

KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
Kasteelpark Arenberg 10, 3001 Leuven (Heverlee)

SEMI-SUPERVISED LEARNING BASED ON KERNEL METHODS AND GRAPH CUT ALGORITHMS

Promotor:
Prof. dr. ir. B. De Moor

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de toegepaste wetenschappen

door

Tijl De Bie

May 2005

KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
Kasteelpark Arenberg 10, 3001 Leuven (Heverlee)

SEMI-SUPERVISED LEARNING BASED ON KERNEL METHODS AND GRAPH CUT ALGORITHMS

Jury:
Prof. dr. ir. H. Neuckermans, voorzitter
Prof. dr. ir. B. De Moor, promotor
Prof. dr. ir. A. Bultheel
Prof. dr. ir. L. De Lathauwer (ETIS, ENSEA)
Prof. dr. ir. K. Marchal
Prof. dr. ir. E. Nyssen (ETRO, VUB)
Prof. dr. ir. J. Suykens

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de toegepaste wetenschappen

door

Tijl De Bie

U.D.C. 681.3*I26 May 2005

© Katholieke Universiteit Leuven – Faculteit Toegepaste Wetenschappen
Arenbergkasteel, B-3001 Heverlee (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotocopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm or any other means without written permission from the publisher.

D/2005/7515/41

ISBN 90-5682-607-7

Voorwoord

Het heeft voeten in de aarde gehad. Het was een zware bevalling. De weg was bezaaid met putten en dalen. Er zijn hoogtes en laagtes geweest, teleurstellingen en kleine victorietjes. Ja, ik heb zelfs zalen doen vollopen... en wellicht ook zalen doen leeglopen. En als ik met deze thesis één record zal gevestigd hebben, dan is het vast en zeker wat betreft het aantal mensen aan wie ik dankbaarheid verschuldigd ben.

De onderzoeksgroep SISTA, later deel van SCD, is een fantastische omgeving om een doctoraat voor te bereiden. Het is mijn promotor Bart De Moor die mij daartoe de kans gegeven heeft. Maar het bleef niet bij deze ene kans: wanneer het mij zinde, heeft hij mij gesteund om ook even buiten de groepsmuren te kijken. En bovenal, ik geloof en beken dat het een ware krachttoer van flexibiliteit en inlevingsvermogen moet geweest zijn om mijn fantasietjes te kunnen volgen. Voor al deze dingen en veel meer, Bart, kan ik je niet genoeg bedanken.

Een groep zo groot als SCD/SISTA heeft sterke steunpilaren nodig. Eén van hen is Johan Suykens, voor wie geen vraag te veel is, geen probleem te groot. Johan, ik bewonder je scherpe inzichten en je onbaatzuchtige bereidwilligheid waarmee je ze telkens opnieuw met mij wilde delen. Ik dank ook Joos Vandewalle, voor zijn werk achter de schermen, en vóór de schermen wanneer het nodig is, voor zijn geduld en begrip als groeps- en departementsvoorzitter. Bij het begin van mijn doctoraat heb ik kunnen rekenen op de hulp en het enthousiasme van Lieven De Lathauwer. Lieven, heel veel dank voor je steun en begeestering. De laatste twee, drie jaar is mijn interesse enigszins verschoven naar de bioinformatica. Zonder Frank De Smet, Yves Moreau en vooral Kathleen Marchal was die overgang voor mij onmogelijk geweest. Kathleen, jij slaagt erin als geen andere de overbrugging te maken tussen biologie en algoritmen. Zowat alles wat ik over bioinformatica heb geleerd, heb jij me bijgebracht. André Barbé ben ik erkentelijk voor de toffe samenwerking rond de oefenzittingen van het vak niet-lineaire en complexe systemen, vier jaar lang. Als laatste steunpilaar van SCD/SISTA wil ik Jan Willems van harte bedanken voor zijn steun in sommige van mijn (misschien) naief-enthousiaste initiatieven. Jan, zonder jou had ik de leesgroep al na twee keer opgegeven. Ik heb veel meer van jou geleerd dan je misschien vermoedt.

Een onderzoeksgroep zou geen groep zijn zonder haar leden. Ik heb veel plezier beleefd aan de compagnie in het 'aquarium', in beide torens en in blok F. Aan heel de groep, hartelijk dank voor de toffe sfeer! Ik wil van deze gelegenheid gebruik maken om Koenraad Audenaert en Frank Verstraete speciaal te bedanken voor het boeiende en vermakelijke eerste jaar (ik denk met een glimlach terug aan al de stoere en waar gebeurde verhalen), ook Luc Hoegaerts, voor enkele interessante discussies die me geholpen hebben bij het schrijven van hoofdstuk 2, en Pieter Monsieurs, Kristof Engelen en Karen Lemmens voor hun onontbeerlijke bijdragen aan hoofdstuk 8.

Een speciaal woordje zou ik willen richten aan de mensen die alles in goede banen leiden: Ida Tassens, Pela Noe, Ilse Pardon, Lut Vanderbracht en Bart Motmans. Ik geloof dat ik meermaals een vervelende administratieve vraag heb gesteld waarop ik het antwoord al lang had moeten weten, en toch kon ik telkens op jullie bereidwillige en efficiënte hulp rekenen. Ook de mensen van de systeemgroep (en de onderzoekers-vrijwilligers) die zich bezighouden met de informatica-infrastructuur verdienen een grote pluim: er zijn zeer weinig groepen die in dergelijke mate op hun computerinfrastructuur kunnen betrouwen. Onvoorstelbaar wat jullie daar vanuit de kelder realiseren.

Without a single doubt, there would be no such thing as this PhD thesis without Nello Cristianini. Nello, you had the most profound impact possible on my research and on my research life. I consider it an enormous honor and pleasure to regard you as a mentor, and as a friend. The times I spent with you at U.C.Davis and U.C.Berkeley were unforgettable, and all the things I learned from you can impossibly be reflected in a text as long as this thesis. You taught me that a lunch consisting of just a coke is more than sufficient when working on a cool project (after all, all a human needs is sugar and water); that the antropic principle can be an explanation for our pattern-finding behavior; about how we are targeted by weapons of mass deception and what we can do about it; that Etruscan is written right to left; and many more fun and interesting things...

I am also indebted to Laurent El Ghaoui, who has been such a friendly and laid-back host during my visit to U.C.Berkeley. Laurent, I had a fantastic time in your group. While in U.C.Berkeley, also Michael Jordan gave me the chance to attend his group meetings, which were a revealing experience to me. Mike, I admire the professional yet personal way you head your research group. I really appreciate you gave me this opportunity much more than I ever said in words. It changed my (view on) research in a drastic way.

During my international visits I met many other researchers that have taught me a lot: many thanks to Peter Bartlett, Aurore Delaigle, Matt Hahn, Michi Momma, Chi Nguyen, Long Nguyen, Bill Noble, Martin Plenio, Wolfgang Polonik, Roman Rosipal, John Shawe-Taylor, and many others.

Mijn lange studieverblijf in Berkeley zou nooit zijn doorgegaan zonder de hulp en begeestering van een goede vriend, Gert Lanckriet. Gert, bij de gelegenheid van het neerleggen van mijn doctoraatsthesis, denk ik dat ik je weer

heel wat Belgisch bier verschuldigd ben. Bij mijn eerstvolgende bezoek aan de Amerikaanse woestijn beloof ik dan ook plechtig mijn bagage nog eens met overgewicht in te checken.

Een andere goede vriend, Pieter Abbeel, heeft met zijn enthousiasme een heel motiverende invloed gehad op mijn onderzoek. Pieter, ik hoop dat we, op een dag, samen zullen bewijzen of P=NP of niet (of dat het onbeslisbaar is, natuurlijk)!

De kwaliteit en accuraatheid van deze thesis zijn in belangrijke mate toegenomen dankzij de nauwkeurige lezing en de suggesties van Adhemar Bultheel, Lieven De Lathauwer, Johan Suykens en mijn promotor Bart De Moor.

Wetenschappers die ik hierboven nog niet heb vernoemd maar die op een of andere manier een bepalende rol hebben gespeeld bij mijn doctoraat zijn Pieter Blomme, Bart De Strooper, Bart Preneel, Johan Schoukens, Peter Schrooten, Tony Van Gestel, Marc Van Ranst en Kris Verstreken. Hen wil ik van harte danken voor hun sturing, ondersteuning, wijze raad,... waarvan ze zich misschien zelf niet ten volle bewust zijn hoe belangrijk die voor mij is geweest.

Ik ben heel veel dank verschuldigd aan het FWO, dat gedurende vier jaar zijn vertrouwen in mij heeft gesteld, niet alleen als aspirant, maar ook meermaals als 'onderzoeksreiziger'. Ik blijf verbaasd staan over de administratieve efficiëntie, de vriendelijkheid en bereidwilligheid die een instantie van dit formaat hanteert.

Had ik gedurende mijn doctoraat enkel onderzoek gedaan, dan trok ik nu wellicht elke dag met enig tegenspartelen een propere dwangbuis aan. Ik ben echter gezegend met een fantastische vriendengroep, die even hecht bleef zelfs na mijn soms lange afwezigheden.

De 'weekly activities', de lange nachtjes (nogal eens dikwijls in docs...), de zeil-, ski- en trekvakanties, de kotfeestjes en de mini-cantusjes,... the extravagant parties and weekend trips in Berkeley and Davis... ik heb er met volle teugen van genoten.

Muziek maken, ernaar luisteren en erover palaveren bij een witkap of leffe (of een rood wijntje) is voor mij steevast een uitlaatklep geweest. Ik blijf me welkom voelen in onze 'Koninklijke Harmonie Moed en Volharding' in Denderhoutem. De verscheidenheid, gedrevenheid, verbondenheid en vriendschap waar deze harmonieuze maatschappij voor staat zijn voor mij enorm belangrijk geweest, en zullen het steeds blijven, waar ik ook ben. Ik denk dan ook met veel respect en dankbaarheid terug aan onze voorzitters André Van Hecke en Jef Kiekens, die er helaas niet meer zijn.

In Leuven heb ik jaren deel mogen uitmaken van de Interfak BigBand, een geweldige groep muzikanten waar ik, in mijn muzikale beperktheid, misschien nauwelijks thuishoorde. Toch bleven de 'thrill' van de repetities, de gezellige palmkes (of iets anders) *op 't goe leven* erna, en de oprechte kameraadschap me lokken zolang het me lukte.

Mama en papa, voor jullie was niets teveel als het mij wel teveel werd. Niets te laat als het al laat was, niets te vroeg als het nog vroeg was. Niets te naief, te zot of te wispelturig om te kunnen begrijpen. Geen dal te diep en geen berg te hoog, jullie hebben me erdoor, erlangs of erover geholpen. En geen vreugde was te klein, of jullie zagen het in mijn ogen en deelden erin.

Tineke en Ward, het is zalig om te weten hoe goed we mekaar ondersteunen en begrijpen, zelfs zonder woorden. Zo'n zus en broer als eeuwige en onvoorwaardelijke vrienden, daar kunnen velen alleen maar van dromen.

Ik heb jullie dikwijls voor lange tijd moeten missen de voorbije jaren... Dank jullie voor deze prachtige Thuis, om steeds naar terug te keren!

Lieve Liesje, wat kan ik zeggen... Weet je wat, ik fluister het straks gewoon in je oor.
En morgen.
En overmorgen...

Leuven, België
April, 2005

Tijl

Abstract

In this thesis, we discuss the application of established and advanced *optimization techniques* in a variety of machine learning problems. More specifically, we demonstrate how fast optimization methods can be of use for the identification of *classes* or *clusters* in sets of data points, and this in general *semi-supervised learning settings*, where the learner is provided with some form of class information (or *supervision*) for some of the data points.

As we will point out, the semi-supervised learning scenario is in fact more tailored to practical problem settings than the traditional supervised and unsupervised approaches (with classification and clustering respectively as their archetypical examples).

The algorithmic machinery we used and extended by the semi-supervised learning methods presented in this thesis, rest on recent achievements in two domains: the *kernel methods* domain, and the domain that studies *graph cut problems* as a means to do data clustering.

Though other divisions of our contributions into coherent parts are possible, we have chosen to partition them based on the type of optimization methods they are built on: eigenvalue problems in Part I, and convex optimization problems in Part II. A smaller Part III reports some results that do not fit nicely in this division.

In the first two chapters of *Part I* we provide a unifying discussion of a series of algorithms that have been developed in the multivariate statistics and machine learning communities in the course of the last century, such as principal component analysis, canonical correlation analysis, partial least squares, Fisher's discriminant analysis and spectral clustering. Wherever possible, we provide the kernel variant, each time derived starting from its primal formulation.

After this overview of eigenvalue problems in multivariate statistics and machine learning, we will be in a position to discuss how such eigenvalue problems can be used to deal with more general learning settings than simple classification or clustering scenarios. We propose ways to circumvent the supervised-versus-unsupervised learning paradigm by introducing new methods that operate in the semi-supervised learning scenario (and in particular in the transduction setting), which holds the middle between both extremes.

Whereas Part I concerns the study of eigenvalue problems in machine learning and in generalized learning settings, in *Part II* we describe our contributions concerning the use of advanced convex optimization techniques for semi-supervised learning. We derive polynomial time approximation methods for transduction, a specific semi-supervised learning problem of which the computational complexity is known to be exponential in the problem size.

Next, using a bioinformatics case study, we show another strength of convex optimization in a transduction setting: it allows to jointly use heterogeneous information sources to make better classifiers.

Lastly, *Part III* contains some of our research results that would not fit in the main part of this thesis without breaking the flow. Still, we believe they are important enough to report on in our thesis.

In the first chapter of Part III we discuss an interpretation of a common way to regularize canonical correlation analysis. The second chapter of part III contains a second bioinformatics application: now the problem is not classification, but inference of regulatory modules. Interestingly, the techniques used here are necessarily radically different from the machine learning methods used in this thesis. Indeed, for most real life applications, a mix of machine learning, artificial intelligence and database techniques should be used.

Korte inhoud

In deze thesis behandelen we het gebruik van gevestigde en geavanceerde optimalisatiemethoden in het domein van machine leren, en meer specifiek in methoden voor het leren van klassen in het semi-gesuperviseerde scenario.

Zoals zal blijken voldoen dergelijke semi-gesuperviseerde benaderingen dikwijls beter aan de noden van de praktijk dan de volledige gesuperviseerde (zoals klassificatie) of volledig ongesuperviseerde (zoals clustering) aanpak.

In ons onderzoek naar nieuwe algoritmen die opereren in het semi-gesuperviseerde scenario hebben we ons laten leiden door recente ontwikkelingen in twee verwante onderzoeksdomeinen: het domein van kernfunctie methodes, en het domein dat zich bezig houdt met het bestuderen van graaf snedes en hun nut voor het clusteren van data.

Hoewel andere onderverdelingen mogelijk zijn, hebben we ervoor gekozen om onze bijdragen op te delen op basis van het type van optimalisatiemethoden waarvan ze gebruik maken:

In Deel I geven we eerst een geunificeerd overzicht van een reeks van lineaire algoritmen voor data analyse, zoals principale componenten analyse, canonieke componenten analyse, Fisher discriminant analyse en spectrale clustering. Waar mogelijk geven we een afleiding voor de primale versie en de kernfunctie gebaseerde versie.

Na deze overzichtsbijdrage kunnen we twee nieuwe methoden introduceren voor het leren van klassen in semi-gesuperviseerde scenario's, en in het bijzonder voor transductie.

Waar Deel I semi-gesuperviseerde scenarios voor klasseleren aanpakt met behulp van eigenwaardeproblemen, maken we in Deel II gebruik van convexe optimalisatiemethoden. We tonen aan hoe bijvoorbeeld SVM transductie (een specifiek geval van semi-gesuperviseerd leren), een probleem met een exponentiële complexiteit, gerelaxeerd kan worden tot een convex optimalisatieprobleem, met een polynomiale complexiteit.

Daarnaast bevat Deel II een praktische gevalstudie in de bioinformatica. We behandelden het klassificeren van genen op basis van heterogene informatiebronnen. Het gebruikte algoritme is gebaseerd op convexe optimalisatie, en opereert in het transductie scenario.

Tenslotte, in Deel III beschrijven we twee bijdragen die minder goed in indeling van de hoofdtekst passen, maar die we belangrijk genoeg achten om ze te rapporteren in deze thesis.

Het eerste hoofdstuk van Deel III behandelt een theoretische studie van de regularisatieparameter in canonieke correlatie analyse. Het tweede hoofdstuk van Deel III beschrijft een benadering van een belangrijk probleem uit de bioinformatica: het zoeken naar transcriptionele modules gebruik makende van heterogene informatiebronnen. Het algoritme dat we ontworpen hebben om dit probleem op te lossen maakt geen gebruik van de traditionele technieken van machine leren waarop de rest van deze thesis is gebaseerd. Het is daarentegen gebaseerd op het apriori algoritme, ontstaan in de data mining onderzoeksgemeenschap.

Symbols and Acronyms

List of symbols

The most important symbols used in this text are (other specific notational conventions will be made at the beginning of the relevant chapter):

- General notation:

matrices, vectors, scalars, sets	All matrices ($\mathbf{M}, \mathbf{N}, \mathbf{X}, \mathbf{Y}, \ldots$) are bold face upper case. Vectors ($\mathbf{w}, \mathbf{x}, \mathbf{y}, \boldsymbol{\alpha}, \ldots$) are bold face lower case. They are column vectors unless stated otherwise. Scalar variables ($c, d, i, j, \ldots$) are standard lower case. The element at the ith row and jth column of $\mathbf{A}$ is denoted by a_{ij}, the ith element of a vector $\mathbf{a}$ is denoted by a_i. Sets ($\mathcal{X}, \mathcal{Y}, \mathcal{F}, \ldots$) are denoted with calligraphic letters.
diagonal matrices	Bold face upper case Greek letters ($\boldsymbol{\Lambda}, \boldsymbol{\Xi}, \ldots$) denote diagonal matrices. Their diagonal elements are denoted by the corresponding standard lower case Greek letters, indexed to indicate which position on the diagonal they come from ($\lambda_i, \xi_i, \ldots$).
$'$, †	A transpose is denoted by a prime $'$, and † indicates the Moore-Penrose pseudo-inverse.
$\begin{pmatrix} \mathbf{a} & \mathbf{b} & \cdots & \mathbf{z} \end{pmatrix}$	The matrix built by stacking the column vectors $\mathbf{a}, \mathbf{b}, \ldots \mathbf{z}$ next to each other.
$d, i, j, k, m, n, \ldots$	Scalar integers, of which d is exclusively used for indicating dimensionalities, and i and j are preferentially used as indices in vectors and matrices or to indicate the iteration number in an iterative algorithm. Where a sample size is meant, k or n is used. In a different context, k may also stand for a kernel function.

$\mathbf{1}, \mathbf{I}$ The vector containing all ones is denoted by $\mathbf{1}$. The identity matrix is denoted by $\mathbf{I}$. The matrix or vector containing all zeros is denoted by $\mathbf{0}$. If their dimensionality is not clear from the context, it will be denoted with a subscript.

$\odot$ Elementwise matrix product operator, also known as the Schur product. For two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times m}$: $\mathbf{A} \odot \mathbf{B} = \mathbf{C}$ means $a_{ij} b_{ij} = c_{ij}$.

$\langle \mathbf{A}, \mathbf{B} \rangle$ The inner product between two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times m}$: $\langle \mathbf{A}, \mathbf{B} \rangle = \sum_{i=1:n} \sum_{j=1:m} a_{ij} b_{ij}$.

$\mathcal{E}$ The expectation operator.

- Algebra and optimization notation:

$\mathbf{u}$, $\mathbf{v}$, $\mathbf{w}$, $\mathbf{u}_i$, $\mathbf{v}_i$, $\mathbf{w}_i$, $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$,... These are the preferred symbols to denote (generalized) eigenvectors or singular vectors, usually of unit norm. When the eigenvectors are indexed, a smaller index value is used for an eigenvector belonging to a larger eigenvalue except when specified otherwise. $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{W}$ are matrices with these eigenvectors as their columns. Thus, for symmetric eigenvalue problems they are orthogonal: $\mathbf{U}'\mathbf{U} = \mathbf{I}$.

λ, λ_i, σ, σ_i λ is the preferred symbol for an eigenvalue. With λ_i the ith largest eigenvalue is meant (eigenvalue problems will be symmetric in this thesis, thus yielding real eigenvalues). I.e., $\lambda_i \geq \lambda_j$ iff $i \leq j$. Similarly, σ usually denotes a singular value, and σ_i denotes the ith largest singular value.

λ, μ, ν, α, λ_i, μ_i, ν_i, α_i, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$, $\boldsymbol{\nu}$, $\boldsymbol{\alpha}$,... These symbols are used as Lagrange multipliers (or as vectors containing Lagrange multipliers).

$\mathcal{L}$ The Lagrangian.

- Notation specific for kernel methods, multivariate statistics, and machine learning:

k A kernel function.

γ The regularization parameter.

c	The number of classes or classes present in a data set.
$\mathbf{x}, \mathbf{x}_i, \mathbf{X}$	The vectors $\mathbf{x}$ and $\mathbf{x}_i$ are column vectors representing vectors in the $\mathcal{X}$-space: $\mathbf{x}, \mathbf{x}_i \in \mathcal{X}$. When there are n samples, the matrix $\mathbf{X}$ is built up as $\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \end{pmatrix}'$.
$\mathbf{y}, \mathbf{y}_i, \mathbf{Y}$	Similarly $\mathbf{y}$ and $\mathbf{y}_i$ are sample vectors from the $\mathcal{Y}$-space: $\mathbf{y}, \mathbf{y}_i \in \mathcal{Y}$. The matrix $\mathbf{Y}$ containing samples $\mathbf{y}_1$ through $\mathbf{y}_n$ is built up as $\mathbf{Y} = \begin{pmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_n \end{pmatrix}'$.
$y, y_i, \mathbf{y}$	When $\mathcal{Y}$ is one-dimensional, a sample from this space is denoted by y or y_i, and the vector containing all samples is $\mathbf{y} = \begin{pmatrix} y_1 & y_2 & \cdots & y_n \end{pmatrix}'$. In that case, very often y_i represents a class label, and $\mathcal{Y} = \{1, 2, \ldots, c\}$ with c the number of classes, or $\mathcal{Y} = \{-1, 1\}$ in the two-class case
$\mathbf{K_X}, \mathbf{K_Y}, \mathbf{K}$	The matrices $\mathbf{K_X}$ and $\mathbf{K_Y}$ are the so-called *kernel* or *Gram matrices* corresponding to the data matrices $\mathbf{X}$ and $\mathbf{Y}$. They are the inner product matrices $\mathbf{K_X} = \mathbf{X}\mathbf{X}'$ and $\mathbf{K_Y} = \mathbf{Y}\mathbf{Y}'$. When it is clear from the context which data the kernel is built from, we just use $\mathbf{K}$. When we want to stress the kernel is centered we use $\mathbf{K}_c$.
$\mathbf{S_{XX}}$, $\mathbf{S_{XY}}$, $\mathbf{S_{YX}}$, $\mathbf{S_{YY}}$, $\mathbf{C_{XX}}$, $\mathbf{C_{XY}}$, $\mathbf{C_{YX}}$, $\mathbf{C_{YY}}$	For centered data matrices $\mathbf{X}$ and $\mathbf{Y}$, the matrices $\mathbf{S_{XX}} = \mathbf{X}'\mathbf{X}$, $\mathbf{S_{XY}} = \mathbf{X}'\mathbf{Y}$, $\mathbf{S_{YY}} = \mathbf{Y}'\mathbf{Y}$ and $\mathbf{S_{YX}} = \mathbf{S_{XY}}'$ are the *scatter matrices*. If $\mathbf{X}$ and $\mathbf{Y}$ contain n samples, the *sample covariance matrices* are denoted by $\mathbf{C_{XX}} = \frac{\mathbf{X}'\mathbf{X}}{n}$, $\mathbf{C_{XY}} = \frac{\mathbf{X}'\mathbf{Y}}{n}$, $\mathbf{C_{YY}} = \frac{\mathbf{Y}'\mathbf{Y}}{n}$ and $\mathbf{C_{YX}} = \mathbf{C_{XY}}'$.
$\mathbf{w}, \mathbf{w_X}, \mathbf{w_Y}$	These vectors will often be referred to as *weight vectors*. Their respective ith coordinates are denoted by $w_i, w_{\mathbf{X},i}, w_{\mathbf{Y},i}$. When an index i is used as a subscript after a boldface $\mathbf{w}$, this refers to a weight vector indexed by i, and not to the ith coordinate.
$\boldsymbol{\alpha}$, $\boldsymbol{\alpha_X}$, $\boldsymbol{\alpha_Y}$, α_i, $\alpha_{\mathbf{X},i}$, $\alpha_{\mathbf{Y},i}$,	These variables will be referred to as *dual vectors* and their respective ith coordinates. When an index i is used as a subscript after a boldface $\boldsymbol{\alpha}$, this refers to a dual vector indexed by i, and not to the ith coordinate.
$\phi(\mathbf{x}_i)$	The feature map from the input space $\mathcal{X}$ to the feature space $\mathcal{F}$.

List of acronyms

AI	Artificial intelligence
BoW	Bag-of-words
CCA	Canonical correlation analysis
ChIPchip	chromatin immunoprecipitation on arrays
ERM	Empirical risk minimization
FDA	Fisher discriminant analysis
LDA	Linear discriminant analysis
LP	Linear programming
LSR	Least squares regression
LS-SVM	Least squares support vector machine
ML	Machine learning
PCA	Principal component analysis
PLS	Partial least squares
QP	Quadratic programming
RCCA	Regularized CCA
RR	Ridge regression
SC	Spectral clustering
SDP	Semi definite programming
SPD	Semi positive definite
SVM	Support vector machine

Glossary

Artificial intelligence (AI). A branch of computer science that is concerned with the simulation of intelligent behavior in machines. Two important approaches can be distinguished:

- one branch of AI studies approaches to implement intelligent actions and responses in a machine, based on existing human experience. Once the intelligent behaviour has been implemented, the machine remains as it is and is unable to learn from experience autonomically. Examples are expert systems, (most) chess computer programs, traditional speech recognition algorithms,...
- in the other main branch of AI, methods are studied and developed that allow a machine to *learn* from examples. Such methods enable initially ignorant machines to autonomically become capable of solving complex tasks after a learning (or training) period, during which the machine is presented several examples of (sometimes partial) solutions to the task to be solved. This branch of AI is called *machine learning*.

Class, class label. An integer value that is associated with each data point in *class learning* problems. Sometimes the class of a data point actually a stochastic variable depending on this data point. Then, if a specific value for the class of a certain point is provided, this means that this specific value is randomly drawn from the conditional probability distribution of the class label, conditioned on the data point.

Class learning. A set of learning problems where one is concerned with learning to predict the *class* of a set of points as accurately as possible, based on the knowledge of these points themselves, and potentially on information that is given on the *class labels* of some of these points. Examples of specific class learning problems are *clustering*, *classification*, and *transduction*. In all these cases, a priori statistical assumptions on the data points and class labels have to be made, in order to be able to prove workability of the methods.

Classification. A particular instance of *class learning*, where a so-called training set of points is given along with a class label associated to each of the data

points, and where one tries to come up with a classification function that allows to estimate the labels of other points drawn from the same distribution. A classification task usually consists of two steps: first, based on the set of labeled points (called the training set), a classification function is proposed for which it is believed that it will perform well on test samples. This step is called the *induction* step or sometimes the *learning* step. Subsequently, in the *deduction* step, this classification function can be evaluated on any unlabeled test sample, which yields an estimate for its label. See Section 0.2.1 for a more formal definition.

Clustering. A particular instance of *class learning*, where only a set of points is given while no information on their class labels is provided. See Section 0.2.1 for a more formal definition.

Kernel methods. A specific approach to machine learning in which algorithms are decomposed into two stages: the kernel computation stage, which computes a similarity measure between the given data points (expressed by the kernel function), and the actual algorithm, which operates on the kernel function. See Section 1.2 for a more elaborate discussion.

Learning. A restrictive definition of learning is the process of observing examples of (potentially partial) solutions to a specific task and, in doing so, detecting statistically stable regularities (the induction phase) that, to some nontrivial extent, enable the learner to solve the same task on new data generated by the same source (the deduction phase). E.g., a *classification* problem is a specific learning problem, where the task to be solved is the assignment of a *class label* to any given data point that is drawn from an unknown but fixed distribution. A broader definition of learning is the process of detecting statistically stable regularities in given data, based on which non-trivial predictions can be made. According to this definition, also *clustering*, *transduction*, principal component analysis, canonical correlation analysis,... are learning problems.

Machine learning (ML). A branch of *artificial intelligence* that aims at making machines capable to learn from examples. A machine in which a machine learning algorithm is implemented usually is initially ignorant, but after going through a learning process, during which it observes examples of (potentially partial) solutions to the task it is meant to solve (the training stage), it becomes capable of solving this task itself, to some nontrivial extent.

Side-information learning. See *semi-supervised learning*.

Semi-supervised learning. A general class of *learning* problems that encompasses all learning methods (in the broad sense) in between *supervised learning* and *unsupervised learning*. In particular in this thesis, we discuss semi-supervised learning in the context of *class learning*.

Supervised learning. A particular class of *learning* problems where a set of data points are given, and where for each of these data points the solution of the task to be solved is provided to the learner. A particular instance of supervised learning, where the solution of the problem to be solved is a *class label* associated to the data point given, is *classification*.

Transduction. A particular type of *class learning* problems, similar to *classification*, but where the entire test set (often called the working set in the *transductive* setting) is given beforehand. As a result, it is not necessary to induce a classification function, but it is possible to combine the induction and deduction steps in one single step, called the transduction step. See Section 0.2.2 for a more formal definition.

Unsupervised learning. A particular class of *learning* problems where only data points are given, but the attributes one is interested in (e.g. *class labels* in *class learning* problems) corresponding to the data points are unknown. A particular instance of unsupervised learning is *clustering*: the data points are given, but for none of the data points the class label is specified.

Contents

Voorwoord **i**

Abstract **v**

Korte inhoud **vii**

Symbols and Acronyms **ix**

Glossary **xiii**

Contents **xvi**

Nederlandstalige samenvatting **xxi**

Introduction **1**
0.1 Motivation . . . 2
0.1.1 What, why and how? . . . 2
0.1.2 Machine learning today: learning from general label information and heterogeneous data, using fast optimization techniques . . . 5
0.2 General learning settings . . . 5
0.2.1 Standard settings for class-learning: classification and clustering . . . 5
0.2.2 Transduction . . . 7
0.2.3 Incorporating general label information . . . 8
0.2.4 Learning from heterogeneous information . . . 8
0.3 Machine learning and optimization . . . 10
0.3.1 Eigenvalue problems . . . 10
0.3.2 Convex optimization problems . . . 10
0.4 Outline of the thesis . . . 11
0.5 Personal contributions . . . 11

I Algorithms based on eigenvalue problems **15**

1 Eigenvalue problems and kernel trick duality: elementary principles **17**
1.1 Some basic algebra . . . 18
1.1.1 Symmetric (Generalized) Eigenvalue Problems . . . 18
1.1.2 Singular Value Decompositions, Duality . . . 21
1.2 Kernel methods: the duality principle . . . 22
1.2.1 Theory . . . 23
1.2.2 Example: least squares and ridge regression . . . 24
1.3 Conclusions . . . 26

2 Eigenvalue problems in machine learning, primal and dual formulations **29**
2.1 Kernels in this chapter . . . 29
2.1.1 Normalizing a kernel matrix . . . 30
2.1.2 Centering a kernel matrix . . . 30
2.1.3 A leading example . . . 32
2.1.4 Kernel Functions . . . 32
2.2 Dimensionality Reduction: PCA, (R)CCA, PLS . . . 35
2.2.1 PCA . . . 36
2.2.2 (R)CCA . . . 38
2.2.3 PLS . . . 46
2.2.4 Illustrative comparison of PCA, CCA and PLS dimensionality reduction . . . 55
2.3 Classification: Fisher Discriminant Analysis (FDA) . . . 56
2.3.1 Cost function . . . 58
2.3.2 Primal . . . 59
2.3.3 Dual . . . 59
2.3.4 Linear discriminant analysis (LDA) . . . 61
2.4 Spectral methods for clustering . . . 61
2.4.1 The affinity matrix . . . 62
2.4.2 Cut, average cut and normalized cut cost functions . . . 62
2.4.3 What to do with the eigenvectors? . . . 65
2.5 Summary . . . 67
2.6 Conclusions . . . 67

3 Eigenvalue problems for semi-supervised learning **71**
3.1 Side-information for dimensionality reduction . . . 72
3.1.1 Learning the Metric . . . 73
3.1.2 Remarks . . . 78
3.1.3 Empirical Results . . . 79
3.1.4 Alternative Derivation . . . 82
3.2 Spectral clustering with constraints . . . 83
3.2.1 Spectral clustering . . . 84
3.2.2 Hard constrained spectral clustering . . . 86
3.2.3 Softly constrained spectral clustering . . . 89
3.2.4 Empirical results . . . 89
3.3 Conclusions . . . 90

II Algorithms based on convex optimization **93**

4 Convex optimization in machine learning: a crash course **95**
4.1 Optimization and convexity . . . 95
4.1.1 Lagrange theory . . . 96
4.1.2 Weak duality . . . 96
4.1.3 Convexity and strong duality . . . 96
4.2 Standard formulations of convex optimization problems . . . 96
4.2.1 LP . . . 97
4.2.2 QP . . . 97
4.2.3 SDP . . . 98
4.3 Convex optimization in machine learning . . . 98
4.3.1 SVM classifier . . . 98
4.3.2 LS-SVM classifier . . . 99
4.4 Conclusions . . . 100

5 Convex optimization and transduction **101**
5.1 Support vector machine transduction by density estimation . . . 102
5.1.1 Weighting errors with the estimated density 102
5.1.2 Weighting the weight vector 104
5.2 A convex relaxation of SVM transduction 107
5.2.1 The transductive SVM 107
5.2.2 Relaxation to an SDP problem 108
5.2.3 Subspace SDP formulation 111
5.2.4 Empirical results 113
5.3 Convex transduction using the normalized cut 113
5.3.1 NCut transduction 115
5.3.2 A spectral and a first SDP relaxation of NCut clustering . 115
5.3.3 Two tractable SDP relaxations for transduction with the NCut 117
5.3.4 Empirical results 120
5.4 Conclusions 120

6 Convex optimization for kernel learning: a bioinformatics example **123**
6.1 Optimizing the kernel 123
6.1.1 The optimization problem 124
6.1.2 Corollary: a convex method for tuning the soft-margin parameter 124
6.2 A case study in bioinformatics 125
6.2.1 The individual kernels 125
6.2.2 Experimental Design 128
6.2.3 Results 129
6.2.4 Discussion 134
6.3 Conclusions 136

III Further topics **137**

7 On regularization, canonical correlation analysis, and beyond **139**
7.1 Regularization: what and why? 139
7.2 From least squares regression to ridge regression 140
7.2.1 Least squares as a maximum likelihood estimator 140
7.2.2 Ridge regression as the maximizer of the expected log likelihood 141
7.2.3 Interpretation 142
7.2.4 Practical aspects 143
7.3 From canonical correlation analysis to its regularized version . . 143
7.3.1 Standard geometrical approach to canonical correlation analysis 144
7.3.2 Canonical correlation analysis as a maximum likelihood estimator 144
7.3.3 Regularized CCA as the maximizer of the expected log likelihood 147
7.3.4 Interpretation 148
7.3.5 Practical aspects 148
7.4 Conclusions 149

8 Integrating heterogeneous data: a database approach **151**
8.1 Introduction . . . 151
8.2 Situation . . . 152
8.3 Materials and Algorithms . . . 153
8.3.1 Data sources . . . 153
8.3.2 Module construction algorithm . . . 154
8.3.3 Calculating overrepresentation of functional classes . . . 157
8.4 Results . . . 157
8.4.1 Cell cycle related modules . . . 157
8.4.2 Non cell cycle related modules . . . 159
8.5 Discussion . . . 159
8.6 Conclusion . . . 161

General conclusions **163**

Bibliography **165**

Publication list **173**

Semi-gesuperviseerd leren gebaseerd op kernfunctie methoden en graaf snede algoritmen

Nederlandstalige samenvatting

Het werk in deze thesis situeert zich in het domein van *machine leren*, dat bestudeert in hoeverre en op welke manier het mogelijk is te leren op basis van voorbeelden. Meer specifiek behandelen we hier het leren van klassen (of: klasse labels) van datapunten. Elke klasse $y \in \mathcal{Y}$ wordt hierbij voorgesteld door een geheel getal (meestal $\mathcal{Y} = \{1, 2, \ldots, c\}$, of $\mathcal{Y} = \{-1, 1\}$, afhankelijk van het aantal clusters c), en elk datapunt $\mathbf{x} \in \mathcal{X}$.

Methoden voor klasse leren worden traditioneel gecatalogeerd in ongesuperviseerde leermethoden en gesuperviseerde leermethoden. Clusteralgoritmen, ongesuperviseerde leeralgoritmen voor het leren van klassen, opereren op een dataset $\{\mathbf{x}_i\}|_{i=1,n}$, zonder klasse labels. Ze zoeken naar coherente groepen van datapunten binnnen de hele dataset, en associëren een verschillend klasse label met elk van deze verschillende coherente groepen. Klassificatiealgoritmen daarentegen werken met een volledig gelabelde dataset $\{(\mathbf{x}_i, y_i)\}|_{i=1,n}$, de *trainingset* genoemd, op basis waarvan ze een functie $h : \mathcal{X} \to \mathcal{Y}$ zoeken die de datapunten afbeeldt op een klasselabel. Goede klassificatiealgoritmen slagen erin een functie h te vinden die voor testpunten $\mathbf{x}_{\text{test}}$ (met bijhorend maar ongekend klasse label y_{test}), lukraak getrokken uit dezelfde waarschijnlijkheidsverdeling als de trainingset, zo vaak mogelijk een predictie $h(\mathbf{x}_{\text{test}})$ maakt gelijk aan het ware klasse label y_{test}.

De meeste nieuwe algoritmen voorgesteld in deze thesis maken deel uit van twee belangrijke klassen van algoritmen binnen het domein van machine leren: de kernfunctie gebaseerde methoden (die voornamelijk goede resultaten hebben geleverd op het gebied van gesuperviseerde methoden), en de klasse van graaf snede methoden (die totnogtoe uitsluitend uit ongesuperviseerde methoden bestond).

In deze thesis slaan we de brug tussen ongesuperviseerde en gesuperviseerde methoden, om te kunnen opereren in het semi-gesuperviseerde scenario waarbij een set van datapunten $\{\mathbf{x}_i\}|_{i=1,n}$ is gegeven, en verder enkel *beperkingen* op de klasse labels van een deel van deze datapunten. We tonen aan hoe kernfunctie gebaseerde methoden, zoals de *support vector machine* (SVM), kunnen worden aangepast om dit soort informatie uit te buiten. Omgekeerd passen we graaf snede methoden voor clustering aan om zij-informatie over de klasse labels te exploiteren.

We kunnen verschillende vormen van zij-informatie onderscheiden. In de eenvoudigste vorm bestaat de zij-informatie uit de volledige specificatie van labels van *een deel* van de datapunten (in klassificatieproblemen zijn alle labels gegeven, in clusteringsproblemen geen). Meer complexe vormen van zij-informatie specifiëren bijvoorbeeld dat twee gegeven datapunten hetzelfde of, omgekeerd, een verschillend klasse label hebben.

Het is belangrijk op te merken dat het in feite vrij eenvoudig is om nieuwe semi-gesuperviseerde methoden voor te stellen. Echter, de computationele kost van voor de hand liggende algoritmen is steeds exponentieel in de grootte van de ontbrekende label informatie, hetgeen ze onbruikbaar maakt in praktijk. De algoritmen die zijn afgeleid in deze thesis zijn daarentegen allemaal gebaseerd op eigenwaardeproblemen (Deel I), of op convexe optimalisatiemethoden die gegarandeerd in polynomiale tijd kunnen worden opgelost (Deel II).

In deze nederlandstalige samenvatting zullen we ons toespitsen op de leidraad door ons onderzoek dat tot deze thesis heeft geleid: *efficiënte methoden voor semi-gesuperviseerd leren*. De hoofdstukken die het minst aansluiten bij het hoofdonderwerp van deze thesis, of die voornamelijk inleidende informatie bevatten, zullen we hier niet (hoofdstukken 7 en 8) of in eerder beperkte mate (hoofdstukken 1, 2 en 4) bespreken.

Deel I: Eigenwaardeproblemen voor semi-gesuperviseerd leren

In het eerste deel van deze nederlandstalige samenvatting zullen we semi-gesuperviseerde problemen aanpakken gebruik makende van eigenwaardeproblemen. Hiervoor stellen we twee verschillende nieuwe benaderingen voor.

De eerste benadering is gebaseerd op kernfuncties, en zoekt naar een laagdimensionale representatie van de datapunten die de zij-informatie reflecteert, waarbij de zij-informatie bestaat uit het feit dat voor bepaalde paren van punten is gegeven dat ze tot dezelfde klasse behoren. Met 'zij-informatie reflecteren' bedoelen we dat datapunten die volgens de zij-informatie hetzelfde label hebben in de nieuwe representatie kort bij mekaar moeten liggen, gemeten met de Euclidische afstand. Het algoritme is gebaseerd op canonieke correlatie analyse (CCA), en vergt slechts het oplossen van een eigenwaardeprobleem.

De tweede benadering is gebaseerd op graaf snede methoden voor clustering, en hun spectrale relaxaties. Zoals we zullen aantonen is het mogelijk op een elegante manier zeer algemene beperkingen op te leggen aan dergelijke algoritmen (ook ongelijkheidsbeperkingen). Op die manier is het mogelijk om datapunten te clusteren rekening houdend met de beperkingen opgelegd door de zij-informatie. Ook hier vergt het algoritme slechts het oplossen van een eigenwaardeprobleem.

1. Dimensionaliteitsreductie gebaseerd op zij-informatie

Dimensionaliteitsreductie gebaseerd op zij-informatie kan een goede eerste verwerking van de gegevens zijn: indien in de lager dimensionale ruimte gelijk gelabelde datapunten korter bij mekaar liggen, zullen standaard clusteralgoritmen beter presteren op deze geprojecteerde versie van de data. Dit is de manier waarop het algoritme in deze sectie moet gebruikt worden: als voorverwerking van de datapunten rekening houdend met de zij-informatie, waarna een standaard clusteralgoritme kan worden aangewend. Vooraleer we het eigenlijke algoritme kunnen uitleggen, moeten we echter kort ingaan op canonieke correlatie analyse.

1.1. Canonieke correlatie analyse

Gegeven is een set van corresponderende datapunten $\{\mathbf{x}_i, \mathbf{z}_i\}|_{i=1,n}$. We nemen hier aan dat de datapunten behoren tot vectorruimten $\mathcal{X} = \mathbb{R}^{d_\mathbf{X}}$ en $\mathcal{Z} = \mathbb{R}^{d_\mathbf{Z}}$ (dus $\mathbf{x}_i$ en $\mathbf{z}_i$ zijn kolomvectoren), en stellen de datasets voor door matrices $\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n \end{pmatrix}'$ en $\mathbf{Z} = \begin{pmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \dots & \mathbf{z}_n \end{pmatrix}'$.

Canonieke correlatie analyse (CCA) zoekt naar richtingen $\mathbf{w_X}$ en $\mathbf{w_Z}$ in beide data ruimten $\mathcal{X}$ en $\mathcal{Y}$ zodanig dat de correlatie tussen $\mathbf{x}_i'\mathbf{w_X}$ en $\mathbf{z}_i'\mathbf{w_Z}$ zo groot mogelijk is. Geschreven als een optimalisatieprobleem wordt dit:

$$\begin{aligned} \{\mathbf{w_X}, \mathbf{w_Z}\} &= \text{argmax}_{\mathbf{w_X},\mathbf{w_Z}} \frac{(\mathbf{X}\mathbf{w_X})'(\mathbf{Z}\mathbf{w_Z})}{\sqrt{(\mathbf{X}\mathbf{w_X})'(\mathbf{X}\mathbf{w_X})}\sqrt{(\mathbf{Z}\mathbf{w_Z})'(\mathbf{Z}\mathbf{w_Z})}} \\ &= \text{argmax}_{\mathbf{w_X},\mathbf{w_Z}} \frac{\mathbf{w_X}'\mathbf{S_{XZ}}\mathbf{w_Z}}{\sqrt{\mathbf{w_X}'\mathbf{S_{XX}}\mathbf{w_X}}\sqrt{\mathbf{w_Z}'\mathbf{S_{ZZ}}\mathbf{w_Z}}}. \end{aligned}$$

waarbij we de notatie $\mathbf{S_{XZ}} = \mathbf{X'Z}$, $\mathbf{S_{XX}} = \mathbf{X'X}$ en $\mathbf{S_{ZZ}} = \mathbf{Z'Z}$ gebruiken. In woorden stellen deze twee gewichtenvectoren $\mathbf{w_X}$ en $\mathbf{w_Z}$ projectierichtingen voor volgens dewelke corresponderende datapunten sterk met mekaar correleren.

De oplossing van dit optimalisatieprobleem kan gevonden worden onder de vorm van de dominante eigenvector van het volgende veralgemeende eigenwaardeprobleem:

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XZ}} \\ \mathbf{S_{ZX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Z} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{S_{XX}} & \mathbf{0} \\ \mathbf{0} & \mathbf{S_{ZZ}} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Z} \end{pmatrix}.$$

De eigenvectoren die horen bij andere grote eigenwaarden zijn dikwijls ook van belang: zolang de eigenwaarde groot is stellen ze richtingen voor waarop projecties van corresponderende datapunten sterk gecorreleerd zijn.

Regularisatie van CCA In feite zijn we geinteresseerd in richtingen $\mathbf{w_X}$ en $\mathbf{w_Z}$ waarlangs ook *testpunten* die uit dezelfde waarschijnlijkheidsverdeling zijn getrokken, sterk gecorreleerd zijn. Zoals in vrijwel alle technieken voor machine leren is daarom een vorm van regularisatie noodzakelijk, om te voorkomen dat teveel ruis in onze datasets $\mathbf{X}$ en $\mathbf{Z}$ een te sterke invloed heeft. De geregulariseerde vorm van CCA kan worden opgelost door middel van het volgende eigenwaardeprobleem:

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XZ}} \\ \mathbf{S_{ZX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Z} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{S_{XX}} + \gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{S_{ZZ}} + \gamma\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Z} \end{pmatrix}$$

waarbij γ de zogenaamde regularisatieparameter voorstelt. Voor meer informatie en een theoretische interpretatie van de regularisatieparameter verwijzen we de lezer naar Hoofdstuk 7 van de hoofdtekst.

