framework has a well defined generalization to out-of-sample examples (SOVs in this case) using the Nyström formula, and hence it does not require retraining the system whenever new data are available.

Notations Before proceeding to the following sections, sets of vectors are formally defined as follows. Let $\mathscr{D} = \{D_i\}_{i=1}^n$ be a set of n objects D_i, where D_i can be a speech utterance or a short video segment for instance. Using a feature extraction function ϕ, the data set $\mathscr{D} = \{D_i\}_{i=1}^n$ is transformed to a set $\mathscr{S} = \{\mathcal{S}_i\}_{i=1}^n$ where $\phi : D_i \longmapsto \mathcal{S}_i = \{\mathbf{x}_1^i, \ldots, \mathbf{x}_{t_i}^i\}$, $\mathbf{x}_j^i \in \mathbb{R}^p$, t_i is the cardinality of set $\mathcal{S}_i$, and $\mathcal{S}_i$ is one set of vectors. Note that $\mathscr{S}$ is now a *set of sets* ; a.k.a. a family or a collection of sets. Note also that it is expected that each SOV $\mathcal{S}_i$ has a different number of vectors in it.

6.2 Related Work

Earlier approaches to hypothesis learning over SOVs focused on directly measuring the (dis)similarity between two SOVs using, for instance, dynamic time warping (DTW) [157], and the earth mover's distance [158]. Instead of measuring the similarity directly on the SOVs, a more popular approach in the computer vision community, is to construct a codebook of words (or visual words) from all the vectors of all SOVs, represent each SOV as a histogram of visual words, and then define kernels over the histograms [150] to be used for classification using support vector machines (SVMs).

A slightly different approach, which is adopted here, is to model each SOV $\mathcal{S}_i$ as a multivariate Gaussian distribution $\mathcal{G}_i$, where the mean vector $\boldsymbol{\mu}_i$ and the covariance matrix $\boldsymbol{\Sigma}_i$ are estimated using the sample mean and the sample covariance matrix for $\mathcal{S}_i$ respectively. Now that the set $\mathscr{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_n\}$ is replaced by the set $\mathscr{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_n\}$, a natural measure of (dis)similarity between two densities are divergence measures such as, the Bhattacharyya divergence d_B, the symmetric Kullback & Leibler (KL) divergence, a.k.a Jeffreys divergence d_J, and the Hellinger distance d_H [92]. For instance, Kondor & Jebara [44] use SVMs with kernels based on d_B to classify images represented as bags of pixels,

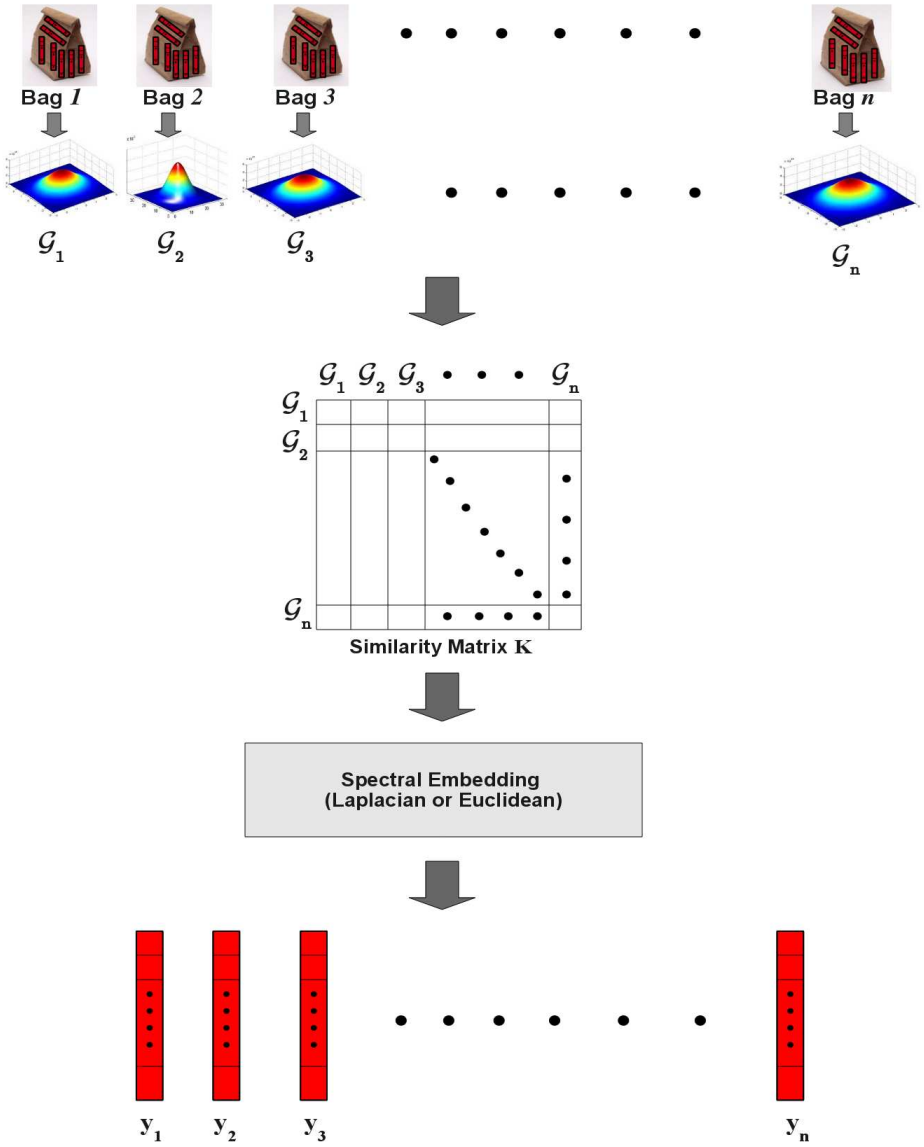


Fig. 6.1 Outline of the proposed framework for unifying the representation of sets of vectors. In the first step, each *bag of features*, or *set of vectors* (SOV) is modelled as a Gaussian distribution. In the second step, the difference or similarity between every pair of Gaussian densities is used to fill a (dis)similarity matrix $\mathbf{K}$. In the third step, spectral embedding methods (Laplacian or Euclidean embeddings) are used to collectively embed all SOVs in a low dimensional Euclidean space. The final result is that each bag i is represented by a single vector $\mathbf{y}_i$.

while Moreno *et al.* [45] uses SVMs with kernels based on d_J to classify multimedia objects (video,audio) represented as bags of features.

In the context of supervised learning over time-series data, Jaakkola & Haussler [156] model each class or category of SOVs using a single hidden Markov model (HMM) [159, 160], followed by extracting the Fisher score for each SOV $\mathcal{S}_i$. The Fisher score is a fixed size high dimensional vector that is extracted from the HMMs' parameters with respect to the pattern $\mathcal{S}_i$, and hence it uniquely represents $\mathcal{S}_i$. In turn, the Fisher scores unify the representation of variable length time-series patterns. Following this representation, the authors define the Fisher kernel over the Fisher scores, and use SVMs to classify these Fisher scores. Note that this framework is completely different from the standard HMM based approach used in speech recognition [155]. The advantage of [156] is that it allows powerful discriminative models such as SVMs, which can not handle variable length input, to be indirectly used for classifying variable length time-series patterns.

6.3 A Framework for Embedding Sets of Vectors

In the same spirit as the above approaches, but without being geared towards classification using SVMs, I propose an application independent framework that focuses on unifying the representation for SOVs, while discovering their latent natural clusters. That is, as shown in Figure 6.1, the first step in the framework is to model each bag of features, or SOV $\mathcal{S}_i$ as a Gaussian distribution $\mathcal{G}_i$, consequently forming the non-empty set of Gaussians $\mathscr{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_n\}$. In the second step, a (dis)similarity measure is used to fill the dis(similarity) matrix $\mathbf{K}$ with the distance (or similarity) between every pair of Gaussian distributions. In the last step, using the metric space learning algorithms presented in the previous chapter, each Gaussian $\mathcal{G}_i$ is finally embedded as a *single vector* $\mathbf{y}_i \in \mathbb{R}^{p_0}$, where in general $p_0 \ll p$, and $p_0 \ll n$.

In other words, instead of relying on kernels to measure the similarity between two probability distributions to be used in an SVM classifier as in [150, 44, 45], the Gaussian distributions $\mathscr{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_n\}$ are collectively embedded in a low dimensional subspace $\mathbb{R}^{p_0}$ in order to unify the representation of SOVs as in the work of Jaakkola & Haussler [156].

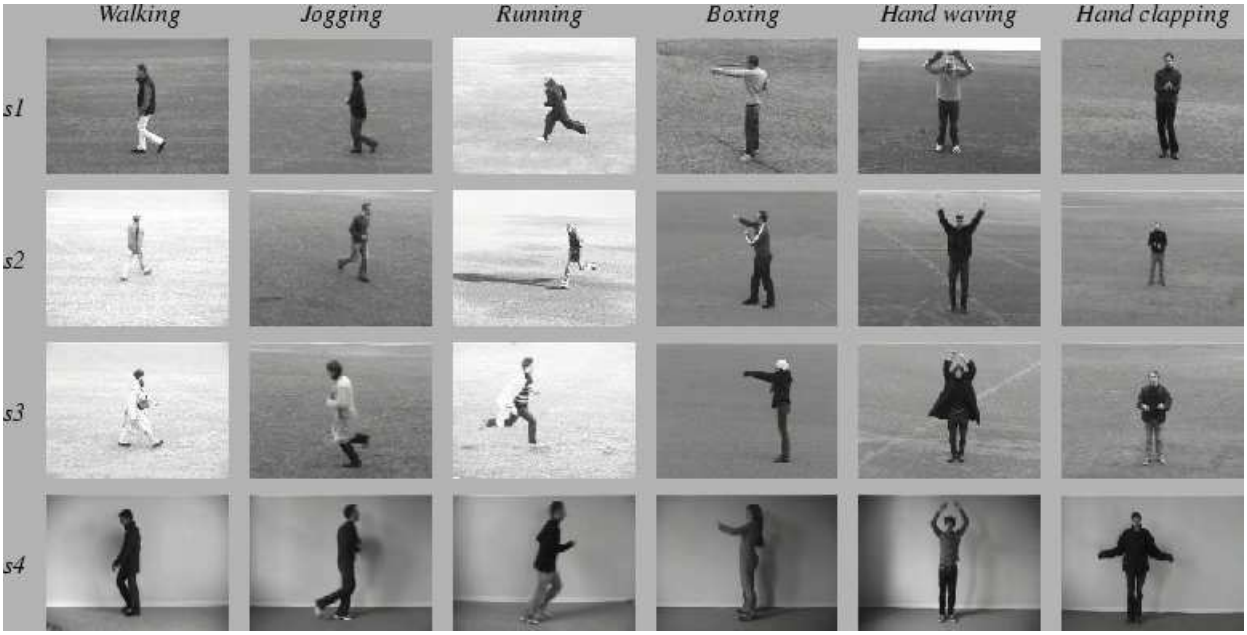


Fig. 6.2 Sample frames from the KTH video data set for human action recognition.

The key idea of the proposed framework is that after modelling each SOV $\mathcal{S}_i$ as Gaussian distribution $\mathcal{G}_i$, we obtain the non-empty set $\mathscr{G} = \{\mathcal{G}_i\}_{i=1}^n$. The set $\mathscr{G}$ of Gaussians, together with the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$ form the metric spaces $(\mathscr{G}, d_{J\mathcal{R}})$ and $(\mathscr{G}, d_{B\mathcal{R}})$ respectively, which are the dual perspective for the augmented space $\mathbb{X}$ introduced in the previous chapter. Therefore, it is straight forward to obtain an embedding for the set $\mathscr{G} = \{\mathcal{G}_i\}_{i=1}^n$ in the lower dimensional space $\mathbb{R}^{p_0}$ using Laplacian embedding and the kernels $K_{J\mathcal{R}}$ and $K_{B\mathcal{R}}$, or using Euclidean embedding and the metrics $d_{J\mathcal{R}}$ and $d_{B\mathcal{R}}$.

6.4 Experiments

The validity of the proposed framework is tested in two different learning contexts; 1) supervised learning for human action recognition from video sequences, and 2) unsupervised learning for clustering human motion in video sequences. For this purpose, we use the KTH video data set for human action recognition shown in Figure 6.2 [161][1]. The data set consists of video clips for 6 types of human actions (boxing, hand clapping, hand waving, jogging, running, and walking) performed by 25 subjects in 4 different scenarios

[1] `http://www.nada.kth.se/cvap/actions/`

(outdoors, outdoors with scale variation, outdoor with different clothes, and indoors), resulting in a total number of video clips $n = 6 \times 25 \times 4 = 600$. All video sequences are taken over homogeneous backgrounds with a static camera with a frame rate of 25 fps. The spatial resolution of the videos is 160×120, and each clip has a length of 20 seconds on average.

In the supervised learning setting, Section 6.4.2, the objective is to differentiate between 6 different types of human actions. To this end, the motion in each video clip is represented as a SOV by means of histograms of gradient orientations, and Euclidean embedding via $d_{B\mathcal{R}}$ is used to embed all SOVs (representing all video clips) in a low dimensional Euclidean space $\mathbb{R}^{p_0}$, where classification of actions is done via a simple k nearest neighbour (k-NN) classifier.

In the unsupervised setting, Sections 6.4.3 and 6.4.4, the objective is to cluster the frames of a long video sequence according to the different types of motion in the video. More specifically, in a long video sequence, there is a human subject performing different types of actions, and the objective is to assign frames with very similar motion content, a unique label. Here, the long video sequence is equally segmented into short video clips, and the motion information from the frames in each clip is represented as a SOV using histograms of gradient orientations. Next, Laplacian embedding is used to embed all SOVs into a low dimensional Euclidean space $\mathbb{R}^{p_0}$, where clustering is performed via the k-Means algorithm.

6.4.1 Representing motion as sets of vectors (SOVs)

To extract the motion information, a dense optical flow is computed for each video clip using the Lucas-Kanade algorithm [162][2], resulting in a large set of spatio-temporal gradients vectors describing the motion of pixels in each frame. The gradient vector is normal to the local spatio-temporal surface generated by the motion in the space–time volume. The gradient direction captures the local surface orientation which depends on the local behavioural properties of the moving object, while its magnitude depends mainly on the photometric properties of the moving object, and it is affected by its spatial appearance

[2]Implemented in Piotr's Image and Video Toolbox for Matlab `http://vision.ucsd.edu/ pdollar/toolbox/doc/`

(color, texture, etc.) [163].

To capture the motion information encoded in the gradient direction, first we apply an adaptive threshold based on the norm of the gradient vectors to eliminate all vectors resulting from slight illumination changes and camera jitter. Second, each video frame is divided into $h \times w$ blocks – typically 3×3 and 4×4 – and the motion in each block is encoded by an m–bins histogram of gradient orientations. In all our experiments, m is set to 4 and 8 bins. The histograms of all blocks for one frame are concatenated to form one vector of dimensionality $p = m \times h \times w$. Therefore, a single video clip D_i with t_i frames is finally represented as a set $\mathcal{S}_i = \{\mathbf{x}_1^i, \ldots, \mathbf{x}_{t_i}^i\}$, where $\mathbf{x}_j^i$ is a p-dimensional vector of the concatenated histograms of frame j. Since histograms of orientations from optical flow vectors can not differentiate between two identical actions performed at different speeds, I excluded the `'walking'` and `'running'` classes from the data set. This resulted in $n = 400$ video clips, for 25 persons performing 4 actions in 4 different scenarios.

6.4.2 Experimental Setting I : Human Action Recognition

After extracting the motion information from each video clip D_i and representing it as a SOV $\mathcal{S}_i$, each $\mathcal{S}_i$ is modelled as a Gaussian distribution $\mathcal{G}_i$ with mean vector $\hat{\boldsymbol{\mu}}_i = \frac{1}{t_i} \sum_{j=1}^{t_i} \mathbf{x}_j^i$, and a covariance matrix $\hat{\boldsymbol{\Sigma}}_i = \frac{1}{t_i - 1} \sum_{j=1}^{t_i} (\mathbf{x}_j^i - \hat{\boldsymbol{\mu}}_i)(\mathbf{x}_j^i - \hat{\boldsymbol{\mu}}_i)^\top + \gamma \mathbf{I}$, where γ is the regularization parameter as introduced in the previous chapter, and $\mathbf{I}$ is the identity matrix. In all the experiments γ was set to 1.

Using the Euclidean embedding algorithm described in Section 5.8, all the Gaussians representing the motion of all video clips were embedded in four low dimensional subspaces $\mathbb{R}^{p_0}$ using four different dissimilarity measures; $d_J(\mathcal{G}_i, \mathcal{G}_j)$ used in [45] which is a semi-metric, $d_B(\mathcal{G}_i, \mathcal{G}_j)$ used in [44] which is also a semi-metric, $d_H(\mathcal{G}_i, \mathcal{G}_j)$ which is a metric, and the metric $d_{B\mathcal{R}}(\mathcal{G}_i, \mathcal{G}_j)$. This resulted in 4 similarity matrices, $\mathbf{K}_J$, $\mathbf{K}_B$, $\mathbf{K}_H$, and $\mathbf{K}_{B\mathcal{R}}$ respectively. Note that p_0, the dimensionality of the embedding space, is a free parameter that is either user defined, or selected by cross validation.

To classify the different actions embedded in the different low dimensional subspaces, a k–NN classifier is used with, $k = \{1, 3, 5, 7\}$. The empirical error rate is measured using

Table 6.1 Empirical error rate (%), with standard deviation, and the dimensionality p_0 of the embedding space obtained by the four different dissimilarity measures on the four feature settings obtained from the KTH data set.

$m \times h \times w$	d_J	d_B	d_H	$d_{B\mathcal{R}}$
$4 \times 3 \times 3$	21.2 (3.8), $p_0 = 11$	20.2 (3.4), $p_0 = 30$	19.7 (3.7), $p_0 = 45$	**17.7 (4.7)**, $p_0 = 38$
$4 \times 4 \times 4$	16.7 (3.6), $p_0 = 15$	17.0 (4.1), $p_0 = 19$	16.9 (3.7), $p_0 = 44$	**15.9 (3.2)**, $p_0 = 47$
$8 \times 3 \times 3$	24.3 (2.9), $p_0 = 43$	23.3 (4.9), $p_0 = 48$	22.1 (3.8), $p_0 = 44$	**19.9 (3.8)**, $p_0 = 45$
$8 \times 4 \times 4$	20.9 (4.6), $p_0 = 20$	20.4 (3.8), $p_0 = 22$	20.4 (3.7), $p_0 = 22$	**18.8 (3.5)**, $p_0 = 47$

a 30 folds double cross validation procedure, in which the data set is randomly split into a training set (80%) and a test set (20%), and then search for k that minimizes the training error of the current split. This optimal k is used to obtain the test error of one trial. This process is repeated 30 times, and the final empirical error (with standard deviation) is the average test error over all the 30 trials. Since p_0 is a free parameter, the optimal p_0 for each embedding is selected based on the lowest empirical error, where $p_0 \in [2, 50]$.

Before proceeding to the results, it is worth recalling that selecting the optimal values for m, h, w, γ, and the parameters for optical flow computation, is again a question of model selection which is not addressed here. Nevertheless, even though when these parameters are not optimized, Euclidean embedding using the metric $d_{B\mathcal{R}}$ appears to be a valid framework for unifying the representation of SOVs with various desirable properties as will be shown below.

Analysis of the results

Table (6.1) shows the empirical error rate on the KTH data set using the experimental setting described above. First of all, it is interesting to have a round figure on the results reported on this data set using more sophisticated systems. State of the art results on this data set with very sophisticated feature descriptors and SVM classifiers are around 20% error rate as reported in [161], and 10% error rate as reported in [164]. Given the very simple features used in our experiments, our error rates seems to be very comparable to these state of the results.

Second, our hypothesis before running the experiments is that the embeddings obtained

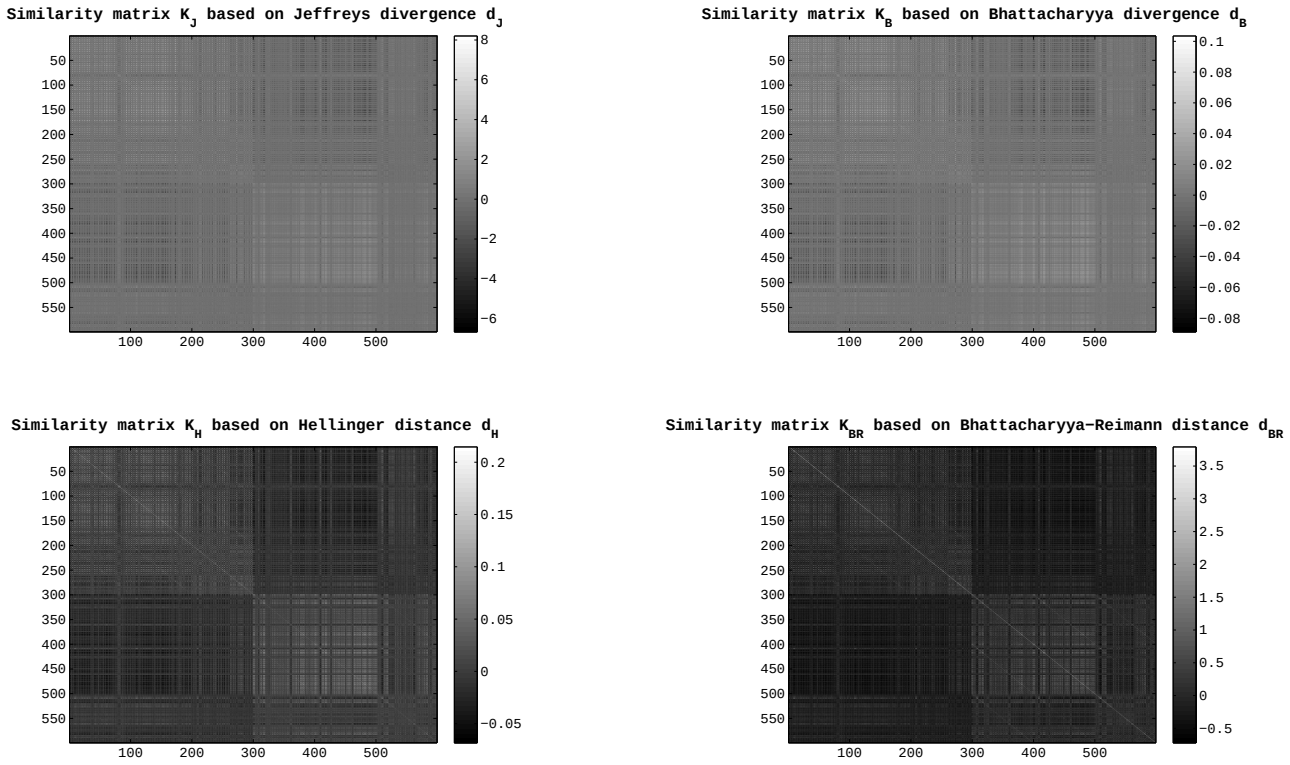


Fig. 6.3 The four similarity matrices $\mathbf{K}_J$ top left, $\mathbf{K}_B$ top right, $\mathbf{K}_H$ bottom left, and $\mathbf{K}_{B\mathcal{R}}$ bottom right. Note the clear block structure for $\mathbf{K}_{B\mathcal{R}}$ compared to other matrices. This figure is better seen on a coloured display.

via d_J and d_B will yield higher classification error than those embeddings obtained via d_H and $d_{B\mathcal{R}}$ since d_J and d_B are semi-metrics. According to Theorem 5.8.1 and the definition of semi-metrics, the resulting similarity matrix $\mathbf{K}$ is not guaranteed to be PSD for semi-metrics, and hence the resulting embedding space will be a semi-metric space in which metric properties and the relative distances between points are violated. Table 6.1 shows the classification error (with standard deviation) and the dimensionality of the embedding space for each dissimilarity measure on the 4 feature sets extracted from the KTH data set. It can be seen that despite the dimensionality p_0, d_H resulted in lower classification error than d_J and d_B did, while the embedding based on the proposed metric $d_{B\mathcal{R}}$ yielded the lowest error among all other dissimilarity measures. Although d_H is a metric, $d_{B\mathcal{R}}$ performs better since it is able to better characterize the natural grouping in the data and separat them in $\mathbb{R}^{p_0}$, which is reflected in the form of low error rates in Table 6.1.