Kernfunctie formulering van CCA Men kan aantonen (zie hoofdtekst) dat de gewichtenvectoren $\mathbf{w_X}$ en $\mathbf{w_Z}$ die gevonden worden met CCA uitgedrukt kunnen worden als lineaire combinaties van de datapunten: $\mathbf{w_X} = \mathbf{X'}\boldsymbol{\alpha_X}$ en $\mathbf{w_Z} = \mathbf{Z'}\boldsymbol{\alpha_Z}$. Als we deze uitdrukkingen voor de gewichtenvectoren substitueren in het bovenstaande eigenwaardeprobleem, dan verkrijgen we de duale versie van CCA:

$$\begin{pmatrix} \mathbf{0} & \mathbf{K_X K_Z} \\ \mathbf{K_Z K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Z} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{K_X^2} + \gamma\mathbf{K_X} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Z^2} + \gamma\mathbf{K_Z} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Z} \end{pmatrix}$$

waarbij we de notatie $\mathbf{K_X} = \mathbf{XX'}$ en $\mathbf{K_Z} = \mathbf{ZZ'}$ gebruiken.

Merk op dat deze zogenaamde Gram matrices op rij i en kolom j het inwendig product tussen $\mathbf{x}_i$ en $\mathbf{x}_j$, respectievelijk tussen $\mathbf{z}_i$ en $\mathbf{z}_j$ bevatten. De projectie van een testpunt $\mathbf{x}_{\text{test}}$ op $\mathbf{w_X} = \mathbf{X'}\boldsymbol{\alpha_X}$ kan worden uitgevoerd als $\mathbf{x}'_{\text{test}}\mathbf{w_X} = \mathbf{x}'_{\text{test}}\mathbf{X'}\boldsymbol{\alpha_X}$, en is dus ook volledig te schrijven in termen van inwendige producten tussen de datapunten.

Het blijkt dus dat elke operatie gepaard gaande met CCA, en het projecteren van een datapunt op een richting verkregen met CCA, kan geschreven worden in termen van inwendige producten alleen. Dit maakt het mogelijk de zogenaamde kernfunctie truuk toe te passen: in plaats van gewoon inwendige producten te berekenen tussen de datapunten zelf, kunnen we inwendige producten berekenen tussen een mapping van deze datapunten in een mogelijks hoog-dimensionale Hilbert ruimte. Dit is zelfs mogelijk als $\mathcal{X}$ en $\mathcal{Y}$ zelf geen Hilbert ruimte vormen. Van fundamenteel belang is de observatie dat de mappings van de datapunten blijkbaar nooit geëxpliciteerd moeten worden: als we de inwendige producten op een efficiënte manier kunnen berekenen is dit voldoende. De functie die dit inwendige product berekent tussen twee datapunten wordt de kernfunctie genoemd, en heeft als eigenschappen dat ze symmetrisch en semi positief definiet is.

De mogelijkheden van het gebruik van kernfuncties in algoritmen voor machine leren zijn enorm: het maakt het mogelijk om CCA (en vele andere algoritmen) toe te passen op niet-vectoriële data, zolang er een relevante kernfunctie voor deze data kan worden gedefinieerd, of om niet-lineaire varianten van deze *in se* lineaire algoritmen te maken (gebruik makende van een niet-lineaire kernfunctie).

CCA tussen datapunten enerzijds, en zij-informatie anderzijds

Laten we nu terugkeren naar het probleem van het leren van klassen. Gegeven zijn een dataset van n punten $(\{\mathbf{x}_i\}|_{i=1,n}$ met $\mathbf{x}_i \in \mathcal{X} = \mathbb{R}^d$, en beperkingen op de klasse labels $y_i \in \mathcal{Y} = \{1, 2, \ldots, c\}$ van deze punten, onder de vorm van gelijkheidsbeperkingen tussen paren van datapunten (dus van de vorm $y_i = y_j$). We noteren de volledige datamatrix $\mathbf{X} = \begin{pmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{pmatrix}$, waarbij het datapunt op de $i$de rij van $\mathbf{X}^{(1)}$ en op de $i$de rij van $\mathbf{X}^{(2)}$ volgens de zij-informatie hetzelfde label hebben. We nemen aan dat $\mathbf{X}$ *gecentreerd* is, d.w.z. dat bij alle kolommen een constante is opgeteld om de kolomsommen gelijk aan nul te maken, hetgeen erop neer komt dat alle datapunten getransleerd worden in $\mathbb{R}^d$ zodat de gemiddelde waarde van elk van de d coördinaten gelijk is aan nul.

We zijn op zoek naar een projectie van de (gecentreerde) datapunten $\mathbf{x}_i$ op een subruimte van $\mathcal{X}$, waarin projecties van gelijk gelabelde datapunten kort bij mekaar liggen, en projecties van ongelijk gelabelde datapunten ver van mekaar verwijderd, in termen van Euclidische afstand.

In het geval van een volledig gelabelde dataset bestaat een dergelijk algoritme: het heet *lineaire discriminant analyse* (LDA), en is equivalent met het uitvoeren van CCA tussen de datapunten in $\mathbf{X}$ zoals hoger gedefinieerd, en een label matrix $\mathbf{Z}$ gedefinieerd als de gecenterde versie van $\widetilde{\mathbf{Z}}$: $\mathbf{Z} = \widetilde{\mathbf{Z}} - \frac{\mathbf{1}\mathbf{1}'}{n}\widetilde{\mathbf{Z}}$, waarbij voor $\widetilde{\mathbf{Z}}_{i,j}$

(het element op rij i en kolom j):

$$\widetilde{\mathbf{Z}}_{i,j} = \begin{cases} 1 & \text{als de klasse van datapunt } \mathbf{x}_i \text{ gelijk is aan } j \\ 0 & \text{anders.} \end{cases}$$

In deze thesis hebben we een benaderende versie van LDA ontwikkeld, gebaseerd op zij-informatie alleen, waarbij de zij-informatie van de vorm is zoals hoger gedefinieerd. Dit kan door de CCA kostfunctie in termen van de volledige label matrix $\mathbf{Z}$ uit te schrijven, en het gemiddelde van deze kostfunctie te berekenen, waarbij het gemiddelde is genomen over alle mogelijke label matrices die de zij-informatie respecteren. Optimalisatie van de resulterende gemiddelde kostfunctie leidt opnieuw tot een veralgemeend eigenwaardeprobleem:

$$(\mathbf{C}_{12} + \mathbf{C}_{21})\mathbf{w} = \lambda(\mathbf{C}_{11} + \mathbf{C}_{22})\mathbf{w}$$

waarbij $\mathbf{C}_{kl} = {\mathbf{X}^{(k)}}'\mathbf{X}^{(l)}$. Projecties op de dominante eigenvectoren van dit eigenwaardeprobleem respecteren de zij-informatie in hoge mate.

In de hoofdtekst beschrijven we diverse uitbreidingen naar meer algemene vormen van zij-informatie, in het bijzonder in het geval de zij-informatie niet enkel voor paren van datapunten specifieert dat ze tot dezelfde klasse behoren, maar ook grotere groepjes datapunten van willekeurige grootte. Voor meer informatie verwijzen we naar Sectie 3.1.2.

Regularisatie Net zoals bij CCA is het dikwijls noodzakelijk een vorm van regularisatie te gebruiken. Dit kan op een analoge manier als bij CCA: het volstaat bij $\mathbf{C}_{11}+\mathbf{C}_{22}$ in het rechterlid van het veralgemeend eigenwaardeprobleem $\gamma\mathbf{I}$ op te tellen.

Kernfunctie formulering Het blijkt mogelijk het resulterende eigenwaardeprobleem volledig uit te schrijven in termen van inwendige producten tussen datapunten. We kunnen ook hier dus de kernfunctie truuk toepassen, hetgeen betekent dat het algoritme kan worden toegepast op niet-vectoriële data, en dat eenvoudig niet-lineaire versies van het algoritme kunnen worden opgesteld. Voor meer informatie verwijzen we naar Sectie 3.1.2.

2. Spectrale clustering met beperkingen

De methode voor semi-gesuperviseerd leren geïntroduceerd die we introduceren in deze sectie is gebaseerd op clusteralgoritmen op basis van graaf snedes, en hun spectrale relaxaties. We zullen eerst een bondig overzicht geven van de basisprincipes van dergelijke benaderingen voor clustering. Daarna zullen we aantonen hoe het mogelijk is om beperkingen op te leggen deze clusteralgoritmen om ervoor te zorgen dat het resultaat beantwoordt aan de gegeven zij-informatie over de labels van de datapunten.

In de afleidingen van clusteralgoritmen voor graaf sneden zullen we ons beperken tot twee-klasse problemen. We spreken dan van een zogenaamde positieve (met label 1) en een negatieve (met label -1) klasse. Uitbreidingen naar multi-klasse clustering toe zijn beschreven in de literatuur, en doorgaans eenvoudig te maken. Waar dit nuttig is zullen we kort aangeven hoe dit kan gebeuren.

2.1. Graaf snedes en spectrale clustering

Gegeven een volledig geconnecteerde gewogen graaf, bestaande uit een set van knopen $\mathcal{S}$, waarvan elke knoop geassocieerd is met een datapunt $\mathbf{x}_i$. Elke tak tussen een paar knopen heeft een positief gewicht. De gewichten worden samengevat in een symmetrische matrix $\mathbf{A}$, met als element op rij i en kolom j het gewicht tussen de knopen overeenkomend met de datapunten $\mathbf{x}_i$ en $\mathbf{x}_j$. De gewichten a_{ij} zijn in de regel groter naarmate de punten $\mathbf{x}_i$ en $\mathbf{x}_j$ 'gelijkaardiger' zijn, en dus met grotere waarschijnlijkheid tot dezelfde klasse behoren.

Gegeven zo een graaf, definiëren we de volgende grootheden:

- de graaf snede tussen de set van knopen $\mathcal{N}$ en de set van knopen $\mathcal{P}$: $\text{cut}(\mathcal{P},\mathcal{N}) = \text{cut}(\mathcal{N},\mathcal{P}) = \sum_{i:\mathbf{x}_i\in\mathcal{P},j:\mathbf{x}_j\in\mathcal{N}} a_{ij}$. Hierbij is $\mathcal{N} = \mathcal{S}\backslash\mathcal{P}$;
- de associatie tussen de set van knopen $\mathcal{P}$ en de volledige set $\mathcal{S}$ van alle knopen in de graaf: $\text{assoc}(\mathcal{P},\mathcal{S}) = \sum_{i:\mathbf{x}_i\in\mathcal{P},j:\mathbf{x}_j\in\mathcal{S}} a_{ij}$.

Gebruik makende van deze definitie en van het feit dat $\mathbf{A}$ een similariteitsmatrix is, is het duidelijk dat een partitie in $\mathcal{P}$ en $\mathcal{N}$ een betere clustering zal voorstellen naarmate de graaf snede $\text{cut}(\mathcal{P},\mathcal{N})$ kleiner is. Dit criterium op zich is echter problematisch: de $\text{cut}(\mathcal{P},\mathcal{N})$ is minimaal voor $\mathcal{P}$ of $\mathcal{N}$ ledig, zodat het minimaliseren van de graaf snede op zich een triviale clustering zou opleveren.

Verschillende oplossingen voor dit probleem zijn reeds voorgesteld in de literatuur. Elk van deze voorstellen houdt een aanpassing van de te optimaliseren kostfunctie in, op dergelijke manier dat opdelingen van de dataset in eerder gelijke delen worden geprefereerd boven sterk asymmetrische opdelingen. Wellicht de empirisch beste kostfunctie is de zogenaamde genormaliseerde graaf snede:

$$\begin{aligned}\text{Ncut}(\mathcal{P},\mathcal{N}) &= \frac{\text{cut}(\mathcal{P},\mathcal{N})}{\text{assoc}(\mathcal{P},\mathcal{S})} + \frac{\text{cut}(\mathcal{N},\mathcal{P})}{\text{assoc}(\mathcal{N},\mathcal{S})} \\ &= \left(\frac{1}{\text{assoc}(\mathcal{P},\mathcal{S})} + \frac{1}{\text{assoc}(\mathcal{N},\mathcal{S})}\right)\cdot \text{cut}(\mathcal{P},\mathcal{N})\end{aligned}$$

met als geassocieerd optimalisatieprobleem voor clustering:

$$\min_{\mathcal{P},\mathcal{N}} \quad \text{Ncut}(\mathcal{P},\mathcal{N}).$$

Helaas vergt het optimaliseren van deze kostfunctie een rekentijd exponentieel in het aantal datapunten, hetgeen onwerkbaar is in alle praktische gevallen. Een manier om toch een benaderende oplossing te vinden voor dergelijke exponentiële problemen wordt geboden door de techniek van het *relaxeren*.

Terwijl de oplossing van het ongerelaxeerde probleem een partitie van $\mathcal{S}$ in $\mathcal{P}$ en $\mathcal{N}$ is (of equivalent hiermee een label vector $\mathbf{y} \in \{-1, 1\}^n$), is de oplossing van het relaxeerde probleem (zoals beschreven in de literatuur en op een alternatieve manier in Sectie 2.4 van deze thesis) een vector $\widetilde{\mathbf{y}} \in \mathbb{R}^n$, die *min of meer* constant is voor datapunten die volgens de optimale label vector $\mathbf{y}$ tot dezelfde klasse behoren. Hoe beter de relaxatie voor het specifieke probleem, hoe kleiner het verschil tussen $\widetilde{y}_i$ en $\widetilde{y}_j$ zal zijn als $y_i = y_j$.

Concreet kan de gerelaxeerde oplossing gevonden worden als de eigenvector horende bij de kleinste van nul verschillende eigenwaarde van het veralgemeende eigenwaardeprobleem $(\mathbf{D} - \mathbf{A})\widetilde{\mathbf{y}} = \lambda \mathbf{D}\widetilde{\mathbf{y}}$, waarbij $\mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1})$ (men kan aantonen dat de kleinste eigenwaarde horende bij $\widetilde{\mathbf{y}}_0$ gelijk is aan nul).

Voor multi-klasse clustering maakt men doorgaans gebruik van de $c-1$ eigenvectoren $\widetilde{\mathbf{y}}_i, i = 1, 2, \ldots, c-1$ horende bij de $c-1$ kleinste van nul verschillende eigenwaarden. Voor elk van deze eigenvectoren zijn de elementen horende bij datapunten van eenzelfde cluster dan min of meer gelijk aan mekaar. Men kan dus een matrix $\widetilde{\mathbf{Y}} = \begin{pmatrix} \widetilde{\mathbf{y}}_1 & \widetilde{\mathbf{y}}_2 & \ldots & \widetilde{\mathbf{y}}_{c-1} \end{pmatrix}$ construeren, waarvan elke rij overeenkomt met een bepaald datapunt, en rijen overeenkomend met datapunten uit dezelfde cluster zullen min of meer gelijk zijn aan mekaar (in die zin dat de Euclidische afstand tussen deze rijen kleiner is voor datapunten die horen tot eenzelfde cluster). Bijgevolg kan men de rijen van de matrix $\widetilde{\mathbf{Y}}$ clusteren met een standaard clusteralgoritme (zoals K-means) om de finale clustering te bekomen.

2.2. Spectrale clustering met beperkingen op de labels

Nu we hebben uitgelegd hoe graaf snede algoritmen voor clustering en hun spectrale relaxaties werken, kunnen we ons richten op de vraag hoe in dergelijke algoritmen zij-informatie aangaande de labels kan worden geïntroduceerd. We nemen aan dat de beperkingen gegeven door de zij-informatie van de vorm zijn dat voor m groepjes van punten (mogelijks singletons) gegeven is dat ze tot dezelfde klasse behoren. In het geval van twee klassen zullen we ook beperkingen toelaten die stellen dat twee zulke groepjes van punten tot verschillende klassen behoren. Zonder verlies van algemeenheid zullen we aannemen dat de rijen van de data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ en dus van $\mathbf{y}$ en $\widetilde{\mathbf{y}}$ zodanig gerangschikt zijn dat datapunten die tot dezelfde klasse behoren volgens de zij-informatie mekaar opvolgen, en dat ook groepjes van datapunten die gespecifieerd zijn als behorende tot een verschillende klasse mekaar opvolgen.

Meerdere klassen Als gegeven is, door zij-informatie, dat twee datapunten $\mathbf{x}_i$ en $\mathbf{x}_j$ tot dezelfde klasse behoren, dan willen we dat in de oplossingsvector $\widetilde{\mathbf{y}}$, $\widetilde{y}_i$ en $\widetilde{y}_j$ aan mekaar gelijk zijn, zeg, gelijk aan z_k.

Dergelijke beperkingen kunnen eenvoudig en constructief worden geintroduceerd in het eigenwaardeprobleem, gebruik makende van de matrix

$$\mathbf{L} = \begin{pmatrix} \mathbf{1}_{s_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{1}_{s_2} & \cdots & \mathbf{0} \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{s_g} \end{pmatrix},$$

waarbij elke blok rij overeenkomt met een groepje van datapunten die tot dezelfde klasse behoren volgens de zij-informatie. Gebruik makende van deze matrix, kunnen we de beperkte versie van $\widetilde{\mathbf{y}}$ schrijven als $\widetilde{\mathbf{y}} = \mathbf{Lz}$, met $\mathbf{z} \in \mathbb{R}^m$. Dit kan eenvoudigweg in het optimalisatieprobleem worden gesubstitueerd, met als resultaat het eigenwaardeprobleem:

$$\mathbf{L}'(\mathbf{D} - \mathbf{A})\mathbf{Lz} = \lambda \mathbf{L}'\mathbf{DLz}.$$

De vector $\widetilde{\mathbf{y}} = \mathbf{Lz}$ zal dan automatisch aan de door de zij-informatie gestelde beperkingen voldoen.

Twee klassen Verder, in het geval van twee klassen, als gegeven is dat $\mathbf{x}_i$ en $\mathbf{x}_j$ tot een verschillende klasse behoren, willen we dat $\widetilde{y}_i = \frac{y_+ - y_-}{2} + \frac{y_+ + y_-}{2}$ en $\widetilde{y}_j = -\frac{y_+ - y_-}{2} + \frac{y_+ + y_-}{2}$ (of omgekeerd). Als we $\mathbf{z}_1 = \frac{y_+ + y_-}{2}$ en $\mathbf{z}_k = \frac{y_+ - y_-}{2}$ definiëren, dan is $\widetilde{y}_i = z_k + z_1$ en $\widetilde{y}_j = -z_k + z_1$. Ook deze beperkingen kunnen worden opgelegd gebruik makende van een eenvoudige substitutie $\widetilde{\mathbf{y}} = \mathbf{Lz}$, nu met de matrix:

$$\mathbf{L} = \begin{pmatrix} \mathbf{1}_{s_1} & \mathbf{1}_{s_1} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_2} & -\mathbf{1}_{s_2} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_3} & \mathbf{0} & \mathbf{1}_{s_3} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_4} & \mathbf{0} & -\mathbf{1}_{s_4} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{1}_{s_{2p-1}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{s_{2p-1}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_{2p}} & \mathbf{0} & \mathbf{0} & \cdots & -\mathbf{1}_{s_{2p}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_{2p+1}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{1}_{s_{2p+1}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_{2p+2}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{1}_{s_{2p+2}} & \cdots & \mathbf{0} \\ \cdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{1}_{s_g} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{s_g} \end{pmatrix}$$

waarbij voor $1 \leq k \leq p$ geldt dat het groepje datapunten behorende bij blok rij $2k$ behoort tot de andere klasse van het groepje behorende bij blok rij $2k - 1$.

Conclusies voor Deel I

We hebben twee semi-gesuperviseerde methoden voorgesteld gebaseerd op eigenwaardeproblemen. De eerste benadering zoekt naar een lager dimensionale representatie in dewelke de zij-informatie gereflecteerd wordt in de afstanden tussen de datapunten. Een clusteralgoritme toegepast op de projectie van de datapunten in deze subruimte zal bijgevolg beter presteren.

De tweede benadering vertrekt van clusteralgoritmen gebaseerd op het optimaliseren van graaf snedes en relaxaties hiervan. We toonden aan hoe het mogelijk is om aan deze algoritmen beperkingen op te leggen overeenkomstig de gegeven zij-informatie. Terwijl de eerste benadering gebaseerd op dimensionaliteitsreductie enkel gelijkheidsbeperkingen tussen labels in rekening kon brengen, kan deze tweede benadering in het geval van twee klassen ook ongelijkheidsbeperkingen uitbuiten.

Deel II: Convexe optimalisatie voor transductie

In het eerste deel van deze nederlandstalige samenvatting hebben we enkel eigenwaardeproblemen behandeld, die de taak van het leren met zij-informatie benaderend kunnen oplossen op een computationeel interessante manier. In een eerste sectie van dit tweede deel zullen we twee alternatieve relaxaties van combinatoriële transductiealgoritmen voorstellen, in dit geval gebaseerd op convexe optimalisatie. De resulterende algoritmen zijn trager dan de spectrale relaxaties, doch, nog steeds op te lossen in polynomiale tijd. Bovendien is de accuraatheid van de methoden gebaseerd op convexe optimalisatie beter, in die zin dat de oplossing die van het ongerelaxeerde probleem beter benadert. In de meeste gevallen betekent dit dan ook de performantie in praktijk beter is.

In een tweede sectie zullen we een gevalstudie toelichten, die we uitgevoerd hebben op een probleem uit de bioinformatica. Het gebruikte algoritme opereert in het transductie scenario, echter, het buit de kennis van de ongelabelde punten niet ten volle uit. Anderzijds maakt het algoritme het mogelijk om, gebruik makende van convexe optimalisatie, verschillende bronnen van informatie te combineren, om zodoende een optimale klassificator te bekomen.

3. SDP relaxaties van transductie

In ons doctoraatsonderzoek hebben we twee methoden voor transductie op basis van semi definiete programmering ontwikkeld: één gebaseerd op support vector machines (SVMs), en één gebaseerd op graaf snedes.

3.1. SDP relaxatie van SVM transductie

Vooraleer we SVM transductie zullen uitleggen, zullen we eerst de details van conventionele SVM inductie bespreken, waarbij enkel een volledig gelabelde

trainingset is gegeven.

SVM inductie De support vector machine (SVM) is een lineaire klassificatiemethode voor problemen met twee klassen, die zoekt naar een hypervlak in de dataruimte $\mathcal{X}$, geparameteriseerd door de gewichtenvector $\mathbf{w}$,[1] zodanig dat punten van beide klassen gescheiden worden door het hypervlak, en dat de minimale afstand van eender welk datapunt tot dit hypervlak zo groot mogelijk is. M.a.w., de *marge* rond dit hypervlak waar zich geen datapunten bevinden is zo groot mogelijk. De motivatie voor de keuze voor dit hypervlak is van statistische aard: men kan aantonen dat, zelfs in hoogdimensionale ruimten $\mathcal{X}$, dergelijk hypervlak met grote waarschijnlijkheid ook ongeziene testpunten correct zal klassificeren, en beter naarmate de marge groter is. [2]

Naast het statistische argument is er ook een computationeel: het optimaliseren van de marge blijkt een vrij eenvoudig convex optimalisatieprobleem te zijn. Het wordt gegeven door:

$$\begin{aligned} \min_{\xi_i, \mathbf{w}} \quad & \frac{1}{2}\mathbf{w}'\mathbf{w} \\ \text{s.t.} \quad & y_i \mathbf{w}'\mathbf{x}_i \geq 1 \end{aligned}$$

Tenslotte kan er voor het SVM optimalisatieprobleem een versie gebaseerd op kernfuncties worden afgeleid. Dit maakt het mogelijk om het SVM algoritme uit te voeren in Hilbertruimten die enkel impliciet zijn gedefinieerd door een kernfunctie (het inwendig product in die Hilbertruimte). Hierdoor kunnen SVMs worden gebruikt voor de klassificatie van vrijwel willekeurige objecten (vectoren, grafen, strings en sequenties,...), zolang er een kernfunctie voor deze objecten kan worden gedefinieerd. In het geval van klassificatieproblemen op vectoren betekent dit dat op eenvoudige manier niet-lineaire klassificatie kan worden uitgevoerd. De kernfunctie formulering van de SVM is als volgt:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & 2\boldsymbol{\alpha}'\mathbf{1} - \boldsymbol{\alpha}'(\mathbf{K} \odot \mathbf{y}\mathbf{y}')\boldsymbol{\alpha} \\ \text{s.t.} \quad & \alpha_i \geq 0 \end{aligned}$$

waarbij $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$ het verband geeft tussen de gewichtenvector en de duale variabelen α_i. Na het oplossen van bovenstaand optimalisatieprobleem op een

[1] Dikwijls staat men ook toe dat het hypervlak niet door de oorsprong gaat, zodat men een bijkomende offset parameter b nodig heeft. In praktijk maakt dit in hoog-dimensionale datasets echter slechts zelden een verschil, en kan het gebruik van deze offset parameter makkelijk vermeden worden.

[2] Dikwijls zal men echter toelaten dat een beperkt aantal punten toch aan de verkeerde kant van het hypervlak terecht komen, of in de marge vallen. Dit is vaak nuttig om niet al te gevoelig te zijn voor uitschieters in de dataset. In dat geval spreekt men van een *zachte marge*. In deze nederlandstalige samenvatting zullen we voor de eenvoud en bondigheid enkel onze resultaten voor *harde* marges rapporteren.

trainingset, kan een test punt geklassificeerd worden in de klasse gegeven door $\text{sign}(\mathbf{w}'\mathbf{x}_{\text{test}}) = \text{sign}(\sum_i \alpha_i y_i \mathbf{x}_i' \mathbf{x}_{\text{test}})$, of, in uitgedrukt in termen van kernfuncties, als $\text{sign}(\sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}_{\text{test}}))$.

SVM transductie Terwijl in het inductiescenario geen testpunten gekend zijn op het moment van training, zijn in het transductiescenario *alle* datapunten gekend waarvoor we het label willen bepalen. De set van deze datapunten wordt in het engels de *working set* genoemd. We zullen hier de vertaling werkset gebruiken.

Het SVM transductie algoritme zoekt, net als in het inductiescenario, naar een hypervlak dat de datapunten van beide klassen scheidt met een zo groot mogelijke marge, echter, dit na het bepalen van de labels van de werkset, zodanig dat de uiteindelijke marge zo groot mogelijk is op de combinatie van trainingset en werkset. Formeel kan het optimalisatieprobleem geassocieerd met SVM transductie dus als volgt worden geschreven:

$$\begin{aligned} \min_{\boldsymbol{\Gamma}} \max_{\boldsymbol{\alpha}} \quad & 2\boldsymbol{\alpha}'\mathbf{1} - \boldsymbol{\alpha}'(\mathbf{K} \odot \boldsymbol{\Gamma})\boldsymbol{\alpha} \\ \text{s.t.} \quad & \alpha_i \geq 0 \\ & \boldsymbol{\Gamma} = \begin{pmatrix} \mathbf{y}^t \\ \mathbf{y}^w \end{pmatrix} \cdot \begin{pmatrix} \mathbf{y}^t \\ \mathbf{y}^w \end{pmatrix}' \\ & y_i^w \in \{1, -1\}, \end{aligned}$$

waarbij we aannemen dat de kernfunctie matrix $\mathbf{K}$ is opgebouwd als $\mathbf{K} = \begin{pmatrix} \mathbf{K}^t & \mathbf{K}^{c\prime} \\ \mathbf{K}^c & \mathbf{K}^w \end{pmatrix}$, met $\mathbf{K}^t$ de kernfunctie evaluaties tussen trainingspunten onderling, $\mathbf{K}^c$ tussen trainingspunten en werkpunten, en $\mathbf{K}^w$ tussen werkpunten onderling. Met $\mathbf{y}^t$ bedoelen we de vector die de gekende labels van de n_t trainingpunten bevat, $\mathbf{y}^w$ bevat de ongekende labels van de n_w werkpunten. De matrix $\boldsymbol{\Gamma}$ noemen we de label matrix.

De rekentijd nodig voor het oplossen van dit optimalisatieprobleem is exponentieel in de grootte n_w van de werkset, door de combinatorische beperking $y_i^w \in \{1, -1\}$ op de werkset labels. Dit maakt het onmogelijk bovenstaand probleem exact op te lossen voor redelijke n_w binnen een aanvaardbare tijd. In deze thesis hebben we daarom een *relaxatie* van bovenstaand optimalisatieprobleem afgeleid. Het relaxeren van een optimalisatieprobleem bestaat erin dat de beperkingen van de variabelen minder strikt worden gemaakt, met als doel op die manier een optimalisatieprobleem te bekomen dat eenvoudiger op te lossen is. Als de relaxatie van de beperkingen voldoende strikt is, blijft het resultaat vaak bruikbaar, zoals ook gebleken is uit onze resultaten.

In bovenstaand optimalisatieprobleem kan men zien dat uit de beperkingen volgt dat $\boldsymbol{\Gamma} \succeq \mathbf{0}$. Bovendien weten we dat $[\boldsymbol{\Gamma}]_{i,j \in \{1:n_t, 1:n_t\}} = y_i^t y_j^t$ en $\text{diag}(\boldsymbol{\Gamma}) =$

1. De relaxatie die we doorvoeren bestaat in het vervangen van de twee laatste beperkingen in de bovenstaande formulering van SVM transductie, door deze drie minder strikte beperkingen. Na enig herschrijven van het resulterende optimalisatieprobleem geeft dit:

$$\begin{aligned} \min_{\mathbf{\Gamma}, \boldsymbol{\nu} \geq \mathbf{0}, t} \quad & t \\ \text{s.t.} \quad & \begin{pmatrix} \mathbf{K} \odot \mathbf{\Gamma} & (\mathbf{1} + \boldsymbol{\nu}) \\ (\mathbf{1} + \boldsymbol{\nu})' & t \end{pmatrix} \succeq \mathbf{0} \\ & \mathbf{\Gamma} \succeq \mathbf{0} \\ & [\mathbf{\Gamma}]_{i,j \in \{1:n_t, 1:n_t\}} = y_i^t y_j^t \\ & \text{diag}(\mathbf{\Gamma}) = \mathbf{1} \end{aligned}$$

Dit is een convex optimalisatieprobleem, met een lineaire objectfunctie en lineaire matrix beperkingen. Dergelijke optimalisatieproblemen zijn zeer goed bestudeerd in de literatuur, en het is bewezen dat ze kunnen opgelost worden in een tijd die polynomiaal is in het aantal variabelen en beperkingen.

Een verdere vereenvoudiging In de hoofdtekst tonen we aan dat de optimale matrix $\mathbf{\Gamma}$ van de volgende vorm is:

$$\mathbf{\Gamma} = \begin{pmatrix} \mathbf{y}^t \mathbf{y}^{t\prime} & \mathbf{y}^t \mathbf{g}' \\ \mathbf{g} \mathbf{y}^{t\prime} & \mathbf{\Gamma}^w \end{pmatrix}$$

met $\mathbf{g} \in \mathbb{R}^{n_w}$. Deze uitdrukking kunnen we dus substitueren in het optimalisatieprobleem. Verder tonen we aan dat een matrix van die vorm semi positief definiet is, enkel en alleen indien:

$$\begin{pmatrix} 1 & \mathbf{g}' \\ \mathbf{g} & \mathbf{\Gamma}^w \end{pmatrix} \succeq \mathbf{0}.$$

In plaats van $\mathbf{\Gamma} \succeq \mathbf{0}$, kunnen we dus deze (eenvoudigere) beperking opnemen in het optimalisatieprobleem.

Schatting van de labels Als gevolg van de relaxatie hebben we nu natuurlijk geen exacte waarde voor de labels meer. Goede schattingen kunnen echter bekomen worden op basis van de dominante eigenvector van $\mathbf{\Gamma}$, of gebruik makende van $\mathbf{g}$ als rechtstreekse schatting voor $\mathbf{y}^w$ (in deze thesis hebben we voor de laatste optie gekozen, aangezien die in praktijk het beste blijkt).

Een subruimte benadering Hoewel het resulterende optimalisatieprobleem gegarandeerd kan opgelost worden in een polynomiale tijd, is de computationele complexiteit nog te groot voor vele realistische problemen. In deze gevallen kan de volgende benadering worden uitgevoerd.

Stel dat we weten dat de label vector zich bevindt in de kolomruimte van de matrix $\mathbf{V} \in \mathbb{R}^{(n_t+n_w)\times d}$. Dan kunnen we de label matrix schrijven als $\mathbf{\Gamma} = \mathbf{VMV}'$, met $\mathbf{M} \in \mathbb{R}^{d\times d}$. Zo een matrix $\mathbf{V}$ kan gevonden worden met behulp van de methode afgeleid in Deel I van deze nederlandstalige samenvatting, voor het leren met zij-informatie gebaseerd op een eigenwaardeprobleem, afgeleid van graaf snede clustering. Door het substitueren van deze parameterisatie voor $\mathbf{\Gamma}$ in het optimalisatieprobleem kan het aantal parameters drastisch worden gereduceerd, zonder teveel aan nauwkeurigheid in te boeten. Bovendien kan de semi positief definiete beperking op $\mathbf{\Gamma}$ dan vervangen worden door de kleinere maar equivalente beperking $\mathbf{M} \succeq \mathbf{0}$.

3.2. SDP relaxatie van transductie gebaseerd op genormaliseerde graaf snedes

In Deel I hebben we een spectrale relaxatie van graaf snede clustering besproken, en op basis hiervan een algoritme voor het leren met zij-informatie afgeleid. Hier rapporteren we een nieuwe relaxatie van het optimalisatieprobleem van de genormaliseerde graaf snede tot een semi definiet programmeringsprobleem. Daarnaast zullen we aantonen hoe het mogelijk is om label informatie in rekening te brengen, om zodoende te kunnen werken in het transductiescenario.

SDP relaxatie van de genormaliseerde graaf snede kost We geven het resultaat, zowel het primale als het duale optimalisatieprobleem, zonder afleiding (hiervoor verwijzen we naar de hoofdtekst):

$$\mathbf{P}^{\mathbf{clust}}_{\mathbf{SDP2}}: \begin{cases} \min\limits_{\widehat{\mathbf{\Gamma}},q} & s\langle \widehat{\mathbf{\Gamma}}, \mathbf{D}-\mathbf{A}\rangle \\ \text{s.t.} & \widehat{\mathbf{\Gamma}} \succeq \mathbf{0} \\ & \operatorname{diag}(\widehat{\mathbf{\Gamma}}) = q\mathbf{1} \\ & \langle \widehat{\mathbf{\Gamma}}, \mathbf{dd}'\rangle = qs^2 - 1 \\ & q \geq \frac{1}{s^2}, \end{cases}$$

$$\mathbf{D}^{\mathbf{clust}}_{\mathbf{SDP2}}: \begin{cases} \max\limits_{\boldsymbol{\lambda},\mu} & \frac{1}{s^2}\mathbf{1}'\boldsymbol{\lambda} \\ \text{s.t.} & s(\mathbf{D}-\mathbf{A}) - \operatorname{diag}(\boldsymbol{\lambda}) \\ & \quad -\mu\mathbf{dd}' \succeq \mathbf{0} \\ & \mu s^2 + \mathbf{1}'\boldsymbol{\lambda} \geq 0. \end{cases}$$

Hierbij is $s = \mathbf{1}'\mathbf{A}\mathbf{1} = \mathbf{1}\mathbf{D}\mathbf{1}$, de som van alle elementen in $\mathbf{A}$. De matrix $\widehat{\mathbf{\Gamma}}$ is de label matrix. Ideaal gezien heeft deze label matrix rang 1, in welk geval de eigenvector horende bij de van nul verschillende eigenwaarde een schatting voor de labels geeft. In praktijk echter zal het nodig zijn om een schatting voor de labels te bekomen gebaseerd op de dominante eigenvector van $\widehat{\mathbf{\Gamma}}$.

We rapporteren hier zowel het primale als het duale probleem, aangezien het duale probleem in dit geval veel minder variabelen bevat, zodat het de

optimalisatie sterk vergemakkelijkt indien gebruik wordt gemaakt van primaal-duale optimalisatiemethoden zoals geimplementeerd in SeDuMi.

Transductie met de genormaliseerde graaf snede kost Zoals in Deel I reeds besproken, is de genormaliseerde graaf snede kost een kostfunctie voor het clusteren van een dataset in twee homogene clusters. Als een deel van de labels echter gekend is, willen we deze informatie in rekening brengen. Dit kan hier op een manier die zeer gelijkaardig is aan hoe dit mogelijk was bij spectrale clustering, zoals beschreven in Deel I van deze nederlandstalige samenvatting. Als we de rijen en kolommen van $\mathbf{A}$ zodanig ordenen dat de gelabelde punten eerst komen en vervolgens pas de ongelabelde, dan kunnen we de label matrix $\widehat{\mathbf{\Gamma}}$ parameteriseren als $\widehat{\mathbf{\Gamma}} = \mathbf{L}\widehat{\mathbf{\Gamma}}_c\mathbf{L}'$, met $\mathbf{L} = \begin{pmatrix} \mathbf{y}_t & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$, waarmee de beperkingen op de label matrix opgelegd door de gegeven labels automatisch en constructief in rekening worden gebracht. Het optimalisatieprobleem wordt dan (primaal en duaal):

$$\mathbf{P}^{\mathbf{trans}}_{\mathbf{SDP2}}: \begin{cases} \min\limits_{\widehat{\mathbf{\Gamma}}_c, q} & s\langle \widehat{\mathbf{\Gamma}}_c, \mathbf{L}'(\mathbf{D}-\mathbf{A})\mathbf{L}\rangle \\ \text{s.t.} & \widehat{\mathbf{\Gamma}}_c \succeq \mathbf{0} \\ & \text{diag}(\widehat{\mathbf{\Gamma}}_c) = q\mathbf{1} \\ & \langle \widehat{\mathbf{\Gamma}}_c, \mathbf{L}'\mathbf{d}\mathbf{d}'\mathbf{L}\rangle = qs^2 - 1 \\ & q \geq \frac{1}{s^2}, \end{cases}$$

$$\mathbf{D}^{\mathbf{trans}}_{\mathbf{SDP2}}: \begin{cases} \max\limits_{\boldsymbol{\lambda}, \mu} & \frac{1}{s^2}\mathbf{1}'\boldsymbol{\lambda} \\ \text{s.t.} & s\mathbf{L}'(\mathbf{D}-\mathbf{A})\mathbf{L} - \text{diag}(\boldsymbol{\lambda}) - \mu\mathbf{L}'\mathbf{d}\mathbf{d}'\mathbf{L} \succeq \mathbf{0} \\ & \mu s^2 + \mathbf{1}'\boldsymbol{\lambda} \geq 0, \end{cases}$$

Een subruimte benadering Zoals bij SVM transductie, kunnen we ook hier een benadering doorvoeren door aan te nemen dat de label vector in een subruimte ligt, opgespannen door de kolommen van een matrix $\mathbf{V}$. Deze matrix kan ook hier bekomen worden door de spectrale transductiemethode afgeleid in Deel I van deze nederlandstalige samenvatting. Voor meer informatie verwijzen we naar de hoofdtekst van deze thesis.

3.3 Welk transductie algoritme in welke situatie?

Het SVM transductiealgoritme blijkt in praktijk het beste te presteren, en heeft bovendien de stevigste statistische basis. Helaas is het het meest rekenintensieve. Daarom is het aan te raden problemen met tot 50 tot 100 ongelabelde datapunten aan te pakken met SVM transductie, en problemen tot 500 of 1000 ongelabelde datapunten met de subruimte benadering van SVM transductie (met een dimensionaliteit d van de subruimte gelijk aan minimaal 3). Van zodra het aantal datapunten de 1000 overschrijdt, is ook de subruimte benadering van SVM transductie te rekenintensief voor een standaard PC, zodat vanaf dan de subruimte benadering van de genormaliseerde graaf snede transductie

aangewezen is. Deze methode is werkbaar op een standaard PC tot ongeveer 5000 datapunten. Voor grotere datasets is de spectrale transductiemethode van Deel I de aangewezen methode.

In alle gevallen is het aangewezen een spaarse affiniteitsmatrix (of kernfunctiematrix) te gebruiken. Spaarsheid kan dikwijls een aanzienlijke snelheidswinst en geheugenbesparing opleveren.

4. Leren met behulp van heterogene informatiebronnen in het transductie scenario: een gevalstudie

In dit hoofdstuk van de nederlandstalige samenvatting bespreken we een toegepast project uitgevoerd tijdens ons doctoraatsonderzoek: het gebruik van een methode die gebaseerd is op convexe optimalisatie om genen te klassificeren. Hierbij hebben we twee klassificatieproblemen beschouwd: het eerste onderscheidt genen die coderen voor membraanproteinen van genen die coderen voor niet-membraanproteinen; het tweede klassificeert genen al naargelang ze coderen voor cytoplasmatische ribosomale proteinen of niet.

Specifiek aan klassificatieproblemen in de bioinformatica is het feit dat dikwijls vele heterogene bronnen van informatie over de te klassificeren objecten (genen in dit geval) voorhanden zijn. Een gen kan gespecifieerd worden door zijn mRNA sequentie, proteine sequentie, gen expressieprofiel,... Op basis van elk van deze bronnen van informatie kan een kernfunctie worden voorgesteld, die genen vergelijkt op basis van de aspecten bepaald door de informatiebron in kwestie. Algoritmen gebaseerd op kernfuncties maken echter gebruik van slechts één kernfunctie. De vraag is dus hoe we, op basis van verschillende kernfuncties, één optimale kernfunctie kunnen construeren.

De methode die we hebben gebruikt in dit project zoekt naar die lineaire combinatie van de gegeven kernfuncties, waarvoor het hypervlak gevonden door een SVM op de resulterende kernfunctie een zo groot mogelijke marge heeft.

Correcter gesteld werkt de methode niet op de kern*functies*, maar op de kernfunctie matrices. M.a.w., gegeven een dataset van punten, en hun verschillende kernfunctie matrices $\mathbf{K}_i$, vindt de methode een kernfunctie matrix $\mathbf{K} = \sum_i \mu_i \mathbf{K}_i$ met $\mathbf{K} \succeq \mathbf{0}$, zodanig dat een SVM toegepast op $\mathbf{K}$ een hypervlak vindt waarvoor de marge maximaal is over alle mogelijke waarden voor μ_i. Aangezien de methode zelf niet ons eigen werk is, verwijzen we voor details naar onze samenvatting ervan in de hoofdtekst van deze thesis.

Zoals hoger gezegd opereert de methode in het transductie scenario. De reden hiervoor is dat de resulterende kernfunctie matrix, geëvalueerd op de trainings-

en testdata samen, semi positief definiet moet zijn om met een geldige kernfunctie overeen te komen. Dit kan slechts gewaarborgd worden als de testpunten (bij transductie meestal werkpunten genoemd) al gekend zijn op het moment van training. We willen wel opmerken dat het algoritme verder de kennis van de testpunten niet uitbuit, zoals onze hoger beschreven algoritmen wel doen.

Resultaten We hebben de volgende experimenten uitgevoerd:

- Voor beide klassificatieproblemen hebben we nagegaan of de lineaire combinatie gevonden door het optimalisatieprobleem beter presteert dan een triviale lineaire combinatie met alle gewichten gelijk aan één. Dit blijkt inderdaad het geval te zijn.

- Voor beide klassificatieproblemen hebben we nagegaan hoe robuust de methode is wanneer random kernfuncties worden toegevoegd aan de set van kernfuncties. Ideaal gezien krijgen deze een gewicht gelijk aan nul. Dit blijkt bij goede benadering ook het geval te zijn.

- We hebben de klassificatie in al dan niet membraanproteinen vergeleken met enkele naïvere methoden en met TMHMM, een state-of-the-art methode voor het herkennen van membraanproteinen. De methode gebaseerd op het combineren van kernfuncties presteert beter dan elk van deze andere methoden.

Voor meer details en numerieke resultaten verwijzen we naar de hoofdtekst.

Conclusies

In deze thesis hebben we ons geconcentreerd rond de vraag hoe we de brug kunnen slaan tussen gesuperviseerde methoden en ongesuperviseerde methoden voor het leren van klassen. Immers, in praktijk komt het dikwijls voor dat bij klassificatieproblemen de testset op voorhand gekend is; dan is het zonde om deze informatie niet te gebruiken. Anderzijds zijn pure clusterproblemen waarbij een hele set van datapunten gegeven is zonder labels in praktijk zeldzaam. Bovendien zijn clusterproblemen, die de data onderverdelen in coherente subgroepen, inherent slecht gedefinieerd (wat is dat, een *coherente* groep van punten?). Het gebruik van eventueel gekende labels of beperkingen op sommige van de labels is dus zeer belangrijk.

Semi-gesuperviseerde problemen zijn intrinsiek echter combinatorieel van aard, zodat ze onoplosbaar zijn voor realistische probleemgroottes. Daarom hebben we ons in een belangrijk deel van deze thesis toegelegd op het herformuleren en het relaxeren van dergelijke problemen tot eigenwaardeproblemen of tot convexe optimalisatieproblemen.

Deze piste lijkt veelbelovend: in de laatste jaren werden belangrijke algoritmische resultaten geboekt aangaande convexe optimalisatiemethoden, en het domein is nog steeds zeer actief. Het mag dus verwacht worden dat optimalisatiemethoden als QCQP, SOCP, en SDP in de toekomst nog aan populariteit zullen winnen. We geloven daarom dat verdere ontwikkelingen in dit domein nog belangrijke aanvullingen gaan leveren aan de gereedschapskist van het domein van machine leren.

Introduction

Roughly 45 years after the introduction of the *perceptron* algorithm by Rosenblatt [90] that spawned the first wave in *neural networks*, 20 years after the invention of the *backpropagation* algorithm (Rumelhart, Hinton and Williams [93]) that led to the second wave of optimism, and about 10 years after the introduction of support vector machines and kernel methods by Vapnik and others [25, 114, 115, 108, 33], the *machine learning* field again seems to be in need of new impulses and ideas. Or did we hit the theoretical limits of *what can be learned*?

Similarly, after the introduction of K-means clustering almost 40 years ago by MacQueen [78], a majority of researchers is still using the same algorithm with reasonable satisfaction. Can we at all do better, does it make sense investigating this problem any further?

After all, improvements over state of the art methods for classification and clustering, arguably the 2 archetypical machine learning problems, seem rather incremental in today's literature, and one might argue that no truly ground breaking conceptual or algorithmic advances in these fields have been seen since long. Does anyone care about the last 0.1% improvement in classification error, or about which of the many clustering algorithms is just a bit better for that one specific data set?

The urgent need for answers to these questions is as imminent as our fear for those answers may be. Somewhere in their minds, many researchers struggle with the feeling that the ultimate goal of research can not (or should not) be to have an impact on their own scientific community, but on society itself. And how is that possible when the end of what is useful in industrial applications and for society has been reached years ago.

Machine learning We have to admit that in the above, we reduced the field of machine learning to just one of its subdomains (see [82] an introductory machine learning book). Indeed, maybe the earliest attempts to build machines that are capable to learn from examples have been made in the field of *inductive inference*. This field is less active in recent years, possibly because of the large number of negative theoretical results, in terms of statistical performance or computational complexity of the algorithms that originated from this domain.

Another interesting line of research was concerned with *learning boolean*

expressions from data, or *concept learning*, with the version space algorithm for learning monomials as its cornerstone. It turns out however that learning more general boolean formula is hard, hence research in this field became less active as well.

Because of the high computational and statistical cost that has to be paid for the generality of learning boolean expressions, it makes sense to focus on a more specific class of problems: the tasks of *learning sets of rules* from data, with decision trees as a special case. The problems considered there are statistically more stable and the algorithms computationally more tractable. This type of methods is often used to solve classification problems, as are the problems in this thesis, however the approach is different than ours.