To see this natural grouping of the data, while being able to compare the difference between the 4 embeddings, we pick the $4 \times 4 \times 4$ feature set from the 4 sets of features shown in Table 6.1 since it yielded the lowest error rate with all dissimilarity measures. Using this feature set, we obtain the 4 similarity matrices $\mathbf{K}_J$, $\mathbf{K}_B$, $\mathbf{K}_H$, and $\mathbf{K}_{B\mathcal{R}}$ shown in Figure 6.3 (better seen on a display). It can be clearly seen that $\mathbf{K}_{B\mathcal{R}}$ has 3 clear block structures along the diagonal, indicating three main categories in the data, which has originally 4 classes. Further, the top-left block of $\mathbf{K}_{B\mathcal{R}}$ has further sub-blocks indicating finer categories within the data. This is less clear for $\mathbf{K}_H$, and obscured in the case of $\mathbf{K}_J$ and $\mathbf{K}_B$.

Further analysis can be made by comparing the eigen-spectrum of the four similarity matrices $\mathbf{K}_J$, $\mathbf{K}_B$, $\mathbf{K}_H$, and $\mathbf{K}_{B\mathcal{R}}$, and in particular, the tail of each eigen-spectrum which reflects the adherence of each dissimilarity measure to the metric properties. From Theorem 5.8.1, we know that only metrics will yield PSD similarity matrices $\mathbf{K}$. This is exactly depicted in Figure 6.4 where the smallest eigenvalues for $\mathbf{K}_H$ and $\mathbf{K}_{B\mathcal{R}}$, generated by d_H and $d_{B\mathcal{R}}$ respectively, are greater than or equal to zero. This is unlike d_J and d_B which resulted in negative definite matrices $\mathbf{K}_H$ and $\mathbf{K}_B$ respectively, and hence the negative eigenvalues in Figure 6.4.

Finally, it is important to consider the reduction in space complexity achieved by the proposed framework. If the minimum representation of a single video frame, using the first feature set in Table 6.1 and a double precision format is $4 \times 3 \times 3$ $(m \times h \times w)$ $\times 4$ (bytes) $= 144$ bytes per frame, then for 400 clips, with 25 fps, and an average length for video clips of 20 seconds, the total space required for the data set is $400 \times 20 \times 25 \times 144 \approx 27$ MB. However, after using the proposed framework, the same data set will require 400 (clips) $\times\ p_0 \times 4 = 73$ KB of memory for $p_0 = 47$ using $d_{B\mathcal{R}}$ (see Table 6.1). This is a significant reduction in space complexity, and indeed learning a hypothesis over the embedded data set will be much faster than learning a similar hypothesis over the original representation.

6.4.3 Experimental Setting II : Motion clustering – An illustrative example

The objective of these experiments is to cluster or group similar motion profiles in a video sequence in an unsupervised manner. This task is usually found as preliminary step in applications such as motion segmentation and tracking [165], event modelling and event based

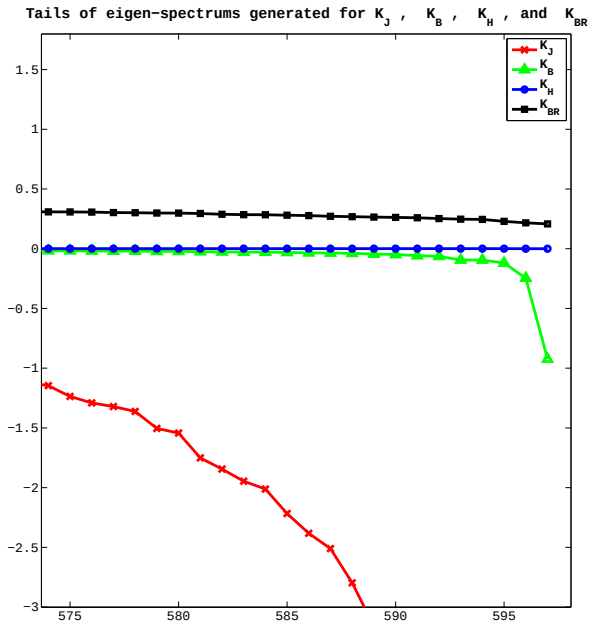
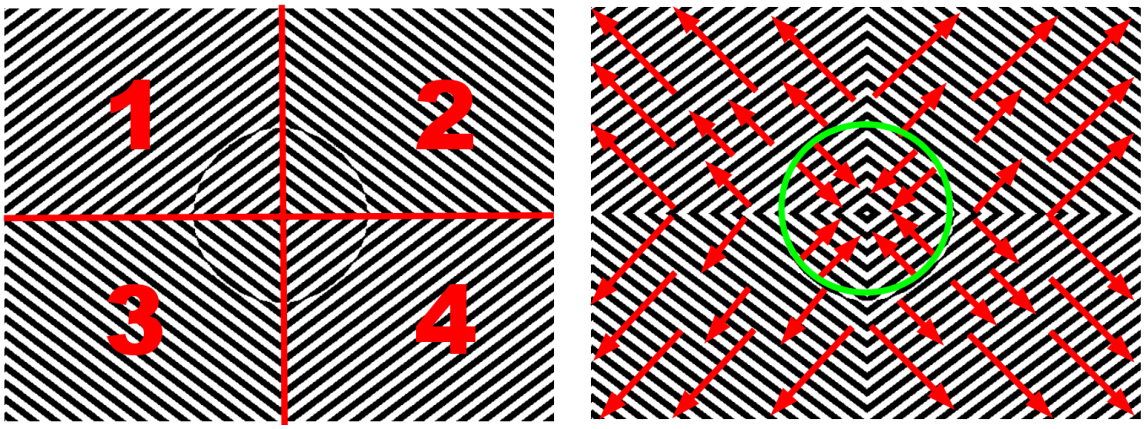


Fig. 6.4 Tails of eigen-spectrums for the four similarity matrices shown in Figure (6.3). Note how the semi-metrics d_J and d_B yield negative eigenvalues, indicating that $\mathbf{K}_J$ and $\mathbf{K}_B$ are negative definite matrices and not PSD as required by Theorem 5.8.1

analysis for video [163], unusual event detection [166, 152], analysis of spatio-temporal patterns in video [151], scene understanding and analysis [153, 154, 167], etc. In this task, the video is usually decomposed into smaller units where it is expected that a motion activity will occur. This smaller unit can be a single frame, a group of contiguous frames, or any smaller spatio-temporal units known as a *cuboids* [151, 152, 153, 154]. In the next step, motion information is extracted from each unit and represented as a p-dimensional vector. Finally, classification and clustering of motion (or events) can be obtained by applying learning algorithms on this vectorial representation of motion activities.

Motion clustering has stronger requirements than motion segmentation [168]. While the latter focuses on detecting and isolating moving pixels from static backgrounds, the former, in addition to motion segmentation, focuses on grouping similar motion profiles into clusters. Note that the representative motion profile (or activity) of each cluster, or the cluster mean, is defined here as a *motion pattern*. In our particular context, for a long video sequence in which a human subject is performing different types of actions, the



(a) (b)

Fig. 6.5 (a) One frame from the illusion sequence with the red lines indicating the four different regions of motion taking place in the sequence. (b) The red arrows indicate the direction of the black strips in each block. The green circle indicates the boundary points at which the motion in each block flips its direction. Note that the two types of motion appearing in each block are considered together as one motion pattern.

objective is to assign frames which contains similar actions, a unique label. An illustrative example for motion clustering in a synthetic video sequence is described in the following, followed by the experiments on longer video sequences with different human actions.

Consider Figure 6.5 which depicts one frame from a sequence of frames known as the *illusion video sequence*[3]. Each frame in the video sequence is divided into 4 blocks, where the black strips in each block are moving in the directions of the red arrows depicted in Figure 6.5(b). In the illusion movie, there are four different motion patterns, one for each block. Given a clustering algorithm and a suitable representation for the motion in the frame sequence, it is desired to define these 4 different groups/clusters of motion activities in the sequence.

To extract the motion features from this sequence of 1200 frames, the video is divided into short video clips of 10 frames/clip; i.e. 120 video clips. Each frame in each video clip is further divided into 2×2 blocks, where the sequence of blocks in the video clip form a

[3] `http://en.wikipedia.org/wiki/File:Illusion_movie.ogg`

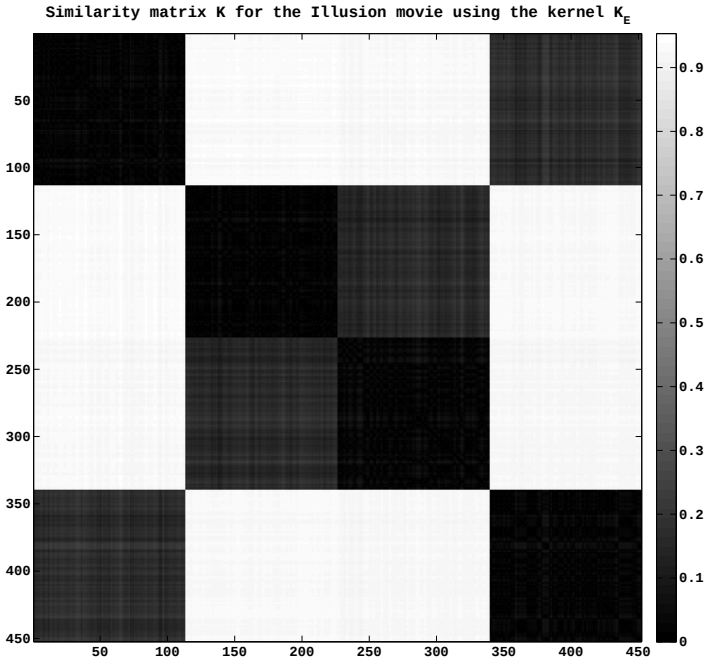


Fig. 6.6 The similarity matrix $\mathbf{K}$ for the illusion video sequence. Note the 4 block structures along the diagonal of the matrix, indicating the 4 different motion patterns in the data.

spatio-temporal unit known as a *cuboid*. That is, for 120 clips and 2×2 blocks, there are 480 cuboids. The motion information in each block is encoded using the m-bins histogram of gradient orientations as described in the previous section, where m is set to 8.

Due to the steady periodic motion of the illusion movie, the covariance matrices for all SOVs (or all cuboids) are equal, and hence they carry no additional information for clustering. Therefore, only the mean vector $\boldsymbol{\mu}$ of each SOV is computed as a representation for the SOV. This resulted in a total of 480 motion descriptor vectors with dimensionality $p = 8$, which describe the motion in the illusion sequence.

We apply the standard spectral clustering (SC) algorithm on the 480 motion descriptor vectors, where the exponential kernel K_E is used as a measure of similarity between the descriptor vectors. Figure 6.6 shows the resulting similarity (or affinity) matrix $\mathbf{K}$ for the descriptor vectors using K_E. It can be seen that the matrix has 4 clear block structures along the diagonal which is equivalent to the number of clusters in the data, or the number

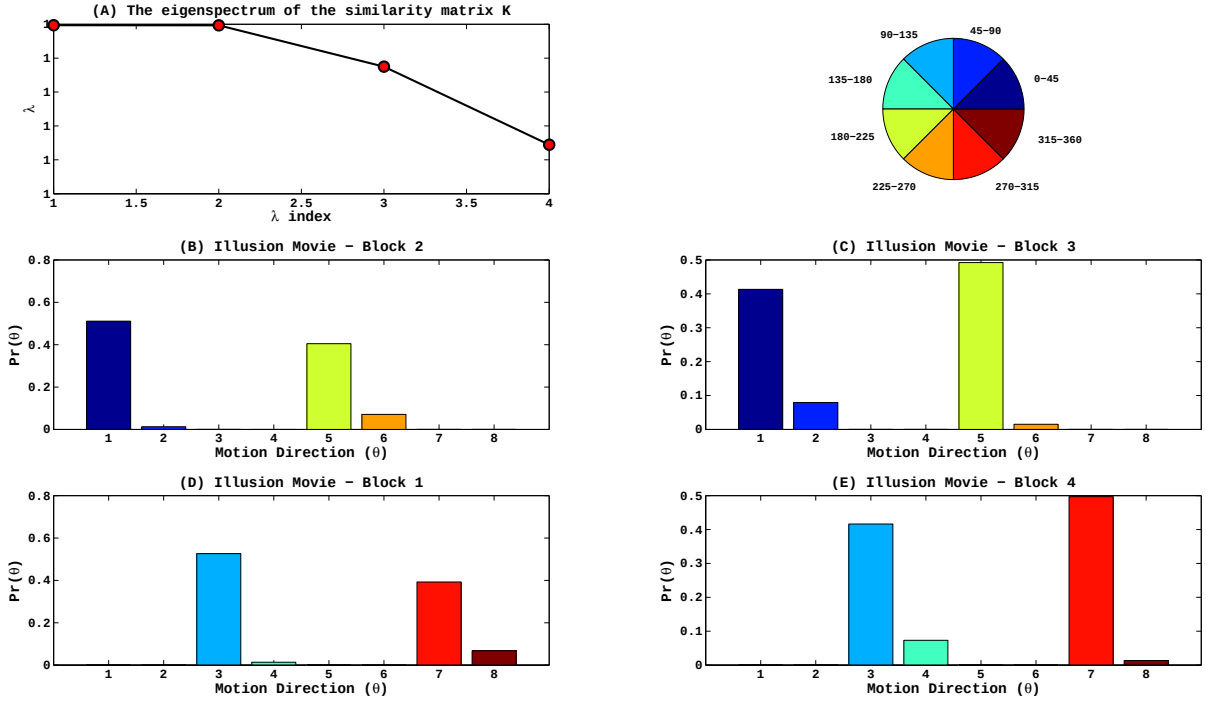


Fig. 6.7 (A) The eigen-spectrum of the similarity matrix **K** in Figure 6.6. (B) The mean histogram of cluster one corresponds the main motion directions in block 2 which are $(0^\circ - 45^\circ)$ and $(180^\circ - 225^\circ)$. These two different orientations for the motion in that particular block form what is called the *motion pattern*. (C) The mean histogram of cluster two corresponds to the main motion directions in block 3 which are $(180^\circ - 225^\circ)$ and $(0^\circ - 45^\circ)$. (D) The mean histogram of cluster three corresponds to the main motion directions in block 1 which are $(90^\circ - 135^\circ)$ and $(270^\circ - 335^\circ)$. (E) The mean histogram of cluster four corresponds to the main motion directions in block 4 which are $(270^\circ - 335^\circ)$ and $(90^\circ - 135^\circ)$.

of motion patterns that exist in the illusion video sequence. Further, Figure 6.7 A shows the eigenvalues of the affinity matrix **K**, in which there are 4 eigenvalues equal to one. This is a clear indication that the data has exactly 4 connected components, equivalently 4 clusters [169], that correspond to the 4 different motion patterns in the data.

To visualize the four motion patterns in the illusion video sequence, the mean histogram of each cluster is shown in Figures 6.7 B, (6.7(C)), (6.7(D)), and (6.7(E)). Note that due to the random initialization for the centres in the k-Means clustering step of SC, the cluster assignments are not in one to one correspondence with the block numbers in Figure 6.5(a).

It can be seen that SC well captured the 4 different motions patterns in the illusion video sequence. In a similar fashion, it is desired to extend the analysis developed above to more complex videos sequences with different types of human actions.

6.4.4 Experimental setting III : Clustering complex human motion

To generalize the analysis developed above for videos with complex human motion, it is essential to have a suitable data set for this purpose. To this end, we create a data set of 100 video sequences, each with 6 different human actions from the KTH data set for human action recognition. For each subject, the 6 video clips from one scenario are concatenated to form one long video sequence. The length of each concatenated sequence is around 120 seconds, with a frame rate of 25 fps. In order to validate any learning algorithm applied on the new video sequences, each frame in the video is labelled by the type of action it contains. Since there are 4 scenarios for each of the 25 subjects, this concatenation resulted in a total number of video sequences $n = 25 \times 4 = 100$.

To extract the motion features from one video sequence D_i, each video is divided into short video clips, with typically 20, 25, 30, and 35 frames/clip. Each frame in each clip is divided into $h \times w$ blocks – typically 3×3 and 4×4 – and the motion in each block is encoded by the m-bins histogram of gradient orientations. In all the experiments, m is set to 4 and 8 bins. The histograms of all blocks in one frame are concatenated to form one vector of dimensionality $p = m \times h \times w$. This makes each short video clip as one SOV, with 20, 25, 30, or 35 p-dimensional vectors in it.

The final representation of the new data set is as follows. There are $n = 100$ video sequences $\mathscr{D} = \{D_i\}_{i=1}^{n}$, and each video D_i is divided into short video clips which are represented as SOVs; $D_i = \{\mathcal{S}_1^i, \ldots, \mathcal{S}_j^i, \ldots, \mathcal{S}_{T_i}^i\}$, where T_i is the number of short clips in video D_i. Similar to the human action recognition experiments, each SOV $\mathcal{S}_j^i$ is represented as a Gaussian distribution $\mathcal{G}_j^i$ with a mean vector $\boldsymbol{\mu}_j^i$ and a regularized covariance matrix $\boldsymbol{\Sigma}_j^i$, where $1 \leq i \leq n$, and $1 \leq j \leq T_i$.

The reader should note that in the following experiments, motion clustering will be applied on each video sequence D_i independently, and not simultaneously on all the sequences.

Further, the accuracy of the clustering algorithm will be measured on each individual video sequence. The final performance of the clustering algorithm over the 100 sequences, will be the average of all the clustering accuracies.

To cluster the motion in one video sequence D_i, first, the Laplacian embedding algorithm described in Section 5.7 is used to embed all the Gaussians $\{\mathcal{G}_1^i \ldots, \mathcal{G}_{T_i}^i\}$ into 4 low dimensional subspaces $\mathbb{R}^{p_0}$ using the 4 relaxed kernels K_J, K_B, K_H, and $K_{B\mathcal{R}}$. Since the number of different actions in a single video is known *a priori* , the dimensionality of the embedding space is set to $p_0 = 6$. Next, the k-Means clustering algorithm is applied on the vectors in $\mathbb{R}^{p_0}$ to cluster the video clips into $k = 6$ clusters. The k-Means algorithm is initialized with 30 different initializations, and the clustering with the minimum distortion is selected as the final result of clustering.

Two clustering accuracy measures are used to assess the quality of clustering for one video sequence; the Hungarian score [147, 148], and the normalized mutual information (NMI) measure [170]. The final performance for the embedding process together with the k-Means algorithm is measured by taking the average Hungarian score and the average NMI measure over the 100 video sequences.

Analysis of the results

Tables 6.2, 6.3, 6.4, and 6.5 show the average clustering accuracies in the embedding spaces obtained from different Laplacian embeddings and different feature settings. Under each feature set, the difference between the embedding spaces is due to the kernel that defines the similarity matrix $\mathbf{K}$.

The first column in each table shows the number of frames per short video clip. The second column shows the accuracy for the standard spectral clustering (SC) algorithm – i.e. Laplacian embedding using the exponential kernel K_E – over the raw features representing each frame in the video sequence. Note that each frame is represented as a p-dimensional vector, where $p = m \times h \times w$. The third column, deonted by $K_E(\mu)$, shows the results of the standard SC algorithm over SOVs when each SOV $\mathcal{S}_j^i$ is represented only by the mean vector $\boldsymbol{\mu}_j^i$ – i.e. the covariance information in each SOV is not considered. Columns

Table 6.2 Average clustering accuracy (%), with standard deviation, over the 100 video sequences in the embedding spaces obtained by Laplacian embedding and the kernels K_E, $K_E(\mu)$, K_J, K_B, K_H, and $K_{B\mathcal{R}}$. The histogram of gradient orientations has the following setting: $m \times h \times w = 4 \times 3 \times 3$.

	$m \times h \times w = 4 \times 3 \times 3$					
	Hungarian score					
frames/clip	K_E	$K_E(\mu)$	K_J	K_B	K_H	$K_{B\mathcal{R}}$
20	37.2 (6.3)	55.9 (11.7)	53.0 (11.4)	53.2 (11.5)	57.6 (11.9)	**63.5 (12.6)**
25	37.2 (6.3)	58.7 (13.2)	54.5 (12.3)	54.8 (12.4)	60.8 (13.3)	**66.3 (13.8)**
30	37.2 (6.3)	60.0 (13.0)	58.1 (13.3)	62.6 (13.1)	62.6 (13.6)	**68.8 (14.0)**
35	37.2 (6.3)	63.2 (13.1)	60.7 (14.1)	60.5 (14.1)	66.0 (13.3)	**70.6 (13.1)**
	NMI measure					
frames/clip	K_E	$K_E(\mu)$	K_J	K_B	K_H	$K_{B\mathcal{R}}$
20	17.6 (7.0)	50.2 (13.8)	47.9 (13.2)	48.2 (13.3)	52.6 (14.0)	**56.5 (14.4)**
25	17.6 (7.0)	53.2 (14.4)	50.3 (13.2)	50.7 (13.3)	56.3 (13.9)	**59.8 (14.5)**
30	17.6 (7.0)	55.1 (14.3)	53.9 (13.7)	54.0 (13.7)	58.4 (13.7)	**62.1 (14.4)**
35	17.6 (7.0)	57.8 (13.6)	56.2 (13.9)	56.3 (13.8)	61.0 (13.2)	**63.6 (14.0)**

4 to column 7 show the results of k-Means clustering in the embedding spaces obtained by Laplacian embedding using the relaxed kernels K_J, K_B, K_H, and $K_{B\mathcal{R}}$. Note that each table shows the average clustering accuracies using two different measures, the Hungarian score and the NMI measure.

The four Tables show a consistent behaviour over the different features settings and he number of frames/clip. As expected, SC using K_E for the raw feature vectors, without grouping them into short clips, yields the lowest accuracies under both measures. Moving from a raw feature representation to sets of vectors via short video clips and using only the mean vector for each SOV with the standard SC algorithm, Column 3, boosts the accuracies by approximately 50%. When considering the covariance information for each SOV, and using the relaxed kernels K_J and K_B, the accuracies drop below those of $K_E(\mu)$. This is unlike the accuracies for K_H which are significantly higher than $K_E(\mu)$. This difference between K_J and K_B on one hand, and K_H on the other hand, shows the impact of adhering to metric properties on the embeddings obtained by Laplacian eigenmaps. This is further emphasized by the accuracies for $K_{B\mathcal{R}}$ which consistently outperforms K_H and all other kernels.