A currently highly active and promising line of machine learning research is based on *probabilistic graphical models*, and relies on Bayesian inference to make probabilistic statements about data and inferred distributions of the data.

And lastly, an approach originally developed as an analogy to the biological neural system, is referred to as *connectionism*. It has its foundations in the perceptron algorithm, neural networks and support vector machines, and is based on a strong statistical and algorithmic foundation. This is the domain to which this thesis belongs. We complement it in this thesis with graph cut algorithms, that originate from the computer science literature and entered the machine learning literature only in more recent years.

It probably needs no arguing that this list is not exhaustive (we did not mention fuzzy set theory, genetic algorithms, swarm intelligence,...). All these domains are strongly connected, and many of the distinctions are merely due to historical artefacts.

0.1 Motivation

0.1.1 What, why and how?

During our PhD research, we have been confronted very much with the doubts mentioned above: the main problems that we tackled here are indeed related to clustering and classification problems (or under one denominator: *class learning problems*). Luckily, however un-easing they may seem, we can complement these questions with a personal and encouraging answer, largely based on the following three main observations.

First observation: the *data set size*

It cannot be missed that practical data sets have been growing increasingly in the last decades, thanks to better technology that allows for fast acquisition, but not less thanks to faster computer technology that allows to exploit the data in machine learning tasks. It seems however that the growth of the data is more explosive these days than the boost in computing power, and this evolution is unlikely to change when Moore's law will saturate as the limits of electronics are

approached. Then only the algorithmics can make further speed-ups possible. We argue that due to the explosion of data, computational and time complexity of algorithms will become more important than accuracy. This seems to be a general evolution in machine learning: nowadays it is the *size* of the data set that takes care of the *accuracy* and the *statistics*, and the *algorithms* take care of the *size* of the data set.[3]

Therefore, the exploitation of recent advances in optimization is of crucial importance. Some researchers and practitioners will remember the impact linear programming had on entire engineering disciplines among which machine learning, after a few unpopular years. A similar fate may be reserved for recent advances in convex optimization, such as semi-definite programming (SDP), second order cone programming (SOCP) and quadratic programming (QP). In this thesis, we bring these modern optimization methods under the spotlight in a machine learning context.

Second observation: the *problem definition*: supervised, unsupervised or semi-supervised?

In no engineering discipline theory ever matches practice. The same is true in machine learning: during machine learning history, a set of template problems have been identified, modeled and studied, in an attempt to cover a wide variety of practical problems to a sufficient approximation. Two of the most important of these template problems are the classification and clustering problems. Also in this thesis, we concentrated on these two and related problems, as explained below.

In the *classification* problem, a number of *samples* $\mathbf{x}_i \in \mathcal{X}$ are given, along with the corresponding *labels* $y_i \in \mathcal{Y}$ where $\mathcal{Y}$ a countable set. A label y_i tells to which *class* the particular sample $\mathbf{x}_i$ belongs. One distinguishes multi-class problems (where the size of $\mathcal{Y}$ is larger than two) and two-class problems (where usually $\mathcal{Y} = \{1, -1\}$). The set of pairs $\{(\mathbf{x}_i, y_i)\}|_{i=1,n}$ is called the training set of the classification problem. Then, the task solved by classification is to learn from the training set what makes a given sample $\mathbf{x} \in \mathcal{X}$ be a member of one or another class. More specifically, a classification algorithm will use the training set $\{(\mathbf{x}_i, y_i)\}$ to come up with a classification function $h \in \mathcal{H} : \mathcal{X} \to \mathcal{Y}$ that takes $h(\mathbf{x})$ to a hypothesized label y. Of course, in practice this classification function will not be perfect. The accuracy of such a classification algorithm is usually measured by how likely h is to misclassify a randomly drawn test sample.

In *clustering*, only a set of unlabeled samples $\mathbf{x}_i \in \mathcal{X}$ is given, and the task is to partition these samples in so-called *clusters*, each labeled with a label $\in \mathcal{Y}$. After the clusters are identified using the clustering algorithm, the samples can be labeled with the label y_i of the cluster they belong to. Evaluating the

[3] Admittedly, this evolution is not total: for example microarray data sets tend to be rather small in terms of number of experiments, although quickly growing in size.

quality of a clustering result is more difficult than for classification. Usually, some measure of coherence of the different clusters is used: the more coherent they are, the better the clustering.

Since basically there is no difference between the notions *class* and *cluster*, both approaches can together be called *class-learning problems*. Because the classification problem is guided or supervised by the *given* labels y_i, classification is called a *supervised class-learning method*. On the other hand, clustering is called an *unsupervised class-learning method*.

Very recently, here and there in the literature people have made the claim that the traditional distinction between supervised and unsupervised learning methods is too limitative and counterproductive. There are several reasons to abolish this distinction, to be found in statistical learning theory, algorithmics, and applications.

Specifically, very recent results in statistical learning theory indicate that *semi-supervised* learning settings in principle are able to provide an improvement over their supervised or unsupervised alternatives [35]. Algorithmic advances (such as the ones reported in this thesis) show that semi-supervised learning settings can actually be dealt with efficiently. And most importantly and maybe surprisingly, semi-supervised problems actually appear to become the *rule rather than the exception* in practical applications. In the next section, we will explain more formally how supervised, unsupervised and semi-supervised problems are defined.

Third observation: the *diversity of data types*

Along with the explosive growth of data availability, an increasing diversity of the data types can be observed. How to deal with this heterogenity? How to weigh the importance of different sources of data and information? How to extract *common* information or on the other hand to find *complementary* information in each of these sources (two problems that need to be distinguished carefully)?

Also here, examples where this problem setting is of practical importance abound, again not in the least in bioinformatics. Think of all the information we could gather for a gene: its expression profile in microarray experiments, its gene ontology classification, its location in an (often only partially known) genetic network, its location in a protein-protein interaction network, its DNA, mRNA and amino acid sequences, and so on. Clearly all these sources of data are of a totally different kind, and integrating them is not trivial.

0.1.2 Machine learning today: learning from general label information and heterogeneous data, using fast optimization techniques

The first observation above hints at the first main thread in this thesis: fast and convex optimization methods in machine learning. The second and third observations provide another main line in this thesis, unrelated to the first: learning from heterogeneous sources of data, and learning in semi-supervised learning settings. Both types of problems can be seen as learning from general types of data.

We will now briefly motivate both topics in greater detail. To conclude this introductory chapter, an outline of the thesis will be given.

0.2 General learning settings

Here we will go a bit deeper into the different learning settings related to our work in this thesis.

0.2.1 Standard settings for class-learning: classification and clustering

As explained in the previous section, the main problems addressed in this thesis are *class-learning* problems. This means that we want to partition the data, seen as well as unseen data, in separate classes or clusters. Traditionally this is done in two different settings: the classification setting and the clustering setting.

Classification: induction and deduction

In the standard classification approach, a *training set* $|\mathbf{x}_i||_{i=1,n}$ with $\mathbf{x}_i \in \mathcal{X}$ is given, for which the labels $y_i \in \mathcal{Y}$ of all samples are given ($\mathcal{Y}$ usually a set of integers, indicating to which class the sample belongs). Then, in a so-called *induction step*, a hypothesis (or model) h is selected from a class of hypotheses (or models) $\mathcal{H}$ that partitions the training data $\{\mathbf{x}_i\}|_{i=1,n}$ such that samples in one partition have the same label y_i, at least most of the time. In a subsequent *deduction step*, the labels of a so-called *test set* can be deduced by evaluating the model h for the samples in this test set. When the number of misclassifications on the test set is small with high probability, one says that the method generalizes well.

One of the most noteworthy approaches to solve this problem is called *empirical risk minimization* (ERM) [114, 115]. It works by searching a hypothesis $h \in \mathcal{H}$ that minimizes a cost function *on the training set*, and use this model to make predictions on the test set. There are two important assumptions for empirical risk minimization to work.

Probabilistic assumption The first fundamental assumption is that the data, training data as well as test data, are independently sampled from the same (fixed but unknown) distribution. This is called the *iid* property of the data (from *independently and identically distributed*). It is this assumption that allows one to analyze the properties of the hypothesis h found by empirical risk minimization. While this assumption is necessary to obtain sensible results, unfortunately, it merely is an approximation in all practical cases.

Finite VC-dimension The second assumption is that the hypothesis class $\mathcal{H}$ is not too large, or in other words, that it has a modeling capacity that is not too large. It can be proved that generalization is guaranteed if and only if the *VC-dimension* of the hypothesis class is finite. (The VC-dimension is a measure of the complexity of this class [114, 115].)

Formal problem definition The classification problem as it is usually tackled within the machine learning framework can formally be stated as follows:

Given:	$\{(\mathbf{x}_i, y_i)\}\|_{i=1,n}$ the training set, with $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ drawn iid from the distribution $\mathcal{D}_{(\mathbf{x},y)}$.
Asked:	a classification function $h \in \mathcal{H} : \mathcal{X} \to \mathcal{Y}$ such that with high probability $h(\mathbf{x}_{\text{test}}) = y_{\text{test}}$, for $(\mathbf{x}_{\text{test}}, y_{\text{test}})$ independently drawn from $\mathcal{D}_{(\mathbf{x},y)}$.

Then, ERM tackles the problem by solving:

$$h = \text{argmin}_{h \in \mathcal{H}} \sum_{i=1,n} (1 - \delta_{h(\mathbf{x}_i), y_i})$$

where

$$\delta_{a,b} = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{if } a \neq b. \end{cases}$$

So far for the induction step; the deduction step is nothing more than the evaluation of h on a newly given test sample $\mathbf{x}_{\text{test}}$.

Clustering

In clustering, no labels are known beforehand, and the task is to come up with a partitioning of the data that seems plausible in some sense. (Since no labels are given here, clustering is often called an *unsupervised* technique, as opposed to *supervised* techniques such as classification.)

Clustering is statistically more difficult to study, and this is largely due to the fact that the problem is ill-defined. In what sense do we want the clustering to be 'plausible'? Different cost functions can be proposed for this, leading to cluster boundaries that go through regions of low density, leading to clusterings that minimize the total within cluster variance, or many other potential approaches.

However, before the formal cost function is defined, which is some measure of incoherence of the clusters, not much can be done. Exactly this cost function is arbitrary and largely depends on the task at hand.

Formal problem definition The cluster problem can be stated as follows:

Given:	$\{\mathbf{x}_i\}\|_{i=1,n}$ with $\mathbf{x}_i \in \mathcal{X}$.
Asked:	a clustering function $h : \{\mathbf{x}_i\}\|_{i=1,n} \to \mathcal{Y}$ such that for some cost function $c : (\mathcal{X} \times \mathcal{Y})^n \to \mathbb{R}$ measuring cluster incoherence, $c(\{\mathbf{x}_i, h(\mathbf{x}_i)\}\|_{i=1,n})$ is as small as possible.

Then, the clustering function can simply solved as:

$$h = \text{argmin}_{h \in \mathcal{H}} c(\{\mathbf{x}_i, h(\mathbf{x}_i))\}|_{i=1,n})$$

For some c, if iid assumptions are made on the samples $\mathbf{x}_i$, and if the function h is defined on $\mathcal{X}$ instead of on $\{\mathbf{x}_i\}|_{i=1,n} \subset \mathcal{X}$, generalization bounds can be proved in the sense that the incoherence function c remains small when evaluated on a test sample. However, in most practical cases, clustering algorithms are applied to the complete data set, after which no test samples have to be classified anymore.

0.2.2 Transduction

In the standard classification setting we face the problem that we need the unrealistic iid assumption to prove generalization. On the other hand, in clustering it is not clear which cost function one should optimize, as it is largely problem dependent. However, we can do something that merges the advantages of both learning paradigms.

This in-between-solution is offered by transduction as introduced by Vapnik. In the transduction setting, the test set is known beforehand, and we are only interested in the labels of the test set (similar to classification), without any ambition to further generalize towards other unseen samples (similar to the traditional clustering setting). This means that in fact we can merge the induction and deduction steps in one step, called *transduction*. Figure 0.1 provides a pictorial illustration of how transduction may combine features of standard classification and clustering approaches. Note that in transduction settings, the test set is sometimes called the *working set*. For clarity, we will do this in this thesis as well.

Probabilistic assumption Importantly, in this setting no iid assumptions are necessary to prove statistical learning bounds [35]. Here it is sufficient to make the weaker assumption that the training set is a randomly selected subset from the union of the training and test set. Even though also this may be too strong in some practical cases, it is a weaker assumption than iid.

Formal problem definition

Given:	a sample $\{(\mathbf{x}_i)\}\|_{i=1,n}$ with $\mathbf{x}_i \in \mathcal{X}$, and for the samples in a randomly drawn subset $\{\mathbf{x}_i^t\}\|_{i=1,n_t} = \{(\mathbf{x}_i)\}\|_{i=1,n} \setminus \{(\mathbf{x}_i^w)\}\|_{i=1,n_w}$ also the labels $y_i \in \mathcal{Y}$ are given.
Asked:	a function $h : \{\mathbf{x}_i^w\}\|_{i=1,n_w} \to \mathcal{Y}$ such that for as many $\mathbf{x}_i^w$ as possible $h(\mathbf{x}_i^w) = y_i^w$.

Tackling algorithmic aspects of transduction is an important challenge nowadays. This is where a large part of this thesis situates itself.

0.2.3 Incorporating general label information

Whereas in the transduction setting part of the data is labeled and part is unlabeled, we can imagine more general settings where different kinds of constraints on the labels of the samples are given: some labels can be constrained to a fixed value (as in the transduction setting), labels of some specified samples can be constrained to be equal to or different from each other (but unknown), etc.

This more general setting has already found its applications in recommender systems, image and video segmentation, and our belief is that the recent development (in this thesis and in other work) of algorithms dealing with these types of label information will motivate researchers and practitioners to spot these problems in real life situations.

Semi-supervised learning In this thesis, we will call *semi-supervised learning* all methods that situate themselves in between classification and clustering. We should point out however that some authors use a more restrictive definition of semi-supervised learning: they define it as a class of methods for selecting a classification function from a hypothesis class (as in the induction step of a classification algorithm), based on labeled and unlabeled information. According to this definition, transduction would not be a semi-supervised learning method, as it only seeks to classify given samples without interest in the classification function itself. Therefore, we choose to adopt our more general definition: all class learning methods that are based on a given set of data points along with some form of label information for some of these points.

0.2.4 Learning from heterogeneous information

Another non-conventional but increasingly popular learning situation is where, for the objects to be classified or clustered, several heterogeneous data sources are available. Then it is unclear how to weigh the importance of each of these sources, and intelligent techniques need to be developed to ensure the best possible generalization of the resulting classifier towards test samples.

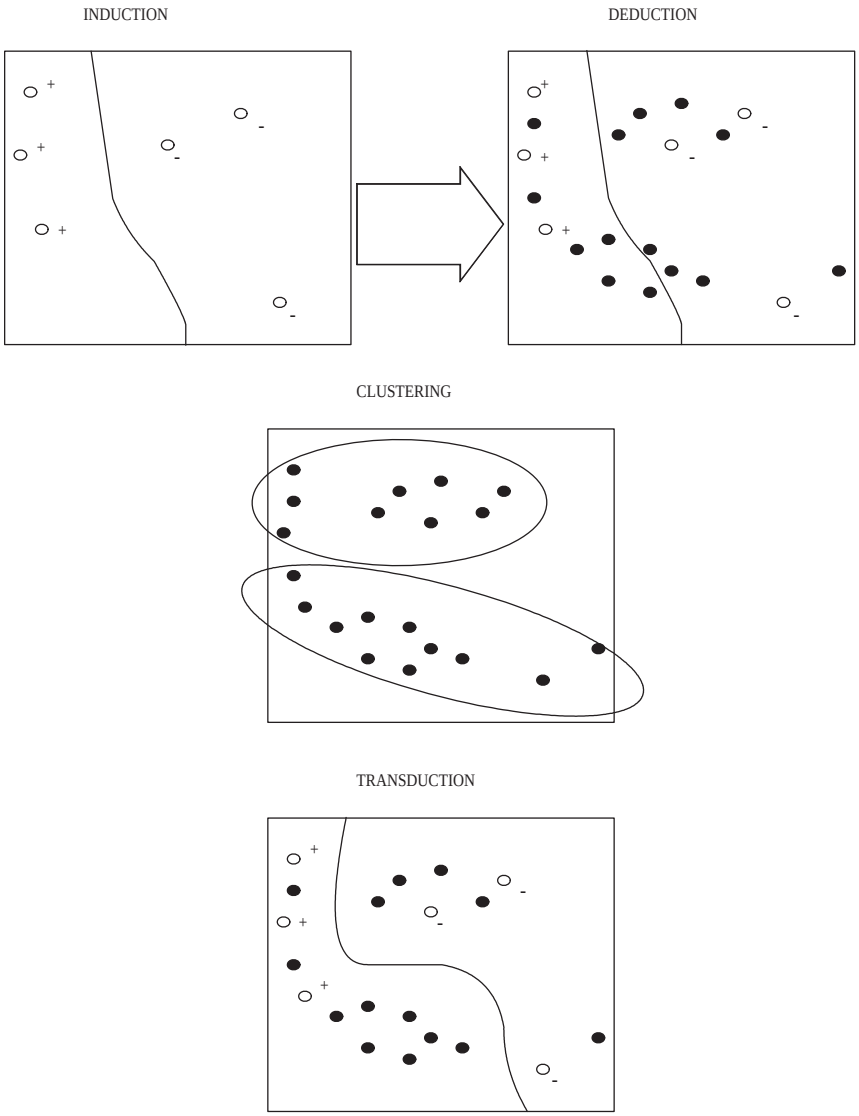


Figure 0.1: The different learning settings on an artificially constructed example. The white dots represent labeled samples (in classification and in transduction). The black dots are the unlabeled samples. The + and - signs indicate to which class the training samples belong to. Of course for this toy example, there is no such thing as a 'best' hypothesis. Still, by looking at these pictures, the reader may get some intuition of why transduction is probably a good (if not the best) choice when the test set is known beforehand. Indeed, transduction comes up with a partitioning of the data that respects the known labels as good as possible, while at the same time keeping the classes as coherent as possible.

0.3 Machine learning and optimization

As argued above, the use of fast optimization techniques becomes more and more a necessity in today's applications. We will distinguish two important types of methods to achieve this speed: eigenvalue problems and recent convex optimization techniques. In this thesis, problems that are reduced to an eigenvalue problem or to a standard formulation of a convex optimization problem will be considered to be solved.

0.3.1 Eigenvalue problems

One of the most well-known techniques used in machine learning is Principal Component Analysis (PCA) [57]. It is used as a preprocessing step to perform dimensionality reduction of high dimensional data sets, in order to improve performance of subsequently used algorithms. The success of PCA is arguably due to its simplicity. It is merely based on a simple eigenvalue problem, which is since long a standard tool in linear algebra.

Interestingly, PCA was developed in the statistics community long before machine learning was born as a field. Another early technique that originated in the statistics community is Canonical Correlation Analysis (CCA) [58]. Again, CCA can be formulated as a (generalized) eigenvalue problem, hence part of its success.

Whereas PCA and CCA are essentially dimensionality reduction techniques, other eigenvalue problems are used in classification and in clustering. Fisher's Linear Discriminant Analysis (FDA) [40] is a popular technique for classification that (again) was invented long ago, while Spectral Clustering (SC) recently was introduced by a number of authors as a promising alternative to standard clustering techniques such as K-means and Bayesian techniques.

Whereas most of these methods (with the exception of SC) are actually quite old and at best rediscovered in the machine learning community, their applicability was greatly extended in recent years: whereas all these techniques are essentially linear, the development of a *kernel* approach to machine learning made it possible to extend each of these methods to find nonlinear relations in data as well. Even more strongly, by using recent sophisticated kernels, not only (real) vectorial data can be analyzed, but virtually any kind of data, such as nodes in a tree or graph, trees and graphs themselves, sequences and strings,... Therefore, the discussion of these eigenvalue problems in relation to their kernel versions is crucial.

0.3.2 Convex optimization problems

Another class of useful optimization problems is the class of convex optimization problems [26] such as the Semi-Definite Programming (SDP), Second Order Cone Programming (SOCP) and Quadratic Programming (QP) problems. These three examples are of extreme importance, since for them it is proved that the global optimum can be found in polynomial time.

Again, many machine learning problems can be reformulated as one of these standard convex optimization problems, the most notable of which is the Support Vector Machine (SVM), a QP problem.

At this point we also want to point out the particular role played by SDP. Importantly, SDP can handle linear matrix inequalities (e.g. constraining a linearly parameterized matrix to be positive definite). Since kernel matrices are semi positive definite, and since kernels play a fundamental role in many state of the art machine learning problems, SDP promises to play a fundamental role in future methods. Apart from this, SDP also allows to come up with tight relaxations of combinatorial problems such as transduction. It is mainly this second aspect of SDP that will be invoked in this thesis.

0.4 Outline of the thesis

The text will be divided into three large Parts. The division into the first two Parts is motivated by the optimization methods used in the algorithms discussed. Part I handles about eigenvalue problems in machine learning; Part II about convex optimization in machine learning. When we say machine learning, in both Parts we aimed at extending the traditional classification and clustering settings towards side-information learning.

In Part III we discuss two other problems addressed during the research that led to this thesis. The first one is about a theoretical study of the meaning of a nowadays commonly used regularization technique for CCA. The second one is about a method to learn from heterogeneous data, applied to a bioinformatics problem. Both chapters are placed in a separate Part, because they either differ in the nature of the problem solved (the first of both chapters: it is theoretical in nature and not algorithmic), or in the technology used (the second of both chapters: no machine learning approach is used there).

The structure of the chapters is shown pictorially in figure 0.2. This structuring in Parts is largely based on the machinery.

An alternative structuring is shown in figure 0.3. It is orthogonal to the physical structure of this text, in the sense that the chapter ordering of this structure is only weakly correlated with the actual chapter ordering. We advise the reader to keep it in mind while reading.

0.5 Personal contributions

Here we give a brief overview of our own research contributions in this thesis. Chapter 1 and 4 are general introductory chapters, introducing the reader to the necessary background. The other chapters contain either tutorial contributions (chapter 2) or research contributions (chapters 3, 5, 6, 7 and 8), as explained below:

- Chapter 2 contains a *tutorial contribution*, about eigenvalue problems in *multivariate statistics* and *machine learning* in the broad sense, seen from

a primal dual perspective. Topics covered are principal component analysis, canonical correlation analysis, partial least squares, Fisher's discriminant analysis and spectral clustering. While this chapter does not contain any research contributions, it should be valued as a comprehensive and unified discussion of these methods.
Relevant publications: [16, 15].

- In Chapter 3 we report our work on the use of *eigenvalue problems to solve semi-supervised learning problems*. Two different approaches are considered, the first of which is based on dimensionality reduction, the second of which directly incorporates general label information in the learning algorithm.
Relevant publications: [18, 21].

- Solving the transduction problem, an important special case of semi-supervised learning, requires a time exponential in the size of the test set. Therefore, approximation or relaxation algorithms are required in order to make its (approximate) solution feasible in practice. In Chapter 5, we introduce several approaches to this problem, some based on density estimation, others based on relaxing the transduction problem to a convex optimization problem.
Relevant publication and technical report: [13, 14].

- Chapter 6 contains an example of another potential use of convex optimization in machine learning, for integrating heterogeneous data sources in a classification context. Our work consists of demonstrating the usefulness of this approach on a bioinformatics problem.
Relevant publication: [70].

- While the previous chapters mostly deal with algorithmic aspects, in Chapter 7 we deal with a theoretical interpretation of the regularization of canonical correlation analysis. More specifically, we show how this regularization can be related to the existence of a noise term in a model based approach to canonical correlation analysis. Furthermore, a connection between canonical correlation analysis and independent component analysis is made clear.
Relevant publication: [20].

- Lastly, in chapter 8 we propose a (non-machine learning) approach to deal with heterogeneous information sources, in the context of *regulatory module inference*, an important bioinformatics problem. The method is based on advanced database techniques.
Relevant publication: [19].

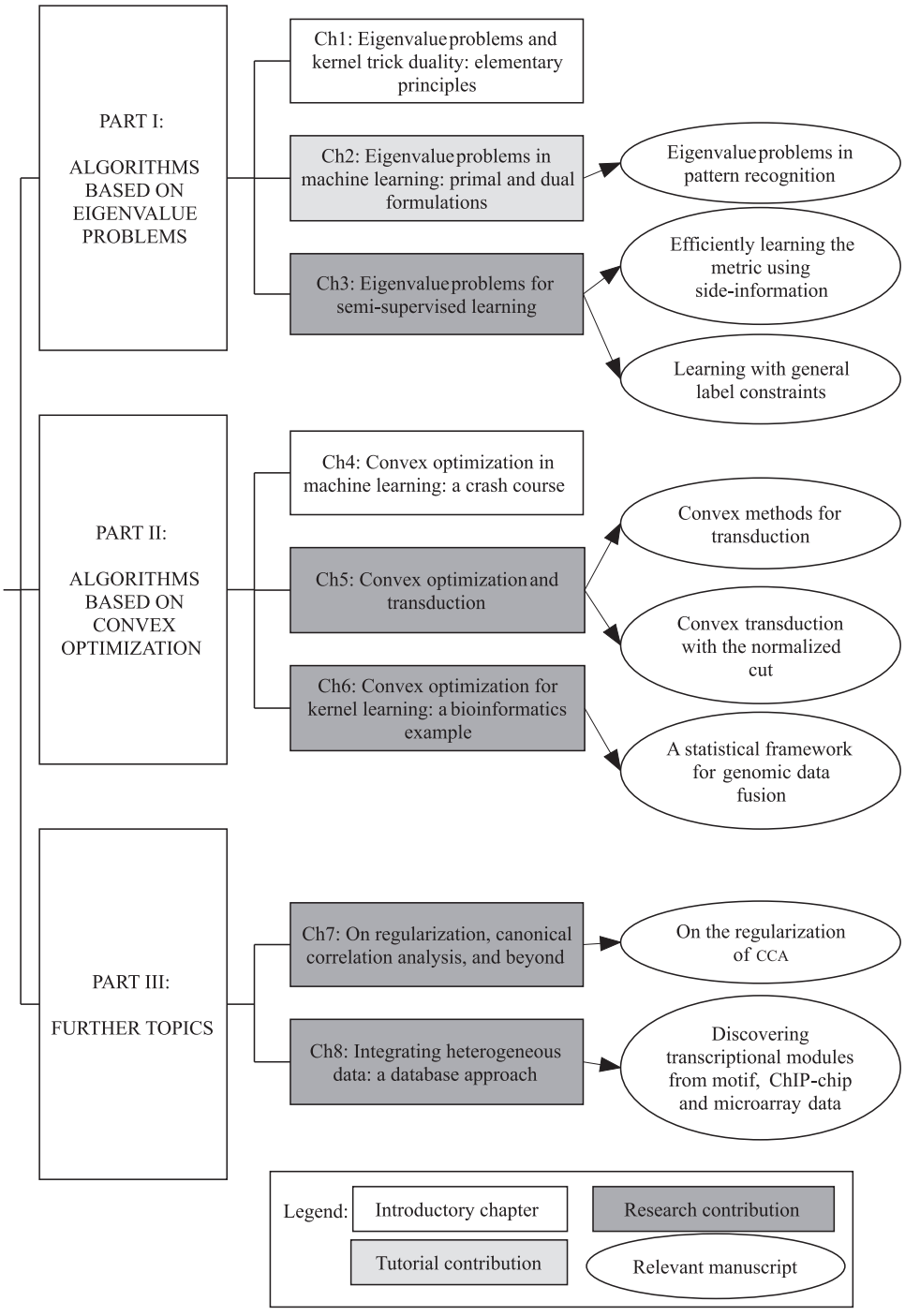


Figure 0.2: A schematic overview of the chapter structure.

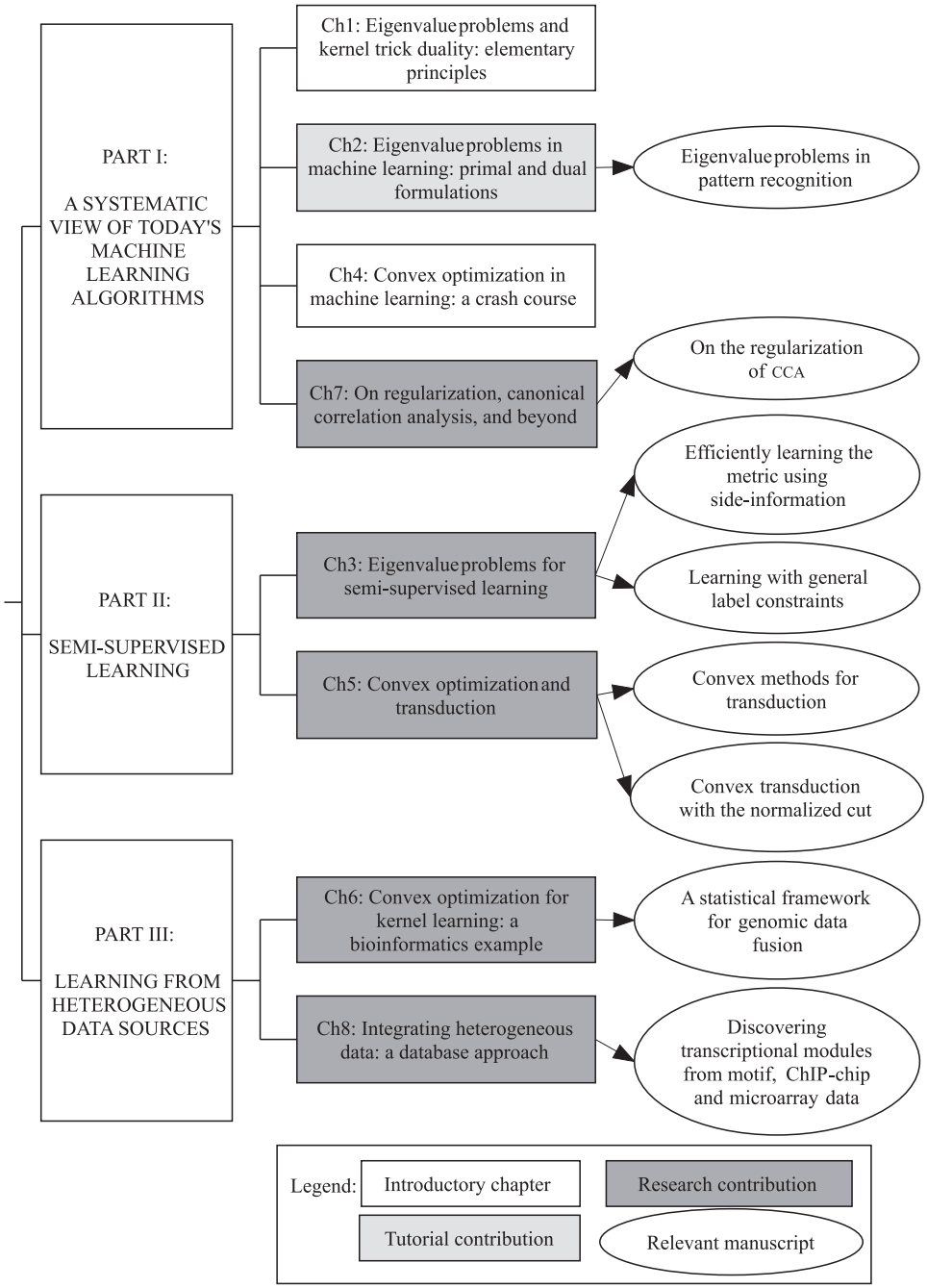


Figure 0.3: An orthogonal structure in the chapters. While this would be a logical chapter ordering as well, it is only weakly correlated with the actual chapter ordering.

Part I

Algorithms based on eigenvalue problems

Chapter 1

Eigenvalue problems and kernel trick duality: elementary principles

Studying the (properties of) configurations of points embedded in a metric space has long been a central task in pattern recognition, but has acquired an even bigger importance after the recent introduction of kernel-based learning methods. These work by virtually embedding general types of data in a vector space, and then analyzing the properties of the resulting data cloud. While a number of techniques for this task have been developed in fields as diverse as multivariate statistics, neural networks, and signal processing, many of them show an underlying unity. In this chapter we describe a large class of pattern analysis methods based on the use of Generalized Eigenproblems, that reduce to solving the equation $\mathbf{Aw} = \lambda\mathbf{Bw}$ with respect to $\mathbf{w}$ and λ, where $\mathbf{A}$ and $\mathbf{B}$ are real symmetric square matrices and $\mathbf{B}$ is semi positive definite.

The problems in this class range from finding a set of directions in the data-embedding space containing the maximum amount of variance in the data (Principal Component Analysis), to finding a hyperplane that separates two classes of data minimizing a certain cost function (Fisher Discriminant), or finding correlations between two different representations of the same data (Canonical Correlation Analysis). Also some important clustering algorithms lead to solving eigenproblems. The importance of this class of algorithms derives from the facts that: generalized eigenproblems provide an efficient way to optimize an important family of cost functions, of the type $f(\mathbf{w}) = \frac{\mathbf{w}'\mathbf{A}\mathbf{w}}{\mathbf{w}'\mathbf{B}\mathbf{w}}$ (known as a *Rayleigh quotient*); they can be studied with standard linear algebra; and they can be solved or approximated efficiently using a number of well known techniques from numerical algebra.

Their statistical behavior has also been studied to some extent (e.g. [99]

and [100]) and to a greater extent for Gaussian distribution [4], allowing us to efficiently design regularization strategies in order to reduce the risk of overfitting. However, methods limited to detecting linear relations among vectors could hardly be considered to constitute state-of-the-art technology, given the nature of the challenges presented by modern data analysis. Therefore it is crucial that all such problems can be cast in a formulation that only makes use of inner products between the samples, after which the kernel trick can be applied. The resulting kernel versions of the generalized eigenproblems can then be applied to the detection of generalized and nonlinear relations on a wide range of data types apart from vectors, such as sequences, text, images, etc.

In this chapter we will first review the general theory of eigenvector problems, then we will give a brief review of kernel methods in general, and finally we will discuss a number of algorithms based in multivariate statistics: Principal Component Analysis, Partial Least Squares, Canonical Correlation Analysis, Fisher Discriminant and Spectral Clustering, where appropriate both in their primal and in their dual form, leading to a version involving kernels.

1.1 Some basic algebra

In this section, we will shortly review some basic properties of linear algebra that prove to be useful in this chapter [43, 54, 55, 88, 122]. We use the standard linear algebra notation in the beginning, and translate the important results to the kernel methods conventions afterwards. An extensive reference for matrix analysis can be found in [54] and [55].

1.1.1 Symmetric (Generalized) Eigenvalue Problems

Variational characterization

The optimization problems we are concerned with in this chapter are all basically of the form (with $\mathbf{M} = \mathbf{M}' \in \mathbb{R}^{n \times n}$, $\mathbf{N} = \mathbf{N}' \in \mathbb{R}^{n \times n}$, and $\mathbf{N} \succ \mathbf{0}$ positive definite):

$$\max_{\mathbf{w}} \frac{\mathbf{w}'\mathbf{M}\mathbf{w}}{\mathbf{w}'\mathbf{N}\mathbf{w}} \text{ with } \mathbf{w} \in \mathbb{R}^n.$$

This is an optimization of a Rayleigh quotient. One can see that the norm of $\mathbf{w}$ does not matter as scaling $\mathbf{w}$ does not change the value of the object function. Thus, one can impose an additional scalar constraint on $\mathbf{w}$, and optimize the object function, without losing any solutions. This constraint is chosen to be $\mathbf{w}'\mathbf{N}\mathbf{w} = 1$. Then the optimization problem becomes a constrained optimization problem of the form:

$$\max_{\mathbf{w}} \quad \mathbf{w}'\mathbf{M}\mathbf{w} \quad \text{s.t.} \quad \mathbf{w}'\mathbf{N}\mathbf{w} = 1,$$

or by using the Lagrangian $\mathcal{L}(\mathbf{w}, \lambda)$:

$$\max_{\mathbf{w}} \qquad \mathcal{L}(\mathbf{w}) = \max_{\mathbf{w},\lambda} \left[\mathbf{w}'\mathbf{M}\mathbf{w} - \lambda(\mathbf{w}'\mathbf{N}\mathbf{w} - 1)\right].$$

Equating the first derivative to zero leads to

$$\mathbf{M}\mathbf{w} = \lambda\mathbf{N}\mathbf{w}. \tag{1.1}$$

The optimal value reached by the object function is equal to the maximal *eigenvalue*, the Lagrange multiplier λ, for which this equation holds.

This is the symmetric generalized eigenvalue problem that will be studied here.

Note that the vector $\mathbf{w}$ with the scalar λ leading to the optimum of the Rayleigh quotient is not the only stationary point of the generalized eigenvalue problem (1.1). There exist other eigenvector–eigenvalue pairs that do not correspond to the optimum of the Rayleigh quotient. For any pair $(\mathbf{w}, \lambda)$ that is a solution of (1.1), $\mathbf{w}$ is called a (generalized) eigenvector and λ is called a (generalized) eigenvalue. In many cases several of these eigenvector–eigenvalue pairs will be of interest.

We will now first discuss the properties of symmetric ordinary eigenvalue problems, where $\mathbf{N} = \mathbf{I}$. Afterwards, it will be easy to derive the properties of symmetric generalized eigenvalue problems by using a simple transformation of the variables.

Symmetric eigenvalue problems

For the *ordinary* symmetric eigenvalue problem we have that $\mathbf{N} = \mathbf{I}$, and equation (1.1) reduces to:

$$\mathbf{M}\mathbf{w} = \lambda\mathbf{w}.$$

Theorem 1.1 *i. Eigenvectors $\mathbf{w}_i$ corresponding to different eigenvalues λ_i are orthogonal to each other.*

ii. Furthermore, the eigenvalues of symmetric matrices are real, and a real eigenvector corresponds to them.

Proof i. For $\lambda_i \neq \lambda_j$,

$$\begin{aligned} \mathbf{M}\mathbf{w}_i &= \lambda_i\mathbf{w}_i \\ \Rightarrow \lambda_i(\mathbf{w}_j'\mathbf{w}_i) &= \mathbf{w}_j'\mathbf{M}\mathbf{w}_i = \mathbf{w}_i'\mathbf{M}'\mathbf{w}_j = \mathbf{w}_i'\mathbf{M}\mathbf{w}_j \\ &= \lambda_j(\mathbf{w}_i'\mathbf{w}_j) \\ \Rightarrow \mathbf{w}_j'\mathbf{w}_i &= 0. \end{aligned}$$

Thus, eigenvectors corresponding to different eigenvalues λ_i and λ_j are orthogonal.

ii. And, with $\cdot^*$ the adjoint operator:

$$\begin{aligned}
\mathbf{M}\mathbf{w}_i &= \lambda_i \mathbf{w}_i \\
\Rightarrow \lambda_i^* \mathbf{w}_i^{*\prime} \mathbf{w}_i &= (\lambda_i \mathbf{w}_i{}^{\prime} \mathbf{w}_i^*)^* = (\mathbf{w}_i^{\prime} \mathbf{M} \mathbf{w}_i^*)^* = \mathbf{w}_i^{\prime} \mathbf{M} \mathbf{w}_i^* \\
&= (\mathbf{w}_i^{*\prime} \mathbf{M}^{\prime} \mathbf{w}_i)^{\prime} = (\mathbf{w}_i^{*\prime} \mathbf{M} \mathbf{w}_i)^{\prime} = \lambda_i (\mathbf{w}_i^{*\prime} \mathbf{w}_i)^{\prime} \\
&= \lambda_i \mathbf{w}_i^{*\prime} \mathbf{w}_i \\
\Rightarrow \lambda_i &= \lambda_i^*,
\end{aligned}$$

so the eigenvalues of a symmetric matrix are real. Furthermore, the eigenvectors are real up to a complex scalar (and can thus be made real by scalar multiplication), since if they were not, we could take the real part and the imaginary part separately, and both would be real eigenvectors corresponding to the same eigenvalue. ∎

When eigenvalues are *degenerate*, that is they are equal but correspond to a different eigenvector, then these eigenvectors can be chosen to be orthogonal to each other. This follows from the fact that they are in a subspace orthogonal to the space spanned by all eigenvectors corresponding to the other eigenvalues. In this subspace an orthogonal basis can be found.

The number of eigenvalues and corresponding orthogonal eigenvectors of a real symmetric matrix thus is equal to the dimensionality n of $\mathbf{M}$.

If we normalize all eigenvectors $\mathbf{w}_i$ to unit length and choose them to be orthogonal to each other, they are said to form an *orthonormal* basis. For $\mathbf{W}$ being the matrix built by stacking these normalized eigenvectors $\mathbf{w}_i$ next to each other, we have

$$\mathbf{W}\mathbf{W}^{\prime} = \mathbf{W}^{\prime}\mathbf{W} = \mathbf{I},$$

that is the matrix $\mathbf{W}$ is *orthogonal*.

Since then $\mathbf{M}\mathbf{w}_i = \mathbf{w}_i \lambda_i$ for all i, we can state that

$$\mathbf{M}\mathbf{W} = \mathbf{W}\mathbf{\Lambda},$$

where $\mathbf{\Lambda}$ contains the corresponding eigenvalues λ_i on its diagonal. Then, taking into account that $\mathbf{W}^{-1} = \mathbf{W}^{\prime}$, we can express the matrix $\mathbf{M}$ as:

$$\mathbf{M} = \mathbf{W}\mathbf{\Lambda}\mathbf{W}^{\prime} = \sum_i \lambda_i \mathbf{w}_i \mathbf{w}_i^{\prime}.$$

This is called the *eigenvalue decomposition* of the matrix $\mathbf{M}$, also known as the *spectral decomposition* of $\mathbf{M}$.

Symmetric generalized eigenvalue problems

In general, we will deal with the generalized eigenvalue problem of the form of equation (1.1):

$$\mathbf{M}\mathbf{w} = \lambda \mathbf{N}\mathbf{w}.$$

This could be solved as an ordinary but non-symmetric eigenvalue problem (by multiplying with $\mathbf{N}^{-1}$ on the left hand side). We can also convert it to a symmetric eigenvalue problem by defining[1] $\mathbf{v} = \mathbf{N}^{1/2}\mathbf{w}$:

$$\mathbf{M}\mathbf{N}^{-1/2}\mathbf{N}^{1/2}\mathbf{w} \quad = \quad \lambda\mathbf{N}^{1/2}\mathbf{N}^{1/2}\mathbf{w}$$

and thus by left multiplication with $\mathbf{N}^{-1/2}$

$$(\mathbf{N}^{-1/2}\mathbf{M}\mathbf{N}^{-1/2})\mathbf{v} \quad = \quad \lambda\mathbf{v}.$$

For this type of problems, we know that the different eigenvectors $\mathbf{v}_i$ can be chosen to be orthogonal and of unit length, thus:

$$\mathbf{V}'\mathbf{V} = \mathbf{I} = \mathbf{W}'\mathbf{N}\mathbf{W},$$

which means that the generalized eigenvectors $\mathbf{w}_i$ of a symmetric eigenvalue problem are orthogonal in the metric defined by $\mathbf{N}$.

1.1.2 Singular Value Decompositions, Duality

The singular value decomposition of a real rectangular matrix $\mathbf{A} \in \mathbb{R}^{n\times m}$ is defined as

$$\mathbf{A} = \begin{pmatrix} \mathbf{U} & \mathbf{U}^\perp \end{pmatrix} \begin{pmatrix} \mathbf{\Sigma} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{V} & \mathbf{V}^\perp \end{pmatrix}' = \mathbf{U}\mathbf{S}\mathbf{V}'$$

where $\mathbf{\Sigma} \in \mathbb{R}^{r\times r}$ contains the r non-zero *singular values* σ_i in non-increasing order (by convention) on the diagonal, r is the rank of $\mathbf{A}$. Dimensions of all blocks are compatible. The matrices $\begin{pmatrix} \mathbf{U} & \mathbf{U}^\perp \end{pmatrix}$ and $\begin{pmatrix} \mathbf{V} & \mathbf{V}^\perp \end{pmatrix}$ are orthogonal matrices, respectively containing the *left* and the *right singular vectors* as their columns. This decomposition exists for any real matrix.

One can see that multiplying $\mathbf{A}$ on the left with a column of $\mathbf{U}^\perp$ gives zero: ${\mathbf{U}^\perp}'\mathbf{A} = \mathbf{0}$. Therefore, the column space of $\mathbf{U}^\perp$ is said to span the *left null space* of $\mathbf{A}$. Similarly, $\mathbf{V}^\perp$ is a basis for the *right null space* of $\mathbf{A}$: $\mathbf{A}\mathbf{V}^\perp = \mathbf{0}$. On the other hand, $\mathbf{U}$ and $\mathbf{V}$ respectively span the *column and row spaces* of $\mathbf{A}$.

Note that $\mathbf{A}\mathbf{A}'$ and $\mathbf{A}'\mathbf{A}$ are symmetric, and their eigenvalue decompositions are:

$$\begin{aligned} \mathbf{A}\mathbf{A}' &= \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}' \\ \mathbf{A}'\mathbf{A} &= \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}'. \end{aligned}$$

[1] The symmetric square root $\mathbf{N}^{-1/2}$ of a symmetric positive definite matrix $\mathbf{N}$ is defined as the symmetric matrix $\mathbf{S} = \mathbf{S}'$ for which: $\mathbf{S}^2 = \mathbf{S}\mathbf{S} = \mathbf{N}$. It can be shown that, given the eigenvalue decomposition of $\mathbf{N}$ as $\mathbf{N} = \mathbf{V}_\mathbf{N}\mathbf{\Lambda}_\mathbf{N}\mathbf{V}'_\mathbf{N}$, this square root exists and is (up to a sign) equal to $\mathbf{S} = \mathbf{V}_\mathbf{N}\mathbf{\Lambda}_\mathbf{N}^{-1/2}\mathbf{V}'_\mathbf{N}$. Thus, the symmetric square root $\mathbf{S}$ is the square root (meaning that $\mathbf{S}'\mathbf{S} = \mathbf{N}$) and the *false* square root (meaning that $\mathbf{S}^2 = \mathbf{N}$) at the same time.

Another important property of singular value decompositions is that the nonzero singular values and corresponding singular vectors are the nonzero eigenvalues and corresponding eigenvectors of the matrix $\begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}' & \mathbf{0} \end{pmatrix}$:

$$\begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}' & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u}_i \\ \pm\mathbf{v}_i \end{pmatrix} = \pm\sigma_i \begin{pmatrix} \mathbf{u}_i \\ \pm\mathbf{v}_i \end{pmatrix}, \tag{1.2}$$

the solution of which leads to the singular value decomposition of $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}'$.[2]

Thus far, we used the standard linear algebra notation. Note that further in this thesis, we will use notation that is commonly used in kernel methods.

1.2 Kernel methods: the duality principle

Kernel Methods (KM) ([115, 33, 94, 108]) form a relatively new family of algorithms that presents a series of useful features for pattern analysis in data sets. In recent years, their simplicity, versatility and efficiency have made them a standard tool for practitioners, and a fundamental topic in many data analysis courses and applications. We will outline some of their important features, referring the interested reader to more detailed articles and books for a deeper discussion (see for example [33] and references therein).