Table 6.3 Average clustering accuracy (%), with standard deviation, over the 100 video sequences in the embedding spaces obtained by Laplacian embedding and the kernels K_E, $K_E(\mu)$, K_J, K_B, K_H, and $K_{B\mathcal{R}}$. The histogram of gradient orientations has the following setting: $m \times h \times w = 4 \times 4 \times 4$.

	$m \times h \times w = 4 \times 4 \times 4$					
	Hungarian score					
frames/clip	K_E	$K_E(\mu)$	K_J	K_B	K_H	$K_{B\mathcal{R}}$
20	39.4 (6.6)	57.6 (12.1)	53.8 (12.0)	54.1 (12.2)	59.4 (12.7)	**65.5 (13.3)**
25	39.4 (6.6)	58.7 (12.1)	55.9 13.4)	56.4 (13.4)	63.5 (13.4)	**68.2 (13.6)**
30	39.4 (6.6)	60.4 (12.6)	58.6 13.5)	58.6 (13.7)	65.3 (13.4)	**69.8 (13.7)**
35	39.4 (6.6)	63.3 (13.6)	60.9 14.2)	60.6 (13.8)	68.0 (13.9)	**71.6 (13.1)**
	NMI measure					
frames/clip	K_E	$K_E(\mu)$	K_J	K_B	K_H	$K_{B\mathcal{R}}$
20	21.0 (8.0)	52.0 (13.8)	48.8 (13.6)	49.4 (13.8)	55.2 (14.1)	**59.4 (14.7)**
25	21.0 (8.0)	54.1 (13.7)	51.5 (13.6)	52.0 (13.4)	59.1 (13.3)	**62.2 (13.7)**
30	21.0 (8.0)	56.0 (13.4)	54.4 (13.7)	54.9 (13.6)	60.8 (13.3)	**63.7 (13.5)**
35	21.0 (8.0)	58.2 (13.2)	56.8 (13.8)	56.5 (13.7)	62.9 (12.6)	**65.2 (12.5)**

6.5 Discussion and Concluding Remarks

In this chapter I have proposed an unsupervised framework for embedding sets of vectors based on the Bhattacharyya-Riemannian metric $d_{B\mathcal{R}}$. Similar to previous ideas in the literature, the framework models each SOV as a multivariate Gaussian distribution, forming by that a non-empty set, or family of Gaussians $\mathscr{G}$. The set of Gaussians $\mathscr{G}$ and the metric $d_{B\mathcal{R}}$ define the metric space $(\mathscr{G}, d_{B\mathcal{R}})$ which is the dual perspective for the augmented space $\mathbb{X}$ introduced in the previous chapter. Therefore, unlike previous methods in the literature which rely on the dis(similarity) between the Gaussians to only learn a classifier for the SOVs, the proposed framework embeds the metric space $(\mathscr{G}, d_{B\mathcal{R}})$ as points in a low dimensional Euclidean space $\mathbb{R}^{p_0}$, which allows any learning algorithm to learn from the low dimensional points instead of the SOVs.

The spectral embedding step offers an implicit clustering for the SOVs based on the metric $d_{B\mathcal{R}}$. That is, the metric space $(\mathscr{G}, d_{B\mathcal{R}})$ reorganizes the proximity between SOVs based on $d_{B\mathcal{R}}$ which explicitly respects the geometry of $\mathbb{R}^p$ and $\mathbb{S}_{++}^{p\times p}$. This reorganization for the SOVs is reflected on the embedding and manifested by the Euclidean distance in $\mathbb{R}^{p_0}$. The spectral embedding step, has two additional advantages; 1) dimensionality reduc-

Table 6.4 Average clustering accuracy (%), with standard deviation, over the 100 video sequences in the embedding spaces obtained by Laplacian embedding and the kernels K_E, $K_E(\mu)$, K_J, K_B, K_H, and $K_{B\mathcal{R}}$. The histogram of gradient orientations has the following setting: $m \times h \times w = 8 \times 3 \times 3$.

	$m \times h \times w = 8 \times 3 \times 3$					
	Hungarian score					
frames/clip	K_E	$K_E(\mu)$	K_J	K_B	K_H	$K_{B\mathcal{R}}$
20	37.5 (6.9)	58.5 (11.1)	55.7 (11.2)	56.0 (10.9)	60.1 (11.5)	**65.1 (13.2)**
25	37.5 (6.9)	60.5 (12.3)	58.2 (12.0)	58.1 (11.9)	63.6 (13.1)	**69.6 (13.6)**
30	37.5 (6.9)	62.4 (12.0)	60.0 (12.7)	59.9 (12.6)	64.8 (12.9)	**70.3 (13.4)**
35	37.5 (6.9)	65.2 (13.2)	63.0 (13.3)	62.9 (13.3)	67.4 (13.1)	**71.8 (13.6)**
	NMI measure					
frames/clip	K_E	$K_E(\mu)$	K_J	K_B	K_H	$K_{B\mathcal{R}}$
20	17.8 (8.9)	53.3 (13.4)	51.2 (13.1)	51.5 (12.9)	55.5 (13.1)	**59.0 (14.1)**
25	17.8 (8.9)	55.6 (13.3)	54.0 (12.6)	53.9 (12.5)	59.0 (13.1)	**62.9 (13.9)**
30	17.8 (8.9)	58.0 (12.5)	56.0 (13.1)	56.1 (12.9)	60.6 (12.8)	**64.0 (13.5)**
35	17.8 (8.9)	60.0 (12.6)	58.3 (13.2)	58.3 (12.9)	62.3 (12.3)	**65.6 (13.0)**

tion which results in faster and more efficient hypothesis learning over the SOVs through their images in $\mathbb{R}^{p_0}$, and 2) generalization via the Nyström formula.

The metric space $(\mathscr{G}, d_{B\mathcal{R}})$ can be tuned using the regularization parameter γ, and the kernel parameter σ. If these parameters are jointly optimized with the parameters of the hypothesis learning algorithm in the embedding space, the final performance of hypothesis is guaranteed to improve. Similar to the augmented space $\mathbb{X}$, the proposed framework can be used in supervised and semi-supervised learning settings. Further, the proposed framework can be generalized to other distributions over SOVs, and combined with other divergence measures that adhere to metric properties such as the Jensen-Shannon divergence [171].

The different experiments on the KTH data set for human action recognition clearly shows the efficacy of the proposed framework and the metric $d_{B\mathcal{R}}$. The experiments used simple low level features to represent the motion in a video clip, the k-NN classifier and the k-Means clustering in the embedding space, which were sufficient to show that the proposed framework is promising when compared to the basic ideas of Kondor & Jebara [44] and Moreno *et al.* [45] . An immediate future research work in that direction is to replace the

Table 6.5 Average clustering accuracy (%), with standard deviation, over the 100 video sequences in the embedding spaces obtained by Laplacian embedding and the kernels K_E, $K_E(\mu)$, K_J, K_B, K_H, and $K_{B\mathcal{R}}$. The histogram of gradient orientations has the following setting: $m \times h \times w = 8 \times 4 \times 4$.

	$m \times h \times w = 8 \times 4 \times 4$					
	Hungarian score					
frames/clip	K_E	$K_E(\mu)$	K_J	K_B	K_H	$K_{B\mathcal{R}}$
20	39.0 (7.2)	58.1 (11.6)	54.0 (12.5)	54.6 (12.7)	60.8 (12.2)	**66.3 (12.7)**
25	39.0 (7.2)	61.1 (12.2)	57.7 (14.0)	57.7 (13.9)	64.7 (13.2)	**69.5 (13.2)**
30	39.0 (7.2)	62.8 (12.6)	59.5 (13.4)	59.5 (13.2)	66.3 (12.5)	**70.5 (12.6)**
35	39.0 (7.2)	65.7 (13.4)	59.5 (13.4)	59.5 (13.2)	66.3 (12.5)	**70.5 (12.6)**
	NMI measure					
frames/clip	K_E	$K_E(\mu)$	K_J	K_B	K_H	$K_{B\mathcal{R}}$
20	20.7 (9.9)	53.2 (13.2)	**73.7 (12.2)**	**73.6 (12.3)**	**73.4 (12.2)**	**73.7 (11.5)**
25	20.7 (9.9)	56.8 (12.8)	52.7 (14.4)	52.9 (14.2)	60.2 (13.0)	**63.7 (13.1)**
30	20.7 (9.9)	58.4 (12.6)	54.9 (13.4)	55.4 (13.0)	62.0 (12.3)	**65.1 (12.3)**
35	20.7 (9.9)	60.5 (12.5)	54.9 (13.4)	55.4 (13.0)	62.0 (12.3)	**65.1 (12.3)**

k-NN classifier with SVMs, and use other multimedia data sets to have comparisons with [44] and [45]. Another interesting direction to explore is to use more complex low level features such as the composite frequency features [168], or features based on the trend of space-time interest points [161] (and similar ideas) together with the proposed framework in applications such as learning motion patterns from surveillance cameras, crowd analysis, etc.

Chapter 7

Conclusions & Future Directions

In practice, there are plethora of instances in which one learns a hypothesis immediately from the data without considering any sort of metric learning nor dimensionality reduction for the data. There are also a plethora of instances in which a sort of preprocessing such as, applying PCA, LDA, whitening transformation, etc. is considered a good practice and encouraged on the grounds of achieving good performance. These situations mainly appear at the systems level in which hypothesis learning becomes one part of a bigger pattern recognition system. Such systems, at a smaller scale for instance, occur in very active areas such as object/scene recognition in computer vision, and equally in speech recognition. Indeed these systems achieve state-of-the-art results, however it usually comes with enormous (and appreciated) efforts in designing feature detectors/descriptors, training, fine tuning parameters, etc.

In simple terms, the main message of this thesis is that one can achieve better systems design with less complexity, and better performance, if hypothesis learning is tied to metric learning and/or dimensionality reduction (linear or nonlinear). This is not just being a good practice, or a good customary habit in a certain application domain. According to this study, hypothesis learning and metric learning should be one unit, and it is not optional to learn a hypothesis without learning the necessary metric for the data.

More formally, for a data set $\mathcal{D} = \{\mathbf{x}\}_{i=1}^{n}$, $\mathbf{x}_i \in \mathbb{R}^p$, instead of imposing $\mathbb{R}^p$ on $\mathcal{D}$, we would like to learn the metric space $(\mathbb{R}^{p_0}, \|\cdot\|_2)$, with $p_0 \ll p$, such that the Euclidean

distance reveals more about the structure and groupings in the data. Note also that with $p_0 \ll p$, as by products, we obtain a reduction in space complexity, and build lower capacity predictors that can improve generalization [67].

An interesting outcome from this research work is the question on whether the linear/nonlinear embedding process preserves the metric properties or not, and whether the embedding space is a metric space, a semi-metric space or any other space with different combinations of properties. For supervised learning, these issues might not constitute serious hazards. However for unsupervised learning, where no labels or side information are available for the data, these issues become of great importance. In unsupervised learning, it is not known *a priori* which points should be close or far away from each other. Therefore, to avoid any misleading results from the embedding process, one should confirm that it preserves all the metric properties.

In addition to the conclusions and future research direction at the end of Chapters 4, 5, and 7, I highlight here more general conclusions and future research questions that I will be interested to pursue. The Pareto discriminant analysis framework in Chapter 4, and in particular Equation (4.25), can be considered an instance from a more general model for linear dimensionality reduction:

$$(\mathbf{B}^*, \mathbf{w}^*) = \underset{\mathbf{B}\in\mathcal{R},\ \mathbf{w}\in\mathbb{R}^c}{\arg\max} \quad E(\mathbf{B}, \mathbf{w}),\ \text{s.t. } \mathbf{B}^\top\mathbf{B} = \mathbf{I},\ \mathbf{w}^\top\mathbf{1} = 1,\ \text{and } \mathbf{w} \succeq 0,$$

where

$$E(\mathbf{B}, \mathbf{w}) = \sum_{i=1}^{c} w_i \text{div}(\mathcal{G}_i, \mathcal{G}'_i; \mathbf{B}) - \lambda\|\mathbf{B}\|_\delta,$$

div is any divergence measure between probability distributions, $\lambda \succeq 0$ is a regularization parameter, and δ decides the type of norm used in the regularization. This optimization problem enforces an orthogonality constraint on $\mathbf{B}$, and includes a regularization term on $\mathbf{B}$ as well. Depending on the matrix norm type, $\mathbf{B}$ can be a low rank or a sparse (orthogonal) matrix. A similar variant can be obtained for Equation (4.26). The final solution will not only depend on δ, but as well as on the divergence measure used and its properties. Note that a loss term based on classification error can be added to this objective function,

which implies combining metric learning and hypothesis learning in one model. The main questions now are which divergence measures to use, which matrix norm to use, and more importantly, how to efficiently optimize this objective function?

The augmented space $\mathbb{X}$ in Chapter 5 was only explored here in the unsupervised learning setting, although it can be easily applied in supervised and semi-supervised learning settings. The advantage of $\mathbb{X}$ is that it captures the local structure and density around each point in the data set. This local information is manifested in the proposed Bhattachrayya-Riemann and Jeffreys-Riemann metrics for the space $\mathbb{X}$. The metric spaces $(\mathbb{X}, d_{B\mathcal{R}})$ and $(\mathbb{X}, d_{J\mathcal{R}})$ give a new meaning for the distance between points that is based on the local structure and the local density around each point. That is, two points are close or similar to each other, when they are physically close to each other in the input space, and the local structure and density around each point are very similar. This is unlike the Euclidean distance that does not take any of these aspects into consideration.

Another advantage of $\mathbb{X}$ is that it is adaptive, and can be tuned using the neighbourhood size m, and the regularization parameter γ which was set to 1 in all experiments. This implies that the metrics $d_{B\mathcal{R}}$ and $d_{J\mathcal{R}}$ can be tuned as well. If these parameters are jointly optimized with a hypothesis learning algorithm, then we obtain a flexible metric that adapts to the task under consideration.

Note that in spectral clustering, or Laplacian eigenmaps, the main input to the algorithm is the Euclidean distance between points on the data neighbourhood graph, originally constructed using Euclidean distance as well. If the data neighbourhood graph is built using the metrics $d_{B\mathcal{R}}$ or $d_{J\mathcal{R}}$, we obtain the experiments in Section 5.11, which show that $d_{B\mathcal{R}}$ or $d_{J\mathcal{R}}$ can better capture the similarity between points based on this new meaning for the distance between points. This significant improvement of the results encourages us to consider the topological stability of all manifold learning algorithm [71] when encapsulated with the augmented space $\mathbb{X}$. That is, a main question for future research work is whether the adaptive augmented space $\mathbb{X}$ can improve, or leverage, the topological stability of spectral manifold learning algorithms?

Appendix A

Preliminaries

Positive Definite and Positive Semi-Definite Functions

A scalar $x \in \mathbb{R}$ is positive definite (PD), or positive semi-definite (PSD), *if and only if* $x > 0$, or $x \geq 0$ respectively.

A real continuous function $f : \mathbb{R}^p \mapsto \mathbb{R}$, that is even : $f(-\mathbf{x}) = f(\mathbf{x})$, is said to be PD if [43] :

$$\sum_{i,j}^{n} f(\mathbf{x}_i - \mathbf{x}_j)\rho_i\rho_j > 0 \ ,$$

for arbitrary real ρ_i, and any n points $\{\mathbf{x}_i\}_{i=1}^n$. Similarly, an even function $f : \mathbb{R}^p \mapsto \mathbb{R}$ is said to be PSD if

$$\sum_{i,j}^{n} f(\mathbf{x}_i - \mathbf{x}_j)\rho_i\rho_j \geq 0 \ ,$$

for arbitrary real ρ_i, and any n points $\{\mathbf{x}_i\}_{i=1}^n$.

Mercer Kernels

A necessary and sufficient condition to guarantee that a symmetric similarity function K is a kernel function over the input space $\mathcal{X}$, is that K should be PSD as defined above. That

is, for the set $\mathcal{X}$ and for any set of real numbers $a_1, \ldots, a_n$, the function K must satisfy the following: $\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0$. This ensures the existence of a mapping $\boldsymbol{\phi} : \mathcal{X} \mapsto \mathcal{H}$, where $\mathcal{H}$ is a Hilbert space called the feature space, in which K turns into an inner product: $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \boldsymbol{\phi}(\mathbf{x}_i), \boldsymbol{\phi}(\mathbf{x}_j) \rangle$. Such symmetric PSD kernels are known as Mercer kernels [123]. Note that the difference between Mercer kernels and the definition of PSD function introduced above is in the extra property of symmetry for Mercer kernels. However, the main definition of PSD is identical in both cases.

Metric Spaces

A metric space [46, p. 3] is an ordered pair $(\mathcal{X}, d)$, where $\mathcal{X}$ is a non-empty abstract set (of any objects/elements whose nature is left unspecified), and d is a distance function, or a metric, defined as :

$$d : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R},$$

and $\forall\ a, b, c \in \mathcal{X}$, the following axioms hold : (i) $d(a, b) \geq 0$, (ii) $d(a, a) = 0$, (iii) $d(a, b) = 0$ iff $a = b$, (iv) Symmetry : $d(a, b) = d(b, a)$, and (v) The triangle inequality : $d(a, c) \leq d(a, b) + d(b, c)$.

A semi-metric distance satisfies Axioms (i), (ii) and (iv) only. That is, the triangle inequality need not hold for semi-metrics, and $d(a, b)$ can be zero for any a, b and $a \neq b$. For instance, the Euclidean distance $\|\mathbf{x} - \mathbf{y}\|_2$ in $\mathbb{R}^p$ is a metric, but $\|\mathbf{x} - \mathbf{y}\|_2^2$ is a semi-metric. Hence, $(\mathbb{R}^p, \|\cdot\|_2)$ is a metric space, while $(\mathbb{R}^p, \|\cdot\|_2^2)$ is a semi-metric space. Note that the definition of a metric space is independent from whether $\mathcal{X}$ is equipped with an inner product or not.

The generalized quadratic distance (GQD): $d(\mathbf{x}, \mathbf{y}; \mathbf{A}) = \sqrt{(\mathbf{x} - \mathbf{y})^\top \mathbf{A} (\mathbf{x} - \mathbf{y})}$, is a metric, where $\mathbf{A}$ is a square symmetric PD matrix, and $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$. Note that $d(\mathbf{x}, \mathbf{y}; \mathbf{A})^2$ is a semi-metric, and if $\mathbf{A}$ is not PD, then $d(\mathbf{x}, \mathbf{y}; \mathbf{A})$ is also a semi-metric.

Axioms (i) & (ii) produce the positive semi–definiteness of d, and hence metrics and semi-metrics are both PSD. This definition of positive semi-definiteness is only valid for metrics

and semi-metrics due to their axiomatic definition above, and can not be generalized to PSD functions formally introduced above.

Intrinsic Dimensionality

The intrinsic dimensionality of a data set is the number of free variables needed to represent the data without any loss of information. More formally, a data set $\mathcal{X} \subset \mathbb{R}^p$ is said to have an intrinsic dimensionality equal to p_0 if its elements lie entirely within a p_0-dimensional subspace of $\mathbb{R}^p$ [172].

References

[1] D. Newman, S. Hettich, C. Blake, and C. Merz, "UCI Repository of Machine Learning Databases," 1998. `www.ics.uci.edu/~mlearn/MLRepository.html`.

[2] T. Cover and P. Hart, "Nearest neighbour pattern classification," *IEEE Trans. on Information Theory*, vol. 13, pp. 21–27, 1967.

[3] J. Moody and C. Darken, "Fast learning in networks of locally tuned processing units," *Neural Computation*, vol. 1, pp. 281–294, 1989.

[4] T. Hastie, R. Tibshirani, and J. Friedman, eds., *Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer, 2001.

[5] F. Rosenblat, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 6, pp. 386–408, 1958.

[6] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford Press, England, 1995.

[7] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[8] N. Cristianin and J. Shawe-Taylor, *An Introduction to support vector machines and other kernel-based learning methods*. Cambridge Univ. Press, Cambridge, England, 2000.

[9] S. P. Loyd, "Least squares quantization in pcm," *IEEE Trans. on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[10] G. Lebanon, *Riemannian geometry and statistical machine learning*. PhD thesis, Carnegie Mellon University, 2004.

[11] E. Xing, A. Ng, M. Jordan, and S. Russell, "Distance metric learning with application to clustering with side–information," in *NIPS 15*, pp. 505–512, MIT Press, 2002.

[12] M. Schultz and T. Joachims, "Learning a distance metric from relative comparisons," in *NIPS 16*, MIT Press, 2004.

[13] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall, "Learning a Mahalanobis metric from equivalence constraints," *JMLR*, vol. 6, pp. 937–965, 2005.

[14] J. Goldberg and S. Roweis, "Neighborhood component analysis," in *NIPS 17*, pp. 513–520, MIT Press, 2005.

[15] A. Globerson, S. Roweis, G. Hinton, and R. Salakhutdinov, "Metric learning by collapsing classes," in *NIPS 18*, pp. 451–458, MIT Press, 2006.

[16] K. Weinberger, J. Blitzer, and L. Saul, "Distance metric learning for large margin nearest neighbour classification," in *NIPS 18*, pp. 1473–1480, MIT Press, 2006.

[17] S. Xiang, F. Nie, and C. Zhang, "Learning a Mahalanobis metric for data clustering and classification," *Pattern Recognition*, vol. 40, no. 12, pp. 3600–3612, 2008.

[18] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding (LLE)," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[19] J. Tenenbaum, V. de Silva, and J. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319–2323, November 2000.

[20] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for data representation," *Neural Computation*, vol. 15, pp. 1373–1396, 2003.

[21] D. Donoho and C. Grimes, "Hessian eigenmaps: Locally linear embedding techniques for high–dimensional data," *Proc. of National Academy of Sciences*, vol. 100, no. 10, pp. 5591–5596, 2003.

[22] Y. Bengio, O. Delalleau, N. L. Roux, J.-F. Paiement, P. Vincent, and M. Ouimet, "Learning eigenfunctions links spectral embedding and kernel PCA," *Neural Computation*, vol. 16, pp. 2197–2219, 2004.

[23] Z. Zhang and H. Zha, "Principal manifolds and nonlinear dimension reduction via local tangent space alignment," *SIAM J. of Scientific Computing*, vol. 26, pp. 313–338, 2004.

[24] K. Weinberger and L. Saul, "Unsupervised learning of image manifolds by semidefinite programming," in *IEEE Proc. of CVPR*, pp. 988–995, 2004.

[25] J. Shi and J. Malik, "Normalized cuts and image segmentation," in *IEEE Proc. of Int. Conference on Computer Vision and Pattern Recognition*, 1997.

[26] Y. Weiss, "Segmentation using eigenvectors: a unifying view," in *IEEE Proc. of ICCV*, pp. 975–982, 1999.

[27] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *NIPS 14*, pp. 849–856, MIT Press, 2002.