KMs combine the simplicity and computational efficiency of linear estimation algorithms, such as the perceptron algorithm or ridge regression, with the flexibility of nonlinear models, such as for example neural networks, and the rigor of statistical approaches such as regularization methods in multivariate statistics. As a result of the special way they represent functions, these algorithms typically reduce the learning step to a simple optimization problem, that can always be solved in polynomial time, avoiding the problem of local minima typical of neural networks, decision trees and other nonlinear approaches.

Their foundation in the principles of Statistical Learning Theory make them remarkably robust to overfitting especially in regimes where other methods are affected by the 'curse of dimensionality'.[3]

Another important feature for applications is that they can naturally accept input data that are not in the form of vectors, such as for example strings, trees and images. Their characteristically modular design makes them amenable to theoretical analysis but also well suited to a software engineering approach: a general purpose learning module is combined with a data specific 'kernel function' that provides the interface with the data and incorporates domain knowledge.

[2] This second eigenvalue problem is numerically the best way to compute the singular value decomposition of $\mathbf{A}$.

[3] The fact that a volume scales exponentially with the number of dimensions, makes it very hard for learning algorithms to learn in large dimensional spaces. Indeed, the larger the space, the more data is needed to gather sufficient information about the entire space, and to find reliable patterns. This problem is referred to as the curse of dimensionality.

Many learning modules can be used depending on whether the task is one of classification, regression, clustering, novelty detection, ranking, etc. At the same time many kernel functions have been designed: for protein sequences, for text and hypertext documents, for images, time series, etc. The result is that this method can be used for dealing with rather exotic tasks, such as ranking strings, or clustering graphs, in addition to such classical tasks as classifying vectors. (See [98] and references therein.)

In the remainder of this section, we will shortly describe the theory behind kernel methods, followed by a brief example of how this can be used in practice: kernelizing least squares regression and ridge regression.

1.2.1 Theory

Kernel-based learning algorithms work by embedding the data into a Hilbert space, and searching for *linear* relations in this Hilbert space. This is generally an easier task than immediately looking for nonlinear relations among the data in the input space directly (the term *input space* is used to denote the space in which the data are explicitly represented). The embedding in the Hilbert space is performed implicitly, that is by specifying the inner product between each pair of points rather than by giving their coordinates explicitly. This approach has several advantages, the most important being the fact that often the inner product in the embedding space can be computed much more easily than the coordinates of the points themselves.

Given an input space $\mathcal{X}$, and an embedding Hilbert space $\mathcal{F}$ (often called the feature space), we consider a map $\phi : \mathcal{X} \to \mathcal{F}$ (often called the feature map). The function that, given two samples $\mathbf{x}_i \in \mathcal{X}$ and $\mathbf{x}_j \in \mathcal{X}$, returns the inner product between their images $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ in the space $\mathcal{F}$ is known as *kernel function*.

Definition: A *kernel* is a function k, such that for all $\mathbf{x}, \mathbf{z} \in \mathcal{X}$, $k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$, where ϕ is a mapping from $\mathcal{X}$ to a Hilbert space $\mathcal{F}$, and $\langle \cdot, \cdot \rangle$ denotes the inner product.

In fact, *Mercer's theorem*, stating that *every symmetric, semi positive definite bilinear function can be represented as an inner product in a feature space*, allows one to make abstraction of this feature space, and to simply use semi positive definite kernel functions. Thus the exact form of the feature vectors $\phi(\mathbf{x})$ does not have to be known.

We also consider the matrix $\mathbf{K}$ with elements $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, called the *kernel matrix* or the *Gram matrix*. Because of the fact that it is built from inner products, it is always a symmetric semi positive definite matrix, and since it specifies the inner products between all pairs of points, it completely determines the relative positions between those points in the embedding space (for example: given such information, it is trivial to recover all the pairwise distances between them).[4]

[4]Notice that we do not need $\mathcal{X}$ to be a vector space. This is because the distances are computed in the Hilbert space into which the data points are mapped by the feature map, and

The pattern functions (be it classification functions, regression functions or something else) sought by kernel-based algorithms are linear functions in the feature space:

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle,$$

for some vector $\mathbf{w} \in \mathcal{F}$, which will be called the *weight vector*. In the linear algebra notation, which is more common in the kernel methods literature and will hence be used throughout this thesis, this would be denoted as:

$$f(\mathbf{x}) = \mathbf{w}'\phi(\mathbf{x}).^5$$

The kernel can be exploited whenever the weight vector can be expressed as a linear combination of the training points, $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \phi(\mathbf{x}_i)$, implying that we can express f as follows:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}).$$

The vector $\boldsymbol{\alpha}$ containing these α_i as its entries will be referred to as the *dual vector*. For all algorithms considered in this chapter, it will indeed be possible to write $f(\mathbf{x})$ in this general form.

1.2.2 Example: least squares and ridge regression

We consider the well known problem of least squares regression to start with, and will derive a *kernelized* version for it.

Consider the vector $\mathbf{y} \in \mathbb{R}^n$ and the data points $\mathbf{X} \in \mathbb{R}^{n \times d}$. We want to find the *weight vector* $\mathbf{w} \in \mathbb{R}^d$ that minimizes $\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$. Taking the gradient of this cost function with respect to $\mathbf{w}$ and equating to zero leads to:

$$\begin{aligned} \nabla_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 &= \nabla_{\mathbf{w}} (\mathbf{y}'\mathbf{y} + \mathbf{w}'\mathbf{X}'\mathbf{X}\mathbf{w} - 2\mathbf{w}'\mathbf{X}'\mathbf{y}) \\ &= 2\mathbf{X}'\mathbf{X}\mathbf{w} - 2\mathbf{X}'\mathbf{y} \\ &= 0 \\ \Rightarrow \mathbf{w} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}, \end{aligned}$$

if $\mathbf{X}$ is of full rank. This is the well known least squares solution.

However, least squares is highly sensitive to overfitting: especially when $\mathbf{X}$ lives in a high dimensional (feature) space, care needs to be taken (ultimately, when the dimensionality $d > n$, regression can always be carried out exactly,

such a map can be defined for a wide variety of objects, including vectors, strings, sequences, graphs and more.

[5] Note that the *bracket* notation $\langle \cdot, \cdot \rangle$ for the inner product is more accurate or at least more general than the linear algebra notation, since a Hilbert space is not necessarily a vector space. However, for conciseness and for consistency with literature, we will somewhat abusively use the linear algebra notation, keeping in mind that results can be transferred towards general Hilbert spaces.

which means that any noise sequence could be fit by the model). In order to avoid overfitting, a standard approach is to reduce the *capacity* of the learner by imposing a prior on the solution, thus introducing a bias. In the case of regression for example, one usually prefers a weight vector with small norm. This is taken into account by introducing an additional term $\gamma\|\mathbf{w}\|^2$ in the cost function, with γ the *regularization parameter*. Minimizing leads to the ridge regression (RR) estimate:[6]

$$\begin{aligned}
\nabla_{\mathbf{w}}\left[\|\mathbf{y}-\mathbf{X}\mathbf{w}\|^2+\gamma\|\mathbf{w}\|^2\right] &= \nabla_{\mathbf{w}}\left[(\mathbf{y}'\mathbf{y}+\mathbf{w}'\mathbf{X}'\mathbf{X}\mathbf{w}-2\mathbf{w}'\mathbf{X}'\mathbf{y})+\gamma(\mathbf{w}'\mathbf{w})\right] \\
&= 2(\mathbf{X}'\mathbf{X}+\gamma\mathbf{I})\mathbf{w}-2\mathbf{X}'\mathbf{y} \\
&= 0 \\
\Rightarrow \mathbf{w} &= (\mathbf{X}'\mathbf{X}+\gamma\mathbf{I})^{-1}\mathbf{X}'\mathbf{y}.
\end{aligned}$$

To evaluate the regression function in a new test sample $\mathbf{x}_{\text{test}}$, it can simply be 'projected' on the weight vector as follows:

$$y_{\text{test}} = \mathbf{x}'_{\text{test}}\mathbf{w}.$$

So far the primal version of the ridge regression method.

The dual version can be derived by noting that the optimal weight vector will always be in the span of the data $\mathbf{X}$ with singular value decomposition $\mathbf{X} = \mathbf{W}\sqrt{\mathbf{\Lambda}}\mathbf{V}'$ (thus, the right singular vectors $\mathbf{V}$ form a basis for the row space of $\mathbf{X}$). This can be seen by observing that $(\mathbf{X}'\mathbf{X}+\gamma\mathbf{I})^{-1}\mathbf{X}' = (\mathbf{V}\mathbf{\Lambda}\mathbf{V}'+\gamma\mathbf{I})^{-1}\mathbf{V}\sqrt{\mathbf{\Lambda}}\mathbf{W}' = \mathbf{V}(\mathbf{\Lambda}+\gamma\mathbf{I})^{-1}\sqrt{\mathbf{\Lambda}}\mathbf{W}'$. Thus the weight vector $\mathbf{w} = \mathbf{V}\left[(\mathbf{\Lambda}+\gamma\mathbf{I})^{-1}\sqrt{\mathbf{\Lambda}}\mathbf{W}'\mathbf{y}\right]$ lies in the column space of $\mathbf{V}$ or equivalently in the row space of $\mathbf{X}$ and can thus be expressed as $\mathbf{w} = \mathbf{X}'\boldsymbol{\alpha}$. Here $\boldsymbol{\alpha} \in \mathbb{R}^n$ is the *dual vector*. Plugging this into the equations leads to:

$$\begin{aligned}
\nabla_{\boldsymbol{\alpha}}\left[\|\mathbf{y}-\mathbf{X}\mathbf{X}'\boldsymbol{\alpha}\|^2+\gamma\|\mathbf{X}'\boldsymbol{\alpha}\|^2\right] &= 2(\mathbf{X}\mathbf{X}'\mathbf{X}\mathbf{X}')\boldsymbol{\alpha}-2\mathbf{X}\mathbf{X}'\mathbf{y}+2\gamma\mathbf{X}\mathbf{X}'\boldsymbol{\alpha} \\
&= 2(\mathbf{K}^2+\gamma\mathbf{K})\boldsymbol{\alpha}-2\mathbf{K}\mathbf{y} \\
&= \mathbf{0} \\
\Rightarrow \mathbf{K}(\mathbf{K}+\gamma\mathbf{I})\boldsymbol{\alpha} &= \mathbf{K}\mathbf{y}. \qquad (1.3)
\end{aligned}$$

In the second step, $\mathbf{X}\mathbf{X}'$, which is the matrix containing the inner products between any two points as its elements, is replaced by the kernel matrix $\mathbf{K}$. Since the inner products in $\mathbf{K}$ can be inner products in a feature space, they can in fact be a nonlinear function of the data points, namely the kernel function. In this way, nonlinearities can be dealt with in a very natural way. This is the essence of the 'kernel trick'. A general solution for equation (1.3) is given by:

$$\boldsymbol{\alpha} = (\mathbf{K}+\gamma\mathbf{I})^{-1}\mathbf{y}+\boldsymbol{\alpha}_0,$$

[6] Another argument for using ridge regression instead of least squares regression is the fact that the least squares problem can be numerically ill-conditioned, when the ratio between the largest and the smallest eigenvalue of $\mathbf{X}'\mathbf{X}$ is very large. Also this can be solved by adding an diagonal matrix $\gamma\mathbf{I}$ to $\mathbf{X}'\mathbf{X}$.

where $\boldsymbol{\alpha}_0$ is any vector in the null space of $\mathbf{K}$: $\mathbf{X}'\boldsymbol{\alpha}_0 = \mathbf{K}\boldsymbol{\alpha}_0 = \mathbf{0}$.

We can now compute the weight vector corresponding to this dual vector $\boldsymbol{\alpha}$ as $\mathbf{w} = \mathbf{X}'\left[(\mathbf{K}+\gamma\mathbf{I})^{-1}\mathbf{y} + \boldsymbol{\alpha}_0\right] = \mathbf{X}'(\mathbf{K}+\gamma\mathbf{I})^{-1}\mathbf{y}$. As one can see, the actual value of $\boldsymbol{\alpha}_0$ does not matter. Given $\boldsymbol{\alpha}$, the projection of a test point $\mathbf{x}_{\text{test}}$ onto the weight vector $\mathbf{w} = \mathbf{X}'\boldsymbol{\alpha}$ can be written as $y_{\text{test}} = \mathbf{x}'_{\text{test}}\mathbf{X}'\boldsymbol{\alpha}$, or, in terms of kernel evaluations:

$$y_{\text{test}} = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}_{\text{test}}).$$

This is indeed the standard form as put forward in Section 1.2.1.

Summary: the kernel trick. As in this example, the general approach to *kernelizing* algorithms is by following these three steps:

1. rewrite all equations in terms of inner product matrices $\mathbf{XX}'$ only,
2. substitute these inner product matrices $\mathbf{XX}'$ by the symbol $\mathbf{K}$,
3. notice that in the derivations, you assumed that the data $\mathbf{X}$ belongs to the input space. However, nothing keeps you from reading $\mathbf{X}$ as being the matrix containing the feature vectors corresponding to the data samples. Then, the matrix $\mathbf{K}$ contains the inner products between the feature vectors. As a result, $\mathbf{K}$ can be any kernel matrix evaluated on the data.

Remark. We would like to point out the difference between this derivation of kernel ridge regression and the LS-SVM classifier [108, 110]. While in the first case, the dual version is derived by explicitly using the so-called kernel trick, the LS-SVM classifier is developed in an optimization framework, and the primal-dual relations can thus be interpreted in the context of Langrange duality.

1.3 Conclusions

We have now introduced the basic principles needed to understand the first part of this thesis, that will contain the study of eigenvalue problems in pattern recognition, and how they originate in a variety of machine learning settings. As we will show throughout this chapter, in many of these cases the kernel trick drastically increases the versatility of these methods.

In the next chapter we will provide a systematic overview with interpretations of eigenvalue problems in pattern recognition. The third chapter of this part consists of personal contributions only, all related to this particular field of eigenvalue problems in machine learning.

Apart from the numerical particularities of these algorithms (namely, they can be solved by solving eigenvalue problems), another common thread in this chapter is the general problem setting that will be dealt with: we will show

how eigenvalue problems can be used to tackle clustering problems where constraints on the labels are given. This kind of problems is sometimes described as *semi-supervised* as opposed to *unsupervised* (clustering) and *supervised* (classification).

Chapter 2

Eigenvalue problems in machine learning, primal and dual formulations

To a wide area of methods and applications in system and control theory, signal processing, mechanical engineering, quantum mechanics, and notably multivariate statistics and machine learning, the *eigenvalue problem* is without any doubt the most valuable tool.

In this chapter (the material of which is published in [16]) we will give an overview of the use of eigenvalue problems in the fields of multivariate statistics and machine learning. Indeed, it is an *overview* indicating nothing new is being told here. On the other hand, we try to give a unified account of this huge field, while at the same time incorporating the discussion of the kernel versions of these algorithms in a smooth way.

Throughout this chapter, we assume no probabilistic model for the data, and we do not study generalization properties of the algorithms. Therefore, all concepts such as correlation, covariance,... should be read as sample correlation, sample covariance,...

After this chapter, we will have introduced all the necessary material to be able to introduce our own contributions in Part I.

2.1 Kernels in this chapter

In this chapter, we will aim at deriving primal and dual versions of spectral algorithms in machine learning tools whose primal versions have been developed in the multivariate statistics literature. It is usually the *primal* formulation that is best known in literature. However, more recently, formulations of these

algorithms in terms of inner products only, the so-called *dual* formulation, have been derived. This is important, since the kernel trick can straightforwardly be applied in the dual formulations, very much in the same way as shown in the example above: the matrix $\mathbf{X}\mathbf{X}'$ containing the inner products between the samples in the rows of $\mathbf{X}$ can simply be replaced with the kernel matrix $\mathbf{K}$. Thus, the methodology adopted here is to reformulate the algorithms into a version that only uses inner products, after which the kernel trick can be applied.[1]

The importance of the kernel trick is that the entries of $\mathbf{K}$ can in fact be inner products between any (potentially nonlinear) mapping of the data points from the so-called *primal space or input space* to the so-called *feature vectors* in a *feature space*. Here, the inner product can be carried out implicitly, without ever actually computing the feature vectors. Indeed, in order to rightfully interpret a real symmetric matrix $\mathbf{K}$ as an inner product matrix, we do not need to know these feature vectors: it is sufficient to know that such feature vectors *exist*, which is ensured by Mercer's theorem if $\mathbf{K}$ is semi positive definite. Therefore, even though in the equation $\mathbf{K} = \mathbf{X}\mathbf{X}'$ the symbol $\mathbf{X}$ usually represents the data in primal space, it can as well be interpreted as representing the *not explicitly specified* feature vectors (or *dual vectors*) in feature space. Note that while the data vectors are typically in a finite dimensional space (or sometimes even in a finite or countable set), the feature space can be infinite dimensional.

After the computation of the kernel matrix, two preprocessing steps can be carried out: normalizing and centering the kernel matrix.

2.1.1 Normalizing a kernel matrix

Sometimes, we are more interested in the correlation between two feature vectors than in their inner product. This is the case when the norm of a feature vector is irrelevant for the task we want to carry out. Then, it is convenient to normalize all samples: $\mathbf{x}_{n,i} = \frac{\mathbf{x}_{n,i}}{\sqrt{\mathbf{x}'_{n,i}\mathbf{x}_{n,i}}}$. This normalization operation can be carried out directly on the kernel matrix, such that we never need to know the actual feature vectors:

$$\mathbf{K}_n = \mathrm{diag}(k_{11}\ k_{22}\ \ldots\ k_{nn})^{-1/2} \cdot \mathbf{K} \cdot \mathrm{diag}(k_{11}\ k_{22}\ \ldots\ k_{nn})^{-1/2}.$$

2.1.2 Centering a kernel matrix

Very often, only the relative positions and distances between the feature vectors are of importance. Therefore, in this chapter we will assume all the data sets are centered after the normalization step (meaning that the sample mean is zero). In primal space (i.e. when $\mathbf{X}$ is known explicitly), this centering is a trivial operation, as it is done by simply subtracting the mean of each of the coordinates (n is the number of samples): $\mathbf{X}_c = \left(\mathbf{X} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{X}\right)$ (where $\mathbf{1} \in \mathbb{R}^n$

[1] In many if not all practical cases, the dual can be motivated using an optimization perspective as well, where duality can then be interpreted as Lagrangian duality. The reader is referred to [108] for an in depth treatment.

is the column vector containing n ones). However centering in feature space deserves some attention since we do not compute the feature vectors explicitly, but only the inner products between them. Thus we have to compute the centered kernel matrix based on the uncentered kernel matrix.

For an uncentered $\mathbf{K}$ corresponding to uncentered $\mathbf{X}$, the centered version $\mathbf{K}_c$ can be computed as the product of the centered matrices $\mathbf{X}_c = \left(\mathbf{X} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{X}\right)$:

$$\begin{aligned} \mathbf{K}_c &= \left(\mathbf{X} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{X}\right)\left(\mathbf{X} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{X}\right)' \\ &= \mathbf{K} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{K} - \mathbf{K}\frac{\mathbf{1}\mathbf{1}'}{n} + \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{K}\frac{\mathbf{1}\mathbf{1}'}{n}. \end{aligned} \tag{2.1}$$

The centering operation is depicted graphically in figure 2.1 In this chapter we assume all kernel matrices are centered as such.

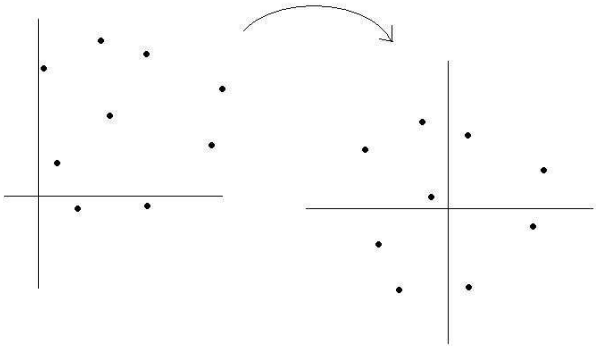


Figure 2.1: The centering operation in a ficticious 2-dimensional feature space graphically depicted: on the left side the original feature space, on the right side the feature space after centering: the *center of mass* of the samples lies in the origin after centering.

Similarly, a test sample $\mathbf{x}_{\text{test}}$ should be centered accordingly. Let $\mathbf{k}_{\text{test}} = [k(\mathbf{x}_{\text{test}}, \mathbf{x}_i)]_{i=1:n}$ be the vector containing the kernel evaluations of $\mathbf{x}_{\text{test}}$ with all n training samples $\mathbf{x}_i$. Then again, we can do the centering implicitly: the properly centered version (in correspondence with the centering of (2.1)) of this vector can be shown to be

$$\mathbf{k}_{\text{test,c}} = \mathbf{k}_{\text{test}} - \mathbf{K}\frac{\mathbf{1}}{n} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{k}_{\text{test}} + \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{K}\frac{\mathbf{1}}{n}.$$

In this chapter we assume all test samples are already centered in this way as well.

2.1.3 A leading example

To visualize the ideas explained in this chapter, we use a demonstration data set [13] that allows to illustrate all algorithms developed in this chapter, in combination with different text and string kernels. The data samples are the articles of the Swiss Constitution, which is available in 4 languages: English, French, German and Italian. Interestingly for this demonstration, the constitution is divided into several groups of articles, each group under a different so-called 'Title' (in the English translation).

All data can be found online at `www.admin.ch/ch/e/rs/c101.html`. A few articles were omitted in this case study (some because they do not have an exact equivalent in the different languages, 2 others because they are considerably different in length than the bulk of the articles), leaving a total of 195 articles per language. The texts are processed by removing punctuation and stop words followed by stemming[2] (where stop word removal and stemming are performed in a language specific way).

2.1.4 Kernel Functions

Here we briefly discuss the various kernels we will use in the demonstrations. All kernels are text and string kernels, and we always normalized them before centering. The reason is that we do not want to distinguish texts based on their length, but on their content. For a detailed description of these kernels, we refer the reader to [98].

Note that at this point, we do not have to decide for which concrete task we will use the kernels. Any kernel can be used with any kernel based algorithm, as long as the kernel is based on features that are relevant for the task to be solved. This modularity is a fundamental property of kernel methods.

Bag of words kernel

A text document can be represented by the words occurring in it, without considering the order in which the words appear. Of course this is a less complete representation than the texts themselves, but for many practical problems this is sufficient. Consider the complete dictionary of words occurring in all texts. Then each text document x could be represented by a *bag of words (BoW)* feature vector $\phi(x)$. The entries in this vector are indexed by the words in the vocabulary, and equal to the number of times the corresponding word occurs in the given text. Then, the BoW kernel between two texts is defined as the inner product of their BoW vectors: $K(x, z) = \langle \phi(x), \phi(z) \rangle$. Of course the

[2]Stemming is the preprocessing step where inflections,... are removed. This means that e.g. the words 'go', 'went', 'gone',... are all replaced by 'go'.

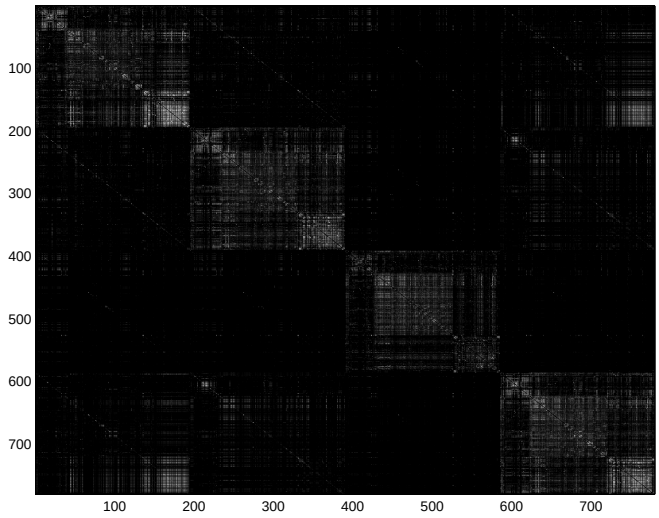


Figure 2.2: A visualization of the full BoW kernel matrix after normalization. The darker, the closer the kernel value is to 0; the brighter, the closer it is to 1. The number of rows and columns of this matrix is equal to 780, 4 times the number of articles in each language. Note that it is obvious from the figure that we have 4 distinct groups of texts, corresponding to the 4 different languages.

feature vectors are usually sparse (since texts are usually much smaller than the dictionary size), and some care has to be taken to efficiently implement the BoW kernel.

Figure 2.2 contains an image of the BoW kernel on all articles (of all languages).[3] Since there are 195 articles for each of the 4 languages, the size of the matrix is 780×780, with 4 blocks of size 195×195. One can actually visually distinguish the block structure corresponding to the 4 languages, as you can see in figure 2.2. Within these blocks, one can see some substructure in the articles, roughly corresponding to the Titles, Chapters, Sections... the articles are arranged in. This substructure reappears in all languages to some extent.

K-mer kernel

Another –more generally applicable– class of kernels is the class of k-mer kernels [75]. For each document a feature vector is constructed indexed by all possible length-k strings (k-mers) of the given alphabet (including white spaces); the value of these entries is equal to the number of times this substring occurs in

[3] To avoid a completely black picture except for a bright diagonal, before visualizing the diagonal is subtracted from the kernel. This is necessary because text kernels generally have a very heavy diagonal.

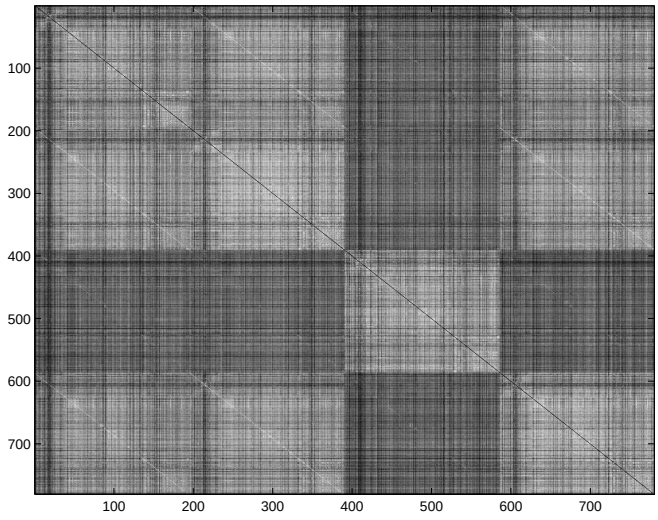


Figure 2.3: The 2-mer kernel matrix after normalization. Again the cluster structure can be seen, however, it is less clear than from the BoW kernel. This is not surprising: a 2-mer kernel only takes into account 1st order Markov properties in the texts, making them probably less suitable for natural language applications. Note that the third group of texts –corresponding to the German language–, sticks out however, indicating that the 1st order Markov properties of German are significantly different from those of the other languages considered.

the given text. The kernel between two texts is then computed in the usual way, as the inner product of their corresponding feature vectors. Note that this kernel is therefore applicable to string data, also where no words can be distinguished, such as in DNA sequences. On the other hand, its power is generally less than a BoW kernel wherever this can be used, such as on natural language. K-mer kernels capture the order $k-1$ Markov properties of the texts, which are specific to natural languages. Therefore, even for small k they are quite powerful already in distinguishing different languages.

Note that the length of the feature vector is exponential in k, therefore a naive implementation would be prohibitively expensive for larger k. However, efficient algorithms have been devised allowing for the computation of this kernel for large scale problems [75].

Figures 2.3 and 2.4 contain the full 2-mer and 4-mer kernels.

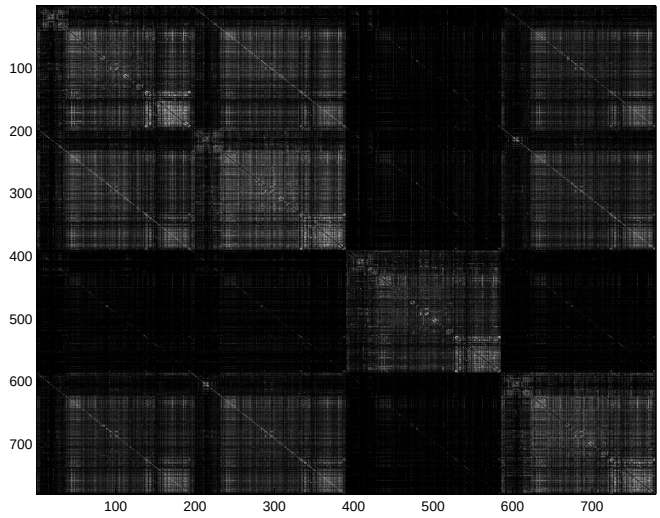


Figure 2.4: The 4-mer kernel matrix after normalization. One can see that the distinction between the different languages is more clear now than for the 2-mer kernel.

2.2 Dimensionality Reduction: PCA, (R)CCA, PLS

The general philosophy that motivates dimensionality reduction techniques is the fact that real life data sets contain redundancies[4] and noise. Dimensionality reduction is often a good way to deal with this: by using a low-dimensional approximate representation, noise can be suppressed, and redundancies removed. The data is replaced by an approximation that still captures as much information as possible. All methods described in this section can be useful as a preprocessing step for other algorithms like clustering, classification, regression, etc.

We will discuss various ways to perform dimensionality reduction. They all share the property that they rely on inner products and on eigenproblems. As a consequence, they can easily be made nonlinear using the kernel trick, and they can be solved efficiently. The difference between them lies in the cost function they optimize.

Therefore, each of the subsections will be structured as follows: first the different cost functions leading to the algorithm are described, subsequently the primal is derived and some properties are given, and finally the dual formulation is presented. For a previous treatment of these algorithms in their primal version, we refer to [24]. On the other hand, whereas the dual versions are not new

[4]Such redundancies are often due to (linear) dependencies. E.g. in a database of people, weight is somewhat redundant with the size.

on themselves, as far as we know this [16] is the first comprehensive overview of these algorithms that includes dual versions as well.

2.2.1 PCA

Cost function

The motivation for performing *principal component analysis* (PCA) [63] is often the assumption that directions of high variance will contain more information than directions of low variance. The rationale behind this could be that the noise can be assumed to be isotropically spread, such that directions of high variance will have a higher signal to noise ratio. Mathematically:

$$\begin{aligned} \mathbf{w} &= \text{argmax}_{\|\mathbf{w}\|=1} \mathbf{w}'\mathbf{X}'(\mathbf{w}'\mathbf{X}')' \\ &= \text{argmax}_{\|\mathbf{w}\|=1} \mathbf{w}'\mathbf{X}'\mathbf{X}\mathbf{w} \\ &= \text{argmax}_{\|\mathbf{w}\|=1} \mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}. \end{aligned} \tag{2.2}$$

Or, for $\mathbf{w}$ not normalized this can be written as:

$$\mathbf{w} = \text{argmax}_{\mathbf{w}} \frac{\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}}{\mathbf{w}'\mathbf{w}}.$$

where $\mathbf{S}_{\mathbf{XX}} = \mathbf{X}'\mathbf{X}$ is called the *scatter matrix* of $\mathbf{X}$.

The solution of (2.2) is also equivalent to minimizing the 2-norm of the residuals. This can be seen by projecting all samples $\mathbf{X}$ on the subspace orthogonal to $\mathbf{w}$ (by left multiplication with $(\mathbf{I}-\mathbf{w}\mathbf{w}')$, and computing the Frobenius norm:

$$\begin{aligned} \mathbf{w} &= \text{argmin}_{\|\mathbf{w}\|=1} \|\mathbf{X}(\mathbf{I} - \mathbf{w}\mathbf{w}')\|_F^2 \\ &= \text{argmin}_{\|\mathbf{w}\|=1} \text{trace}\left([\mathbf{X}(\mathbf{I} - \mathbf{w}\mathbf{w}')]'[\mathbf{X}(\mathbf{I} - \mathbf{w}\mathbf{w}')]\right) \\ &= \text{argmin}_{\|\mathbf{w}\|=1} \text{trace}\left(\mathbf{X}'\mathbf{X} + \mathbf{w}\mathbf{w}'\mathbf{X}'\mathbf{X}\mathbf{w}\mathbf{w}' - 2\mathbf{X}'\mathbf{X}\mathbf{w}\mathbf{w}'\right) \\ &= \text{argmin}_{\|\mathbf{w}\|=1} \text{trace}(\mathbf{S}_{\mathbf{XX}}) + \|\mathbf{w}\|^2\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w} - 2\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w} \\ &= \text{argmin}_{\|\mathbf{w}\|=1} - \mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}. \end{aligned}$$

Primal

Differentiating the Lagrangian $\mathcal{L}(\mathbf{w}, \lambda) = \mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w} - \lambda(\mathbf{w}'\mathbf{w} - 1)$ corresponding to (2.2) with respect to $\mathbf{w}$, and equating to zero, leads to

$$\begin{aligned} & \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}, \lambda) = \nabla_{\mathbf{w}}(\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w} - \lambda\mathbf{w}'\mathbf{w}) = 0 \\ \Leftrightarrow \quad & \mathbf{S}_{\mathbf{XX}}\mathbf{w} = \lambda\mathbf{w}. \end{aligned}$$

This is a symmetric eigenvalue problem as presented in section 1.1. Such an eigenvalue problem has d eigenvectors. All are called principal directions, corresponding to their variance λ.

Properties

- All principal directions are orthogonal to each other.
- The principal directions can all be obtained by optimizing the same cost function, where the orthogonality property is explicitly imposed.
- The projections of the data onto different principal directions are *uncorrelated*: $(\mathbf{X}\mathbf{w}_i)'\mathbf{X}\mathbf{w}_j = 0$ for $i \neq j$. Note that one could as well say the projections are *orthogonal*. This is equivalent, but we will use the notion of correlation when we are talking about projections of data onto a weight vector. Because of this property of PCA, it is sometimes called *linear decorrelation*.
- The PCA solution is equivalent to, and can thus be obtained by computing the singular value decomposition of $\mathbf{X}$.

Dual

To derive the dual, we use the key fact that the dominant eigenvector $\mathbf{w}$ will always be a linear combination of the columns of $\mathbf{X}'$, since $\mathbf{w} = \frac{1}{\lambda}\mathbf{S}_{\mathbf{XX}}\mathbf{w} = \mathbf{X}' \cdot \frac{\mathbf{X}\mathbf{w}}{\lambda}$. We can thus replace $\mathbf{w}$ with $\mathbf{X}'\boldsymbol{\alpha}$, where $\boldsymbol{\alpha}$ is a vector containing the dual variables. The dual problem is then:

$$\begin{aligned} \mathbf{S}_{\mathbf{XX}}\mathbf{X}'\boldsymbol{\alpha} &= \lambda\mathbf{X}'\boldsymbol{\alpha} \\ \Rightarrow \mathbf{X}\mathbf{S}_{\mathbf{XX}}\mathbf{X}'\boldsymbol{\alpha} &= \lambda\mathbf{X}\mathbf{X}'\boldsymbol{\alpha} \\ \Rightarrow \mathbf{K}_{\mathbf{X}}^2\boldsymbol{\alpha} &= \lambda\mathbf{K}_{\mathbf{X}}\boldsymbol{\alpha}. \end{aligned} \tag{2.3}$$

When $\mathbf{K}_{\mathbf{X}}$ has full rank, we can multiply (2.3) by $\mathbf{K}_{\mathbf{X}}^{-1}$ on the left hand side, leading to:

$$\mathbf{K}_{\mathbf{X}}\boldsymbol{\alpha} = \lambda\boldsymbol{\alpha}. \tag{2.4}$$

On the other hand, when $\mathbf{K}_{\mathbf{X}}$ is rank deficient, a solution for (2.3) is not always a solution for (2.4) anymore (the converse is still true however). Then for $\boldsymbol{\alpha}_0$ lying in the null space of $\mathbf{K}_{\mathbf{X}}$, and $\boldsymbol{\alpha}$ a solution of (2.4) (and thus also of (2.3)), also $\boldsymbol{\alpha}+\boldsymbol{\alpha}_0$ is a solution of (2.3) but generally not of (2.4). But, since $\mathbf{K}_{\mathbf{X}}\boldsymbol{\alpha}_0 = \mathbf{0}$ and thus $\mathbf{X}'\boldsymbol{\alpha}_0 = \mathbf{0}$, the component $\boldsymbol{\alpha}_0$ will have no effect on $\mathbf{w} = \mathbf{X}'(\boldsymbol{\alpha}+\boldsymbol{\alpha}_0) = \mathbf{X}'\boldsymbol{\alpha}$ anyway, and we can ignore the null space of $\mathbf{K}_{\mathbf{X}}$, by simply solving (2.4) also in the case $\mathbf{K}_{\mathbf{X}}$ is rank deficient.

Since $\mathbf{K}_{\mathbf{X}}$ is a symmetric matrix, the dual eigenvectors will be orthogonal to each other. The projections of the training samples onto the weight vector $\mathbf{w}$ are $\mathbf{X}\mathbf{w} = \mathbf{X}\mathbf{X}'\boldsymbol{\alpha} = \lambda\boldsymbol{\alpha}$. Thus, the vector $\boldsymbol{\alpha}$ is proportional with (and thus up to a normalization equal to) the projections of the training samples onto this weight vector. The fact that different dual vectors are orthogonal is thus equivalent to the observation that the projections of the data onto different weight vectors are uncorrelated.

Note that in order to normalize the weight vector in feature space, one has to normalize the dual vector $\boldsymbol{\alpha}$ such that $\mathbf{w}'\mathbf{w} = \boldsymbol{\alpha}'\mathbf{K}_\mathbf{X}\boldsymbol{\alpha} = 1$.

Projection of a feature vector belong to $\mathbf{x}_{\text{test}}$ onto the PCA direction corresponding to a dual vector $\boldsymbol{\alpha}$ can be carried out as

$$y_{\text{test}} = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}_{\text{test}}).$$

Demonstration

Do the large variance directions found by PCA effectively help us in handling and in visualizing our data? How powerful is the *kernel trick*? This can be demonstrated on our leading example in two ways.

First, we applied PCA on the entire dataset consisting of all 4 languages. The two dominant components of all samples are plotted in figure 2.5.

As a second demonstration, we applied PCA to all English texts only. The result is pictured in figure 2.6, showing the 2 dominant principal components. Samples coming from different Titles in the constitution are given another symbol. Clearly, they are grouped somehow, indicating that PCA indeed captures some relevant information in the data.

2.2.2 (R)CCA

While PCA deals with only one data space $\mathcal{X}$ where it identifies directions of high variance, *canonical correlation analysis* (CCA) (first introduced in [58]) proposes a way for dimensionality reduction by taking into account relations between samples coming from *two* spaces $\mathcal{X}$ and $\mathcal{Y}$. The assumption is that the data points coming from these two spaces contain some joint information that is reflected in correlations between them. Directions along which this correlation is high are thus assumed to be relevant directions when these relations are to be captured.

Again a primal and a dual form are available. The dual form makes it possible to capture nonlinear correlations as well thanks to the kernel trick ([41], [2], [6]).

When data is scarce as compared to the dimensionality of the problem, it is important to regularize the problem in order to avoid overfitting. This regularization is incorporated in the RCCA algorithm: *regularized CCA*.

A small example

To make things more specific, consider the following example described in [118]. Suppose we have two text corpora, one containing English texts, and another one containing the same texts but translated in French. The text corpora can be represented by the matrices $\mathbf{X}$ and $\mathbf{Y}$ containing vectors that are the BoW representations of the texts as its rows. Now, since we know that the same basic semantic information must be present in both the English text and the French

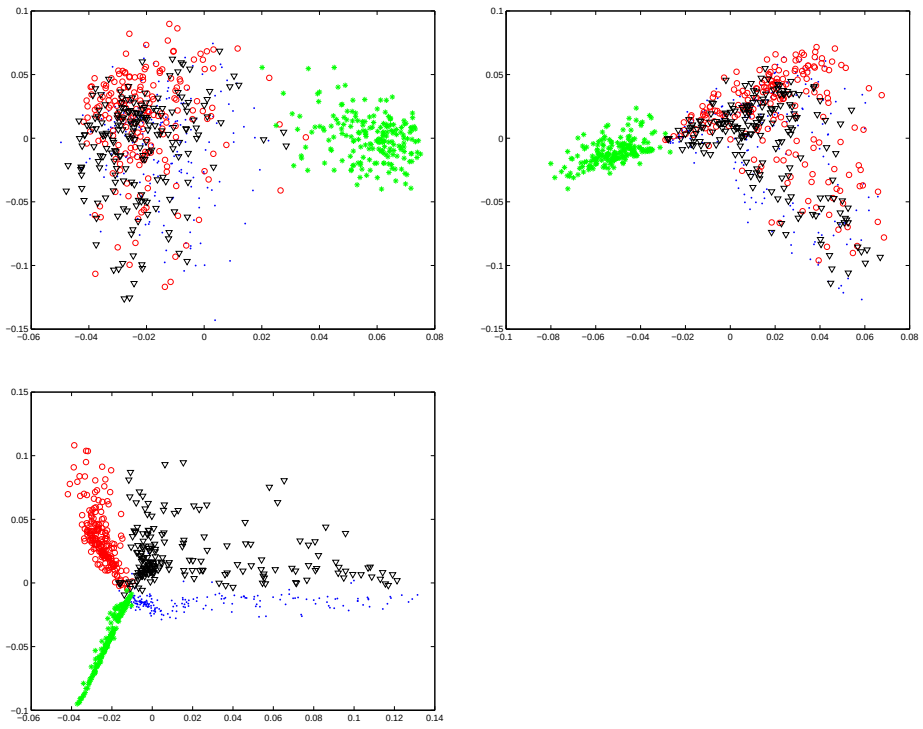


Figure 2.5: Scatter plots of the first (horizontal axis) and second (vertical axis) eigenvector components of all constitution articles, in the feature spaces corresponding to the 2mer (upper left), 4mer (upper right) and BoW kernels (lower). Articles in different languages are plotted with a different symbol. Notice how the BoW kernel apparently performs best in providing a 2-dimensional representation capturing the language of the articles: indeed, the articles from different languages seem to be most distant from each other in the BoW representation.

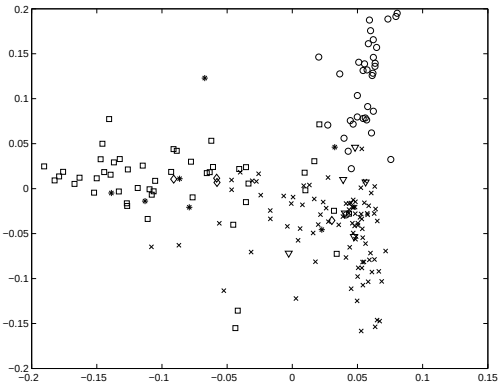


Figure 2.6: PCA applied to the BoW kernel on the English texts: the 2 dominant components are given (with the component along the first eigenvector on the horizontal axis, for the second eigenvector on the vertical axis). Points corresponding to articles under different Titles in the constitution are given a different symbol.

translation, we must be able to extract some information from every row of $\mathbf{X}$ that is similar to information extracted from the rows of $\mathbf{Y}$. If we do this in a linear way, this would mean that $\mathbf{X}\mathbf{w}_\mathbf{X}$ and $\mathbf{Y}\mathbf{w}_\mathbf{Y}$ are similar in a way, for some $\mathbf{w}_\mathbf{X}$ and $\mathbf{w}_\mathbf{Y}$ representing a certain semantic meaning. This could be: $\mathbf{X}\mathbf{w}_\mathbf{X}$ and $\mathbf{Y}\mathbf{w}_\mathbf{Y}$ are correlated, thus motivating the cost function introduced below. In [118], it is pointed out that many of the $\mathbf{w}_\mathbf{X}$–$\mathbf{w}_\mathbf{Y}$ pairs found can indeed be related to an intuitively satisfying semantic meaning.

Other successful applications of RCCA can be found in literature, notably in bioinformatics ([116] and [128]).

Cost function

We thus want to maximize the correlation between the projections $\mathbf{X}\mathbf{w}_\mathbf{X}$ of $\mathbf{X}$ and the projections $\mathbf{Y}\mathbf{w}_\mathbf{Y}$ of $\mathbf{Y}$. Or, another geometrical interpretation is: find directions $\mathbf{X}\mathbf{w}_\mathbf{X}, \mathbf{Y}\mathbf{w}_\mathbf{Y}$ in the column space of $\mathbf{X}$ and $\mathbf{Y}$ with a minimal angle between each other:

$$\begin{aligned}
\{\mathbf{w}_\mathbf{X}, \mathbf{w}_\mathbf{Y}\} &= \text{argmax}_{\mathbf{w}_\mathbf{X},\mathbf{w}_\mathbf{Y}} \cos\left(\angle(\mathbf{X}\mathbf{w}_\mathbf{X}, \mathbf{Y}\mathbf{w}_\mathbf{Y})\right) \\
&= \text{argmax}_{\mathbf{w}_\mathbf{X},\mathbf{w}_\mathbf{Y}} \frac{(\mathbf{X}\mathbf{w}_\mathbf{X})'(\mathbf{Y}\mathbf{w}_\mathbf{Y})}{\sqrt{(\mathbf{X}\mathbf{w}_\mathbf{X})'(\mathbf{X}\mathbf{w}_\mathbf{X})}\sqrt{(\mathbf{Y}\mathbf{w}_\mathbf{Y})'(\mathbf{Y}\mathbf{w}_\mathbf{Y})}} \\
&= \text{argmax}_{\mathbf{w}_\mathbf{X},\mathbf{w}_\mathbf{Y}} \frac{\mathbf{w}_\mathbf{X}'\mathbf{S}_{\mathbf{XY}}\mathbf{w}_\mathbf{Y}}{\sqrt{\mathbf{w}_\mathbf{X}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}_\mathbf{X}}\sqrt{\mathbf{w}_\mathbf{Y}'\mathbf{S}_{\mathbf{YY}}\mathbf{w}_\mathbf{Y}}}.
\end{aligned}$$

Where we use the notation $\mathbf{S}_{\mathbf{XY}} = \mathbf{X}'\mathbf{Y}$, referred to as the *cross-scatter* matrix.