[28] K. T. Abou-Moustafa, F. De La Torre, and F. P. Ferrie, "Pareto discriminant analysis," in *IEEE Proc. 23rd Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[29] K. T. Abou-Moustafa, F. De La Torre, and F. P. Ferrie, "Designing a metric for the difference between two Gaussian densities," in *Advances in Intelligent and Soft Computing*, vol. 83, pp. 57 – 70, Springer, 2010.

[30] K. Abou-Moustafa, M. Shah, F. D. L. Torre, and F. Ferrie, "Relaxed exponenrial kernels for unsupervised learning," in *LNCS 6835, Pattern Recognition, Proc. of the 33rd DAGM Symposium*, pp. 335–344, Springer, 2011.

[31] K. Abou-Moustafa and F. Ferrie, "A framework for hypothesis learning over sets of vectors," in *Proc. of 9th SIGKDD Workshop on Mining and Learning with Graphs*, pp. 335–344, ACM, 2011.

[32] K. T. Abou-Moustafa and F. Ferrie, "The minimum volume ellipsoid metric," in *LNCS 4713, Pattern Recognition, Proc. of the 29th DAGM Symposium*, pp. 335–344, Springer, 2007.

[33] K. T. Abou-Moustafa and F. Ferrie, "Fast and regularized local metric for query–based operations," in *IEEE Proc. of the 19th Int. Conf. on Pattern Recognition (ICPR)*, 2008.

[34] K. T. Abou-Moustafa and F. Ferrie, "Local metric learning on manifolds with application to query–based operations," in *LNCS 5342, IAPR Proc. of Int. Workshop on Structural, Syntactic, and Statistical Pattern Recognition (SSSPR)*, pp. 872–883, Springer, 2008.

[35] K. T. Abou-Moustafa and F. Ferrie, "Regularized minimum volume ellipsoid metric for query–based operations," in *IEEE Proc. of the 7th Int. Conf. on Machine Learning and Applications (ICMLA)*, pp. 183–193, 2008.

[36] M. Meila and J. Shi, "A random walks view of spectral segmentation," in *Proc. of AI & Statistics*, 2001.

[37] C. Hillermeier, *Nonlinear multiobjective optimization*. Birkhäuser Verlag, 2001.

[38] M. Ehrgott, *Multicriteria Optimization*. Springer, 2005.

[39] G. Young and A. Householder, "Discussion of a set of points in terms of their mutual distances," *Psychometrika*, vol. 3, no. 1, pp. 19–22, 1938.

[40] J. Gower and P. Legendre, "Metric and Euclidean properties of dissimilarity coefficients," *Journal of Classification*, vol. 3, pp. 5–48, 1986.

[41] W. S. Torgerson, *Theory and Methods of Scaling*. John Wiley & Sons, New York, 1958.

[42] T. F. Cox and M. A. Cox, *Multidimensional Scaling*. Chapman and Hall, 2001.

[43] I. Schoenberg, "Metric spaces and positive definite functions," *Trans. of the American Mathematical Society*, vol. 44, no. 3, pp. 522–536, 1938.

[44] R. Kondor and T. Jebara, "A kernel between sets of vectors," in *ACM Proc. of ICML*, 2003.

[45] P. Moreno, P. Ho, and N. Vasconcelos, "A Kullback–Leibler divergence based kernel for svm classification in multimedia applications," in *NIPS 16*, 2003.

[46] E. Kreyszig, ed., *Introductory functional Analysis with Applications*. Wiley Classics Library, 1989.

[47] L. Bottou and V. Vapnik, "Local learning algorithms," *Neural Computation*, vol. 4, no. 6, pp. 888–900, 1992.

[48] W. Boothby, *An Introduction to Differentiable Manifolds and Riemannian Geometry*. Academic Press, 2003.

[49] H. Chang and D.-Y. Yeung, "Locally linear metric adaptation for semi–supervised clustering," in *ACM Proc. of ICML*, pp. 153–160, 2004.

[50] L. Yang, "Distance metric learning: A comprehensive review," tech. rep., Dept. of Computer Science and Engineering, Michigan State University, 2006.

[51] R. Short and K. Fukunaga, "The optimal distance measure for nearest neighbour classification," *IEEE Trans. on Information Theory*, vol. 27, no. 5, pp. 622–627, 1981.

[52] J. Friedman, "Flexible metric nearest neighbor classification," tech. rep., Department of Statistics, Stanford University, 1994.

[53] T. Hastie and R. Tibshirani, "Discriminant adaptive nearest neighbor classification," *IEEE Trans. PAMI*, vol. 18, no. 6, pp. 607–615, 1996.

[54] C. Domeniconi and D. Gunopulos, "Adaptive nearest neighbor classification using support vector machines," in *NIPS 14*, 2002.

[55] C. Domeniconi, J. Peng, and D. Gunopulos, "Locally adaptive metric nearest neighbor classification," *IEEE Trans. PAMI*, vol. 24, no. 9, pp. 1281–1285, 2002.

[56] J. Peng, D. Heisterkamp, and H. Dai, "Adaptive quasi–conformal kernel for nearest neighbour classification," *IEEE Trans. PAMI*, vol. 26, no. 5, pp. 656–661, 2004.

[57] H. Chang and D.-Y. Yeung, "Locally smooth metric learning with application to image retrieval," in *IEEE Proc. of ICCV*, pp. 1–7, 2007.

[58] S. Hoi, W. Liu, M. Lyu, and W.-Y. Ma, "Learning distance metrics with contextual constraints for image retrieval," in *IEEE Proc. of CVPR*, pp. 2072–2078, 2006.

[59] I. Tsang, P.-M. Cheung, and J. Kwok, "Kernel relevant component analysis for distance metric learning," in *Int. Joint Conf. on Neural Networks*, pp. 954–958, 2005.

[60] S. Shalev-Shwartz, Y. Singer, and A. Ng, "Online and batch learning of pseudo–metrics," in *ACM Proc. of ICML*, 2004.

[61] I. Tsang and J. Kwok, "Learning with idealized kernels," in *ACM Proc. of ICML*, pp. 400–407, 2003.

[62] L. Saul and S. Roweis, "Think globally, fit locally: Unsupervised learning of low dimensional manifolds," *JMLR*, vol. 4, pp. 119–155, 2003.

[63] I. T. Jolliffe, *Principle Component Analysis*. Springer–Verlag, New York, 1986.

[64] R. Shepard, "The analysis of proximities: Multidimensional scaling with an unknown distance function," *Psychometrika*, vol. 27, pp. 219–246, 1962.

[65] J. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, pp. 1–27, 1964.

[66] J. Kruskal, "Nonmetric multidimensional scaling: A numerical method," *Psychometrika*, vol. 29, pp. 115–129, 1964.

[67] Y. Bengio, O. Delalleau, N. L. Roux, J.-F. Paiement, P. Vincent, and M. Ouimet, *Features Extraction. Foundations and Applications*, ch. Spectral Dimensionality Reduction. Springer, 2006.

[68] D. Zhao, "Formulating lle using alignment technique," *Pattern Recognition*, vol. 39, no. 11, pp. 2233 – 2235, 2006.

[69] F. Chung, *Spectral Graph Theory. Regional Conference Series in Mathematics (82)*. American Mathematical Society, 1997.

[70] A. Rahimi, *Learning to transform time series with a few examples*. PhD thesis, Massachusetts Institute of Technology, 2005.

[71] M. Balasubramanian, Mukund, and E. Schwartz, "The Isomap algorithm and topological stability," *Science*, vol. 295, no. 5552, p. 7, 2002.

[72] R. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, pp. 179–188, 1936.

[73] K. Fukunaga, ed., *Introduction to Statistical Pattern Recognition*. Academic Press, 1972.

[74] C. R. Rao, *Linear Statistical Inference and its Applications*. John Wiley & Sons, New York, 1965.

[75] O. Hamsici and A. Martinez, "Bayes optimality in linear discriminant analysis," *IEEE Trans. PAMI*, vol. 30, pp. 647–657, Apr 2008.

[76] N. A. Campbell, "Canonical variate analysis – a general formulation," *Australian J. of Statistics*, vol. 26, pp. 86–96, 1984.

[77] T. Hastie, R. Tibshirani, and B. Andreas, "Flexible discriminant and mixture models," in *Statistics and neural networks: advances at the interface*, pp. 1–23, 1999.

[78] N. Kumar and A. Andreou, "Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition," *Speech Communication*, vol. 26, no. 4, pp. 283 – 297, 1998.

[79] M. Zhu and T. Hastie, "Feature extraction for nonparametric discriminant analysis," *J. of Computational and Graphical Statistics*, vol. 12, no. 1, pp. 101–120, 2003.

[80] G. Baudat and F. Anouar, "Generalized discriminant analysis using a kernel approach," *Neural Computation*, vol. 12, no. 10, pp. 2385–2404, 2000.

[81] S. Mika, A. Smola, and B. Schlkopf, "An improved training algorithm for kernel Fisher discriminants," in *Proc. of AISTATS*, pp. 98–104, 2001.

[82] M. Loog, R. Duin, and R. Haeb-Umbach, "Multiclass linear dimension reduction by weighted pairwise Fisher criteria," *IEEE Trans. PAMI*, vol. 23, pp. 762–766, Jul 2001.

[83] J. Lu, K. Plataniotis, and A. Venetsanopoulos, "Face recognition using LDA–based algorithms," *IEEE Trans. Neural Networks*, vol. 14, pp. 195–200, Jan 2003.

[84] M. Zhu and A. Martinez, "Subclass discriminant analysis," *IEEE Trans. PAMI*, vol. 28, pp. 1274–1286, Aug 2006.

[85] S. Zhang and T. Sim, "Discriminant subspace analysis: A Fukunaga–Koontz approach," *IEEE Trans. PAMI*, vol. 29, pp. 1732–1745, Oct 2007.

[86] F. De La Torre and T. Kanade, "Multimodal oriented discriminant analysis," in *ACM Proc. of ICML*, pp. 177–184, 2005.

[87] D. Tao, X. Li, X. Wu, and S. Maybank, "General averaged divergence analysis," in *IEEE Proc. of Seventh ICDM*, pp. 302–311, 2007.

[88] R. Chen, "Solution of MINIMAX problems using equivalent differentiable functions," *Computers & Mathematics with Applications*, vol. 12, pp. 1165–1169, 1985.

[89] Y. Sawaragi, H. Nakayam, and T. Tanino, eds., *Theory of Multiobjective Optimization*. Academic Press, 1985.

[90] T. Hastie and R. Tibshirani, "Discriminant analysis by Gaussian mixtures," *J. of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 155–176, 1996.

[91] J. Tou and R. Heyden, "Some approaches to optimum feature selection," *Computer and Information Sciences*, vol. II, pp. 57–89, 1967.

[92] S. Kullback, *Information Theory and Statistics – Dover Edition*. Dover, New York, 1997.

[93] G. Saon, M. Padmanabhan, R. Gopinath, and S. Chen, "Maximum likelihood discriminant feature spaces.," in *ICASSP*, 2000.

[94] J. Friedman, "Regularized discriminant analysis," *J. of the American Statistical Assoc.*, vol. 84, no. 405, pp. 165–175, 1989.

[95] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. Fisherfaces: recognition using class specific linear projection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 711 –720, 7 1997.

[96] L. Chen, H. Liao, M. Ko, J. Lin, and G. Yu, "A new LDA–based face recognition system which can solve the small sample size problem," *Pattern Recognition*, vol. 33, no. 10, pp. 1713 – 1726, 2000.

[97] H. Yu and J. Yang, "A direct LDA algorithm for high–dimensional data – with application to face recognition," *Pattern Recognition*, vol. 34, pp. 2067–2070, 2001.