Since the objective is independent of the norm of the weight vectors, we can maximize correlation along the weight vectors (or the 'fit') subject to constraints that fix the value of these weight vectors:

$$\begin{aligned}\{\mathbf{w_X}, \mathbf{w_Y}\} &= \text{argmax}_{\mathbf{w_X},\mathbf{w_Y}} \mathbf{w'_X S_{XY} w_Y} \\ &\text{s.t.} \quad \|\mathbf{X w_X}\|^2 = \mathbf{w'_X S_{XX} w_X} = 1, \|\mathbf{Y w_Y}\|^2 = \mathbf{w'_Y S_{YY} w_Y} = 1.\end{aligned}$$

This is equivalent to the minimization of a 'misfit' subject to these constraints:

$$\begin{aligned}\{\mathbf{w_X}, \mathbf{w_Y}\} &= \text{argmin}_{\mathbf{w_X},\mathbf{w_Y}} \|\mathbf{X w_X} - \mathbf{Y w_Y}\|^2 \\ &\text{s.t.} \quad \|\mathbf{X w_X}\|^2 = 1, \|\mathbf{Y w_Y}\|^2 = 1.\end{aligned}$$

Primal

We solve the second formulation of the problem. Differentiating the Lagrangian $\mathcal{L}(\mathbf{w}_X, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) = \mathbf{w'_X S_{XY} w_Y} - \lambda_\mathbf{X}(\mathbf{w'_X S_{XX} w_X} - 1) - \lambda_\mathbf{Y}(\mathbf{w'_Y S_{YY} w_Y} - 1)$ with respect to $\mathbf{w_X}$ and $\mathbf{w_Y}$, and equating to 0, gives

$$\begin{aligned}&\begin{cases} \frac{\partial}{\partial \mathbf{w_X}} \mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) &= 0 \\ \frac{\partial}{\partial \mathbf{w_Y}} \mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) &= 0 \end{cases} \\ \Rightarrow &\begin{cases} \mathbf{S_{XY} w_Y} &= \lambda_\mathbf{X} \mathbf{S_{XX} w_X} \\ \mathbf{S_{YX} w_X} &= \lambda_\mathbf{Y} \mathbf{S_{YY} w_Y} \end{cases}.\end{aligned}$$

Now, since from this

$$\lambda_\mathbf{X} \mathbf{w'_X S_{XX} w_X} = \mathbf{w'_X S_{XY} w_Y} = \mathbf{w'_Y S_{YX} w_X} = \lambda_\mathbf{Y} \mathbf{w'_Y S_{YY} w_Y},$$

and since $\mathbf{w'_X S_{XX} w_X} = \mathbf{w'_Y S_{YY} w_Y} = 1$, we find that $\lambda_\mathbf{X} = \lambda_\mathbf{Y} = \lambda$, and thus

$$\begin{cases} \mathbf{S_{XY} w_Y} &= \lambda \mathbf{S_{XX} w_X} \\ \mathbf{S_{YX} w_X} &= \lambda \mathbf{S_{YY} w_Y} \end{cases}. \tag{2.5}$$

Or, stated in another way as a generalized eigenvalue problem,

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XY}} \\ \mathbf{S_{YX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{S_{XX}} & \mathbf{0} \\ \mathbf{0} & \mathbf{S_{YY}} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}. \tag{2.6}$$

This generalized eigenvalue problem has $2d$ eigenvalues. But for each positive eigenvalue λ and corresponding eigenvector $\begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}$ $-\lambda$ is an eigenvalue too with corresponding eigenvector $\begin{pmatrix} \mathbf{w_X} \\ -\mathbf{w_Y} \end{pmatrix}$. Thus, we get all information by only looking at the d positive eigenvalues. The largest one with its eigenvector corresponds to the optimum of the cost function described higher. The weight vectors making up the other eigenvectors will be referred to as other canonical directions, corresponding to a smaller canonical correlation quantized by their corresponding eigenvalue.

Properties

- CCA not only finds pairs of weight vectors that capture maximal correlations between each other. Projections onto canonical directions corresponding to a different canonical correlation are *uncorrelated*:

$$\begin{aligned}\lambda_i \mathbf{w}'_{\mathbf{Y},j}(\mathbf{S_{YY}}\mathbf{w}_{\mathbf{Y},i}) &= \mathbf{w}'_{\mathbf{Y},j}(\mathbf{S_{YX}}\mathbf{w}_{\mathbf{X},i}) \\ &= \mathbf{w}'_{\mathbf{X},i}(\mathbf{S_{XY}}\mathbf{w}_{\mathbf{Y},j}) \\ &= \lambda_j \mathbf{w}'_{\mathbf{X},i}(\mathbf{S_{XX}}\mathbf{w}_{\mathbf{X},j}) \\ &= \lambda_j \mathbf{w}'_{\mathbf{X},j}(\mathbf{S_{XX}}\mathbf{w}_{\mathbf{X},i})\end{aligned}$$

  and similarly

$$\lambda_i \mathbf{w}'_{\mathbf{X},j}(\mathbf{S_{XX}}\mathbf{w}_{\mathbf{X},i}) = \lambda_j \mathbf{w}'_{\mathbf{Y},j}(\mathbf{S_{YY}}\mathbf{w}_{\mathbf{Y},i}).$$

  So for $\lambda_i \neq \lambda_j$, the projection of $\mathbf{Y}$ onto $\mathbf{w}_{\mathbf{Y},j}$ is uncorrelated with the projection of $\mathbf{X}$ onto $\mathbf{w}_{\mathbf{X},i}$: $\mathbf{w}'_{\mathbf{Y},j}\mathbf{S_{YX}}\mathbf{w}_{\mathbf{X},i} = 0$. Similarly, $\mathbf{w}'_{\mathbf{X},j}\mathbf{S_{XX}}\mathbf{w}_{\mathbf{X},i} = 0$ and $\mathbf{w}'_{\mathbf{Y},j}\mathbf{S_{YY}}\mathbf{w}_{\mathbf{Y},i} = 0$. Another way to state this is to say that $\mathbf{w}_{\mathbf{X},i}$ is orthogonal to $\mathbf{w}_{\mathbf{X},j}$ in the metric defined by $\mathbf{S_{XX}}$; similarly, $\mathbf{w}_{\mathbf{Y},i}$ is orthogonal to $\mathbf{w}_{\mathbf{Y},j}$ in the metric defined by $\mathbf{S_{YY}}$.

- All canonical directions can be captured by a constrained optimization problem in which the above property is explicitly imposed:

$$\begin{aligned}\{\mathbf{w}_{\mathbf{X},i}, \mathbf{w}_{\mathbf{Y},i}\} &= \text{argmax}_{\mathbf{w}_{\mathbf{X},i},\mathbf{w}_{\mathbf{Y},i}} \mathbf{w}'_{\mathbf{X},i}\mathbf{S_{XY}}\mathbf{w}_{\mathbf{Y},i} \\ \text{s.t.} \quad & \|\mathbf{X}\mathbf{w}_{\mathbf{X},i}\| = \mathbf{w}'_{\mathbf{X},i}\mathbf{S_{XX}}\mathbf{w}_{\mathbf{X},i} = 1 \\ & \|\mathbf{Y}\mathbf{w}_{\mathbf{Y},i}\| = \mathbf{w}'_{\mathbf{Y},i}\mathbf{S_{YY}}\mathbf{w}_{\mathbf{Y},i} = 1 \\ & \text{and for } j < i: \quad \begin{aligned}\mathbf{w}'_{\mathbf{X},j}\mathbf{S_{XX}}\mathbf{w}_{\mathbf{X},i} &= 0 \\ \mathbf{w}'_{\mathbf{Y},j}\mathbf{S_{YY}}\mathbf{w}_{\mathbf{Y},i} &= 0.\end{aligned}\end{aligned}$$

- The CCA problem can be reformulated as an ordinary eigenvalue problem:

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XX}}^{-1}\mathbf{S_{XY}} \\ \mathbf{S_{YY}}^{-1}\mathbf{S_{YX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}.$$

  This eigenvalue problem can be made symmetric by introducing $\mathbf{v_X} = \mathbf{S_{XX}}^{1/2}\mathbf{w_X}$ and $\mathbf{v_Y} = \mathbf{S_{YY}}^{1/2}\mathbf{w_Y}$ (where we are using the symmetric square root):

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XX}}^{-1/2}\mathbf{S_{XY}}\mathbf{S_{YY}}^{-1/2} \\ \mathbf{S_{YY}}^{-1/2}\mathbf{S_{YX}}\mathbf{S_{XX}}^{-1/2} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{v_X} \\ \mathbf{v_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{v_X} \\ \mathbf{v_Y} \end{pmatrix}.$$

  Note that this eigenvalue problem is of the form (1.2), so here $\mathbf{v_X}$ and $\mathbf{v_Y}$ are the left and right singular vectors of $\mathbf{S_{XX}}^{-1/2}\mathbf{S_{XY}}\mathbf{S_{YY}}^{-1/2}$. The weight vectors can be found as $\mathbf{w_X} = \mathbf{S_{XX}}^{-1/2}\mathbf{v_X}$ and $\mathbf{w_Y} = \mathbf{S_{YY}}^{-1/2}\mathbf{v_Y}$.

By the orthogonality of the singular vectors, we can derive in an alternative way that projections onto non-corresponding canonical directions are uncorrelated: $0 = \mathbf{v}'_{\mathbf{X},i}\mathbf{v}_{\mathbf{X},j} = \mathbf{w}'_{\mathbf{X},i}\mathbf{S}_{\mathbf{XX}}\mathbf{w}_{\mathbf{X},j}$, and $0 = \mathbf{v}'_{\mathbf{Y},i}\mathbf{v}_{\mathbf{Y},j} = \mathbf{w}'_{\mathbf{Y},i}\mathbf{S}_{\mathbf{YY}}\mathbf{w}_{\mathbf{Y},j}$. Also, we find that $0 = \mathbf{v}'_{\mathbf{X},i}{\mathbf{S}_{\mathbf{XX}}}^{-1/2}\mathbf{S}_{\mathbf{XY}}{\mathbf{S}_{\mathbf{YY}}}^{-1/2}\mathbf{v}_{\mathbf{Y},j} = \mathbf{w}'_{\mathbf{X},i}\mathbf{S}_{\mathbf{XY}}\mathbf{w}_{\mathbf{Y},j}$.

- As a last remark, we note that CCA where one of both data spaces is one dimensional is equivalent to Least Squares Regression (LSR).

Dual

To derive the dual, again note that the (minimum norm[5]) $\mathbf{w}_{\mathbf{X}}$ and $\mathbf{w}_{\mathbf{Y}}$ will lie in the column space of $\mathbf{X}$ and $\mathbf{Y}$ respectively: $\mathbf{w}_{\mathbf{X}} = \mathbf{X}'\boldsymbol{\alpha}_{\mathbf{X}}$ and $\mathbf{w}_{\mathbf{Y}} = \mathbf{Y}'\boldsymbol{\alpha}_{\mathbf{Y}}$). Thus we can write

$$\begin{aligned}
\begin{pmatrix} \mathbf{0} & \mathbf{S}_{\mathbf{XY}} \\ \mathbf{S}_{\mathbf{YX}} & \mathbf{0} \end{pmatrix}\begin{pmatrix} \mathbf{X}'\boldsymbol{\alpha}_{\mathbf{X}} \\ \mathbf{Y}'\boldsymbol{\alpha}_{\mathbf{Y}} \end{pmatrix} &= \lambda\begin{pmatrix} \mathbf{S}_{\mathbf{XX}} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{\mathbf{YY}} \end{pmatrix}\begin{pmatrix} \mathbf{X}'\boldsymbol{\alpha}_{\mathbf{X}} \\ \mathbf{Y}'\boldsymbol{\alpha}_{\mathbf{Y}} \end{pmatrix} \\
&\Downarrow \quad \text{multiplying left with } \begin{pmatrix} \mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{Y} \end{pmatrix} \\
\begin{pmatrix} \mathbf{0} & \mathbf{X}\mathbf{S}_{\mathbf{XY}}\mathbf{Y}' \\ \mathbf{Y}\mathbf{S}_{\mathbf{YX}}\mathbf{X}' & \mathbf{0} \end{pmatrix}\begin{pmatrix} \boldsymbol{\alpha}_{\mathbf{X}} \\ \boldsymbol{\alpha}_{\mathbf{Y}} \end{pmatrix} &= \lambda\begin{pmatrix} \mathbf{X}\mathbf{S}_{\mathbf{XX}}\mathbf{X}' & \mathbf{0} \\ \mathbf{0} & \mathbf{Y}\mathbf{S}_{\mathbf{YY}}\mathbf{Y}' \end{pmatrix}\begin{pmatrix} \boldsymbol{\alpha}_{\mathbf{X}} \\ \boldsymbol{\alpha}_{\mathbf{Y}} \end{pmatrix} \\
&\Downarrow \\
\begin{pmatrix} \mathbf{0} & \mathbf{K}_{\mathbf{X}}\mathbf{K}_{\mathbf{Y}} \\ \mathbf{K}_{\mathbf{Y}}\mathbf{K}_{\mathbf{X}} & \mathbf{0} \end{pmatrix}\begin{pmatrix} \boldsymbol{\alpha}_{\mathbf{X}} \\ \boldsymbol{\alpha}_{\mathbf{Y}} \end{pmatrix} &= \lambda\begin{pmatrix} \mathbf{K}_{\mathbf{X}}^2 & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{\mathbf{Y}}^2 \end{pmatrix}\begin{pmatrix} \boldsymbol{\alpha}_{\mathbf{X}} \\ \boldsymbol{\alpha}_{\mathbf{Y}} \end{pmatrix}.
\end{aligned}$$

Projections of test points $\mathbf{x}_{\text{test}}$ and $\mathbf{y}_{\text{test}}$ onto the CCA directions corresponding to $\boldsymbol{\alpha}_{\mathbf{X}}$ and $\boldsymbol{\alpha}_{\mathbf{Y}}$ can then be carried out as

$$\sum_{i=1}^{n} \alpha_{\mathbf{X},i} k(\mathbf{x}_i, \mathbf{x}_{\text{test}}) \quad \text{and} \quad \sum_{i=1}^{n} \alpha_{\mathbf{Y},i} k(\mathbf{y}_i, \mathbf{y}_{\text{test}}). \tag{2.7}$$

Regularization

Primal problem Regularization is often necessary in doing CCA. The main reason for this is the following. The scatter matrices $\mathbf{S}_{\mathbf{XX}}$ and $\mathbf{S}_{\mathbf{YY}}$ are proportional to finite sample estimates of the covariance matrices. This generally leads to poor performance in case of small eigenvalues of these covariances, which corresponds to the case of (nearly) coplanar samples. Remember the generalized eigenvalue problem is (theoretically) equivalent with a standard eigenvalue

[5]The motivation for taking the minimum norm solution is as follows: first of all, we need to make a choice in cases where there is an indeterminacy as is when the rows of $\mathbf{X}$ and/or $\mathbf{Y}$ do not span the whole space. And a component of the weight vectors orthogonal to the data would never contribute to the correlation of a projection of the data onto this weight vector anyway: the projection onto this orthogonal direction would be zero. We do not get any information concerning the orthogonal subspace, and thus do not want $\mathbf{w}$ to make any unmotivated predictions on this. In this chapter we always look for minimum norm solutions.

problem where the right hand side matrix containing the scatter matrices are inverted. Any fluctuation of the smallest eigenvalue will thus be blown up in the inverse. To counteract this effect, one often adds a diagonal to the scatter matrices, or equivalently to each of their eigenvalues [6]. In this way, a bias is introduced, but it is hoped that for a certain bias, the total variance will be lower than in case no bias is present.

The primal regularized problem is thus

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XY}} \\ \mathbf{S_{YX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{S_{XX}} + \gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{S_{YY}} + \gamma\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}.$$

Intuitively, this type of regularization boils down to trusting correlations along high variance directions more than along low variance directions. Or, equivalently, it corresponds to a modified optimization problem where the constraints contain an additional term constraining the norm of $\mathbf{w_X}$ and $\mathbf{w_Y}$, similarly to the ridge regression cost function. (This type of regularization is often called Tikhonov regularization.)

Note that RCCA with one of both spaces one-dimensional is equivalent to Ridge Regression (RR).

Dual problem The dual of this generalized eigenvalue problem can be derived in the same way as the unregularized problem, leading to:[6]

$$\begin{aligned} &\begin{pmatrix} \mathbf{0} & \mathbf{K_X K_Y} \\ \mathbf{K_Y K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix} \\ &= \lambda \begin{pmatrix} \mathbf{K_X^2} + \gamma\mathbf{K_X} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y^2} + \gamma\mathbf{K_Y} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix}. \end{aligned} \tag{2.8}$$

In the dual case, the need for regularization is often even stronger than in the primal case. This is because the feature space is often infinite dimensional, so that the freedom to find correlations is much too high: all correlations would be equal to 1, which means no generalization is possible at all. Penalizing a large weight vector as above thus makes sense to improve generalization.

When both the kernels have full rank, this generalized eigenproblem is equivalent with (by left-multiplication with $\begin{pmatrix} \mathbf{K_X^{-1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y^{-1}} \end{pmatrix}$ of both sides in equation

[6] Now, we do not need to use the minimum norm argument, but we can actually prove that any solution $\mathbf{w_X}$ and $\mathbf{w_Y}$ have to be of the form $\mathbf{w_X} = \mathbf{X}'\boldsymbol{\alpha_X}$ and $\mathbf{w_Y} = \mathbf{Y}'\boldsymbol{\alpha_Y}$. Indeed, given that $\mathbf{S_{XY}w_Y} = \lambda(\mathbf{S_{XX}} + \gamma\mathbf{I})\mathbf{w_X}$ and with $\mathbf{X} = \mathbf{U}\boldsymbol{\Lambda}^{1/2}\mathbf{V}$ the singular value decomposition of $\mathbf{X}$ where $\mathbf{V}$ is the row space of $\mathbf{X}$ and with $\mathbf{V}^{\perp}$ its right null space, we see that $\mathbf{w_X} = \frac{1}{\lambda}(\mathbf{S_{XX}} + \gamma\mathbf{I})^{-1}\mathbf{S_{XY}w_Y} = \frac{1}{\lambda}\left[\mathbf{V}\boldsymbol{\Lambda}\mathbf{V}' + \gamma(\mathbf{V}\mathbf{V}' + \mathbf{V}^{\perp}\mathbf{V}^{\perp\prime})\right]^{-1}\mathbf{V}\boldsymbol{\Lambda}^{1/2}\mathbf{U}'\mathbf{Y}\mathbf{w_Y}$, which is, by using that $\mathbf{V}'\mathbf{V}^{\perp} = \mathbf{0}$ and $\mathbf{V}'\mathbf{V} = \mathbf{I}$, equal to $\frac{1}{\lambda}\left[\mathbf{V}(\boldsymbol{\Lambda} + \gamma\mathbf{I})^{-1}\mathbf{V}' + \gamma^{-1}\mathbf{V}^{\perp}\mathbf{V}^{\perp\prime}\right]\mathbf{V}\boldsymbol{\Lambda}^{1/2}\mathbf{U}'\mathbf{Y}\mathbf{w_Y} = \frac{1}{\lambda}\mathbf{V}(\boldsymbol{\Lambda} + \gamma\mathbf{I})^{-1}\boldsymbol{\Lambda}^{1/2}\mathbf{U}'\mathbf{Y}\mathbf{w_Y}$. Thus, indeed, $\mathbf{w_X}$ lies in the column space of $\mathbf{V}$ or equivalently in the row space of $\mathbf{X}$. An analogous derivation can be made for $\mathbf{w_Y}$. Note that this provides yet another argument for the use of the minimum norm solution in the unregularized case: it can be seen as the limit of the regularized case for γ tending to zero.

(2.8)):

$$\begin{pmatrix} \mathbf{0} & \mathbf{K_Y} \\ \mathbf{K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{K_X} + \gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y} + \gamma\mathbf{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix}, \quad (2.9)$$

yielding the solution as directly obtained using an optimization approach in [108].

Kernel matrices are often rank deficient however (e.g. when they are centered). In that case the solutions of (2.9) are still solutions for (2.8), but the converse is not always true anymore. The reason is that for any generalized eigenvector $\begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix}$ of (2.9) and thus of (2.8), $\begin{pmatrix} \boldsymbol{\alpha_X} + \boldsymbol{\alpha_{X0}} \\ \boldsymbol{\alpha_Y} + \boldsymbol{\alpha_{Y0}} \end{pmatrix}$ where $\boldsymbol{\alpha_{X0}}$ and $\boldsymbol{\alpha_{Y0}}$ are arbitrary vectors lying respectively in the null spaces of $\mathbf{K_X}$ and $\mathbf{K_Y}$, is also an eigenvector with the same eigenvalue of (2.8) but generally not of (2.9). However similarly as in the ridge regression derivation, it can be seen that these components $\boldsymbol{\alpha_{X0}}$ and $\boldsymbol{\alpha_{Y0}}$ play no role in the calculation of (2.7). This because the weight vectors $\mathbf{w_X} = \mathbf{X}'(\boldsymbol{\alpha_X} + \boldsymbol{\alpha_{X0}}) = \mathbf{X}'\boldsymbol{\alpha_X}$ and $\mathbf{w_Y} = \mathbf{Y}'(\boldsymbol{\alpha_Y} + \boldsymbol{\alpha_{Y0}}) = \mathbf{Y}'\boldsymbol{\alpha_Y}$ are unaffected by the components in the null spaces of $\mathbf{K_X}$ and $\mathbf{K_Y}$. Therefore, we can choose to solve either (2.8) or (2.9).

Demonstration

In our leading example, for each sample we have 4 corresponding samples in the 4 different languages. Could we extract some dimensions in the BoW spaces of these languages that capture the common semantical information? In this case, since more than 2 (namely 4) source of information are available for each of the articles, it is appropriate to use an extension of CCA, called multi-way CCA, that allows to exploit all this information simultaneously. For details we refer to the literature [6]. Then indeed it turns out to be possible as shown in figure 2.7, where the 2 dominant CCA components for the English texts are shown, as computed based all 4 languages. The CCA components separate the English texts according to the Title they belong to, indicating that part of the semantics related to the Title structure in which the articles are divided is captured by the CCA components.

If we compare this figure with figure 2.6 it is hard to tell in which of both representations the Titles are separated most clearly. Still we would expect an improvement should be achieved, as we use more information in the CCA algorithm (the translated documents). To assess if CCA effectively performs better in selecting semantically interesting directions, we do the above for PCA and CCA, now selecting 1 up to 10 dimensions. A performance function indicating how well articles under the same Title are grouped is plotted in figure 2.8. The performance function is the between class variance divided by the total variance (BCV/TV), and lies in between 0 and 1. The larger it is, the better the clusters are separated.

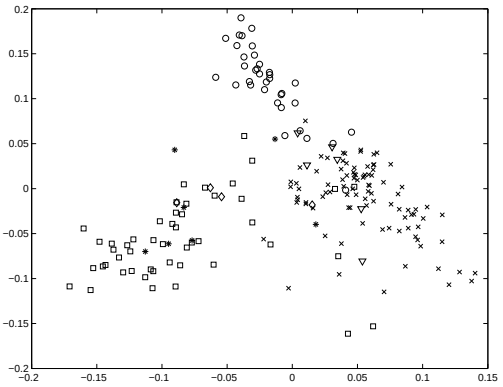


Figure 2.7: The two dominant (large regularized correlation) components of all English samples, in the feature space corresponding to bag of words kernel. (On the horizontal axis the first canonical component and vertical axis the second canonical component.) Articles belonging to different Titles are shown with different symbols. Compare with the right picture in figure 2.6.

2.2.3 PLS

Partial least squares (PLS, introduced in [124, 125], see also [56] for a good review) can be interpreted in two ways. The first PLS component is the maximally regularized version of the first CCA component (the case where $\gamma \to \infty$, after rescaling the eigenvalues by multiplying them with γ). Another view is as a covariance maximizer instead of a correlation maximizer. This again for the first PLS component. Whereas all PLS formulations compute the first component in the same way, there is consensus on which method is best to compute the other components. We will present two alternatives: so-called EZ-PLS, which consists of only one eigenvalue decomposition (or a singular value decomposition) and which is used mainly for exploratory purposes (similar to CCA), and Regression-PLS which is a more involved version that is most widely used in (multivariate) regression applications.

Due to the iterative way PLS components are computed in, and due to the fact that there exist several variants of PLS, the discussion is somewhat more involved. We will first give a general discussion on the cost function optimized in all PLS formulations, followed by the eigenproblem optimizing this cost function. Next, we will shortly go into some computational aspects. Finally, we will show the particularities of the two PLS formulations EZ-PLS and Regression-PLS, followed by a discussion of the regression step in Regression-PLS. Again a primal and a dual (see [91] where this was first derived) formulation will be provided.

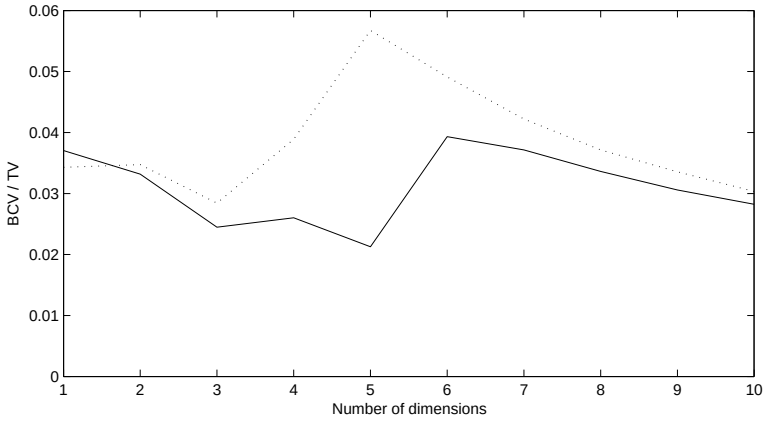


Figure 2.8: Between class variance divided by total variance (BCV/TV) for PCA (full line) and CCA (dotted line) as a function of the dimension of the subspace. Note that overall CCA performs better in separating the classes (since a larger BCV/TV indicates a better separation of the classes), indicating it effectively exploits the extra information given by the translations in order to extract interesting semantic directions.

Cost function

Maximize the sample *covariance*[7] between a projection of $\mathbf{X}$ and a projection of $\mathbf{Y}$:

$$\begin{aligned} \{\mathbf{w_X}, \mathbf{w_Y}\} &= \text{argmax}_{\mathbf{w_X},\mathbf{w_Y}} \frac{(\mathbf{X}\mathbf{w_X})'(\mathbf{Y}\mathbf{w_Y})}{\sqrt{\mathbf{w}'_\mathbf{X}\mathbf{w_X}}\sqrt{\mathbf{w}'_\mathbf{Y}\mathbf{w_Y}}} \\ &= \text{argmax}_{\mathbf{w_X},\mathbf{w_Y}} \frac{\mathbf{w}'_\mathbf{X}\mathbf{S_{XY}}\mathbf{w_Y}}{\sqrt{\mathbf{w}'_\mathbf{X}\mathbf{w_X}}\sqrt{\mathbf{w}'_\mathbf{Y}\mathbf{w_Y}}}. \end{aligned}$$

This is equivalent to maximize sample covariance, or the 'fit' subject to constraints:

$$\begin{aligned} \{\mathbf{w_X}, \mathbf{w_Y}\} &= \text{argmax}_{\mathbf{w_X},\mathbf{w_Y}} \mathbf{w}'_\mathbf{X}\mathbf{S_{XY}}\mathbf{w_Y} \\ &\text{s.t.} \quad \|\mathbf{w_X}\|^2 = \mathbf{w}'_\mathbf{X}\mathbf{w_X} = 1, \|\mathbf{w_Y}\|^2 = \mathbf{w}'_\mathbf{Y}\mathbf{w_Y} = 1, \end{aligned}$$

and equivalent to minimizing the misfit subject to these constraints:

$$\begin{aligned} \{\mathbf{w_X}, \mathbf{w_Y}\} &= \text{argmin}_{\mathbf{w_X},\mathbf{w_Y}} \|\mathbf{X}\mathbf{w_X} - \mathbf{Y}\mathbf{w_Y}\|^2 \\ &\text{s.t.} \quad \|\mathbf{w_X}\|^2 = 1, \|\mathbf{w_Y}\|^2 = 1. \end{aligned}$$

[7] Note the difference between CCA where *correlation* was maximized.

Primal

We solve the second formulation of the problem. Differentiating the Lagrangian $\mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) = \mathbf{w}'_\mathbf{X}\mathbf{S_{XY}}\mathbf{w_Y} - \lambda_\mathbf{X}\mathbf{w}'_\mathbf{X}\mathbf{w_X} - \lambda_\mathbf{Y}\mathbf{w}'_\mathbf{Y}\mathbf{w_Y}$ with respect to $\mathbf{w_X}$ and $\mathbf{w_Y}$, and equating to 0, gives

$$\begin{array}{rl} & \left\{ \begin{array}{rcl} \frac{\partial}{\partial \mathbf{w_X}}\mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) & = & 0 \\ \frac{\partial}{\partial \mathbf{w_Y}}\mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) & = & 0 \end{array} \right. \\ \Rightarrow & \left\{ \begin{array}{rcl} \mathbf{S_{XY}}\mathbf{w_Y} & = & \lambda_\mathbf{X}\mathbf{w_X} \\ \mathbf{S_{YX}}\mathbf{w_X} & = & \lambda_\mathbf{Y}\mathbf{w_Y} \end{array} \right. . \end{array}$$

Since from this

$$\lambda_\mathbf{X}\mathbf{w}'_\mathbf{X}\mathbf{w_X} = \mathbf{w}'_\mathbf{X}\mathbf{S_{XY}}\mathbf{w_Y} = \mathbf{w}'_\mathbf{Y}\mathbf{S_{YX}}\mathbf{w_X} = \lambda_\mathbf{Y}\mathbf{w}'_\mathbf{Y}\mathbf{w_Y},$$

and since $\mathbf{w}'_\mathbf{X}\mathbf{w_X} = \mathbf{w}'_\mathbf{Y}\mathbf{w_Y} = 1$, we find that $\lambda_\mathbf{X} = \lambda_\mathbf{Y} = \lambda$ and thus

$$\left\{ \begin{array}{rcl} \mathbf{S_{XY}}\mathbf{w_Y} & = & \lambda\mathbf{w_X} \\ \mathbf{S_{YX}}\mathbf{w_X} & = & \lambda\mathbf{w_Y} \end{array} \right. . \tag{2.10}$$

Or, stated in another way as an eigenvalue problem,

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XY}} \\ \mathbf{S_{YX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}. \tag{2.11}$$

This eigenvalue problem has d eigenvalues, corresponding to a covariance between projections onto $\mathbf{w_X}$ and $\mathbf{w_Y}$. The largest one with its eigenvector, corresponds to the optimum of the cost function described higher.

Note that (2.11) is of the form (1.2). Thus the EZ-PLS problem can be solved by calculating the singular value decomposition of $\mathbf{S_{XY}}$.

Dual

The dual problem can easily be found by using $\mathbf{w_X} = \mathbf{X}'\boldsymbol{\alpha_X}$ and $\mathbf{w_Y} = \mathbf{Y}'\boldsymbol{\alpha_Y}$ (based on similar arguments as for the kernelization of CCA):

$$\begin{pmatrix} \mathbf{0} & \mathbf{K_X}\mathbf{K_Y} \\ \mathbf{K_Y}\mathbf{K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{K_X} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix},$$

which includes all solutions of

$$\begin{pmatrix} \mathbf{0} & \mathbf{K_Y} \\ \mathbf{K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix} = \lambda \begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix} \tag{2.12}$$

as its solutions as well. Similarly as in CCA this is the formulation of the dual problem that is solved, since it does not suffer from indeterminacies.

Projections of test points $\mathbf{x}_{\text{test}}$ and $\mathbf{y}_{\text{test}}$ onto the PLS directions corresponding to $\boldsymbol{\alpha}_X$ and $\boldsymbol{\alpha}_Y$ can then be computed as

$$\sum_{i=1}^{n} \alpha_{\mathbf{X},i} k(\mathbf{x}_i, \mathbf{x}_{\text{test}}) \quad \text{and} \quad \sum_{i=1}^{n} \alpha_{\mathbf{Y},i} k(\mathbf{y}_i, \mathbf{y}_{\text{test}}).$$

It is important to note that the first component corresponds to maximally regularized RCCA. Taking more than one component lessens this regularization in an alternative way in comparison to RCCA. This will be subject of the remainder of this section on PLS.

NIPALS (Nonlinear Iterative PArtial Least Squares) and the primal-dual symmetry in PLS

A straightforward way to solve for the largest eigenvector of (2.11) could be by using the power method. However, thanks to the structure of the eigenvalue problems at hand, it can be solved by using the so-called NIPALS method [124]. Note that, from (2.11) and (2.12):

- $\mathbf{Y}\mathbf{Y}'\mathbf{X}\mathbf{X}'\boldsymbol{\alpha}_{\mathbf{X}} = \lambda^2\boldsymbol{\alpha}_{\mathbf{X}}$
- $\mathbf{X}'\mathbf{Y}\mathbf{Y}'\mathbf{X}\mathbf{w}_{\mathbf{X}} = \lambda^2\mathbf{w}_{\mathbf{X}}$
- $\mathbf{X}\mathbf{X}'\mathbf{Y}\mathbf{Y}'\boldsymbol{\alpha}_{\mathbf{Y}} = \lambda^2\boldsymbol{\alpha}_{\mathbf{Y}}$
- $\mathbf{Y}'\mathbf{X}\mathbf{X}'\mathbf{Y}\mathbf{w}_{\mathbf{Y}} = \lambda^2\mathbf{w}_{\mathbf{Y}}$.

Thus follows that both the primal and the dual eigenvalue problem are actually solved at the same time, using the following 'power' method:

0. Fix initial value $\mathbf{w}_{\mathbf{Y}}$, normalize. Then iterate over 1.-4.
1. $\boldsymbol{\alpha}_{\mathbf{X}} = \mathbf{Y}\mathbf{w}_{\mathbf{Y}}$
2. $\mathbf{w}_{\mathbf{X}} = \mathbf{X}'\boldsymbol{\alpha}_{\mathbf{X}}$, normalize $\mathbf{w}_{\mathbf{X}}$ to unit length
3. $\boldsymbol{\alpha}_{\mathbf{Y}} = \mathbf{X}\mathbf{w}_{\mathbf{X}}$
4. $\mathbf{w}_{\mathbf{Y}} = \mathbf{Y}'\boldsymbol{\alpha}_{\mathbf{Y}}$, normalize $\mathbf{w}_{\mathbf{Y}}$ to unit length.

After convergence, the normalizations carried out in steps 2. and 4. both amount to a division by λ: then $\mathbf{w}_{\mathbf{X}} = \frac{1}{\lambda}\mathbf{X}'\boldsymbol{\alpha}_{\mathbf{X}}$ and $\mathbf{w}_{\mathbf{Y}} = \frac{1}{\lambda}\mathbf{Y}'\boldsymbol{\alpha}_{\mathbf{Y}}$.

In case the feature vectors $\mathbf{X}$ is only implicitly determined by a kernel function, steps 2. and 3. must be combined in one step:

2,3. $\boldsymbol{\alpha}_{\mathbf{Y}} = \mathbf{K}_{\mathbf{X}}\boldsymbol{\alpha}_{\mathbf{X}}$, normalize.

It can be seen that each of these weight vectors or dual vectors converge to the eigenvector of the above four eigenvalue problems (combining four steps following each other gives the power method for one of these four eigenvalue problems). Since these are equivalent with (2.11) and (2.12), they converge to the PLS weight vectors and dual vectors.

In this way, we can solve efficiently for the largest singular value and singular vectors. Only this one component is not enough to solve most practical problems however. We discuss two ways to extract more information present in the data: what we call EZ-PLS and Regression-PLS. Both times first the primal versions will be discussed, afterwards the dual.

EZ-PLS

Primal In EZ-PLS, the other PLS directions are the other eigenvectors corresponding to a different covariance (eigenvalue) λ. This can be accomplished by using an iterative deflation scheme:

1. Initialize: $\mathbf{S_{XY}}^0 \leftarrow \mathbf{S_{XY}}$.
2. Compute the largest singular value of $\mathbf{S_{XY}}^i$ with NIPALS. This gives the ith PLS component. Normalize so that $\|\mathbf{w}_{\mathbf{X},i}\| = \|\mathbf{w}_{\mathbf{Y},i}\| = 1$.
3. Deflate the scatter matrices:
$$\mathbf{S_{XY}}^{i+1} \leftarrow \mathbf{S_{XY}}^i - \lambda_i \mathbf{w}_{\mathbf{X},i} \mathbf{w}'_{\mathbf{Y},i}.$$
The rank of $\mathbf{S_{XY}}^{i+1}$ is 1 less than the rank of $\mathbf{S_{XY}}^i$.
4. When the number of desired components (necessarily lower than the rank of $\mathbf{S_{XY}}$) is not reached yet, go to step 2.

The deflation of the $\mathbf{X}^i$ matrix for EZ-PLS, in order to get the desired deflation of the cross-scatter matrix, is
$$\mathbf{X}^{i+1} \leftarrow \mathbf{X}^i - \mathbf{X}^i \mathbf{w}_{\mathbf{X},i} \mathbf{w}'_{\mathbf{X},i}.$$
Similarly, one could do the deflation of the $\mathbf{Y}^i$ matrix
$$\mathbf{Y}^{i+1} \leftarrow \mathbf{Y}^i - \mathbf{Y}^i \mathbf{w}_{\mathbf{Y},i} \mathbf{w}'_{\mathbf{Y},i}.$$
also leading to the same desired deflation of the cross-scatter matrix.

Dual Taking the NIPALS iteration into account, the deflation of the kernel matrices corresponding to the EZ-PLS deflation is found to be
$$\mathbf{K}_{\mathbf{X}}^{i+1} \leftarrow \mathbf{K}_{\mathbf{X}}^i - \frac{1}{\lambda_i^2} \mathbf{K}_{\mathbf{X}}^i \boldsymbol{\alpha}_{\mathbf{X},i} \boldsymbol{\alpha}'_{\mathbf{X},i} \mathbf{K}_{\mathbf{X}}^i = \mathbf{K}_{\mathbf{X}}^i - \boldsymbol{\alpha}_{\mathbf{Y},i} \boldsymbol{\alpha}'_{\mathbf{Y},i}.$$

Properties

- Since the $\mathbf{w}_{\mathbf{X},i}$ and the $\mathbf{w}_{\mathbf{Y},i}$ are the left and right singular vectors of $\mathbf{S_{XY}}$, all $\mathbf{w}_{\mathbf{X},i}$ are orthogonal to each other and all $\mathbf{w}_{\mathbf{Y},i}$ are orthogonal to each other.
- For the same reason, if $i \neq j$: $\mathbf{w}'_{\mathbf{X},i} \mathbf{S_{XY}} \mathbf{w}_{\mathbf{Y},j} = 0$. In other words, projections onto non-corresponding $\mathbf{w}_{\mathbf{X},i}$ and $\mathbf{w}_{\mathbf{Y},j}$ are uncorrelated.
- All EZ-PLS components can be calculated at once by optimizing the same cost function as for the first component, taking the first (orthogonality) property into account as an additional constraint.

The EZ-PLS form is the easiest, in the sense that because of the nature of the deflation, it is in fact not more than solving for the most important singular vectors of $\mathbf{S_{XY}}$. That's the reason why it is discussed here: in practice it has little use.

Regression-PLS

Whereas EZ-PLS is not often used for regression (note that it is entirely symmetric between $\mathbf{X}$ and $\mathbf{Y}$, whereas regression is not; it is rather used for modelling though), Regression-PLS is the PLS formulation that is generally preferred for (multivariate) regression (see [56]). We will first discuss the deflations that are characteristic for Regression-PLS. Further on we will explain how regression can be carried out using the results from these deflations.

Primal The difference between EZ-PLS and Regression-PLS lies in the way the deflation is carried out. Regression-PLS has the intention of modelling one (possibly) vectorial variable $\mathbf{Y}$ with the other vectorial variable $\mathbf{X}$, hence the name[8]. It is thus asymmetric between the two spaces, which expresses itself in the deflation step:

2,4. Deflate by orthogonalizing $\mathbf{X}^i$ to its projection onto the weight vector $\mathbf{w}_{\mathbf{X},i}$, $\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}$, and recomputing the scatter matrix:

$$\mathbf{X}^{i+1} \leftarrow \left(\mathbf{I} - \frac{\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}}{\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}}\right)\mathbf{X}^i = \mathbf{X}^i - \frac{\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}}{\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}}\mathbf{X}^i \quad (2.13)$$

$$= \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\right)\mathbf{X}^i. \quad (2.14)$$

Finally (see later, equation (2.24)) we will perform a regression of $\mathbf{Y}$ based on the $\boldsymbol{\alpha}_{\mathbf{Y},i}$. (The $\boldsymbol{\alpha}_{\mathbf{Y},i}$ can be computed from $\mathbf{X}$ as will become clear later, see equation (2.23).) Therefore, we also deflate $\mathbf{Y}^i$ with $\boldsymbol{\alpha}_{\mathbf{Y},i}$ to remove the information captured by the ith iteration:

$$\mathbf{Y}^{i+1} \leftarrow \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\right)\mathbf{Y}^i. \quad (2.15)$$

This boils down to the following deflation of the scatter matrix:

$$\mathbf{S_{XY}}^{i+1} \leftarrow \mathbf{S_{XY}}^i - \frac{\lambda_i}{\mathbf{w}'_{\mathbf{X},i}\mathbf{S_{XX}}^i\mathbf{w}_{\mathbf{X},i}}\mathbf{S_{XX}}^i\mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{Y},i}.$$

The philosophy behind this kind of deflation is as follows: after step i, part of the information in $\mathbf{X}^i$, namely its projection $\boldsymbol{\alpha}_{\mathbf{Y},i}$ onto the ith PLS direction $\mathbf{w}_{\mathbf{X},i}$ is captured already: the component $\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\mathbf{X}^i$ of $\mathbf{X}^i$ (along $\boldsymbol{\alpha}_{\mathbf{Y},i}$) perfectly models the component $\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\mathbf{Y}^i$ of $\mathbf{Y}^i$. This information should not be used or modelled again in next steps, so it is 'subtracted' from both $\mathbf{X}^i$ and $\mathbf{Y}^i$. In the next step, the direction of maximal covariance between the remaining information $\mathbf{X}^{i+1}$ and $\mathbf{Y}^{i+1}$ is found, and so on.

[8] In literature this form of PLS is best known as PLS2, or PLS1 for the case where $\mathbf{Y}$ is one-dimensional.

Dual Using equations (2.14) and (2.15) , the deflation of the kernel matrices corresponding to the regression-PLS deflation can be shown to be

$$\mathbf{K}_{\mathbf{X}}^{i+1} \leftarrow \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\right)\mathbf{K}_{\mathbf{X}}^{i}\left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\right).$$

Analogously

$$\mathbf{K}_{\mathbf{Y}}^{i+1} \leftarrow \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\right)\mathbf{K}_{\mathbf{Y}}^{i}\left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\right).$$

Properties

- The different weight vectors $\mathbf{w}_{\mathbf{Y},i}$ are *not* orthogonal (it is even possible that they are all collinear, e.g. in the case where $\mathbf{Y}$ is one-dimensional). The different weight vectors $\mathbf{w}_{\mathbf{X},i}$ however are orthogonal: using (2.13),

  $$\mathbf{w}'_{\mathbf{X},i}\mathbf{S}_{\mathbf{XY}}^{i+1} = \mathbf{w}'_{\mathbf{X},i}\left(\left(\mathbf{I} - \frac{\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}}{\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}}\right)\mathbf{X}^i\right)'\mathbf{Y}^{i+1} = \mathbf{0},$$

  so that $\mathbf{w}_{\mathbf{X},i}$ is in the left null space of ${\mathbf{S}_{\mathbf{XY}}}^{i+1}$. Since $\mathbf{w}_{\mathbf{X},i+1}$ is a left singular vector of ${\mathbf{S}_{\mathbf{XY}}}^{i+1}$ this means that $\mathbf{w}_{\mathbf{X},i+1}$ will be orthogonal to $\mathbf{w}_{\mathbf{X},i}$. By replacing the left most $\mathbf{X}^i$ in the above equation by $\left(\mathbf{I} - \frac{\mathbf{X}^{i-1}\mathbf{w}_{\mathbf{X},i-1}\mathbf{w}'_{\mathbf{X},i-1}\mathbf{X}^{i-1\prime}}{\mathbf{w}'_{\mathbf{X},i-1}\mathbf{X}^{i-1\prime}\mathbf{X}^{i-1}\mathbf{w}_{\mathbf{X},i-1}}\right)\mathbf{X}^{i-1}$, and so on for $\mathbf{X}^{i-1}, \ldots$ one can see that also for $j < i$, $\mathbf{w}_{\mathbf{X},j}$ is orthogonal to $\mathbf{w}_{\mathbf{X},i}$. Thus, all $\mathbf{w}_{\mathbf{X},i}$ are mutually orthogonal:

  $$\mathbf{W}'_{\mathbf{X}}\mathbf{W}_{\mathbf{X}} = \mathbf{I},$$

  where $\mathbf{W}_{\mathbf{X}}$ represents the matrix built by stacking the vectors $\mathbf{w}_{\mathbf{X},i}$ next to each other.

- The vectors $\boldsymbol{\alpha}_{\mathbf{Y},i}$ are mutually orthogonal: using (2.14), for $i \leq j$ one has:

  $$\mathbf{X}^{j\prime}\boldsymbol{\alpha}_{\mathbf{Y},i} \quad = \quad \mathbf{X}^{i\prime}\left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\right)\ldots\left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},j-1}\boldsymbol{\alpha}'_{\mathbf{Y},j-1}}{\boldsymbol{\alpha}'_{\mathbf{Y},j-1}\boldsymbol{\alpha}_{\mathbf{Y},j-1}}\right)\boldsymbol{\alpha}_{\mathbf{Y},i}.$$

  For $j = i+1$, this is immediately proven to be zero. When this product is zero for all $j : i < j < j*$, $\boldsymbol{\alpha}'_{\mathbf{Y},j}\boldsymbol{\alpha}_{\mathbf{Y},i} = \mathbf{w}'_{\mathbf{X},j}\mathbf{X}^{j\prime}\boldsymbol{\alpha}_{\mathbf{Y},i} = 0$ and the matrices between brackets in the above product commute. Since this is indeed true for $j = i+1$, by induction it is proved for all $i < j$ that:

  $$\mathbf{X}^{j\prime}\boldsymbol{\alpha}_{\mathbf{Y},i} = \mathbf{0}, \tag{2.16}$$

  and thus by left multiplication with $\mathbf{w}_{\mathbf{X},j}$

  $$\boldsymbol{\alpha}'_{\mathbf{Y},j}\boldsymbol{\alpha}_{\mathbf{Y},i} = 0. \tag{2.17}$$

Note that since $\boldsymbol{\alpha}_{\mathbf{Y},i} = \mathbf{X}^i \mathbf{w}_{\mathbf{X},i}$, this means that the projections $\boldsymbol{\alpha}_{\mathbf{Y},i}$ of $\mathbf{X}^i$ onto their weight vectors $\mathbf{w}_{\mathbf{X},i}$ are uncorrelated with each other. This property may remind you of CCA.

- This orthogonality property (2.17) of the $\boldsymbol{\alpha}_{\mathbf{Y},i}$ leads to the fact that

$$\begin{aligned} & \mathbf{w}_{\mathbf{Y},i} = \mathbf{Y}^{i\prime} \boldsymbol{\alpha}_{\mathbf{Y},i} = \mathbf{Y}' \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},1} \boldsymbol{\alpha}'_{\mathbf{Y},1}}{\boldsymbol{\alpha}'_{\mathbf{Y},1} \boldsymbol{\alpha}_{\mathbf{Y},1}} \right) \ldots \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i-1} \boldsymbol{\alpha}'_{\mathbf{Y},i-1}}{\boldsymbol{\alpha}'_{\mathbf{Y},i-1} \boldsymbol{\alpha}_{\mathbf{Y},i-1}} \right) \boldsymbol{\alpha}_{\mathbf{Y},i} \\ \Rightarrow \quad & \mathbf{w}_{\mathbf{Y},i} = \mathbf{Y}' \boldsymbol{\alpha}_{\mathbf{Y},i}. \end{aligned} \tag{2.18}$$

  up to a normalization.

- Furthermore, one finds that for $i < j$:

$$\begin{aligned} \mathbf{X}^j \mathbf{w}_{\mathbf{X},i} &= \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},j-1} \boldsymbol{\alpha}'_{\mathbf{Y},j-1}}{\boldsymbol{\alpha}'_{\mathbf{Y},j-1} \boldsymbol{\alpha}_{\mathbf{Y},j-1}} \right) \ldots \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i} \boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i} \boldsymbol{\alpha}_{\mathbf{Y},i}} \right) \mathbf{X}^i \mathbf{w}_{\mathbf{X},i} \\ &= \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},j-1} \boldsymbol{\alpha}'_{\mathbf{Y},j-1}}{\boldsymbol{\alpha}'_{\mathbf{Y},j-1} \boldsymbol{\alpha}_{\mathbf{Y},j-1}} \right) \ldots \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i} \boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i} \boldsymbol{\alpha}_{\mathbf{Y},i}} \right) \boldsymbol{\alpha}_{\mathbf{Y},i} \\ &= \mathbf{0}. \end{aligned} \tag{2.19}$$

  This generally does not hold for $i \geq j$.

- Another consequence of (2.17) is, for $i < j$:

$$\begin{aligned} \mathbf{Y}^{j\prime} \boldsymbol{\alpha}_{\mathbf{Y},i} &= \mathbf{Y}^{i\prime} \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i} \boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i} \boldsymbol{\alpha}_{\mathbf{Y},i}} \right) \ldots \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},j-1} \boldsymbol{\alpha}'_{\mathbf{Y},j-1}}{\boldsymbol{\alpha}'_{\mathbf{Y},j-1} \boldsymbol{\alpha}_{\mathbf{Y},j-1}} \right) \boldsymbol{\alpha}_{\mathbf{Y},i} \\ &= \mathbf{0}. \end{aligned} \tag{2.20}$$

- And thus also, for $i < j$:

$$\begin{aligned} \boldsymbol{\alpha}'_{\mathbf{X},j} \boldsymbol{\alpha}_{\mathbf{Y},i} &= \mathbf{w}_{\mathbf{X},j} \mathbf{Y}^{j\prime} \boldsymbol{\alpha}_{\mathbf{Y},i} \\ &= 0. \end{aligned} \tag{2.21}$$

- From this follows that

$$\begin{aligned} & \mathbf{w}_{\mathbf{X},i} = \mathbf{X}^{i\prime} \boldsymbol{\alpha}_{\mathbf{X},i} = \mathbf{X}' \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},1} \boldsymbol{\alpha}'_{\mathbf{Y},1}}{\boldsymbol{\alpha}'_{\mathbf{Y},1} \boldsymbol{\alpha}_{\mathbf{Y},1}} \right) \ldots \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i-1} \boldsymbol{\alpha}'_{\mathbf{Y},i-1}}{\boldsymbol{\alpha}'_{\mathbf{Y},i-1} \boldsymbol{\alpha}_{\mathbf{Y},i-1}} \right) \boldsymbol{\alpha}_{\mathbf{X},i} \\ \Rightarrow \quad & \mathbf{w}_{\mathbf{X},i} = \mathbf{X}' \boldsymbol{\alpha}_{\mathbf{X},i} \end{aligned} \tag{2.22}$$

  up to a normalization factor.

Thus as a summary:

$$\begin{aligned}
\mathbf{w}_{\mathbf{X},i} &\propto \mathbf{X}'\boldsymbol{\alpha}_{\mathbf{X},i} \\
\mathbf{w}_{\mathbf{Y},i} &\propto \mathbf{Y}'\boldsymbol{\alpha}_{\mathbf{Y},i} \\
\mathbf{w}'_{\mathbf{X},j}\mathbf{w}_{\mathbf{X},i} &= 0 \\
\boldsymbol{\alpha}'_{\mathbf{Y},j}\boldsymbol{\alpha}_{\mathbf{Y},i} &= 0 \\
\boldsymbol{\alpha}'_{\mathbf{X},j}\boldsymbol{\alpha}_{\mathbf{Y},i} &= 0 \text{ for } i<j \\
\mathbf{X}^j\mathbf{w}_{\mathbf{X},i} &= \mathbf{0} \text{ for } i<j \\
\mathbf{Y}^{j\prime}\boldsymbol{\alpha}_{\mathbf{Y},i} &= \mathbf{0} \text{ for } i<j.
\end{aligned}$$

Final regression in Regression-PLS

Primal The entire Regression-PLS algorithm is composed of a (generally non-invertible) linear mapping of $\mathbf{X}$ towards k so-called *latent variables* (in the current context we would rather call them dual variables) $\boldsymbol{\alpha}_{\mathbf{Y},i} = \mathbf{X}^i\mathbf{w}_{\mathbf{X},i}$, followed by a regression of $\mathbf{Y}$ on $\mathbf{A}_\mathbf{Y}$ where $\mathbf{A}_\mathbf{Y}$ contains $\boldsymbol{\alpha}_{\mathbf{Y},i}$ as its columns.

The part of $\mathbf{X}$ that has been deflated and thus will be used for regression is equal to the sum $\sum_{i=1}^{k}\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\mathbf{X}^i = \mathbf{A}_\mathbf{Y}\mathbf{P}'$ where the vectors $\mathbf{p}_i = \mathbf{X}^{i\prime}\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}$ make up the columns of $\mathbf{P}$. Analogously, define $\mathbf{c}_i = \mathbf{Y}^{i\prime}\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}$ making up the columns of $\mathbf{C}$.

Now, if we would go on with the deflations until the rank of $\mathbf{X}^i$ is zero[9], the space spanned by the orthogonal vectors $\boldsymbol{\alpha}_{\mathbf{Y},i}$ would be complete and we would have that

$$\mathbf{X} = \mathbf{A}_\mathbf{Y}^{\text{tot}}\mathbf{P}^{\text{tot}\prime} = \mathbf{A}_\mathbf{Y}\mathbf{P}' + \mathbf{A}_\mathbf{Y}^{\text{rem}}\mathbf{P}^{\text{rem}\prime} = \mathbf{A}_\mathbf{Y}\mathbf{P}' + \mathbf{E}_\mathbf{X},$$

with $\mathbf{E}_\mathbf{X}$ the part of $\mathbf{X}$ that is not used in regression when the components corresponding to $\mathbf{A}_\mathbf{Y}^{\text{rem}}$ are not kept. Also, because of (2.19) and the definition of $\mathbf{P}$: $\mathbf{p}_j{}'\mathbf{w}_{\mathbf{X},i} = 0$ for $i<j$, and thus:

$$\mathbf{P}^{\text{rem}\prime}\mathbf{W}_\mathbf{X} = \mathbf{0}.$$

This leads to the linear mapping from $\mathbf{X}$ to $\mathbf{A}_\mathbf{Y}$:

$$\begin{aligned}
\mathbf{A}_\mathbf{Y}\mathbf{P}'\mathbf{W}_\mathbf{X} &= \mathbf{X}\mathbf{W}_\mathbf{X} \\
\Rightarrow \mathbf{A}_\mathbf{Y} &= \mathbf{X}\mathbf{W}_\mathbf{X}\left(\mathbf{P}'\mathbf{W}_\mathbf{X}\right)^{-1}, \qquad (2.23)
\end{aligned}$$

where the matrix to be inverted is lower triangular (again because $\mathbf{p}_j{}'\mathbf{w}_{\mathbf{X},i} = 0$ for $i<j$), so the inversion can be carried out efficiently.

The regression from the latent variables $\boldsymbol{\alpha}_\mathbf{Y}$ towards $\mathbf{Y}$ is given by

$$\mathbf{Y} = \sum_{i=1}^{k}\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\mathbf{Y}^i + \mathbf{Y}^{k+1} = \mathbf{A}_\mathbf{Y}\mathbf{C}' + \mathbf{E}_\mathbf{Y}, \qquad (2.24)$$

[9]Note that the number of deflations k will always be smaller (or equal, in full LSR) than the rank of $\mathbf{X}$. This results in matrices $\mathbf{W}_\mathbf{X}, \mathbf{W}_\mathbf{Y}, \mathbf{A}'_\mathbf{X}, \mathbf{A}'_\mathbf{Y}, \mathbf{P}$ and $\mathbf{C}$ all having k columns.

where $\mathbf{E_Y} = \mathbf{Y}^{k+1}$ is the part of $\mathbf{Y}$ that is not predicted by the first k PLS-components (the misfit).

Thus, the entire PLS regression formula is given by

$$\mathbf{y}_{\text{pred}} = \left[\mathbf{W_X}\left(\mathbf{P}'\mathbf{W_X}\right)^{-1}\mathbf{C}'\right]'\mathbf{x}_{\text{pred}} = \left[\mathbf{C}\left(\mathbf{W}'_{\mathbf{X}}\mathbf{P}\right)^{-1}\mathbf{W}'_{\mathbf{X}}\right]\mathbf{x}_{\text{pred}}.$$

Dual Let us define $\mathbf{A_X}$ as the matrix containing $\boldsymbol{\alpha}_{\mathbf{X},i}$ as its columns. Now we use the properties (2.22) and (2.19), showing that $\mathbf{W_X} = \mathbf{X}'\mathbf{A_X}$ and $\mathbf{X}^{k+1}\mathbf{W_X} = \mathbf{0}$ leading to $\mathbf{W}'_{\mathbf{X}}\mathbf{P} \propto \mathbf{W}'_{\mathbf{X}}\mathbf{X}'\mathbf{A_Y} = \mathbf{A}'_{\mathbf{X}}\mathbf{K_X}\mathbf{A_Y}$, where the proportionality is an equality up to a diagonal normalization matrix $\mathbf{A}'_{\mathbf{Y}}\mathbf{A_Y}$ on the right hand side. Furthermore, using (2.20) it is seen that $\mathbf{E}'_{\mathbf{Y}}\mathbf{A_Y} = \mathbf{0}$ and thus (from (2.24)) that with the same diagonal normalization matrix as proportionality factor (which will thus be cancelled out), $\mathbf{C} \propto \mathbf{C}\mathbf{A}'_{\mathbf{Y}}\mathbf{A_Y} = \mathbf{Y}'\mathbf{A_Y}$. This leads to the complete dual form of Regression-PLS:

$$\mathbf{y}_{\text{pred}} = \left[\mathbf{Y}'\mathbf{A_Y}\left(\mathbf{A}'_{\mathbf{X}}\mathbf{K_X}\mathbf{A_Y}\right)^{-1}\mathbf{A}'_{\mathbf{X}}\mathbf{X}\right]\mathbf{x}_{\text{pred}}.$$

Note that the entire algorithm only requires the evaluation of kernel functions, since $\mathbf{X}\mathbf{x}_{\text{pred}}$ also consists of inner products only (or equivalently kernel evaluations $k(\cdot,\cdot)$). Using this fact, the solution can be cast in the standard form of kernel based pattern recognition algorithms:

$$\mathbf{y}_{\text{pred}} = \sum_i \beta_i k(\mathbf{x}_i, \mathbf{x}_{\text{pred}}), \tag{2.25}$$

where $\boldsymbol{\beta}_i$ are the columns of $\boldsymbol{\beta} = \mathbf{Y}'\mathbf{A_Y}\left(\mathbf{A}'_{\mathbf{X}}\mathbf{K_X}\mathbf{A_Y}\right)^{-1}\mathbf{A}'_{\mathbf{X}}$.

Demonstration

A demonstration of PLS is difficult to give for our leading example, and therefore omitted. The reason is that PLS is mainly developed in a regression context (although its use for classification has been proposed in literature). On the other hand, since PLS dimensionality reduction in fact corresponds to a maximally regularized RCCA, a demonstration of the dimensionality reduction capabilities would be redundant as well. However, a visual comparison with CCA is offered below.

2.2.4 Illustrative comparison of PCA, CCA and PLS dimensionality reduction

In each of the discussions of PCA, (R)CCA and PLS, their features have been discussed. In Figure 2.9 a visual illustration of these properties is offered. (The samples are randomly generated, after which the different techniques are applied.)

The first row shows what the result of PCA would be: the dominant direction is the direction along which the data shows the largest variance. No

interaction between the dimensionality reductions of both data sets is present. Note that the second PCA direction is orthogonal to the first.

The second row shows the result of CCA on this data set. Now the directions are not orthogonal anymore, but the projections of the samples on them are uncorrelated (this is hard to show pictorially, however can mentally project the samples onto the CCA directions and verify that they are visually uncorrelated). Also note that the variance does not matter here: CCA directions are not biased towards high variance directions whatsoever, since correlations are normalized by the variance.

The third row shows the PLS directions. As in PCA, PLS directions in one space are orthogonal to each other. Furthermore, the dominant PLS direction searches a compromise between a direction of large variance, and a direction that strongly correlates with the corresponding direction in the other data representation. Therefore, the PLS directions can be thought of being situated between the PCA and CCA directions.

The fourth row contains the RCCA directions for some non-trivial value of the regularization parameter. The PLS and CCA directions are repeated in dash-dotted and dashed line. Indeed, RCCA interpolates in some way between PLS and CCA, finding an optimal compromise between directions of large correlation and (more robust) directions of large covariance.

2.3 Classification: Fisher Discriminant Analysis (FDA)

Definitions We first define some symbols necessary to develop the theory. Since these quantities are defined in general for uncentered data, first this general definition is given. Afterwards when appropriate the simplified formula will be provided for centered data. The latter formulas are the ones used in this section.

- Mean (n is the total number of samples $\mathbf{x}_i$)

$$\mathbf{m} = \frac{1}{n} \sum_i \mathbf{x}_i.$$

- Class mean ($\mathcal{S}_k$ is the set of samples belonging to cluster k, and $n_k = |\mathcal{S}_k|$, the number of samples in cluster k; thus $n = \sum_k n_k$)

$$\mathbf{m}_k = \frac{1}{n_k} \sum_{i:\mathbf{x}_i \in \mathcal{S}_k} \mathbf{x}_i.$$

- Total scatter matrix

$$\mathbf{S}_T = \sum_k \sum_{\mathbf{x}_i \in \mathcal{S}_k} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})'.$$

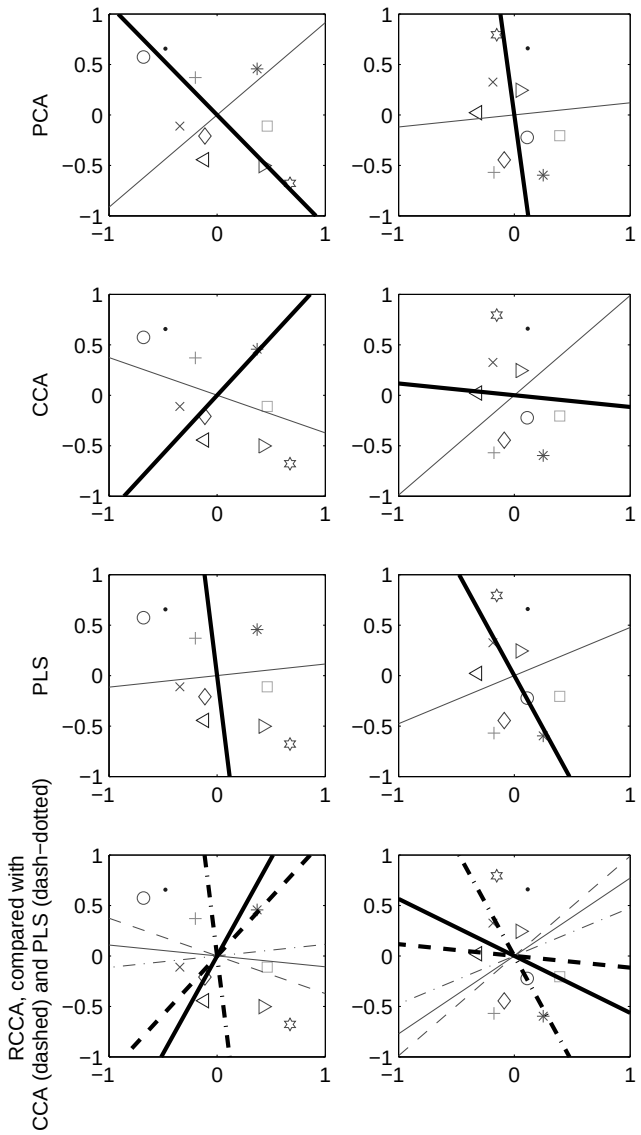


Figure 2.9: A visual comparison of the different techniques for dimensionality reduction. In the left and right column, two different 2-dimensional spaces are shown, each corresponding to a different representation of the data (e.g., the left figure could contain a representation of text documents in Dutch, on the right a representation of the same documents in French). Corresponding samples are represented in both panels by the same symbol. Each row demonstrates one of the dimensionality reduction techniques on this data, where the dominant (generalized) eigenvector is shown with a bold line and the second (generalized) eigenvector with a fine line. For interpretations and explanations, see the main text.

- Within class k scatter matrix

$$\mathbf{S}_k = \sum_{\mathbf{x}_i \in \mathcal{S}_k} (\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)'.$$

- Within class scatter matrix

$$\mathbf{S}_W = \sum_k \mathbf{S}_k. \tag{2.26}$$

- Between class scatter matrix

$$\mathbf{S}_B = \sum_k n_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})'.$$

For centered data we get:

$$\begin{aligned} \mathbf{m} &= \mathbf{0} \\ \frac{1}{n}\sum_k n_k \mathbf{m}_i &= \mathbf{0} \\ \mathbf{S}_T &= \sum_k \sum_{\mathbf{x}_i \in \mathcal{S}_k} \mathbf{x}_i \mathbf{x}_i' = \mathbf{X}'\mathbf{X} = \mathbf{S}_{\mathbf{XX}} \\ \mathbf{S}_B &= \sum_k n_k \mathbf{m}_k \mathbf{m}_k'. \end{aligned}$$

The following properties hold:

- $\mathbf{S}_T = \mathbf{S}_B + \mathbf{S}_W$.
- When the number of classes is 2, they can be indexed as + and -, and:

$$\mathbf{S}_B = \frac{n_+ n_-}{n}(\mathbf{m}_+ - \mathbf{m}_-)(\mathbf{m}_+ - \mathbf{m}_-)'. \tag{2.27}$$

2.3.1 Cost function

Fisher discriminant analysis (FDA) [40] is designed for discrimination between 2 classes, indexed by + and -. It finds the direction $\mathbf{w}$ along which the between class variance divided by within class variance is maximized:

$$\mathbf{w} = \text{argmax}_{\mathbf{w}} \frac{\mathbf{w}'\mathbf{S}_B\mathbf{w}}{\mathbf{w}'\mathbf{S}_W\mathbf{w}}. \tag{2.28}$$

Note that when $\mathbf{w}$ is a solution, $c\mathbf{w}$ with c a real number is a solution too. In fact, we are not interested in the norm of $\mathbf{w}$, but only in the direction it is pointing at. Thus, equivalently, we could optimize the constrained optimization problem

$$\begin{aligned} \mathbf{w} &= \text{argmax}_{\mathbf{w}} \mathbf{w}'\mathbf{S}_B\mathbf{w} \\ &\text{s.t.} \quad \mathbf{w}'\mathbf{S}_W\mathbf{w} = 1. \end{aligned} \tag{2.29}$$

2.3.2 Primal

This optimization problem can be solved by differentiating the Lagrangian $\mathcal{L}(\mathbf{w}, \mu) = \mathbf{w}'\mathbf{S}_B\mathbf{w} - \mu\mathbf{w}'\mathbf{S}_W\mathbf{w}$ with respect to $\mathbf{w}$ and equating to zero:

$$\begin{aligned} \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}, \mu) &= \mathbf{0} \\ \Rightarrow \mathbf{S}_B\mathbf{w} &= \mu\mathbf{S}_W\mathbf{w}. \end{aligned} \tag{2.30}$$

This is again a generalized eigenvalue problem. Furthermore, $\mathbf{S}_B$ and $\mathbf{S}_W$ are both symmetric and positive semi definite as is obvious from the definition.

Connection with CCA

Another way to get the same result, is by optimizing the correlation between the data projected on a weight vector $\mathbf{w}$, with the labels $\mathbf{y}$ (for each sample being 1 or -1 depending on the class the sample belongs to) of the corresponding data points. This is in fact CCA, applied on the data vectors on the one hand, and the labels on the other hand:

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{Xy}} \\ \mathbf{S_{yX}} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ w_\mathbf{y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{S_{XX}} & \mathbf{0} \\ \mathbf{0} & \mathbf{S_{yy}} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ w_\mathbf{y} \end{pmatrix},$$

from which $\mathbf{w_X}$ can be solved as

$$\mathbf{S_{XX}}^{-1}\mathbf{S_{Xy}}\mathbf{S_{yy}}^{-1}\mathbf{S_{yX}}\mathbf{w_X} = \lambda^2\mathbf{w_X}.$$

To see that $\mathbf{w_X} = \mathbf{w}$, note that for centered data $\mathbf{X}$ (so $\mathbf{m}$ is made equal to $\mathbf{0}$ by centering), $\mathbf{S_{XX}} = \mathbf{S}_T = \mathbf{S}_B + \mathbf{S}_W$, $\mathbf{S_{yy}} = n$ is a scalar, and $\mathbf{S_{Xy}} = \mathbf{X}'\mathbf{y} = n_+\mathbf{m}_+ - n_-\mathbf{m}_-$. One can then show that $\mathbf{S_{Xy}}\mathbf{S_{yy}}^{-1}\mathbf{S_{yX}} = \frac{4n_+n_-}{n^2}\mathbf{S}_B$, and thus

$$\begin{aligned} \frac{4n_+n_-}{n^2}\mathbf{S}_B\mathbf{w_X} &= \lambda^2(\mathbf{S}_B + \mathbf{S}_W)\mathbf{w_X} \\ \Rightarrow \mathbf{S}_B\mathbf{w_X} &= \frac{\lambda^2}{\frac{4n_+n_-}{n^2} - \lambda^2}\mathbf{S}_W\mathbf{w_X}. \end{aligned}$$

This is exactly the Fisher discriminant generalized eigenvalue problem, with $\mu = \frac{\lambda^2}{\frac{4n_+n_-}{n^2} - \lambda^2}$ and $\mathbf{w} = \mathbf{w_X}$.

2.3.3 Dual

Define $\mathbf{y}_+$ as $(\mathbf{y}_+)_i = \delta_{y_i,1}$ and $\mathbf{y}_-$ as $(\mathbf{y}_-)_i = \delta_{y_i,-1}$ (where we use the Kronecker delta $\delta_{i,j}$, which is equal to 1 if $i = j$ and to 0 if $i \neq j$). The dual can again be

Table 2.1: Classification error rates of FDA on the noise free data, averaged over 100 randomizations with balanced 80/20 splits in training and test data (*balanced* meaning that the proportion of positive samples in the training set is the same as in the test set). The 2-mer kernel is used with a fixed kernel width, so no (cross-)validation has been performed.

	English vs French	English vs German	English vs Italian
FDA	5.0 ± 0.2	0 ± 0	1.2 ± 0.1

derived by using $\mathbf{w} = \mathbf{X}'\boldsymbol{\alpha}$:

$$\begin{array}{c} \mathbf{S}_B\mathbf{w} = \mu\mathbf{S}_W\mathbf{w} \\ \Downarrow (2.26), (2.27) \\ \frac{n_+n_-}{n}\mathbf{X}(\mathbf{m}_+ - \mathbf{m}_-)(\mathbf{m}_+ - \mathbf{m}_-)'\mathbf{X}'\boldsymbol{\alpha} \\ = \quad \mu\mathbf{X}\sum_{k=+,-}\sum_{\mathbf{x}_i \in \mathcal{S}_k}(\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)'\mathbf{X}'\boldsymbol{\alpha} \\ \Downarrow \\ \frac{n_+n_-}{n}\mathbf{K}_\mathbf{X}\left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right)\left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right)'\mathbf{K}_\mathbf{X}\boldsymbol{\alpha} \\ = \mu\mathbf{K}_\mathbf{X}\left(\mathbf{I} - \frac{1}{n_+}\mathbf{y}_+\mathbf{y}_+' - \frac{1}{n_-}\mathbf{y}_-\mathbf{y}_-'\right)\mathbf{K}_\mathbf{X}\boldsymbol{\alpha} \\ \Downarrow \\ \mathbf{M}\boldsymbol{\alpha} = \mu\mathbf{N}\boldsymbol{\alpha}, \end{array}$$

where we substituted $\mathbf{M} = \frac{n_+n_-}{n}\mathbf{K}_\mathbf{X}\left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right)\left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right)\mathbf{K}_\mathbf{X}'$ and $\mathbf{N} = \mathbf{K}_\mathbf{X}\left(\mathbf{I} - \frac{1}{n_+}\mathbf{y}_+\mathbf{y}_+' - \frac{1}{n_-}\mathbf{y}_-\mathbf{y}_-'\right)\mathbf{K}_\mathbf{X}$.

For centered data as is assumed here, the projection of a test point $\mathbf{x}_{\text{test}}$ onto the FDA direction corresponding to $\boldsymbol{\alpha}$ can again be computed as

$$\sum_{i=1}^{n}\alpha_i k(\mathbf{x}_i, \mathbf{x}_{\text{test}}).$$

Demonstration

We can demonstrate the use of FDA on the classification of texts from our leading example into their respective language classes. The kernel we use here is the 2-mer kernel. We considered 3 binary classification problems, discriminating English texts from the texts in other languages (averaged over 100 random balanced splits in training (80%) and test sets (20%)). Error rates are in table 2.3.3.

English and French are hardest to distinguish based on the 2-mer kernel, which is probably due to many loan words present in English that have been adopted from French in the recent past. Also, English and Italian are not

perfectly distinguished (probably due to the same fact, and due to the fact that many English words have a Roman origin). German sticks out most clearly, which is to be expected.

2.3.4 Linear discriminant analysis (LDA)

While Fisher discriminant analysis is originally designed for the two-class problem, optimization of the very same cost function ((2.28) and (2.29)) leading to the same generalized eigenvalue problem (2.30) can be used for solving the multi-class problem (e.g. [36]). In that case, a few generalized eigenvector may be necessary to do the classification (typically the number of clusters minus one).

The intuition behind this is to maximize the total between class covariance for a certain amount of within class covariance. This amounts to maximizing the signal to noise ratio present in the projections of the samples onto the discriminant directions. Here, the distance between the projected clusters is the signal one is interested in, and the variance in the projections of the clusters is the noise. Interestingly, it has been shown that PLS also maximizes the between class covariance when computed on a class indicator matrix $\mathbf{Y}$, however, this is done without considering the within class covariance [8, 92].

Deriving the dual version of LDA can be done in a similar way as for FDA. We will return to the LDA problem in Section 3.1.

2.4 Spectral methods for clustering

Clustering is arguably the standard problem in pattern recognition: identify groups of points (samples) that supposedly belong to the same class, without any information on the class labels (unsupervised).

A large number of classical algorithms are available, of which the K-means algorithm is the best known. However, while most of them are primarily designed for data with Gaussian class distributions, this is an oversimplification in many cases. Furthermore, many well known algorithms are not convex.

Therefore in recent years a significant amount of research has been done on spectral methods for clustering [95, 120, 85, 103, 65, 5, 34, 11]: the clustering problem is relaxed or restated, leading to efficient algorithms requiring no more than solving an eigenvalue problem. And maybe even more importantly, no Gaussianity assumptions are made for spectral clustering algorithms.

We want to point out that unlike the other algorithms in this chapter, spectral clustering algorithms are not studied in a primal-dual context. Still, their formulation is similar to the dual versions of the algorithms described thus far, which is why we think their discussion deserves a place in this chapter.

2.4.1 The affinity matrix

Whereas standard clustering methods often assume Gaussian class distributions (or make other assumptions on the class distributions), spectral clustering methods do not. In order to achieve this goal, one avoids the use of the Euclidian distance or inner product. Instead, a measure is used that is sophisticated on a small scale (i.e. locally it is accurate), but on a larger scale it views all samples as being equally distant from each other. More specifically, the similarity measure used in spectral clustering algorithms is very often a radial basis function (RBF):

$$\mathrm{RBF}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right).$$

This similarity measure is positive definite; therefore, one often speaks of an *RBF kernel*.

Note that for $\mathbf{x}_i \simeq \mathbf{x}_j$, the RBF kernel is $\mathrm{RBF}(\mathbf{x}_i, \mathbf{x}_j) \simeq 1 - \frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}$. Thus, locally, the RBF kernel is in some sense similar to the Euclidian metric. However, for two points at a larger Euclidian distance from each other, their affinity rapidly approaches 0. The result is that it will not see if a larger group of points is normally distributed or not. It only sees if two given points are lying close to each other. This is desirable: it allows us to cluster points that are stretched out in a nonlinear shape.

Even though in spectral clustering often an RBF kernel is used, in fact the positive definiteness of the RBF kernel is not a requirement here. Instead, the matrix containing the pairwise similarities should be symmetric and all entries must be positive (which is indeed also true for the RBF kernel). This is why, in this context a different terminology is appropriate: the matrix containing the similarities between all pairs of samples will be referred to as the *affinity matrix* (and not as the *kernel matrix*). Therefore we will use the notation $\mathbf{A}$ for the affinity matrix and $a(\mathbf{x}_i, \mathbf{x}_j) = a_{ij}$ for its elements.

We want to point out that many other affinity matrices are used in literature, among which the *k-nearest neighbor affinity* matrix appears particularly useful in many practical cases.

2.4.2 Cut, average cut and normalized cut cost functions

Now we know how to define an affinity matrix, we will explain how to use it to deal with complex cluster shapes.

Spectral clustering methods originate as (spectral) relaxations of graph cut problems on a fully connected graph, where the nodes in the graph represent the samples, and the edges between them are assigned weights equal to the affinities. Clustering then corresponds to dividing the nodes of this graph into groups. This partitioning of the graph is technically called a *graph cut*. Of course, a graph cut that separates nodes that are interconnected by an edge with large affinity is undesirable. To quantify this, several graph cut cost functions for clustering

have been proposed in literature, among which the *cut cost*, the *average cut cost (ACut)* and the *normalized cut cost (NCut)* [103].

The Cut cost is computationally the easiest to handle [23], however as clearly motivated in [62], it often leads to degenerate results (where all but one of the clusters are trivially small). This problem can largely be solved by using the ACut or NCut cost functions, of which the ACut cost seems to be more vulnerable to outliers (distant samples, meaning that they have low affinity to all other points). However, both optimizing the ACut and NCut costs are NP-complete problems. To get around this, *spectral* relaxations of the ACut and NCut optimization problems have been proposed [103, 86, 34].

Below, we provide a new concise derivation of the NCut optimization problem in the *two-class setting* (we will speak of the positive and negative class). Extending this to the multi-class setting is possible. The ACut cost function can be treated very similarly (we will outline this in a next paragraph).

We will need the following shorthand notations for *cuts* and *associations* between complementary sets $\mathcal{P}$ and $\mathcal{N}$ of nodes in the graph:

$$\text{cut}(\mathcal{P},\mathcal{N}) = \text{cut}(\mathcal{N},\mathcal{P}) = \sum_{i:\mathbf{x}_i\in\mathcal{P},j:\mathbf{x}_j\in\mathcal{N}} a_{ij}$$

is the cut between sets $\mathcal{P}$ and $\mathcal{N}$, and

$$\text{assoc}(\mathcal{P},\mathcal{S}) = \sum_{i:\mathbf{x}_i\in\mathcal{P},j:\mathbf{x}_j\in\mathcal{S}} a_{ij}$$

the association between sets $\mathcal{P}$ and the full sample $\mathcal{S}$. Note that in fact $\text{cut}(\mathcal{P},\mathcal{N}) = \text{assoc}(\mathcal{P},\mathcal{N})$.

The NCut optimization problem

The NCut cost function for a partitioning of the sample $\mathcal{S}$ into a positive set $\mathcal{P}$ and a negative set $\mathcal{N}$ is given by (as originally formulated in [103]):

$$\frac{\text{cut}(\mathcal{P},\mathcal{N})}{\text{assoc}(\mathcal{P},\mathcal{S})} + \frac{\text{cut}(\mathcal{N},\mathcal{P})}{\text{assoc}(\mathcal{N},\mathcal{S})} = \left(\frac{1}{\text{assoc}(\mathcal{P},\mathcal{S})} + \frac{1}{\text{assoc}(\mathcal{N},\mathcal{S})}\right)\cdot \text{cut}(\mathcal{P},\mathcal{N}). \quad (2.31)$$

We do not discuss here its statistical properties. Intuitively, however, it is clear that the second factor $\text{cut}(\mathcal{P},\mathcal{N})$ defines how well the two clusters separate. The first factor $\left(\frac{1}{\text{assoc}(\mathcal{P},\mathcal{S})} + \frac{1}{\text{assoc}(\mathcal{N},\mathcal{S})}\right)$ measures how well the clusters are balanced in some sense. This specific measure of balancedness can be seen to be relatively insensitive to distant samples:[10] such outliers have a small cut cost with the other samples, making it beneficial to separate them out into a cluster of their own, which would lead to a useless result in our two-class setting. However, they also have a small association with the rest of the sample $\mathcal{S}$, which

[10] This property seems even more important in the relaxations of NCut based methods: the variables then have even more freedom, often making the methods more vulnerable to outliers.

on the other hand increases the cost function. In other words, the NCut cost function promotes partitions that are balanced in the sense that both clusters are roughly equally 'coherent'. It is this feature that makes it preferable over the ACut cost function, which is more vulnerable to outliers (see below).

To optimize this cost function, we reformulate it into algebraic terms using the unknown label vector $\mathbf{y} \in \{-1, 1\}^n$, the affinity matrix $\mathbf{A}$, the degree vector $\mathbf{d} = \mathbf{A1}$ and associated matrix $\mathbf{D} = \text{diag}(\mathbf{d})$, and shorthand notations $s_+ = \text{assoc}(\mathcal{P}, \mathcal{S})$ and $s_- = \text{assoc}(\mathcal{N}, \mathcal{S})$.

Observe that $\text{cut}(\mathcal{P}, \mathcal{N}) = \frac{(\mathbf{1}+\mathbf{y})'}{2}\mathbf{A}\frac{(\mathbf{1}-\mathbf{y})}{2} = \frac{1}{4}\left(-\mathbf{y}'\mathbf{A}\mathbf{y} + \mathbf{1}'\mathbf{A}\mathbf{1}\right) = \frac{1}{4}\mathbf{y}'(\mathbf{D} - \mathbf{A})\mathbf{y}$. Furthermore, $s_+ = \text{assoc}(\mathcal{P}, \mathcal{S}) = \frac{1}{2}\mathbf{1}'\mathbf{A}(\mathbf{1} + \mathbf{y}) = \frac{1}{2}\mathbf{d}'(\mathbf{1} + \mathbf{y})$ and $s_- = \frac{1}{2}\mathbf{d}'(\mathbf{1} - \mathbf{y})$. Then we can write the combinatorial optimization problem as

$$\begin{aligned} \min_{\mathbf{y}, s_+, s_-} \quad & \frac{1}{4}\left(\frac{1}{s_+} + \frac{1}{s_-}\right) \cdot \mathbf{y}'(\mathbf{D} - \mathbf{A})\mathbf{y} \\ \text{s.t.} \quad & \mathbf{y} \in \{-1, 1\}^n, \\ & \begin{cases} s_+ = \frac{1}{2}\mathbf{d}'(\mathbf{1} + \mathbf{y}) \\ s_- = \frac{1}{2}\mathbf{d}'(\mathbf{1} - \mathbf{y}) \end{cases} \Leftrightarrow \begin{cases} \mathbf{d}'\mathbf{y} = s_+ - s_- \\ \mathbf{d}'\mathbf{1} = s_+ + s_- \end{cases} \end{aligned} \tag{2.32}$$

This algebraic formulation is easier to handle as will become clear below. Note the problem here: while this looks like a nice optimization problem, the binary constraint on $\mathbf{y}$ makes the time complexity of solving it exponential.

Spectral relaxation We now provide a remarkably short derivation of a spectral relaxation of this optimization problem, as first derived in a less transparent way in [103]. We introduce the variable $\widetilde{\mathbf{y}} = \frac{1}{2}\left(\mathbf{y} - \mathbf{1}\frac{s_+ - s_-}{s_+ + s_-}\right) \cdot \sqrt{\frac{1}{s_+} + \frac{1}{s_-}}$ and rewrite the optimization problem in terms of this variable, s_+ and s_-:

$$\begin{aligned} \min_{\widetilde{\mathbf{y}}, s_+, s_-} \quad & \widetilde{\mathbf{y}}'(\mathbf{D} - \mathbf{A})\widetilde{\mathbf{y}} \\ \text{s.t.} \quad & \widetilde{\mathbf{y}} \in \left\{-\sqrt{\frac{s_+}{s_-}}\sqrt{\frac{1}{s_+ + s_-}}, \sqrt{\frac{s_-}{s_+}}\sqrt{\frac{1}{s_+ + s_-}}\right\}^n, \\ & \mathbf{d}'\widetilde{\mathbf{y}} = 0 \quad \text{and} \quad \mathbf{d}'\mathbf{1} = s_+ + s_-. \end{aligned} \tag{2.33}$$

Now, observe that the $\mathbf{D}$ weighted 2-norm of $\widetilde{\mathbf{y}}$ is constant here, and equal to $\widetilde{\mathbf{y}}'\mathbf{D}\widetilde{\mathbf{y}} = 1$. The spectral relaxation is obtained by relaxing the combinatorial constraint on $\widetilde{\mathbf{y}}$ to this norm constraint that is compatible with it. The result is

$$\textbf{Spectral}: \begin{cases} \min\limits_{\widetilde{\mathbf{y}}} & \widetilde{\mathbf{y}}'(\mathbf{D} - \mathbf{A})\widetilde{\mathbf{y}} \\ \text{s.t.} & \widetilde{\mathbf{y}}'\mathbf{D}\widetilde{\mathbf{y}} = 1 \quad \text{and} \quad \mathbf{d}'\widetilde{\mathbf{y}} = 0, \end{cases} \tag{2.34}$$

which can be solved by taking the eigenvector corresponding to the second smallest generalized eigenvalue of $(\mathbf{D} - \mathbf{A})\widetilde{\mathbf{y}} = \lambda\mathbf{D}\widetilde{\mathbf{y}}$.

The ACut cost function

We only briefly state a similar result for the ACut cost. (Note that all derivations are in fact easier.) The cost function is (with n_+ the number of positively labeled samples and n_- the number of negatively labeled samples):

$$\frac{\text{cut}(\mathcal{P},\mathcal{N})}{n_+} + \frac{\text{cut}(\mathcal{N},\mathcal{P})}{n_-} = \left(\frac{1}{n_+} + \frac{1}{n_-}\right) \cdot \text{cut}(\mathcal{P},\mathcal{N}), \tag{2.35}$$

Or, in algebraic notation:

$$\sum_{i,j:y_i \neq y_j} a_{ij} \left(\frac{2}{\mathbf{1}'(\mathbf{1}+\mathbf{y})} + \frac{2}{\mathbf{1}'(\mathbf{1}-\mathbf{y})}\right) = \frac{1}{2}\mathbf{y}'(\mathbf{D}-\mathbf{A})\mathbf{y},$$

and gives rise to a similar eigenvalue problem to be solved after relaxation:

$$(\mathbf{D}-\mathbf{A})\widetilde{\mathbf{y}} = \lambda\widetilde{\mathbf{y}}.$$

The eigenvector $\widetilde{\mathbf{y}}$ corresponding to the second smallest eigenvalue contains a relaxation (and thus approximation) of the labels.

Multi-class clustering

The above was for when we want to perform a bipartitioning, or clustering into two clusters. When one needs $c > 2$ clusters, typically one will simply use $c-1$ eigenvectors instead of just 1, as explained in the following subsection. While this approach can also be motivated based on relaxations of a cost function, we will not go into this here.

2.4.3 What to do with the eigenvectors?

Thus far, we have discussed how to compute eigenvectors that reflect the clustering in some way. In the case of two classes, the clustering can be obtained by simply thresholding the entries in the eigenvector (most often thresholded around zero). Every entry in this thresholded eigenvector corresponds to one of the data points, and indicates which of both classes it is assigned to. For multi-class clustering, one constructs a matrix $\widetilde{\mathbf{Y}} = (\widetilde{\mathbf{y}}_1\widetilde{\mathbf{y}}_2\cdots\widetilde{\mathbf{y}}_{c-1})$ containing the eigenvectors as its columns. Then some Euclidian distance based clustering (such as K-means) is performed on the rows of $\widetilde{\mathbf{Y}}$ in this $c-1$-dimensional space. For further reading on different possible approaches we refer to the literature, see e.g. [95, 85, 130].

Demonstration

We now return to our leading example, to show the performance of spectral clustering in comparison to K-means on various clustering problems.

Table 2.2: Adjusted Rand index performances for spectral clustering and K-means clustering of the documents. The ideal clustering is clustering per language.

	BoW	2-mer	4-mer
Spectral clustering	0.966 ± 0	0.437 ± 0	0.337 ± 0
K-means	0.38 ± 0.04	0.17 ± 0.03	0.26 ± 0.04

Clustering the articles in their language clusters Using the spectral clustering method based on the normalized cut, we cluster the articles of all languages, and check how well they are clustered into their respective language clusters. To assess the performance we use the adjusted Rand index [59], which is 1 for perfect clustering and has an expected value of 0 for random clustering. The final step consists of K-means on the eigenvectors, the clustering corresponding to the minimal K-means cost is taken over 10 starting values, chosen as described in [86].

For comparison, we perform kernel K-means on the documents. After 100 random initializations of K-means, the one with the best K-means cost is taken, and its adjusted Rand index is computed.

The results are summarized in Table 2.4.3. The numbers in the table are averages over 10 runs along with the standard deviations on these averages. Note that spectral clustering (virtually) always returns the same optimal value (very small standard deviation), i.e. it is quite independent of the starting values in the K-means iterations, whereas K-means appears more sensitive to the start value.

Somewhat surprisingly the 2-mer kernel performs better than the 4-mer kernel with the spectral clustering. As expected the best performance is achieved with the BoW kernel. The spectral method outperforms K-means in all cases.

Clustering the articles into coherent groups The articles in the constitution are organized into groups, called 'Titles'. Can we use clustering to automatically categorize the articles into their Titles?

See table 2.4.3 for the adjusted Rand scores achieved on this clustering problem for the different languages, kernels and methods. The performances are much less than for clustering articles into their language classes. This is of course to be expected: now the number of samples is smaller, and the distinction between languages is an objective criterion, while the distinction between Titles in the constitution is man-made and thus subjective in nature. Still, the performance is well above what a random clustering would do.

Table 2.3: Adjusted Rand indices for spectral clustering of the English articles into the Titles they appear in.

		BoW	2-mer	4-mer
English	Spectral clustering	0.326 ± 0	0.231 ± 0	0.328 ± 0.001
	K-means	0.24 ± 0.02	0.24 ± 0.02	0.27 ± 0.02
French	Spectral clustering	0.372 ± 0	0.206 ± 0	0.340 ± 0
	K-means	0.23 ± 0.03	0.17 ± 0.01	0.30 ± 0.02
German	Spectral clustering	0.559 ± 0	0.136 ± 0	0.241 ± 0
	K-means	0.13 ± 0.02	0.12 ± 0.01	0.19 ± 0.02
Italian	Spectral clustering	0.508 ± 0.001	0.214 ± 0	0.308 ± 0
	K-means	0.26 ± 0.02	$0.0.19 \pm 0.01$	0.31 ± 0.03

2.5 Summary

Table 2.4 contains the cost functions optimized for most of the algorithms described in this chapter. Tables 2.5 and 2.6 give the primal and the dual eigenproblems to be solved in order to optimize these cost functions. These tables contain a column $\mathbf{M}$, $\mathbf{N}$ and $\mathbf{v}$, each indicating which matrices and eigenvector to use in the generalized eigenproblem of the form $\mathbf{Mv} = \lambda\mathbf{Nv}$.

Given this, for PCA, CCA, PLS and FDA we still need to know how to project data on the directions found by solving these generalized eigenproblems. This is summarized as:

- Projection of a test sample onto weight vector in primal space $\mathbf{w}$: $\mathbf{w}'\mathbf{x}_{\text{test}}$.
- Projection of a test sample onto weight vector in feature space corresponding to the dual vector $\boldsymbol{\alpha}$: $\sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}_{\text{test}})$.

2.6 Conclusions

Among the algorithms discussed in this chapter, there are a number of classic methods from multivariate statistics, such as PCA and CCA; some methods that are virtually unknown in that field but are hugely popular in specific application domains, such as PLS in chemometrics; and finally some methods that are typically the product of the machine learning community, such as the clustering methods presented here, and all the extensions based on the use of kernels. Despite coming from so many different fields, the algorithms clearly display their common features, and we have emphasized them by casting them in a common notation and with a common language. From those comparisons, and from the comparison with the family of Kernel Methods based on Quadratic Programming, it is clear that this approach based on spectral methods can be considered another major branch of the KM family. The duality that emerges

Table 2.4: Cost functions optimized by the different methods.

PCA	Maximize variance	$\frac{\mathbf{w}'\mathbf{S_{XX}}\mathbf{w}}{\mathbf{w}'\mathbf{w}}$
		$\mathbf{w}'\mathbf{S_{XX}}\mathbf{w}$ s.t. $\|\mathbf{w}\|^2 = 1$
	Minimize residuals	$\|(\mathbf{I} - \mathbf{w}\mathbf{w}')\mathbf{X}\|_F^2$
CCA	Maximize correlation	$\frac{\mathbf{w}_\mathbf{X}'\mathbf{S_{XY}}\mathbf{w_Y}}{\sqrt{\mathbf{w}_\mathbf{X}'\mathbf{S_{XX}}\mathbf{w_X}}\sqrt{\mathbf{w}_\mathbf{Y}'\mathbf{S_{YY}}\mathbf{w_Y}}}$
	Maximize fit	$\mathbf{w}_\mathbf{X}'\mathbf{S_{XY}}\mathbf{w_Y}$ s.t. $\|\mathbf{X}\mathbf{w_X}\|^2 = \|\mathbf{Y}\mathbf{w_Y}\|^2 = 1$
	Minimize misfit	$\|\mathbf{w}_\mathbf{X}'\mathbf{X} - \mathbf{w}_\mathbf{Y}'\mathbf{Y}\|^2$ s.t. $\|\mathbf{X}\mathbf{w_X}\|^2 = \|\mathbf{Y}\mathbf{w_Y}\|^2 = 1$
PLS	Maximize covariance	$\frac{\mathbf{w}_\mathbf{X}'\mathbf{S_{XY}}\mathbf{w_Y}}{\sqrt{\mathbf{w}_\mathbf{X}'\mathbf{w_X}}\sqrt{\mathbf{w}_\mathbf{Y}'\mathbf{w_Y}}}$
	Maximize fit	$\mathbf{w}_\mathbf{X}'\mathbf{S_{XY}}\mathbf{w_Y}$ s.t. $\|\mathbf{w_X}\|^2 = \|\mathbf{w_Y}\|^2 = 1$
	Minimize misfit	$\|\mathbf{w}_\mathbf{X}'\mathbf{X} - \mathbf{w}_\mathbf{Y}'\mathbf{Y}\|^2$ s.t. $\|\mathbf{w_X}\|^2 = \|\mathbf{w_Y}\|^2 = 1$
FDA	Maximize between to within class covariance	$\frac{\mathbf{w}'\mathbf{S}_B\mathbf{w}}{\mathbf{w}'\mathbf{S}_W\mathbf{w}}$
		$\mathbf{w}'\mathbf{S}_B\mathbf{w}$ s.t. $\mathbf{w}'\mathbf{S}_W\mathbf{w}$
NCut[103]	normalized cut-cost	$(\mathbf{1}+\mathbf{y})'\mathbf{A}(\mathbf{1}-\mathbf{y}) \cdot \left(\frac{1}{s_+} + \frac{1}{s_-}\right)$
ACut[34]	cut-cost	$(\mathbf{1}+\mathbf{y})'\mathbf{A}(\mathbf{1}-\mathbf{y}) \cdot \left(\frac{1}{n_+} + \frac{1}{n_-}\right)$

Table 2.5: Primal forms (not for the spectral clustering algorithms).