[98] P. Howland and H. Park, "Generalizing discriminant analysis using the generalized singular value decomposition," *IEEE Trans. PAMI*, vol. 26, pp. 995–1006, Aug 2004.

[99] J. Ye and Q. Li, "A two–stage linear discriminant analysis via QR–decomposition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 929–941, Jun 2005.

[100] R. Lotlikar and R. Kothari, "Fractional–step dimensionality reduction," *IEEE Trans. PAMI*, vol. 22, pp. 623–627, Jun 2000.

[101] Y. Zhang and D.-Y. Yeung, "Worst-case linear discriminant analysis," in *NIPS 23*, pp. 2568–2576, 2010.

[102] Y. Yu, J. Jiang, and L. Zhang, "Distance metric learning by minimal distance maximization," *Pattern Recognition*, pp. 639–649, 2011.

[103] W. Bian and D. Tao, "Max-min distance analysis by using sequential SDP relaxation for dimension reduction," *IEEE Trans. PAMI*, vol. 33, pp. 1037–1050, may 2011.

[104] J. C. Principe, *Information Theoretic Learning. Renyi's entropy and kernel perspectives*. Springer, 2010.

[105] K. Torkkola, "Feature extraction by non–parametric mutual information maximization," *JMLR*, vol. 3, pp. 1415–1438, 2003.

[106] A. Renyi, "On measures of entropy and information," in *Proc. of Fourth Berkeley Symp. on Math. Statist. and Prob.*, vol. 1, pp. 547–561, 1961.

[107] S. Kaski and J. Peltonen, "Informative discriminant analysis," in *ACM Proc. of the 20th Int. Conf. on Machine Learning (ICML)*, pp. 329–336, 2003.

[108] L. M. Bregman, "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming," *USSR Computational Mathematics and Mathematical Physics*, vol. 7, no. 3, pp. 200 – 217, 1967.

[109] L. Zadeh, "Optimality and non–scalar–valued performance criteria," *IEEE Trans. on Automatic Control*, vol. 8, pp. 59 – 60, Jan. 1963.

[110] C. Ding and T. Li, "Adaptive dimension reduction using discriminant analysis and K–means clustering," in *ACM Proc. of the 24th ICML*, 2007.

[111] S. Schffler, R. Schultz, and K. Weinzierl, "Stochastic method for the solution of unconstrained vector optimization problems," *Journal of Optimization Theory and Applications*, vol. 114, no. 1, pp. 209–222, 2002.

[112] T. Sim and T. Kanade, "Combining models and exemplars for face recognition: An illuminating example," in *CVPR Workshop on models vs. exemplars in computer vision*, 2001.

[113] Y. LeCun, "The MNIST database of handwritten digits," 1998. `http://yann.lecun.com/exdb/mnist/`.

[114] "United States Postal Service (USPS) data set," 1998. `www.gaussianprocess.org/gpml/data/`.

[115] A. Georghiades, P. Belhumeur, and D. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. PAMI*, vol. 23, no. 6, pp. 643–660, 2001.

[116] B. Leibe and B. Schiele, "Analyzing appearance and contour based methods for object categorization," in *IEEE Proc. of CVPR*, vol. 2, pp. 409–15, 2003.

[117] M. Zhu and A. Martinez, "Pruning noisy bases in discriminant analysis," *IEEE Trans. Neural Networks*, vol. 19, no. 1, pp. 148–157, 2008.

[118] V. N. Vapnik, *Statistical Learning Theory*. John Wiley & Sons, Sussex, England, 1998.

[119] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[120] P. Vincent and Y. Bengio, "Manifold Parzen windows," in *NIPS 15*, pp. 825–832, MIT Press, 2003.

[121] M. Brand, "Charting a manifold," in *Advances in NIPS 15*, pp. 961–968, MIT Press, 2003.

[122] T. D. G. Shakhnarovich and P. Indyk, *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. The MIT Press, 2006.

[123] J. Mercer, "Functions of positive and negative type, and their connection with the theory of integral equations," *Philosophical Trans. of the Royal Society of London. Series A*, vol. 209, pp. 415–446, 1909.

[124] S. M. Ali and S. D. Silvey, "A general class of coefficients of divergence of one distribution from another," *J. of the Royal Statistical Society.* Series B, vol. 28, no. 1, pp. 131–142, 1966.

[125] I. Csiszár, "Information–type measures of difference of probability distributions and indirect observations," *Studia Scientiarium Mathematicarum Hungarica*, vol. 2, pp. 299–318, 1967.

[126] M. Hein and O. bousquet, "Hilbertian metrics and positive definite kernels on probability measures," in *Proc. of AISTATS*, pp. 136–143, 2005.

[127] M. Cuturi, K. Fukumizu, and J.-P. Vert, "Semigroup kernels on measures," *JMLR*, vol. 6, pp. 1169–1198, 2005.

[128] A. Martins, N. Smith, E. Xing, P. Aguiar, and M. Figueiredo, "Nonextensive information theoretic kernels on measures," *JMLR*, vol. 10, pp. 935–975, 2009.

[129] T. Kailath, "The divergence and Bhattacharyya distance measures in signal selection," *IEEE Trans. on Communication Technology*, vol. 15, no. 1, pp. 52–60, 1967.

[130] S.-I. Amari and H. Nagaoka, *Methods of Information Geometry*. AMS Translations of Mathematical Monographs, Vol. 191, Oxford University Press, 2000.

[131] C. R. Rao, "Information and the accuracy attainable in the estimation of statistical parameters," *Bull. Calcutta Math. Soc.*, no. 58, pp. 326–337, 1945.

[132] C. Atkinson and A. F. S. Mitchell, "Rao's distance measure," *The Indian J. of Statistics, Series A*, vol. 43, no. 3, pp. 345–365, 1945.

[133] W. Förstner and B. Moonen, "A metric for covariance matrices," tech. rep., Dept. of Geodesy and Geo–Informatics, Stuttgart University, 1999.

[134] X. Pennec, P. Fillard, and N. Ayache, "A Riemannian Framework for Tensor Computing," Tech. Rep. RR-5255, INRIA, 7 2004.

[135] O. Tuzel, F. Porikli, and P. Meer, "Pedestrian detection via classification on Riemannian manifolds," *IEEE Trans. PAMI*, vol. 30, no. 10, pp. 1713–1727, 2008.

[136] X. Pennec, P. Fillard, and N. Ayache, "A Riemannian framework for tensor computing," *Int. J. Computer Vision*, vol. 66, no. 1, pp. 41–66, 2006.

[137] E. Levina and P. Bickel, "Maximum likelihood estimation of intrinsic dimension," in *NIPS 17*, pp. 777–784, MIT Press, 2005.

[138] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola, "On kernel-target alignment," in *NIPS 14*, MIT Press, 2002.

[139] X. He and P. Niyogi, "Locality preserving projections," pp. 153–160, 2003.

[140] C. Baker, ed., *The Numerical Treatment of Integral Equations*. Clarendon Press, Oxford, 1977.

[141] V. Roth, J. Laub, and J. Buhmann, "Optimal cluster preserving embedding of nonmetric proximity data," *IEEE Trans. PAMI*, vol. 25, no. 12, pp. 1540–1551, 2003.

[142] C. K. Williams, "On a connection between kernel PCA and metric multidimensional scaling," *Machine Learning*, vol. 46, pp. 11–19, 2002.

[143] P. Simard, Y. LeCun, J. Denker, and B. Victorri, "Transformation invariance in pattern recognition – tangent distance and tangent propagation," in *LNCS 1524, Neural Networks: Tricks of the trade*, pp. 239–274, Springer, 1998.

[144] A. Ghodsi, J. Huang, F. Southey, and D. Schuurmans, "Tangent-corrected embedding," in *IEEE Proc. of CVPR*, pp. 518–525, 2005.

[145] P. Vincent and Y. Bengio, "K–Local hyperplane and convex distance nearest neighbor algorithms," in *NIPS 14*, pp. 985–992, MIT Press, 2002.

[146] L. Wolf and A. Shashua, "Learning over sets using kernel principal angles," *JMLR*, vol. 4, pp. 913–931, Dec. 2003.

[147] H. Zha, C. Ding, M. Gu, X. He, and H. Simon, "Spectral relaxation for k–means clustering," in *NIPS 13*, MIT Press, 2001.

[148] F. D. L. Torre and T. Kanade, "Discriminative cluster analysis," in *ACM Proc. of ICML*, pp. 241–248, 2006.

[149] R. Coifman and S. Lafon, "Diffusion maps," *Applied and Computational Harmonic Analysis*, vol. 21, pp. 5–30, July 2006.

[150] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," *Int. J. of Computer Vision*, vol. 73, pp. 213–238, June 2007.

[151] M. Hu, S. Ali, and M. Shah, "Detecting global motion patterns in complex video," in *IEEE Proc. of ICPR*, 2008.

[152] L. Kratz and K. Nishino, "Anomaly detection in extremely crowded scenes using spatio–temporal motion pattern models," in *IEEE Proc. of CVPR*, pp. 1446 – 1453, 2009.

[153] Y. Yang, J. Liu, and M. Shah, "Video scene understanding using multi–scale analysis," in *IEEE Proc. of ICCV*, 2009.

[154] I. Saleemi, L. Hartung, and M. Shah, "Scene understanding by statistical modelling of motion patterns," in *IEEE Proc. of CVPR*, pp. 2069 – 2076, 2010.

[155] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition*. Prentice–Hall, Inc., 1993.

[156] T. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," in *NIPS 11*, pp. 487–493, MIT Press, 1999.

[157] H. Sakeo and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[158] Y. Rubner, C. Tomasi, and L. Guibas, "The earth mover's distance as a metric for image retrieval," *Int. J. of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.

[159] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *Ann. Math. Stat.*, vol. 37, pp. 1554–1563, 1966.

[160] L. E. Baum and T. Petrie, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. of Math. and Stat.*, vol. 41, no. 1, pp. 164–171, 1970.

[161] C. Schldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," in *In Proc. of ICPR*, pp. 32–36, 2004.

[162] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. of IJCAI*, pp. 674–679, 1981.

[163] Z. Li and Y.-P. Tan, "Event–based analysis of video," in *IEEE Proc. of CVPR*, pp. 1063–6919, 2001.

[164] I. Laptev, M. Marszaek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *IEEE Proc. of CVPR*, 2008.

[165] J. Shi and J. Malik, "Motion segmentation and tracking using normalized cuts," in *IEEE Proc. of ICCV*, pp. 1154–1160, 1998.

[166] H. Zhong and J. Shi, "Detecting unusual activity in video," in *IEEE Proc. of CVPR*, pp. 1161–1166, 2004.

[167] L. Kratz and K. Nishino, "Tracking with local spatio–temporal motion patterns in extremely crowded scenes," in *IEEE Proc. of CVPR*, pp. 693 – 700, 2010.

[168] R. Dosil, X. Fdez-Vidal, and M. Pardo, "Motion representation using composite energy features," *Pattern Recognition*, vol. 41, no. 3, pp. 1110–1123, 2008.

[169] U. v. Luxburg, "A tutotrial on spectral clustering," Tech. Rep. TR–149, Max Plank Institute for Biological Cybernetics, 2006.

[170] A. Strehl and J. Ghosh, "Cluster ensembles – A knowledge reuse framework for combining multiple partitions," *JMLR*, vol. 3, pp. 583–617, 2002.

[171] B. Fuglede and F. Topsoe, "Jensen-shannon divergence and hilbert space embedding," in *Proc. of the International Symposium on Information Theory*, 2004.

[172] K. Fukunaga, "Intrinsic dimensionality extraction," in *Classification Pattern Recognition and Reduction of Dimensionality* (P. Krishnaiah and L. Kanal, eds.), vol. 2 of *Handbook of Statistics*, pp. 347 – 360, Elsevier, 1982.