Algorithm	$\mathbf{M}$	$\mathbf{N}$	$\mathbf{v}$
PCA	$\mathbf{S_{XX}}$	$\mathbf{I}$	$\mathbf{w}$
RCCA	$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XY}} \\ \mathbf{S_{YX}} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{S_{XX}} + \gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{S_{YY}} + \gamma\mathbf{I} \end{pmatrix}$	$\begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}$
PLS	$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XY}} \\ \mathbf{S_{YX}} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$	$\begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}$
FDA	$\mathbf{S}_B$	$\mathbf{S}_W$	$\mathbf{w}$

Table 2.6: Dual forms.

	$\mathbf{M}$	$\mathbf{N}$	$\mathbf{v}$
PCA	$\mathbf{K}$	$\mathbf{I}$	$\boldsymbol{\alpha}$
RCCA	$\begin{pmatrix} \mathbf{0} & \mathbf{K_X}\mathbf{K_Y} \\ \mathbf{K_Y}\mathbf{K_X} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} {\mathbf{K_X}}^2 + \gamma\mathbf{K_X} & \mathbf{0} \\ \mathbf{0} & {\mathbf{K_Y}}^2 + \gamma\mathbf{K_Y} \end{pmatrix}$	$\begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix}$
PLS	$\begin{pmatrix} \mathbf{0} & \mathbf{K_X}\mathbf{K_Y} \\ \mathbf{K_Y}\mathbf{K_X} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$	$\begin{pmatrix} \boldsymbol{\alpha_X} \\ \boldsymbol{\alpha_Y} \end{pmatrix}$
FDA	$\frac{n_+ n_-}{n}\mathbf{K_X}\left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right) \cdot \left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right)' \mathbf{K_X}$	$\mathbf{K_X}\left(\mathbf{I} - \frac{\mathbf{y}_+\mathbf{y}_+'}{n_+} - \frac{\mathbf{y}_-\mathbf{y}_-'}{n_-}\right)\mathbf{K_X}$	$\boldsymbol{\alpha}$
SC[103]	$\mathbf{D} - \mathbf{A}$	$\mathbf{D}$	$\widetilde{\mathbf{y}}$
SC[34]	$\mathbf{D} - \mathbf{A}$	$\mathbf{I}$	$\widetilde{\mathbf{y}}$

here from SVD approaches naturally matches the duality derived by the Kuhn-Tucker Lagrangian theory developed for those methods, and the statistical study demonstrates similar properties as shown in [108] and [109].

Some properties of this class of algorithms are already appealing to machine learning practitioners, while others still need research attention. PLS for example, is designed precisely to operate with input data that are high dimensional and present highly correlated features, exactly the situation created by the use of kernel functions. The match between the two concepts is perfect, and in a way PLS can be better suited to the use of kernels than maximal-margin methodologies. Furthermore it is easily extendible towards multivariate regression. On the other hand, one of the major properties of support vector machines is not naturally present in eigen-algorithms: sparseness. Deliberate design choices can be made in order to enforce it, but the optimal way to include sparseness in this class of methods still remains an open question (see [50] for one way to achieve it). Another important point of research is the stability and statistical convergence of general eigenproblems for finite sample sizes. For work on the stability of the spectrum of Gram matrices, we refer to [99] and [100].

The synthesis offered by this unified view has immediate practical consequences, allowing for unified statistical analysis and for unified implementation strategies.

Chapter 3

Eigenvalue problems for semi-supervised learning

In the previous chapter, we have given an overview of some of the most important algorithms in machine learning that are based on eigenvalue problems. Each of these algorithms has proven its usefulness in numerous practical applications.

However, undeniably the success of these and other machine learning algorithms relies to a large extent on the availability of a *good representation* of the data, which is often the result of human design choices. More specifically, a 'suitable' distance measure between data items needs to be specified, so that a meaningful notion of 'similarity' is induced. The notion of 'suitable' is inevitably task dependent, since the same data might need very different representations for different learning tasks. This is clearly stated by the Ugly Duckling Theorem [36]: in the absence of assumptions, there is no privileged or 'best' feature representation; the notion of similarity between patterns thus implicitly depends on the assumptions.

This means that automating the task of choosing a representation will necessarily require some type of information (e.g. some of the labels, or less refined forms of information about the task at hand). Labels may be too expensive, while a less refined and more readily available source of information can be used (known as side-information). For example, one may want to define a metric over the space of movies descriptions, using data about customers associations (such as sets of movies liked by the same customer in [52]) as side-information. This type of side-information is commonplace in marketing data, image segmentation, recommendation systems, bioinformatics and web data.

Two approaches to semi-supervised learning Many recent papers have dealt with these and related problems; some by imposing hard or soft constraints to an existing algorithm without actually learning a metric, as in [115, 27, 129,

62, 64, 13, 101, 121]; others by implicitly learning a metric, like [52], [116] or explicitly by [102, 126].

Notably [126] provide a conceptually elegant algorithm for learning the metric based on semi-definite programming (SDP). Unfortunately, the algorithm has complexity $O(d^6)$ for d-dimensional data[1]. Therefore, there is a need for more scalable algorithms such as the one we will propose in the *first section of this chapter*.

In the *second section of this chapter*, we will describe a method of the first category: it does not learn a metric, however, an existing algorithm is adapted to take general label constraints into account.

3.1 Side-information for dimensionality reduction

In this section we present an algorithm for the problem of finding a suitable metric, using side-information that consists of n example pairs $(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}), i = 1, \ldots, n_p$ belonging to the *same but unknown* class. Furthermore, we place our algorithm in a general framework, in which also the methods described in [118] and [116] would fit. More specifically, we show how these methods can all be related to Linear Discriminant Analysis (LDA, see [40] or [36]), a method for dimensionality reduction looking for a subspace in which the projected clusters are maximally separated in some sense (measured in the Euclidian metric in the subspace).

For reference, we will first give a brief review of LDA. Next we show how our method can be derived as an approximation for LDA in case only side-information is available. Furthermore, we provide a derivation similar to the one in [126] in order to show the correspondence between the two approaches. Empirical results include a toy example, and UCI data sets also used in [126].

Specific notation To denote the side-information that consists of n_p pairs $(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})$ for which is known that $\mathbf{x}_i^{(1)}$ and $\mathbf{x}_i^{(2)} \in \mathbb{R}^d$ belong to the same class, we will use the matrices $\mathbf{X}^{(1)} \in \mathbb{R}^{n_p \times d}$ and $\mathbf{X}^{(2)} \in \mathbb{R}^{n_p \times d}$. These contain $\mathbf{x}_i^{(1)\prime}$ and $\mathbf{x}_i^{(2)\prime}$ as their ith rows: $\mathbf{X}^{(1)} = \begin{pmatrix} \mathbf{x}_1^{(1)\prime} \\ \mathbf{x}_2^{(1)\prime} \\ \cdots \\ \mathbf{x}_{n_p}^{(1)\prime} \end{pmatrix}$ and $\mathbf{X}^{(2)} = \begin{pmatrix} \mathbf{x}_1^{(2)\prime} \\ \mathbf{x}_2^{(2)\prime} \\ \cdots \\ \mathbf{x}_{n_p}^{(2)\prime} \end{pmatrix}$. This means that for any $i = 1, \ldots, n_p$, it is known that the samples at the ith rows of $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ belong to the same class. For ease of notation (but

[1]The authors of [126] see this problem, and they try to circumvent it by developing a gradient descent algorithm instead of using standard Newton algorithms for solving SDP problems. However, this may cause prohibitively slow convergence, especially for data sets in large dimensional spaces.

without loss of generality) we will construct the full data matrix[2] $\mathbf{X} \in \mathbb{R}^{2n_p \times d}$ as $\mathbf{X} = \begin{pmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{pmatrix}$. When we want to denote the sample corresponding to the ith row of $\mathbf{X}$ without regarding the side-information, it is denoted as $\mathbf{x}_i \in \mathbb{R}^d$ (without superscript, and $i = 1, \ldots, 2n_p$). The data matrix should be *centered*, that is $\mathbf{1}'\mathbf{X} = \mathbf{0}$ (the mean of each column is zero). We use $\mathbf{w} \in \mathbb{R}^d$ to denote a weight vector in this d-dimensional data space.

Although the labels for the samples are *not* known in our problem setting, we will consider the label matrix $\mathbf{Z} \in \mathbb{R}^{2n_p \times c}$ corresponding to $\mathbf{X}$ in our derivations. (The number of classes is denoted by c.) It is defined as (where $\widetilde{\mathbf{Z}}_{i,j}$ indicates the element at row i and column j):

$$\widetilde{\mathbf{Z}}_{i,j} = \begin{cases} 1 & \text{when the class of the sample } \mathbf{x}_i \text{ is } j \\ 0 & \text{otherwise,} \end{cases}$$

followed by a centering to make all column sums equal to zero: $\mathbf{Z} = \widetilde{\mathbf{Z}} - \frac{\mathbf{1}\mathbf{1}'}{n}\widetilde{\mathbf{Z}}$. We use $\mathbf{w}_\mathbf{Z} \in \mathbb{R}^c$ to denote a weight vector in the c-dimensional label space.

The matrices $\mathbf{C}_{\mathbf{ZX}} = \mathbf{C}'_{\mathbf{XZ}} = \mathbf{Z}'\mathbf{X}, \mathbf{C}_{\mathbf{ZZ}} = \mathbf{Z}'\mathbf{Z}, \mathbf{C}_{\mathbf{XX}} = \mathbf{X}'\mathbf{X}$ are called *total scatter matrices* of $\mathbf{X}$ or $\mathbf{Z}$ with $\mathbf{X}$ or $\mathbf{Z}$. The total scatter matrices for the subset data matrices $\mathbf{X}^{(k)}$, $k = 1, 2$, are indexed by integers: $\mathbf{C}_{kl} = {\mathbf{X}^{(k)}}'\mathbf{X}^{(l)}$.

Again if the labels were known, we could identify the sample sets $\mathcal{C}_i = \{\text{all } \mathbf{x}_j \text{ in class } i\}$. Then we could also compute the following quantities for the samples in $\mathbf{X}$: the number of samples in each class: $n_i = |\mathcal{C}_i|$; the class means $\mathbf{m}_i = \frac{1}{n_i}\sum_{j:\mathbf{x}_j \in \mathcal{C}_i} \mathbf{x}_j$; the between class scatter matrix $\mathbf{C}_B = \sum_{i=1}^{c} n_i \mathbf{m}_i \mathbf{m}_i'$. The within class scatter matrix $\mathbf{C}_W = \sum_{i=1}^{c} \sum_{j:\mathbf{x}_j \in \mathcal{C}_i} (\mathbf{x}_j - \mathbf{m}_i)(\mathbf{x}_j - \mathbf{m}_i)'$. Since the labels are not known in our problem setting, we will only use these quantities in our derivations, not in our final results.

3.1.1 Learning the Metric

In this subsection (which is published in [18]), we will show how the LDA formulation that requires labels, can be adapted for cases where no labels but only side-information is available. The resulting formulation can be seen as an approximation of LDA with labels available. This will lead to an efficient algorithm to learn a metric: given the side-information, solving just a generalized eigenproblem is sufficient to identify a subspace in which the projections of the data clusters are maximally separated. Hence the Euclidian metric in this lower dimensional projection is more suitable for use in standard clustering algorithms than the original metric in the original data space.

[2] In all derivations, the only data samples involved are the ones that appear in the side-information. It is not until the empirical results subsection that also data not involved in the side-information is dealt with: the side-information is used to learn a metric, corresponding to the Euclidian distance in a subspace in which the original data is projected. Only subsequently, this metric is used to cluster any other available sample. We also assume no sample appears twice in the side-information.

Motivation

Canonical Correlation Analysis (CCA) formulation of Linear Discriminant Analysis (LDA) for classification When given a data matrix $\mathbf{X}$ and a label matrix $\mathbf{Z}$, LDA [40] provides a way to find a projection of the data that has the largest between class variance to within class variance ratio. This can be formulated as a maximization problem of the Rayleigh quotient $\rho(\mathbf{w}) = \frac{\mathbf{w}'\mathbf{C}_B\mathbf{w}}{\mathbf{w}'\mathbf{C}_W\mathbf{w}}$. In the optimum $\nabla_{\mathbf{w}}\rho = \mathbf{0}$, $\mathbf{w}$ is the eigenvector corresponding to the largest eigenvalue of the generalized eigenvalue problem $\mathbf{C}_B\mathbf{w} = \rho\ \mathbf{C}_W\mathbf{w}$. Furthermore, it is shown that LDA can also be computed by performing CCA between the data and the label matrix [10, 42, 9, 92]. In other words, LDA maximizes the correlation between a projection of the coordinates of the data points and a projection of their class labels. This means the following CCA generalized eigenvalue problem formulation can be used:

$$\begin{pmatrix} \mathbf{0} & \mathbf{C_{XZ}} \\ \mathbf{C_{ZX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ \mathbf{w_Z} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{C_{XX}} & \mathbf{0} \\ \mathbf{0} & \mathbf{C_{ZZ}} \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ \mathbf{w_Z} \end{pmatrix}.$$

The optimization problem corresponding to CCA is (as shown in e.g. [24]):

$$\max_{\mathbf{w},\mathbf{w_Z}} \mathbf{w}'\mathbf{X}'\mathbf{Z}\mathbf{w_Z} \quad \text{s.t.} \quad \|\mathbf{X}\mathbf{w}\|^2 = 1 \text{ and } \|\mathbf{Z}\mathbf{w_Z}\|^2 = 1. \tag{3.1}$$

This formulation for LDA is the starting point for our derivations.

Maximizing the *expected* LDA cost function

Parameterization As explained before, the side-information is such that we get pairs of samples $(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})$ that have the same class label. Using this side-information we stack the corresponding vectors $\mathbf{x}_i^{(1)}$ and $\mathbf{x}_i^{(2)}$ at the same row in their respective matrices

$$\mathbf{X}^{(1)} = \begin{pmatrix} \mathbf{x}_1^{(1)\prime} \\ \mathbf{x}_2^{(1)\prime} \\ \cdots \\ \mathbf{x}_{n_p}^{(1)\prime} \end{pmatrix} \quad \text{and} \quad \mathbf{X}^{(2)} = \begin{pmatrix} \mathbf{x}_1^{(2)\prime} \\ \mathbf{x}_2^{(2)\prime} \\ \cdots \\ \mathbf{x}_{n_p}^{(2)\prime} \end{pmatrix}.$$

The full matrix containing all samples for which side-information is available, is then equal to $\mathbf{X} = \begin{pmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{pmatrix}$. Now, since we know each row of $\mathbf{X}^{(1)}$ has the same label as the corresponding row of $\mathbf{X}^{(2)}$, a parameterization of the label matrix $\mathbf{Z}$ is easily found to be $\mathbf{Z} = \begin{pmatrix} \mathbf{L} \\ \mathbf{L} \end{pmatrix}$. Note that $\mathbf{Z}$ is centered iff $\mathbf{L}$ is centered. The matrix $\mathbf{L}$ is in fact just the label matrix of both $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ on themselves. (We want to stress $\mathbf{L}$ is not known, but used in the equations as an unknown matrix parameter for now.)

The Rayleigh quotient cost function that incorporates the side-information Using this parameterization we apply LDA on the matrix $\begin{pmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{pmatrix}$ with the label matrix $\begin{pmatrix} \mathbf{L} \\ \mathbf{L} \end{pmatrix}$ to find the optimal directions for separation of the classes. For this we use the CCA formulation of LDA. This means we want to solve the CCA optimization problem (3.1) where we substitute the values for $\mathbf{Z}$ and $\mathbf{X}$:

$$\max_{\mathbf{w},\mathbf{w_Z}} \quad \mathbf{w}' \begin{pmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{pmatrix}' \begin{pmatrix} \mathbf{L} \\ \mathbf{L} \end{pmatrix} \mathbf{w_Z} = \max_{\mathbf{w},\mathbf{w_Z}} \mathbf{w}'{\mathbf{X}^{(1)}}'\mathbf{L}\mathbf{w_Z} + \mathbf{w}'{\mathbf{X}^{(2)}}'\mathbf{L}\mathbf{w_Z} \tag{3.2}$$

$$\text{s.t.} \quad \left\| \begin{pmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{pmatrix} \mathbf{w} \right\|^2 = \|\mathbf{X}^{(1)}\mathbf{w}\|^2 + \|\mathbf{X}^{(2)}\mathbf{w}\|^2 = 1 \tag{3.3}$$

$$\|\mathbf{L}\mathbf{w_Z}\|^2 = 1$$

The Lagrangian of this constrained optimization problem is:

$$\mathcal{L} = \mathbf{w}'{\mathbf{X}^{(1)}}'\mathbf{L}\mathbf{w_Z} + \mathbf{w}'{\mathbf{X}^{(2)}}'\mathbf{L}\mathbf{w_Z} - \widehat{\lambda}\mathbf{w}'({\mathbf{X}^{(1)}}'\mathbf{X}^{(1)} + {\mathbf{X}^{(2)}}'\mathbf{X}^{(2)})\mathbf{w} - \mu\mathbf{w}_\mathbf{Z}'\mathbf{L}'\mathbf{L}\mathbf{w_Z}$$

Differentiating with respect to $\mathbf{w_Z}$ and $\mathbf{w}$ and equating to $\mathbf{0}$ yields

$$\nabla_{\mathbf{w_Z}}\mathcal{L} = 0 \quad \Rightarrow \quad \mathbf{L}'(\mathbf{X}^{(1)} + \mathbf{X}^{(2)})\mathbf{w} = 2\mu\mathbf{L}'\mathbf{L}\mathbf{w_Z} \tag{3.4}$$

$$\nabla_{\mathbf{w}}\mathcal{L} = 0 \quad \Rightarrow \quad (\mathbf{X}^{(1)} + \mathbf{X}^{(2)})'\mathbf{L}\mathbf{w_Z} = 2\widehat{\lambda}({\mathbf{X}^{(1)}}'\mathbf{X}^{(1)} + {\mathbf{X}^{(2)}}'\mathbf{X}^{(2)})\mathbf{w} \tag{3.5}$$

From (3.4) we find that $\mathbf{w_Z} = \frac{1}{2\mu}(\mathbf{L}'\mathbf{L})^\dagger\mathbf{L}'(\mathbf{X}^{(1)} + \mathbf{X}^{(2)})\mathbf{w}$. Filling this into equation (3.5) and choosing $\widetilde{\lambda} = 4\widehat{\lambda}\mu$ gives that

$$(\mathbf{X}^{(1)} + \mathbf{X}^{(2)})' \left[\mathbf{L}(\mathbf{L}'\mathbf{L})^\dagger\mathbf{L}'\right] (\mathbf{X}^{(1)} + \mathbf{X}^{(2)})\mathbf{w} = \widetilde{\lambda}({\mathbf{X}^{(1)}}'\mathbf{X}^{(1)} + {\mathbf{X}^{(2)}}'\mathbf{X}^{(2)})\mathbf{w}.$$

It is well known that solving for the dominant generalized eigenvector is equivalent to maximizing the Rayleigh quotient:

$$\frac{\mathbf{w}'(\mathbf{X}^{(1)} + \mathbf{X}^{(2)})' \left[\mathbf{L}(\mathbf{L}'\mathbf{L})^\dagger\mathbf{L}'\right] (\mathbf{X}^{(1)} + \mathbf{X}^{(2)})\mathbf{w}}{\mathbf{w}'({\mathbf{X}^{(1)}}'\mathbf{X}^{(1)} + {\mathbf{X}^{(2)}}'\mathbf{X}^{(2)})\mathbf{w}}. \tag{3.6}$$

Until now, for the given side-information, *there is still an exact equivalence between LDA and maximizing this Rayleigh quotient.* The important difference between the standard LDA cost function and (3.6) however, is that in the latter the side-information is imposed explicitly by using the reduced parameterization for $\mathbf{Z}$ in terms of $\mathbf{L}$.

The expected cost function As pointed out, we do not know the term between $[\cdot]$. What we will do then is compute the expected value of the cost function (3.6) by averaging over all possible label matrices $\mathbf{Z} = \begin{pmatrix} \mathbf{L} \\ \mathbf{L} \end{pmatrix}$, possibly

weighted with any symmetric[3] a priori probability for the label matrices. Since the only part that depends on the label matrix is the factor between $[\cdot]$, and since it appears linearly in the cost function, we just need to compute the expectation of this factor. This expectation is proportional to $\mathbf{I} - \frac{\mathbf{1}\mathbf{1}'}{n}$. To see this we only have to use symmetry arguments (all values on the diagonal should be equal to each other, and all other values should be equal to each other), and the observation that $\mathbf{L}$ is centered and thus $\left[\mathbf{L}(\mathbf{L}'\mathbf{L})^{\dagger}\mathbf{L}'\right]\mathbf{1} = \mathbf{0}$. Now, since we assume that the data matrix $\mathbf{X}$ containing the samples in the side-information is centered too, $(\mathbf{X}^{(1)} + \mathbf{X}^{(2)})'\frac{\mathbf{1}\mathbf{1}'}{n}(\mathbf{X}^{(1)} + \mathbf{X}^{(2)})$ is equal to the null matrix. Thus the expected value of $(\mathbf{X}^{(1)} + \mathbf{X}^{(2)})'\left[\mathbf{L}(\mathbf{L}'\mathbf{L})^{\dagger}\mathbf{L}'\right](\mathbf{X}^{(1)} + \mathbf{X}^{(2)})$ is proportional to $(\mathbf{X}^{(1)} + \mathbf{X}^{(2)})'(\mathbf{X}^{(1)} + \mathbf{X}^{(2)})$. The expected value of the LDA cost function in equation (3.6), where the expectation is taken over all possible label assignments $\mathbf{Z}$ constrained by the side-information, is then shown to be

$$\frac{\mathbf{w}'(\mathbf{C}_{11} + \mathbf{C}_{12} + \mathbf{C}_{22} + \mathbf{C}_{21})\mathbf{w}}{\mathbf{w}'(\mathbf{C}_{11} + \mathbf{C}_{22})\mathbf{w}} = 1 + \frac{\mathbf{w}'(\mathbf{C}_{12} + \mathbf{C}_{21})\mathbf{w}}{\mathbf{w}'(\mathbf{C}_{11} + \mathbf{C}_{22})\mathbf{w}}.$$

The vector $\mathbf{w}$ maximizing this cost is the dominant generalized eigenvector of

$$(\mathbf{C}_{12} + \mathbf{C}_{21})\mathbf{w} = \lambda(\mathbf{C}_{11} + \mathbf{C}_{22})\mathbf{w} \tag{3.7}$$

where $\mathbf{C}_{kl} = \mathbf{X}^{(k)'}\mathbf{X}^{(l)}$.

(Note that the side-information is symmetric in the sense that one could replace an example pair $(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})$ with $(\mathbf{x}_i^{(2)}, \mathbf{x}_i^{(1)})$ without losing any information. However, this operation does not change $\mathbf{C}_{12} + \mathbf{C}_{21}$ nor $\mathbf{C}_{11} + \mathbf{C}_{22}$, so that the eigenvalue problem to be solved does not change either, which is of course a desirable property.)

In subsection 3.1.4 we provide an alternative derivation leading to the same eigenvalue problem (Eq. (3.7)). This derivation is based on a cost function that is close to the cost function used in [126].

Interpretation and Dimensionality Selection

Interpretation Given the eigenvector $\mathbf{w}$, the corresponding eigenvalue λ is equal to $\frac{\mathbf{w}'(\mathbf{C}_{12}+\mathbf{C}_{21})\mathbf{w}}{\mathbf{w}'(\mathbf{C}_{11}+\mathbf{C}_{22})\mathbf{w}}$. The numerator $\mathbf{w}'(\mathbf{C}_{12} + \mathbf{C}_{21})\mathbf{w}$ is twice the covariance of the projections $\mathbf{X}^{(1)}\mathbf{w}$ with $\mathbf{X}^{(2)}\mathbf{w}$ (up to a factor equal to the number of samples in $\mathbf{X}^{(k)}$). The denominator normalizes with the sum of their variances (up to the same factor). This means λ is very close to the correlation between

[3] That is, the a priori probability of a label assignment $\mathbf{L}$ is the same as the probability of the label assignment $\mathbf{PL}$ where $\mathbf{P}$ can be any permutation matrix. Remember every row of $\mathbf{L}$ corresponds to the label of a pair of points in the side-information. Thus, this invariance means we have no discriminating prior information on which pair belongs to which of the classes. Using this ignorant prior is clearly the most reasonable we can do, since we assume only the side-information is given here.

$\mathbf{X}^{(1)}\mathbf{w}$ and $\mathbf{X}^{(2)}\mathbf{w}$ (it becomes equal to their correlation when the variances of $\mathbf{X}^{(1)}\mathbf{w}$ and $\mathbf{X}^{(2)}\mathbf{w}$ are equal, which will often be close to true as both $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ are drawn from the same distribution). This makes sense: we want $\mathbf{X}\mathbf{w} = \begin{pmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{pmatrix} \mathbf{w}$ and thus both $\mathbf{X}^{(1)}\mathbf{w}$ and $\mathbf{X}^{(2)}\mathbf{w}$ to be strongly correlated with a projection $\mathbf{Z}\mathbf{w}_\mathbf{Z}$ of their (common but unknown) labels in $\mathbf{Z}$ on $\mathbf{w}_\mathbf{Z}$ (see equation (3.1); this is what we actually wanted to optimize, but could not do exactly since $\mathbf{Z}$ is not known). Now, when we want $\mathbf{X}^{(1)}\mathbf{w}$ and $\mathbf{X}^{(2)}\mathbf{w}$ to be strongly correlated with the *same* labels, they necessarily have to be strongly correlated with each other.

Some of the eigenvalues may be negative however. This means that along these eigenvectors, samples that should be co-clustered according to the side-information are *anti*-correlated. This can only be caused by features in the data that are irrelevant for the clustering problem at hand (which can be seen as noise).

Dimensionality selection As with LDA, one will generally not only use the dominant eigenvector, but a dominant eigen*space* to project the data on. The number of eigenvectors used should depend on the signal to noise ratio along these components: when it is too low, noise effects will cause poor performance of a subsequent clustering. So we need to make an estimate of the noise level.

This is provided by the negative eigenvalues: they allow us to make a good estimate of the noise level present in the data, thus motivating the strategy adopted here: only retain the k directions corresponding to eigenvalues larger than the largest absolute value of the negative eigenvalues.

The Metric Corresponding to the Subspace Used

Since we will project the data onto the k dominant eigenvectors $\mathbf{w}$, this finally boils down to using the distance measure

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{W}'(\mathbf{x}_i - \mathbf{x}_j))' (\mathbf{W}'(\mathbf{x}_i - \mathbf{x}_j)) = \|\mathbf{x}_i - \mathbf{x}_j\|^2_{\mathbf{W}\mathbf{W}'}.$$

where $\mathbf{W}$ is the matrix containing the k eigenvectors as its columns.

Normalization of the different eigenvectors could be done so as to make the variance equal to 1 along each of the directions. However as can be understood from the interpretation in 3.1.1, along directions with a high eigenvalue λ a better separation can be expected. Therefore, we applied the heuristic to scale each of the eigenvectors by multiplying them with their corresponding eigenvalue. In doing that, a subsequent clustering like K-means will preferentially find cluster separations orthogonal to directions that will probably separate well (which is desirable).

Computational Complexity

Operations to be carried out in this algorithm are the computation of the $d \times d$ scatter matrices, and solving a symmetric generalized eigenvalue problem of

size d. The computational complexity of this problem is thus $O(d^3)$. Since the approach in [126] is basically an SDP with d^2 parameters, its complexity is $O(d^6)$. Thus a massive speedup can be achieved.

3.1.2 Remarks

Relation with Existing Literature

Actually, $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ do not have to belong to the same space, they can be of a different kind: it is sufficient when corresponding samples in $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ belong to the same class to do something similar as above. Of course then we need different weight vectors in both spaces: $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$.

If we replace $\mathbf{w}$ in optimization problem (3.2) subject to (3.3) once by $\mathbf{w}^{(1)}$ and one by $\mathbf{w}^{(2)}$:

$$\begin{aligned} & \max_{\mathbf{w}^{(1)},\mathbf{w}^{(2)}} \mathbf{w}^{(1)\prime}\mathbf{X}^{(1)\prime}\mathbf{L}\mathbf{w}_\mathbf{Z} + \mathbf{w}^{(2)\prime}\mathbf{X}^{(2)\prime}\mathbf{L}\mathbf{w}_\mathbf{Z} \\ \text{s.t.} \quad & \|\mathbf{X}^{(1)}\mathbf{w}^{(1)}\|^2 + \|\mathbf{X}^{(2)}\mathbf{w}^{(2)}\|^2 = 1 \\ & \|\mathbf{L}\mathbf{w}_\mathbf{Z}\|^2 = 1 \end{aligned}$$

where $\mathbf{L}$ corresponds to the common label matrix for $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ (both centered). In a similar way as previous derivation, this can be shown to amount to solving the eigenvalue problem:

$$\begin{pmatrix} \mathbf{0} & \mathbf{X}^{(1)\prime}\left[\mathbf{L}(\mathbf{L}'\mathbf{L})^{-1}\mathbf{L}'\right]\mathbf{X}^{(2)} \\ \mathbf{X}^{(2)\prime}\left[\mathbf{L}(\mathbf{L}'\mathbf{L})^{-1}\mathbf{L}'\right]\mathbf{X}^{(1)} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w}^{(1)} \\ \mathbf{w}^{(2)} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{C}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{w}^{(1)} \\ \mathbf{w}^{(2)} \end{pmatrix}$$

which again corresponds to a Rayleigh quotient. Since also here in fact we do not know the matrix $\mathbf{L}$, we again take the expected value (as above). This leads to an expected Rayleigh quotient that is maximized by solving the generalized eigenproblem corresponding to CCA:

$$\begin{pmatrix} \mathbf{0} & \mathbf{C}_{12} \\ \mathbf{C}_{21} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w}^{(1)} \\ \mathbf{w}^{(2)} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{C}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{w}^{(1)} \\ \mathbf{w}^{(2)} \end{pmatrix}.$$

This is exactly what is done in [118] and [116] (in both papers in a kernel induced feature space).

More General Types of Side-Information

Using similar approaches, also general types of side-information may be utilized. We will only briefly mention them:

- When the groups of samples for which is known they belong to the same class is larger than 2 (let us call them $\mathbf{X}^{(i)}$ again, but now i is not restricted

to only 1 or 2). This can be handled very analogously to our previous derivation. Therefore we just state the resulting generalized eigenvalue problem:

$$\left(\sum_k \mathbf{X}^{(k)\prime} \sum_k \mathbf{X}^{(k)}\right) \mathbf{w} = \lambda \left(\sum_k \mathbf{X}^{(k)\prime} \mathbf{X}^{(k)}\right) \mathbf{w}.$$

- Also in case we are dealing with more than 2 data sets that are of a different nature (eg analogous to [118]: we could have more than 2 data sets, each consisting of a text corpus in a different language), but for which is known that corresponding samples belong to the same class (as described in the previous subsubsection), the problem is easily shown to reduce to the extension of CCA towards more data spaces, as is e.g. used in [6]. Space restrictions do not permit us to go into this.
- It is possible to keep this approach completely general, allowing for any type of side-information of the form of constraints that express for any number of samples they belong to the same class, or on the contrary do not to belong to the same class. Also knowledge of some of the labels can be exploited. For doing this, we have to use a different parameterization for $\mathbf{Z}$ than the one proposed in this chapter. In principle also any prior distribution on the parameters can be taken into account. However, sampling techniques will be necessary to estimate the expected value of the LDA cost function in these cases. We will not go into this in this thesis.

The Dual Eigenproblem

As a last remark, the dual or *kernelized* version of the generalized eigenvalue problem can be derived as follows. The solution $\mathbf{w}$ can be expressed in the form $\mathbf{w} = \begin{pmatrix} \mathbf{X}^{(1)\prime} & \mathbf{X}^{(2)\prime} \end{pmatrix} \boldsymbol{\alpha}$ where $\boldsymbol{\alpha} \in \mathbb{R}^{2n_p}$ is a vector containing the dual variables. Now, with Gram matrices $\mathbf{K}_{kl} = \mathbf{X}^{(k)} \mathbf{X}^{(l)\prime}$, and after introducing the notation

$$\mathbf{G}_1 = \begin{pmatrix} \mathbf{K}_{11} \\ \mathbf{K}_{21} \end{pmatrix} \quad \text{and} \quad \mathbf{G}_2 = \begin{pmatrix} \mathbf{K}_{12} \\ \mathbf{K}_{22} \end{pmatrix}$$

the $\boldsymbol{\alpha}$'s corresponding to the weight vectors $\mathbf{w}$ are found as the generalized eigenvectors of

$$(\mathbf{G}_1 \mathbf{G}_2' + \mathbf{G}_2 \mathbf{G}_1') \boldsymbol{\alpha} = \lambda (\mathbf{G}_1 \mathbf{G}_1' + \mathbf{G}_2 \mathbf{G}_2') \boldsymbol{\alpha}.$$

This motivates that it will be possible to extend the approach to learning non-linear metrics with side-information as well.

3.1.3 Empirical Results

The empirical results reported here will be for clustering problems with the type of side-information described above. Thus, with our method we learn a suitable

metric based on a set of samples for which the side-information is known, i.e. $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$. Subsequently a K-means clustering of *all* samples (including those that are not in $\mathbf{X}^{(1)}$ or $\mathbf{X}^{(2)}$) is performed, making use of the metric that is learned.

Evaluation of Clustering Performance

We use the same measure of accuracy as is used in [126], namely, defining $I(\mathbf{x}_i, \mathbf{x}_j)$ as the function being 1 when $\mathbf{x}_i$ and $\mathbf{x}_j$ are clustered in the same cluster by the algorithm,

$$\text{Acc} = \frac{\sum_k \sum_{i,j>i;\mathbf{x}_i,\mathbf{x}_j \in \mathcal{C}_k} I(\mathbf{x}_i, \mathbf{x}_j)}{2 \sum_k \sum_{i,j>i;\mathbf{x}_i,\mathbf{x}_j \in \mathcal{C}_k} 1} + \frac{\sum_{i,j>i;\neg\exists k:\mathbf{x}_i,\mathbf{x}_j \in \mathcal{C}_k} (1 - I(\mathbf{x}_i, \mathbf{x}_j))}{2 \sum_{i,j>i;\neg\exists k:\mathbf{x}_i,\mathbf{x}_j \in \mathcal{C}_k} 1}.$$

Regularization

To deal with inaccuracies, numerical instabilities and influences of finite sample size, we apply regularization to the generalized eigenvalue problem. This is done in the same spirit as for CCA in [6], namely by adding a diagonal to the scatter matrices $\mathbf{C}_{11}$ and $\mathbf{C}_{22}$. This can be justified by the CCA-based derivation of our algorithm. To train the regularization parameter, a cost function described below is minimized via 10-fold cross validation.

In choosing the right regularization parameter, there are two things to consider: firstly, we want the clustering to be 'good'. This means that the side-information should be reflected as well as possible by the clustering. Secondly we want this clustering to be informative. This means, we don't want one very large cluster, the others being very small (the probability to fulfil the side-information would be too easy then). Therefore, the cross-validation cost minimized here, is the probability for the measured performance on the test set side-information, given the sizes of the clusters found. (More exactly, we maximized the difference of this performance with its expected performance, divided by its standard deviation.) This approach incorporates both considerations in a natural way.

Performance on a Toy Data Set

The effectiveness of the method is illustrated by using a toy example, in which each of the clusters consists of two parts lying far apart (figure 3.1). Standard K-means has an accuracy of 0.50 on this data set, while the method developed here gives an accuracy of 0.92.

Performance on some UCI Data Sets

The empirical results on some UCI data sets, reported in table 3.1, are comparable to the results in [126]. The first column contains the K-means clustering accuracy without any side-information and preprocessing, averaged over 30 different initial conditions. In the second column, results are given for 'small'

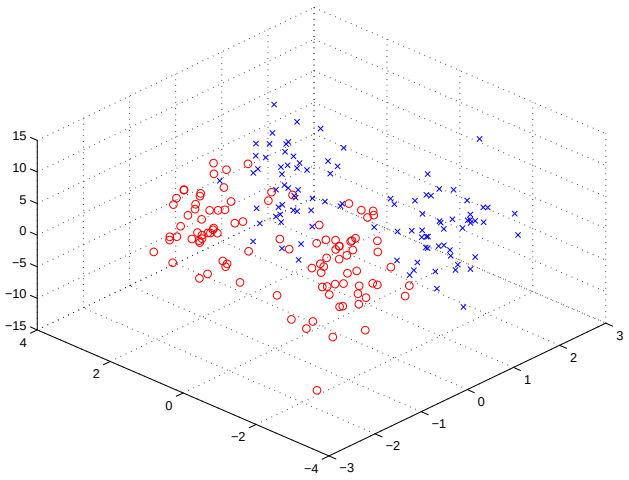


Figure 3.1: A toy example whereby the two clusters each consist of two distinct clouds of samples, that are widely separated. Ordinary K-means obviously has a very low accuracy of 0.5, whereas when some side-information is taken into account as described chapter, the performance goes up to 0.92.

side-information leaving 90 percent of the connected components[4], in the third column for 'large' side-information leaving 70 percent of the connected components. For these two columns, averages over 30 randomizations of the side-information are shown. The side-information is generated by randomly picking pairs of samples belonging to the same cluster. The number between brackets indicates the standard deviation over these 30 randomizations.

Table 3.2 contains the accuracy on the UCI wine data set and on the protein data set, for different amounts of side-information. To quantize the amount of side-information, we used (as in [126]) the number of pairs in the side-information, divided by the total number of pairs of samples belonging to the same class (the *ratio of constraints*.)

These results are comparable with those reported in [126]. Like in [126], *constrained K-means* [27], a variant of K-means that is able to take label constraints into account, will allow for a further improvement. (It is important to note that constrained K-means on itself does not learn a metric, it still uses the standard Euclidian metric; the side-information is not used for learning which directions in the data space are important in the clustering process.)

[4]We use the notion *connected component* as defined in [126]. That is, for given side-information, a set of samples makes up one connected component, if between each pair of samples in this set, there exists a path via edges corresponding to pairs given in the side-information. For no side-information given, the number of connected components is thus equal to the total number of samples.

Table 3.1: Accuracies for on UCI data sets, for different numbers of connected components. (The more side-information, the less connected components. The fraction f is the number of connected components divided by the total number of samples.) Note that there is no actual distinction between training and test set in side-information learning; therefore, the accuracy is always computed on the entire sample.

DATA SET	$f = 1$	$f = 0.9$	$f = 0.7$
WINE	0.69 (0.00)	0.92 (0.05)	0.95 (0.03)
PROTEIN	0.62 (0.02)	0.71 (0.04)	0.72 (0.06)
IONOSPHERE	0.58 (0.02)	0.69 (0.09)	0.75 (0.05)
DIABETES	0.56 (0.02)	0.60 (0.02)	0.61 (0.02)
BALANCE	0.56 (0.02)	0.66 (0.01)	0.67 (0.03)
IRIS	0.83 (0.06)	0.92 (0.03)	0.92 (0.04)
SOY	0.80 (0.08)	0.85 (0.09)	0.91 (0.10)
BREAST CANCER	0.83 (0.01)	0.89 (0.02)	0.91 (0.02)

3.1.4 Alternative Derivation

More in the spirit of [126], we can derive the algorithm by solving the constrained optimization problem (where $\dim(\mathbf{W}) = k$ means that the dimensionality of $\mathbf{W}$ is k, that is, $\mathbf{W}$ has k columns):

$$\begin{aligned} \max_{\mathbf{W}} \quad & \operatorname{trace}({\mathbf{X}^{(1)}}'\mathbf{W}\mathbf{W}'\mathbf{X}^{(2)}) \\ \text{s.t.} \quad & \dim(\mathbf{W}) = k \\ & \mathbf{W}' \begin{pmatrix} \mathbf{X}^{(1)} & \mathbf{X}^{(2)} \end{pmatrix} \begin{pmatrix} {\mathbf{X}^{(1)}}' \\ {\mathbf{X}^{(2)}}' \end{pmatrix} \mathbf{W} = \mathbf{I}_k \end{aligned}$$

so as to find a subspace of dimension k that optimizes the correlation between samples belonging to the same class.

This can be reformulated as

$$\begin{aligned} \max_{\mathbf{W}} \quad & \operatorname{trace}(\mathbf{W}'(\mathbf{C}_{12} + \mathbf{C}_{21})\mathbf{W}) \\ \text{s.t.} \quad & \dim(\mathbf{W}) = k \\ & \mathbf{W}'(\mathbf{C}_{11} + \mathbf{C}_{22})\mathbf{W} = \mathbf{I}_k \end{aligned}$$

Solving this optimization problem amounts to solving for the eigenvectors corresponding to the k largest eigenvalues of the generalized eigenvalue problem described above (3.7).

The proof involves the following theorem by Ky Fan (see e.g. [55]):

Theorem 3.1 *Let* $\mathbf{H}$ *be a symmetric matrix with eigenvalues* $\lambda_1 > \lambda_2 > \ldots >$

Table 3.2: Accuracies on the wine and the protein data sets, as a function of the ratio of constraints.

RATIO OF CONSTR.	ACCURACY FOR WINE	RATIO OF CONSTR.	ACCURACY FOR PROTEIN
0	0.69 (0.00)	0	0.62 (0.03)
0.0015	0.73 (0.08)	0.012	0.59 (0.04)
0.0023	0.78 (0.11)	0.019	0.60 (0.05)
0.0034	0.87 (0.08)	0.028	0.62 (0.04)
0.0051	0.91 (0.05)	0.041	0.67 (0.05)
0.0075	0.93 (0.05)	0.060	0.71 (0.05)
0.011	0.96 (0.05)	0.099	0.75 (0.05)
0.017	0.97 (0.017)	0.14	0.77 (0.05)
0.025	0.97 (0.018)	0.21	0.79 (0.06)
0.037	0.98 (0.015)	0.31	0.78 (0.07)

λ_n, and the corresponding eigenvectors $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n)$. Then

$$\lambda_1 + \ldots + \lambda_k = \max_{\mathbf{P}'\mathbf{P}=\mathbf{I}} \text{trace}(\mathbf{P}'\mathbf{H}\mathbf{P}).$$

Moreover, the optimal $\mathbf{P}^*$ is given by $\mathbf{P}^* = (\mathbf{u}_1, \ldots, \mathbf{u}_k)\mathbf{Q}$ where $\mathbf{Q}$ is an arbitrary orthogonal matrix.

Since $(\mathbf{C}_{11} + \mathbf{C}_{22})$ is positive definite, we can take $\mathbf{P} = (\mathbf{C}_{11} + \mathbf{C}_{22})^{1/2}\mathbf{W}$, so that the constraint $\mathbf{W}'(\mathbf{C}_{11} + \mathbf{C}_{22})\mathbf{W} = \mathbf{I}_k$ becomes $\mathbf{P}'\mathbf{P} = \mathbf{I}_k$. Also put $\mathbf{H} = (\mathbf{C}_{11}+\mathbf{C}_{22})^{-1/2}(\mathbf{C}_{12}+\mathbf{C}_{21})(\mathbf{C}_{11}+\mathbf{C}_{22})^{-1/2}$, so that the objective function (3.8) becomes trace($\mathbf{P}'\mathbf{H}\mathbf{P}$). Applying the Ky Fan theorem and choosing $\mathbf{Q} = \mathbf{I}_k$, leads to the fact that $\mathbf{P}^* = (\mathbf{u}_1, \ldots, \mathbf{u}_k)$, with $\mathbf{u}_1, \ldots, \mathbf{u}_k$ the k eigenvectors corresponding of the k largest eigenvalues of $\mathbf{H}$. Thus, the optimal $\mathbf{W}^* = (\mathbf{C}_{11}+\mathbf{C}_{22})^{-1/2}\mathbf{P}^*$. For $\mathbf{P}^*$ an eigenvector of $\mathbf{H} = (\mathbf{C}_{11} + \mathbf{C}_{22})^{-1/2}(\mathbf{C}_{12} + \mathbf{C}_{21})(\mathbf{C}_{11} + \mathbf{C}_{22})^{-1/2}$, this $\mathbf{W}^*$ is exactly the generalized eigenvector (corresponding to the same eigenvalue) of (3.7). The result is thus exactly the same as obtained in the derivation in Appendix A.

3.2 Spectral clustering with constraints

Whereas in the first section of this chapter the side-information is exploited by performing a dimensionality reduction onto a subspace that reflects the constraints as well as possible, here we propose a method that deals with side-information by adapting an existing clustering algorithm to satisfy the constraints that are given (published in [21]).

Concretely, we address the (spectral) clustering problem as in section 2.4, where additionally general information on the class labels y_i of some of the samples $\mathbf{x}_i$ $(i = 1, \ldots n)$ is given. This label information can be of two general forms: in the first setting subsets of samples are given for which is specified that they belong to the same class; in the second setting, similarly subsets of samples with the same label are given, but now additionally, some of these subsets are constrained to be differently labeled than another such subset. Note that the standard transduction setting, where part of the samples are labelled, is in fact a special case of this type of label information.

The methodology presented here elegantly handles the first type of label information in both the two class and the multi class learning settings. Furthermore, we show how the second type of label information can be dealt with in full generality for the two class case (and in the special case of transduction also indirectly for the multi class case).

In a first subsection we will review spectral clustering as a relaxation of a combinatorial problem. In the second subsection, we will show how to enforce the constraints to the spectral clustering method, first for the two class case, and subsequently for the multi class case. Then, without going into detail, we will point out how the constraints can be imposed in a soft way as well. Finally, empirical results are reported and compared to another recently proposed approach [64] that is able to deal with similar settings.

3.2.1 Spectral clustering

As seen before in section 2.4, spectral clustering methods can best be derived as relaxations of graph cut problems, as first presented in [103]. We will briefly go into the derivation again from a slightly different perspective.

Two class clustering

Consider a weighted graph over the nodes each representing a sample $\mathbf{x}_i$. The edge weights correspond to some positive similarity measure to be defined in an appropriate way. These similarities can be arranged in a symmetric *affinity* matrix $\mathbf{A}$: its entry at row i and column j, denoted by a_{ij}, represents the similarity between sample $\mathbf{x}_i$ and $\mathbf{x}_j$.[5]

A graph cut problem searches for a partition of the nodes in two sets (corresponding clusters of the samples $\mathbf{x}_i$) such that a certain cost function is minimized. Several cost functions are proposed in literature, among which the *average cut cost* and the *normalized cut cost* are best known and most widely used.

For the normalized cut cost, the discrete optimization problem can be written

[5] In many practical cases this similarity will be given by a kernel function $k(\mathbf{x}_i, \mathbf{x}_j) = a_{ij}$ in which case $\mathbf{A}$ is a semi positive definite kernel matrix.

in the form (see e.g. [103]):[6]

$$\min_{\widetilde{\mathbf{y}}} \quad \frac{\widetilde{\mathbf{y}}'(\mathbf{D}-\mathbf{A})\widetilde{\mathbf{y}}}{\widetilde{\mathbf{y}}'\mathbf{D}\widetilde{\mathbf{y}}} \tag{3.8}$$

$$\text{s.t.} \quad \mathbf{1}'\mathbf{D}\widetilde{\mathbf{y}} = 0 \tag{3.9}$$

$$\widetilde{y}_i \in \{\widetilde{y}_+, \widetilde{y}_-\} \tag{3.10}$$

where $\widetilde{y}_+$ and $\widetilde{y}_-$ are the two possible values the $\widetilde{y}_i$ take depending on the class $\mathbf{x}_i$ is assigned to, and $\mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1})$ is a diagonal matrix containing all row sums d_i of $\mathbf{A}$ as its diagonal entries. The matrix $\mathbf{D} - \mathbf{A}$ is generally known as the Laplacian of the graph associated with $\mathbf{A}$. Note that the Laplacian is always semi positive definite, no matter if $\mathbf{A}$ is semi positive definite.

It is constraint (3.10) that causes this problem to be combinatorial. However, the relaxed problem obtained by dropping this constraint can be solved very easily as we will show now. Furthermore, using the resulting vector $\widetilde{\mathbf{y}}$ as an approximation has been shown to be very effective in practical problems. Note that since the scale of $\widetilde{\mathbf{y}}$ in fact does not matter, we can as well solve

$$\begin{aligned} \min_{\widetilde{\mathbf{y}}} \quad & \widetilde{\mathbf{y}}'(\mathbf{D}-\mathbf{A})\widetilde{\mathbf{y}} \\ \text{s.t.} \quad & \widetilde{\mathbf{y}}'\mathbf{D}\widetilde{\mathbf{y}} = 1 \\ & \mathbf{1}'\mathbf{D}\widetilde{\mathbf{y}} = 0 \end{aligned} \tag{3.11}$$

If we would drop constraint (3.11), the minimization becomes equivalent to solving for the minimal eigenvalue of

$$(\mathbf{D}-\mathbf{A})\widetilde{\mathbf{y}} = \lambda\mathbf{D}\widetilde{\mathbf{y}} \tag{3.12}$$

or, after left multiplication with $\mathbf{D}^{-1/2}$

$$\begin{aligned} \mathbf{D}^{-1/2}(\mathbf{D}-\mathbf{A})\mathbf{D}^{-1/2}\mathbf{v} &= \lambda\mathbf{v} \\ \text{with} \qquad \mathbf{v} &= \mathbf{D}^{1/2}\widetilde{\mathbf{y}}. \end{aligned} \tag{3.13}$$

Now note that this is an ordinary symmetric eigenvalue problem, of which the eigenvectors are orthogonal. Since the Laplacian and thus also $\mathbf{D}^{-1/2}(\mathbf{D}-\mathbf{A})\mathbf{D}^{-1/2}$ is always semi positive definite, none of its eigenvalues can be smaller than 0. We can see immediately that a 0 eigenvalue is achieved by the eigenvector $\mathbf{v}_0 = \mathbf{D}^{1/2}\mathbf{1}$. This means that all other eigenvectors $\mathbf{v}_i$ of $\mathbf{D}^{-1/2}(\mathbf{D}-\mathbf{A})\mathbf{D}^{-1/2}$ are orthogonal to $\mathbf{v}_0$, such that for $\widetilde{\mathbf{y}}_i = \mathbf{D}^{-1/2}\mathbf{v}_i$, we have that $\mathbf{1}'\mathbf{D}\widetilde{\mathbf{y}}_i = \mathbf{v}_0'\mathbf{v}_i = 0$. It thus follows that constraint (3.11) is automatically taken into account by simply solving for the *second* smallest eigenvalue of (3.13) or (3.12). This is the final version of the spectral clustering method as a relaxation of the normalized cut problem.[7]

[6] Note that we use a tilde on top of the $\mathbf{y}$ and y variables, to indicate that they are not integer label variables (i.e. not $\{-1, 1\}$ as is usual in the two-class case), in line with our notation in Section 2.4.

[7] We can follow a similar derivation for the average cut cost function, ultimately leading to solving for the second smallest eigenvalue of $(\mathbf{D}-\mathbf{A})\widetilde{\mathbf{y}} = \lambda\widetilde{\mathbf{y}}$. All results presented in this chapter can immediately be transferred to the average cut variant of spectral clustering.

Multi class clustering

In the c-class case, one usually extracts the eigenvectors $\widetilde{\mathbf{y}}_1, \widetilde{\mathbf{y}}_2, \ldots, \widetilde{\mathbf{y}}_{c-1}$ corresponding to the smallest $c-1$ eigenvalues (excluding the 0 eigenvalue). Then, these vectors are put next to each other in a matrix $\widetilde{\mathbf{Y}} = \begin{pmatrix} \widetilde{\mathbf{y}}_1 & \widetilde{\mathbf{y}}_2 & \cdots & \widetilde{\mathbf{y}}_{c-1} \end{pmatrix}$, and subsequently any clustering algorithm can be applied to the rows of this matrix.[8] Every sample $\mathbf{x}_i$ is then assigned a label corresponding to which cluster row i of $\widetilde{\mathbf{Y}}$ is assigned to.

3.2.2 Hard constrained spectral clustering

Our results derive from the observations that

- constraining the labels according to the information as specified in the introduction can be seen as constraining the label vector $\widetilde{\mathbf{y}}$ to belong to some subspace, and
- it is easy, in principle and computationally, to constrain the vector $\widetilde{\mathbf{y}}$ to this subspace, while optimizing the Rayleigh quotient (3.8) subject to the constraint (3.9).

We will first tackle the two class learning problem subject to general label equality and inequality constraints. Afterwards, we show how equality constraints can be handled in the multi class setting.

Two class learning

Consider again the unrelaxed graph cut problem (3.8),(3.9),(3.10). We would now like to solve it with respect to the label information as additional constraints. For this we introduce the label constraint matrix $\mathbf{L} \in \{-1, 0, 1\}^{n \times m}$ (with $n \geq m$) associated with the label equality and inequality constraints:

$$\mathbf{L} = \begin{pmatrix} \mathbf{1}_{s_1} & \mathbf{1}_{s_1} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_2} & -\mathbf{1}_{s_2} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_3} & \mathbf{0} & \mathbf{1}_{s_3} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_4} & \mathbf{0} & -\mathbf{1}_{s_4} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{1}_{s_{2p-1}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{s_{2p-1}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_{2p}} & \mathbf{0} & \mathbf{0} & \cdots & -\mathbf{1}_{s_{2p}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_{2p+1}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{1}_{s_{2p+1}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_{2p+2}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{1}_{s_{2p+2}} & \cdots & \mathbf{0} \\ \cdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{1}_{s_g} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{s_g} \end{pmatrix}.$$

Hereby, every row i of $\mathbf{L}$ corresponds to sample $\mathbf{x}_i$, in such a way that samples corresponding to one block row of size s_k are given to belong to the same class

[8] In [85] it is suggested to first normalize the rows of $\mathbf{Y}$ before performing the clustering.

(i.e., for ease of notation the samples are sorted accordingly; of course also the rows of $\mathbf{A}$ need to be sorted in the same way). On the other hand, inequality constraints are encoded by the first $2p$ block rows: for all $i \leq p$, samples from block row $2k-1$ are given to belong to a different class as samples from block row $2k$. For the last $g-2p$ block rows no inequality constraints are given. Note that in most practical cases, many block row heights s_k will be equal to 1, indicating that no constraint for the corresponding sample is given.

Using the label constraint matrix $\mathbf{L}$, it is possible to impose the label constraints explicitly, by introducing an auxiliary vector $\mathbf{z}$ and equating

$$\widetilde{\mathbf{y}} = \mathbf{L}\mathbf{z}.$$

Then again constraint (3.10) is dropped, leading to

$$\begin{aligned} \min_{\mathbf{z}} \quad & \frac{\mathbf{z}'\mathbf{L}'(\mathbf{D}-\mathbf{A})\mathbf{L}\mathbf{z}}{\mathbf{z}'\mathbf{L}'\mathbf{D}\mathbf{L}\mathbf{z}} \\ \text{s.t.} \quad & \mathbf{1}'\mathbf{D}\mathbf{L}\mathbf{z} = 0 \end{aligned}$$

or equivalently

$$\begin{aligned} \min_{\mathbf{z}} \quad & \mathbf{z}'\mathbf{L}'(\mathbf{D}-\mathbf{A})\mathbf{L}\mathbf{z} \\ \text{s.t.} \quad & \mathbf{z}'\mathbf{L}'\mathbf{D}\mathbf{L}\mathbf{z} \\ & \mathbf{1}'\mathbf{D}\mathbf{L}\mathbf{z} = 0. \end{aligned} \tag{3.14}$$

Note that (similarly as in the derivation on standard spectral clustering above) after dropping the constraint (3.14), we would only have to solve the following eigenvalue problem

$$\mathbf{L}'(\mathbf{D}-\mathbf{A})\mathbf{L}\mathbf{z} = \lambda \mathbf{L}'\mathbf{D}\mathbf{L}\mathbf{z} \tag{3.15}$$

or, by left multiplication with $(\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}$ and an appropriate substitution:

$$\begin{aligned} (\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}[\mathbf{L}'(\mathbf{D}-\mathbf{A})\mathbf{L}](\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}\mathbf{v} &= \lambda\mathbf{v} \\ \text{with} \qquad \mathbf{v} &= (\mathbf{L}'\mathbf{D}\mathbf{L})^{1/2}\mathbf{z} \quad . \end{aligned} \tag{3.16}$$

Again, we can show that the extra constraint is taken into account automatically by picking the *second* smallest eigenvalue and associated eigenvector of this eigenvalue problem. To this end, note that $(\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}[\mathbf{L}'(\mathbf{D}-\mathbf{A})\mathbf{L}](\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}$ is semi positive definite, such that its smallest eigenvalue is 0. One can see[9] that this zero eigenvalue is achieved for

$$\mathbf{v}_0 = (\mathbf{L}'\mathbf{D}\mathbf{L})^{1/2} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

[9] To see this note that $\mathbf{L}\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \mathbf{1}$.

Now, since (3.16) is an ordinary symmetric eigenvalue problem, all other eigenvectors $\mathbf{v}_i$ have to be orthogonal: $\mathbf{v}_0'\mathbf{v}_i = 0$. This means that

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \end{pmatrix} (\mathbf{L}'\mathbf{D}\mathbf{L})^{1/2} \cdot (\mathbf{L}'\mathbf{D}\mathbf{L})^{1/2}\mathbf{z} = 0$$

and thus $\mathbf{1}'\mathbf{D}\mathbf{L}\mathbf{z} = 0$.

Thus, it suffices to solve (3.16) or equivalently (3.15) for its second smallest eigenvalue, and constraint (3.14) is taken into account automatically.

In summary, the procedure is as follows

- Compute the affinity matrix $\mathbf{A}$ and the matrix $\mathbf{D} = \mathbf{A}\mathbf{1}$.
- Compute the label constraint matrix $\mathbf{L}$.
- Compute the eigenvector $\mathbf{z}$ corresponding to the second smallest eigenvalue of the eigenvalue problem $\mathbf{L}'(\mathbf{D} - \mathbf{A})\mathbf{L}\mathbf{z} = \lambda\mathbf{L}'\mathbf{D}\mathbf{L}\mathbf{z}$.
- Compute the resulting label vector as a thresholded version of $\widetilde{\mathbf{y}} = \mathbf{L}\mathbf{z}$.

All operations can be carried out very efficiently, thanks to the sparsity of $\mathbf{L}$. The most expensive step is the eigenvalue problem, which is even smaller than in the unconstrained spectral clustering algorithm: the size of the matrices is only $m \times m$ instead of $n \times n$ (where m is the number of columns of $\mathbf{L}$).

As a last remark in this section, note that we are actually not interested in the component of $\widetilde{\mathbf{y}}$ along $\mathbf{1}$. Thus, we could choose to take

$$\widehat{\mathbf{y}} = \mathbf{L} \begin{pmatrix} 0 \\ z_2 \\ \vdots \\ z_m \end{pmatrix}$$

as an estimate for the labels, instead of $\widetilde{\mathbf{y}} = \mathbf{L}\mathbf{z}$. This results in the fact that estimates for labels $\widetilde{y}_i$ that were specified to be different are actually *opposite in sign*. Therefore thresholding the vector $\widehat{\mathbf{y}}$ around 0 in fact makes more sense than thresholding $\widetilde{\mathbf{y}}$ around 0.

Multi class learning

In the multi class setting, it is not possible anymore to include label inequality constraints in the same straightforward elegant way. The reason is that the true values of the labels can not be made equal to 1 and -1 anymore.

We can still take the equality constraints into account however. This means we would use a label constraint matrix of the form

$$\mathbf{L} = \begin{pmatrix} \mathbf{1}_{s_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{1}_{s_2} & \cdots & \mathbf{0} \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{s_g} \end{pmatrix},$$

constructed in a similar way. Note that this time we don't need a column containing all ones, as such vector $\mathbf{1}$ is included in its column space already.

As in the unconstrained spectral clustering algorithm, c eigenvectors will be calculated when $c-1$ clusters are expected, leading to $\widetilde{\mathbf{Y}} = \begin{pmatrix} \widetilde{\mathbf{y}}_1 & \widetilde{\mathbf{y}}_2 & \cdots & \widetilde{\mathbf{y}}_{c-1} \end{pmatrix}$. Finally, the clustering of $\mathbf{x}_i$ is obtained by clustering the rows of $\widetilde{\mathbf{Y}}$.

3.2.3 Softly constrained spectral clustering

In both the two class case and the multi class case, the constraints could be imposed in a soft way as well. This can be done by adding a cost term to the cost function that penalizes the distance between the weight vector and the column space of $\mathbf{L}$, in the following way (we give it without derivation or empirical results)

$$\begin{array}{ll} \min_{\widetilde{\mathbf{y}}} & \gamma \widetilde{\mathbf{y}}'(\mathbf{D}-\mathbf{A})\widetilde{\mathbf{y}} + (1-\gamma)\widetilde{\mathbf{y}}'(\mathbf{D}-\mathbf{D}\mathbf{L}(\mathbf{L}'\mathbf{D}\mathbf{L})^{-1}\mathbf{L}'\mathbf{D})\widetilde{\mathbf{y}} \\ \text{s.t.} & \widetilde{\mathbf{y}}'\mathbf{D}\widetilde{\mathbf{y}} = 1 \ \text{ and } \ \mathbf{1}'\mathbf{D}\widetilde{\mathbf{y}} = 0 \end{array}$$

where $0 < \gamma < 1$ is called the regularization parameter. Again the same reasoning can be applied, leading to the conclusion that one needs to solve for the second smallest eigenvalue (i.e. the smallest eigenvalue different from 0) of the eigenvalue equation:

$$\left[\gamma(\mathbf{D}-\mathbf{A}) + (1-\gamma)(\mathbf{D}-\mathbf{D}\mathbf{L}(\mathbf{L}'\mathbf{D}\mathbf{L})^{-1}\mathbf{L}'\mathbf{D})\right]\widetilde{\mathbf{y}} = \lambda \mathbf{D}\widetilde{\mathbf{y}}. \tag{3.17}$$

For γ close to 0, the side-information is enforced very strongly, and $\widetilde{\mathbf{y}}$ will satisfy the constraints nearly exactly. In the limit for $\gamma \to 0$, the soft constraints become hard constraints.

3.2.4 Empirical results

We report experiments for the transduction setting, and this for the three newsgroup dataset (2995 samples) as also used in [64], and for the training subset of the USPS dataset [48] (7291 samples). Comparisons are shown with the method proposed in [64]. We construct the affinity matrix in the same way as in that paper, namely by equating the entries a_{ij} equal to 1 if j is among the 20 samples lying closest (in euclidian distance) to i or vice versa.

Since spectral clustering methods provide eigenvectors on which subsequently a clustering of the rows has to be performed, and since we are only interested in evaluating the spectral clustering part, we used a cost function defined on the eigenvectors themselves (without doing the actual clustering step). Specifically, the within cluster variance divided by the total variance in the eigenvectors is used as a quality measure. It can attain values in between 0 and 1. All experiments are averaged over 10 randomizations of the labelled part of the training set; each time the standard deviation on the estimated average is shown on the figures.

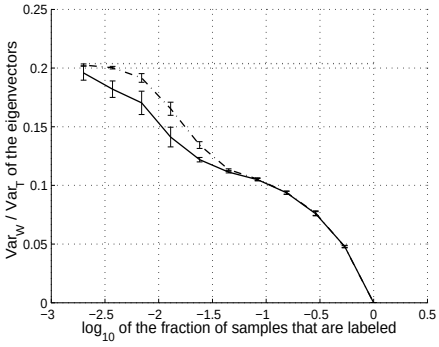


Figure 3.2: The cost (within class variance divided by total variance of the eigenvectors) for spectral learning [64] in dash-dotted line, as compared with our method in full line as a function of fraction of labelled samples (on a log scale). The unconstrained cost is plotted in dotted line.

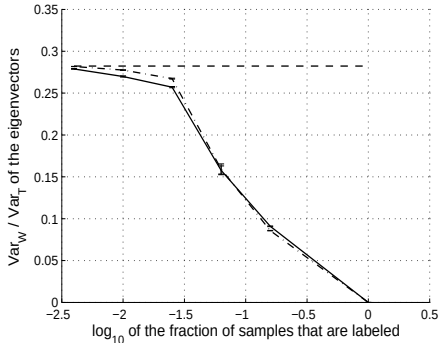


Figure 3.3: The cost for spectral learning [64] in dash-dotted line, as compared with our constrained spectral clustering method in full line as a function of fraction of labelled samples (on a log scale), for the 10-class USPS dataset. The unconstrained cost is plotted in dotted line.

Figures 3.2 and 3.3 show that the performance of both methods effectively increases (the cost decreases) for increasing fraction of labelled samples on the three newsgroup dataset as well as on the USPS dataset. Moreover, expecially for small fractions of labelled samples, the newly proposed constrained spectral clustering method performs significantly better.

Subsequently, we solve a binary classification problem derived from the USPS dataset, where one class contains the samples representing numbers from 0 up to 4, and the other class from 5 up to 9. In figure 3.4 we see that the performance of both methods is comparable in this case. Note however that relatively few information is already sufficient to provide a significant improvement over the clustering with no label information at all.

3.3 Conclusions

In this chapter, we presented two complementary ways to deal with general information on the labels of a given sample, a general class of learning settings of which transduction is a special case. Methods handling these kinds of information allow us to break out of the standard induction framework, which is often not flexible enough to fully satisfy practical needs.

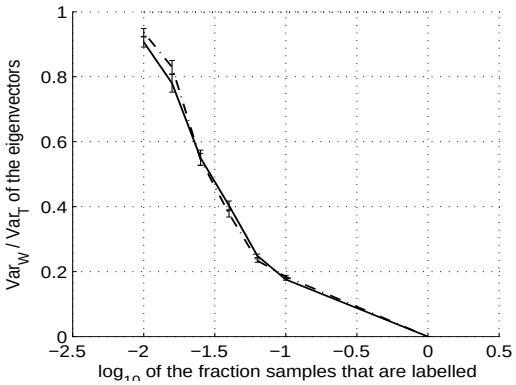


Figure 3.4: Similar experiment as in figure 3.3, but now in the binary classification setting. One class contains all handwritten digits from 0 to 4, the other class contains the digits from 5 to 9. As can be expected the score for no labelled data at all is really bad.

Learning a metric from side-information

Finding a good representation of the data is of crucial importance in many machine learning tasks. However, without any assumptions or side-information, there is no way to find the 'right' metric for the data. Therefore in the first section of this chapter we presented a way to learn an appropriate metric based on examples of co-clustered pairs of points. This type of side-information is often much less expensive or easier to obtain than full information about the label.

The proposed method is justified in two ways: as a maximization of the expected value of a Rayleigh quotient corresponding to LDA, and another way showing connections with previous work on this type of problems. The result is a very efficient algorithm, being much faster than, while showing similar performance as the algorithm derived in [126].

Importantly, the method is put in a more general context, showing it is only one example of a broad class of algorithms that are able to incorporate different forms of side-information. It is pointed out how the method can be extended to deal with basically any kind of side-information.

Furthermore, the result of the algorithm presented here is a lower dimensional representation of the data, just like in other dimensionality reduction methods such as PCA (Principal Component Analysis), PLS (Partial Least Squares), CCA and LDA, that try to identify interesting subspaces for a given task. This often comes as an advantage, since algorithms like K-means and constrained K-means will run faster on lower dimensional data.

Incorporating general label constraints in spectral clustering

In a second section of this chapter we presented an alternative approach to deal with (even more) general types of label information. The result is an efficient, fast and performant method that compares well with a recently proposed related approach that is designed to deal with the same general learning settings.

Note that the soft version can be seen as the application of the spectral clustering method to a sum of two affinity matrices, where one of both is derived from the label constraints. In principle, one may be able to construct a label affinity matrix for very general label information, also for the the multi class case.

Part II

Algorithms based on convex optimization

Chapter 4

Convex optimization in machine learning: a crash course

In this chapter we give a basic introduction to optimization and convex optimization in particular. We will focus on three important classes of optimization problems, linear programming (LP), (convex) quadratic programming (QP) and semi-definite programming (SDP). Interestingly, an LP problem is a special case of a convex QP problem, which in turn is a special case of an SDP problem.

For most of the standard convex optimization problems, dedicated algorithms with proofs of polynomial time convergence towards the global optimum exist. This is what makes it so attractive to cast optimization problems into these standard forms. For a profound introduction to the subject, we refer the reader to [84, 26], on which the first two sections of this chapter are based.

4.1 Optimization and convexity

In this section we will give a brief survey of the theory developed in the context of convex optimization. We do not attempt to be rigorous. Our concern here is only to introduce the reader in the necessary concepts to understand the remainder of this thesis.

The general form of the optimization problems we will consider is (for the time being, we do not assume convexity):

$$\min_{\mathbf{x}} \quad f_0(\mathbf{x}) \tag{4.1}$$

$$\text{s.t.} \quad f_i(\mathbf{x}) \leq 0, i = 1, \ldots, m \tag{4.2}$$

$$h_i(\mathbf{x}) = 0, i = 1, \ldots, p \tag{4.3}$$

where the f_i can be any real function defined on $\mathbb{R}^n$. In what follows, the domain of $\mathbf{x}$ as specified by the constraints will be denoted by $\mathcal{D}$.

4.1.1 Lagrange theory

The Lagrange dual function associated with optimization problem (4.1–4.2), with Lagrange multipliers λ_i and $\boldsymbol{\mu}_i$ is:

$$g(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \min_{\mathbf{x} \in \mathcal{D}} \quad f_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i f_i(\mathbf{x}) + \sum_{i=1}^{p} \mu_i h_i(\mathbf{x}) \tag{4.4}$$

and the objective $\mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{\mu}) = f_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i f_i(\mathbf{x}) + \sum_{i=1}^{p} \mu_i h_i(\mathbf{x})$ is defined as the Lagrangian. When the Lagrangian is unbounded from below in $\mathbf{x}$, the Lagrangian is $-\infty$. Note that since the Lagrangian is the pointwise minimum of a set of affine functions in $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, it is always concave.

4.1.2 Weak duality

Now, one can easily verify that the Lagrange dual function is never larger than the minimum p^* of problem (4.1–4.3), as long as $\boldsymbol{\lambda} \geq \mathbf{0}$. Indeed, for $\boldsymbol{\lambda} \geq \mathbf{0}$, we have that $\sum_{i=1}^{m} \lambda_i f_i(\mathbf{x}) + \sum_{i=1}^{p} \mu_i h_i(\mathbf{x}) \leq 0$ when the constraints of (4.2–4.3) are satisfied. This leads us to the weak duality theorem:

$$d^* = \max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\mu}} g(\boldsymbol{\lambda}, \boldsymbol{\mu}) \leq \min_{\mathbf{x} \in \mathcal{D}} f_0(\mathbf{x}) = p^*. \tag{4.5}$$

In words, the dual maximum is not larger than the primal minimum. The difference $p^* - d^*$ is called the duality gap.

4.1.3 Convexity and strong duality

When the objective and the constraints are convex[1] and when the primal inequality constraints are strictly feasible (the so-called Slater condition), the inequality in the weak duality theorem becomes an equality. In other words, the duality gap is zero. Then one speaks about strong duality.

This means that when the dual is easier to optimize, we can as well do this and obtain the same optimal value.

4.2 Standard formulations of convex optimization problems

Depending on the nature of f_i and h_i, convex optimization problems are further categorized. This is useful since for some of these classes efficient algorithms

[1] A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex iff for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ and $0 \leq \gamma \leq 1$: $f(\gamma \mathbf{x}_1 + (1-\gamma)\mathbf{x}_2) \leq \gamma f(\mathbf{x}_1) + (1-\gamma) f(\mathbf{x}_2)$. A constraint $f_i(\mathbf{x}) \leq 0$ is convex iff f_i is a convex function.

have been developed, such that it helps to cast a problem into one of these standard forms whenever possible.

4.2.1 LP

When f_i and h_i are linear functions of $\mathbf{x}$, the optimization problem is called a *linear program* (LP). Algorithms exist that efficiently solve most practical LP problems (e.g. the simplex method, or interior point methods).

The primal formulation is:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{f}_0'\mathbf{x} + f_0 \\ \text{s.t.} \quad & \mathbf{F}'\mathbf{x} + \mathbf{f} \leq \mathbf{0} \\ & \mathbf{H}'\mathbf{x} + \mathbf{h} = \mathbf{0} \end{aligned}$$

and the corresponding dual:

$$\begin{aligned} \max_{\boldsymbol{\lambda},\boldsymbol{\mu}} \quad & \mathbf{f}'\boldsymbol{\lambda} + \mathbf{h}\boldsymbol{\mu} + f_0 \\ \text{s.t.} \quad & \mathbf{f}_0 + \mathbf{F}\boldsymbol{\lambda} + \mathbf{H}\boldsymbol{\mu} = \mathbf{0} \\ & \boldsymbol{\lambda} \geq \mathbf{0} \end{aligned}$$

which is a linear programming itself.

4.2.2 QP

When f_0 is quadratic and the other f_i and h_i are linear, one speaks of a *quadratic program* (QP). This type of problem is not necessarily convex: it only is when the object function f_0 is convex.

If this is the case, efficient methods exist to solve this optimization problem (e.g. interior point methods). Concretely, for $\mathbf{F}_0 \succ \mathbf{0}$, the following is the primal formulation of a general convex QP problem:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \tfrac{1}{2}\mathbf{x}'\mathbf{F}_0\mathbf{x} + \mathbf{f}_0'\mathbf{x} + f_0 \\ \text{s.t.} \quad & \mathbf{F}'\mathbf{x} + \mathbf{f} \leq \mathbf{0} \\ & \mathbf{H}'\mathbf{x} + \mathbf{h} = \mathbf{0}. \end{aligned}$$

The corresponding dual is:

$$\begin{aligned} \max_{\boldsymbol{\lambda},\boldsymbol{\mu}} \quad & -\tfrac{1}{2}(\mathbf{f}_0 + \mathbf{F}\boldsymbol{\lambda} + \mathbf{H}\boldsymbol{\mu})'\mathbf{F}_0^{-1}(\mathbf{f}_0 + \mathbf{F}\boldsymbol{\lambda} + \mathbf{H}\boldsymbol{\mu}) + \mathbf{f}\boldsymbol{\lambda} + \mathbf{h}\boldsymbol{\mu} + f_0 \\ \text{s.t.} \quad & \boldsymbol{\lambda} \geq \mathbf{0}, \end{aligned}$$

again a convex QP problem.

4.2.3 SDP

Sometimes instead of scalar inequalities in terms of f_i, generalized linear matrix inequalities are used, denoted by $f_i(\mathbf{x}) \preceq \mathbf{0}$. In that case, f_i is a linear function of $\mathbf{x}$ yielding a matrix, and the inequality $f_i(\mathbf{x}) \preceq \mathbf{0}$ means that the matrix $f_i(\mathbf{x})$ is negative semi definite.

SDP problems are convex, and can be solved in polynomial time (which is generally considered as efficient) using (primal-dual) interior point methods.

Lagrangian duality theory can easily be adapted to deal with this type of constraints. The primal formulation of the standard SDP problem is:

$$\begin{array}{ll} \min\limits_{\mathbf{x}} & \mathbf{f}_0'\mathbf{x} \\ \text{s.t.} & \mathbf{F}(\mathbf{x}) = \mathbf{F}_0 + \sum_i x_i \mathbf{F}_i \preceq \mathbf{0} \\ & \mathbf{H}'\mathbf{x} = \mathbf{0} \end{array}$$

and the dual:

$$\begin{array}{ll} \max\limits_{\mathbf{\Gamma},\boldsymbol{\mu}} & \operatorname{tr}(\mathbf{\Gamma} \odot \mathbf{F}_0) \\ \text{s.t.} & \mathbf{\Gamma} \succeq \mathbf{0} \\ & \mathbf{f}_0 + \begin{pmatrix} \operatorname{tr}(\mathbf{\Gamma} \odot \mathbf{F}_1) \\ \operatorname{tr}(\mathbf{\Gamma} \odot \mathbf{F}_2) \\ \vdots \\ \operatorname{tr}(\mathbf{\Gamma} \odot \mathbf{F}_n) \end{pmatrix} + \mathbf{H}\boldsymbol{\mu} = \mathbf{0}. \end{array}$$

Elegantly, convex QP and LP problems can be reduced to SDP problems. These conversions make use of the Schur complement lemma. We refer the reader to the relevant literature [26] for more information.

4.3 Convex optimization in machine learning

4.3.1 SVM classifier

The *support vector machine* (SVM) [114] is nowadays a standard machine learning method for classification. It has its foundations in statistical learning theory and optimization, both grounds contributing to its success, complemented with numerous studies reporting successful empirical results.

The primal optimization problem associated with the *1-norm soft-margin* SVM is (where $\mathbf{K} = \mathbf{X}\mathbf{X}'$ is the kernel matrix):

$$\min_{\xi_i, \mathbf{w}} \quad \frac{1}{2}\mathbf{w}'\mathbf{w} + C\sum_i \xi_i \tag{4.6}$$

$$\text{s.t.} \quad y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 1 - \xi_i \tag{4.7}$$

$$\xi_i \geq 0. \tag{4.8}$$

This is clearly a QP problem. In the object function, we can distinguish two terms: $C\sum_i \xi_i$, which is a measure of the empirical error on the training set, and $\frac{1}{2}\mathbf{w}'\mathbf{w}$, which is a *capacity* or *regularization* term. While the error term ensures the classifier has a good performance on the training set, the regularization term ensures that the classifier is not *overfitting* the training set, such that also a good performance on a *test point* can be expected. We will not go into deeper detail about the statistical aspects here, but refer the reader to the relevant literature (see e.g. [115]).

The dual problem, which is widely used because it allows to use the kernel trick for nonlinear classification and classification of non-vectorial data, is:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & 2\boldsymbol{\alpha}'\mathbf{1} - \boldsymbol{\alpha}'(\mathbf{K} \odot \mathbf{y}\mathbf{y}')\boldsymbol{\alpha} \\ \text{s.t.} \quad & C \geq \alpha_i \geq 0 \\ & \mathbf{y}'\boldsymbol{\alpha} = 0 \end{aligned} \tag{4.9}$$

where $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$ is the relation between the primal and the dual variables.

Note that apart from classification, an SVM formulation for function estimation is developed as well [114].

4.3.2 LS-SVM classifier

The least squares SVM (LS-SVM) classifier [110] is another classifier that is based on an optimization approach:

$$\begin{aligned} \min_{\xi_i, \mathbf{w}} \quad & \mathbf{w}'\mathbf{w} + C\sum_i \xi_i^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}'\mathbf{x}_i + b) = 1 - \xi_i. \end{aligned} \tag{4.10}$$

Constructing the Lagrangian

$$\mathcal{L}(\mathbf{w}, \xi_i, \boldsymbol{\alpha}) = \mathbf{w}'\mathbf{w} + C\sum_i \xi_i^2 + \sum_i \alpha_i \left[1 - \xi_i - y_i(\mathbf{w}'\mathbf{x}_i + b)\right]$$

and optimizing with respect to $\mathbf{w}$, ξ_i and b gives:

$$\begin{aligned} 2\mathbf{w} &= \sum_i \alpha_i y_i \mathbf{x}_i \\ 2C\xi_i &= \alpha_i \\ \mathbf{y}'\boldsymbol{\alpha} &= 0, \end{aligned}$$

and the solution of the dual variables (the Lagrange multipliers) is given by:

$$\begin{pmatrix} 0 & \mathbf{y}' \\ \mathbf{y} & \mathbf{K} \odot \mathbf{y}\mathbf{y}' + \mathbf{I}/C \end{pmatrix} \begin{pmatrix} b \\ \frac{1}{2}\boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{1} \end{pmatrix} \tag{4.11}$$

which is just a linear system.

It is important to note that the LS-SVM formulation can be used for many algorithms other than for classification, such as ridge regression, Fisher's discriminant analysis, CCA, PCA, PLS, recurrent networks etc. The advantage of these formulations is that the *duality* can be interpreted in an optimization context, and the dual variables correspond to the Lagrange multipliers [108]. In contrast, for most of the algorithms discussed in this thesis, we derived the dual version using linear algebra rather than using optimization duality.

4.4 Conclusions

In fact, the backpropagation algorithm has been the first large success of optimization in machine learning, more specifically in neural networks: it was the necessary spark that lit the neural networks candle again. Still, it remained an ad hoc method to solve a non-convex optimization problem. Many other non-convex optimization techniques have been used, among which genetic algorithms and simulated annealing approaches, both mainly used in combinatorial problems, grammar inference and more.

Here, we focus on the parallel line of research that tries to cast or formulate problems as convex optimization problems, often having better properties, or at least, properties that are *provably* good. While the SVM, LS-SVM and some other machine learning methods form the first big successes of convex optimization, other algorithms exploiting the recent evolution in the convex optimization literature have been proposed in the last few years. In the next two chapters, we will discuss a few of our own contributions.

Chapter 5

Convex optimization and transduction

In this chapter we will discuss methods for *transduction* (see Section 0.2.2 for a definition). Since the transduction problem is combinatorial in nature, it is impossible to solve it exactly for reasonable data set sizes. However, in this chapter we report several approaches to transduction that can be tackled using standard convex optimization problems.

In a first section, we approach the transduction problem by making some easy modifications to the standard *inductive* SVM algorithm, exploiting the knowledge of the working set for estimating the marginal distribution of the samples. Two approaches are considered: the first one reweighs the error of the training samples according to the estimated distribution, the second one makes use of a weighted regularization term.

In a second section, we discuss the actual SVM transduction problem. As the problem itself is combinatorial and thus very hard to solve, we developed a relaxation that allows one to solve it in polynomial time and memory.

In a third section, an approach to transduction based on a new convex relaxation of the normalized cut cost function for graphs is discussed. Where the relaxation of the SVM transduction problem discussed in the previous section is still rather expensive in practical cases, the relaxed normalized cut cost may provide a practical solution.

5.1 Support vector machine transduction by density estimation

In this section[1] we will propose two easily understandable and implementable algorithms for transduction. While they do not fully exploit the information available in the *working set*, they do take some advantage of its knowledge. We should be careful in calling these methods *transduction* methods though: strictly speaking, in transduction the *induction* and *deduction* steps are merged, and the labels for the *working set* are directly estimated without picking a hypothesis from a hypothesis class. Still, in the two methods proposed in this section a hypothesis is inferred, be it in a way that is influenced by the working set.[2]

Both approaches we will discuss here are based on a relatively small modification of the SVM object function $\frac{1}{2}\mathbf{w}'\mathbf{w} + C\sum_i \xi_i$ (see equation (4.6)).

The first approach exploits the knowledge of the working set by using it to estimate the marginal distribution $P_{\mathbf{x}}$ of the data points (we define $P_{\mathbf{x}}(\mathbf{x})$ to be the probability density function evaluated in $\mathbf{x} \in \mathcal{X}$; i.e., $\int_{\mathcal{X}} P_{\mathbf{x}}(\mathbf{x})d\mathbf{x} = 1$, and $0 \leq P_{\mathbf{x}}(\mathbf{x}) \leq 1$). Using this marginal distribution, an estimate for the *expected error rate* can be obtained, that is better than $C\sum_i \xi_i$ in the standard SVM objective. Very few assumptions are needed for this approach to hold.

The second approach also exploits the knowledge of the working set by estimating the marginal distribution $P_{\mathbf{x}}$ from it. However, now the *regularization term* $\frac{1}{2}\mathbf{w}'\mathbf{w}$ in the SVM cost function is modified, in such a way that (loosely speaking) the resulting classifier gets more freedom in regions of high density, and less freedom in regions of low density (i.e., it is regularized stronger in regions of lower density). The result is that under the usual smoothness assumptions of the discriminant function, it can only change sign in regions of low density.

A note on the applicability We should note that the second method described in this section is only applicable for use with a kernel function that can be expressed as an inner product between two functions defined on $\mathcal{X}$, the space of the data points. E.g., in this section always an RBF kernel is used.

5.1.1 Weighting errors with the estimated density

All statistical bounds on the expected test set error are expressed in terms of the empirical error on the training set $\sum_i \xi_i$, and a model complexity error

[1] The research that led to the (unpublished) results in this section took place during and after interesting discussions with Xuanlong Nguyen from U.C.Berkeley.

[2] This is immediately the reason why the problems solved in this section are not inherently combinatorial, whereas they are combinatorial in the standard transduction settings.

that depends on the strength of the regularization (i.e., on C in our SVM formulation). In fact, the empirical error is assumed to be a good estimate for the generalization error, while the model complexity term takes into account to what extent the training set is likely to be overfitted. While we did not derive similar generalization bounds for our approaches developed in this section, it makes sense to replace the term $\sum_i \xi_i$ for a better estimate of the training error whenever it is available.

Rationale

Of course the marginal distribution $P_{\mathbf{x}}(\mathbf{x}) = \int_{\mathcal{Y}} P_{\mathbf{x},y}(\mathbf{x}, y)dy$ (with $P_{\mathbf{x},y}(\mathbf{x}, y)$ the joint probability distribution of $\mathbf{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$) is normally not known, instead we can only find an estimate for it using the working set (potentially in conjunction with the training set); in the standard induction case, implicitly only the training set is used however. This means that in the transduction setting we should be able to make a better estimate for the generalization error $\mathcal{E}_{P_{\mathbf{x},y}}\{g(\mathbf{x}, y)\}$ of a cost function $g(\mathbf{x}, y)$ namely as:

$$\sum_i \frac{\hat{P}^{t,w}_{\mathbf{x}}(\mathbf{x}^t_i)}{\hat{P}^t_{\mathbf{x}}(\mathbf{x}^t_i)} g(\mathbf{x}^t_i, y^t_i)$$

where $\hat{P}^t_{\mathbf{x}}$ is the marginal distribution estimate based on the training set only, and $\hat{P}^{t,w}_{\mathbf{x}}$ is the estimate based on the training set together with the working set (in both cases e.g. using a Parzen window estimator). The estimation can be done by using a Parzen window density estimator, for example using an RBF kernel.

Method

Concretely, we propose to modify the SVM object function from:

$$\frac{1}{2}\mathbf{w}'\mathbf{w} + C\sum_i \xi_i$$

into:

$$\frac{1}{2}\mathbf{w}'\mathbf{w} + C\sum_i \frac{\hat{P}^{t,w}_{\mathbf{x}}(\mathbf{x}^t_i)}{\hat{P}^t_{\mathbf{x}}(\mathbf{x}^t_i)} \xi_i,$$

where

$$\begin{aligned}
\hat{P}^{t,w}_{\mathbf{x}}(\mathbf{x}) &= \frac{1}{n^t + n^w}\left(\sum_i K(\mathbf{x}, \mathbf{x}^t_i) + \sum_i K(\mathbf{x}, \mathbf{x}^w_i)\right) \\
\hat{P}^{t,w}_{\mathbf{x}}(\mathbf{x}) &= \frac{1}{n^t}\sum_i K(\mathbf{x}, \mathbf{x}^t_i).
\end{aligned}$$

In the dual this will boil down to using the following constraints on the dual variables α_i

$$C\frac{\hat{P}^{t,w}_{\mathbf{x}}(\mathbf{x}^t_i)}{\hat{P}^t_{\mathbf{x}}(\mathbf{x}^t_i)} \geq \alpha_i \geq 0.$$

In this way an error in a high density region contributes more to the cost than an error in a low density region. In this way training set outliers are dealt with in an elegant way.

Of course a similar strategy can be applied to the 2-norm soft margin SVM and LS-SVM classifiers.

5.1.2 Weighting the weight vector

While the first approach forces the algorithm to focus on regions of high density in minimizing the estimate of the generalization error, the approach that we will discuss now modifies the regularization term $\frac{1}{2}\mathbf{w}'\mathbf{w}$.

We will show that this approach effectively boils down to using a standard SVM but *with a modified kernel*. This modification of the kernel is such that distances between samples separated by densely populated regions become relatively smaller and distances in sparsely populated regions become relatively larger.

Rationale

The approach is applicable to only a specific kind of feature spaces though: the feature space should consist of the square integrable functions defined from the input space $\mathcal{X}$ to the positive real line, parameterized by a vector $\mathbf{x}$: $\phi(\mathbf{x}) : \mathcal{X} \to \mathbb{R}$, and the value of this function in $\mathbf{z} \in \mathcal{X}$ is denoted by $\phi(\mathbf{x})(\mathbf{z})$. The associated inner product between two such feature vectors $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ is defined as $k(\mathbf{x}_i, \mathbf{x}_j) = \int_{\mathcal{X}} \phi(\mathbf{x}_i)(\mathbf{z})\phi(\mathbf{x}_j)(\mathbf{z})d\mathbf{z} = \phi(\mathbf{x}_i)'\phi(\mathbf{x}_j)$. In particular, the RBF kernel can be written in this way. Indeed:

$$\begin{aligned} k(\mathbf{x}_i, \mathbf{x}_j) &= \phi(\mathbf{x}_i)'\phi(\mathbf{x}_j) \\ &= \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \\ &\propto \int \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{z}\|^2}{\sigma^2}\right)\exp\left(-\frac{\|\mathbf{x}_j - \mathbf{z}\|^2}{\sigma^2}\right)d\mathbf{z} \end{aligned}$$

and thus for the RBF kernel:

$$\phi(\mathbf{x})(\mathbf{z}) \propto \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}\|^2}{\sigma^2}\right)$$

(we do not care about the irrelevant normalization).

In the standard SVM formulation, the final discriminant function is of the form $f(\mathbf{x}) = \int_{\mathbf{z}} \phi(\mathbf{x})(\mathbf{z})\mathbf{w}(\mathbf{z})d\mathbf{z} + b = \phi(\mathbf{x})'\mathbf{w} + b$. Based on this, how can we exploit the knowledge of the marginal distribution? This can be done by assuming that a fully connected region of high density is usually not divided into two classes by the classification function. Thus, it should be continuously large or small on regions of high density. On the other hand, in sparsely populated regions, it should get more freedom to change sign easily. Thus: we will modify the cost function, such that the resulting classification function is larger in absolute value in regions of high density and closer to zero in regions of small density. This boils down to penalizing a high $\mathbf{w}(\mathbf{z})$ value where the marginal distribution $P_{\mathbf{x}}(\mathbf{z})$ is large. The way to achieve this is by replacing the regularization term $\mathbf{w}'\mathbf{w} = \int_{\mathbf{z}} \mathbf{w}(\mathbf{z})\mathbf{w}(\mathbf{z})d\mathbf{z}$ by $\mathbf{w}P_{\mathbf{x}}^{-1}\mathbf{w} = \int_{\mathbf{z}} \mathbf{w}(\mathbf{z})'P_{\mathbf{x}}(\mathbf{z})^{-1}\mathbf{w}(\mathbf{z})d\mathbf{z}$. Note that in practice $P_{\mathbf{x}}$ is unknown, so we will have to use an estimate for the density $\hat{P}_{\mathbf{x}}^{t,w}$ instead.

Method

Thus, the optimization problem we propose to solve is

$$\begin{aligned} \min_{\mathbf{w},\xi_i,b} \quad & \frac{1}{2}\mathbf{w}'(\hat{P}_{\mathbf{x}}^{t,w})^{-1}\mathbf{w} + \sum_i \xi_i \\ \text{s.t.} \quad & y_i\left(\mathbf{w}'\phi(\mathbf{x}_i) + b\right) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned}$$

As motivated above, this allows for a larger complexity in regions of high density. The dual optimization problem is

$$\begin{aligned} \max_{\alpha_i} \quad & 2\sum_i \alpha_i - \sum_i \sum_j \alpha_i \alpha_j y_i y_j \left[\phi(\mathbf{x}_i)'\hat{P}_{\mathbf{x}}^{t,w}\phi(\mathbf{x}_j)\right] \\ \text{s.t.} \quad & \alpha_i \geq 0 \\ & \sum_i \alpha_i y_i = 0 \end{aligned}$$

where

$$\mathbf{w} = \sum_i \alpha_i y_i \left[\hat{P}_{\mathbf{x}}^{t,w}\phi(\mathbf{x}_i)\right]$$

such that an evaluation on a new sample $\mathbf{x}$ can be written as

$$f(\mathbf{x}) = \phi(\mathbf{x})'\mathbf{w} + b = \sum_i \alpha_i y_i \left[\phi(\mathbf{x})'\hat{P}_{\mathbf{x}}^{t,w}\phi(\mathbf{x}_i)\right] + b.$$

Equivalent kernel function From the above it can be seen that this change of the object function effectively boils down to performing a standard SVM,

using a modified kernel function $\phi(\mathbf{x}_i)'\hat{P}_{\mathbf{x}}^{t,w}\phi(\mathbf{x}_j)$ instead of $\phi(\mathbf{x}_i)'\phi(\mathbf{x}_j)$. Specifically for the RBF kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2\sigma^2}\right)$, we have (as shown above):

$$\phi(\mathbf{x})(\mathbf{z}) \propto \exp\left(-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{\sigma^2}\right).$$

Then, if we use a Parzen window estimator for the density making use of the same functions $\phi(\mathbf{x}_i)(\mathbf{z})$ as basis functions (this is not required, we just work this out as an example), we get

$$P_{\mathbf{x}}(\mathbf{z}) \propto \sum_i \phi(\mathbf{x}_i)(\mathbf{z}).$$

Then, the kernel effectively used by the approach proposed is

$$\begin{aligned} k_1(\mathbf{x}_i, \mathbf{x}_j) &= \phi(\mathbf{x}_i)'P_{\mathbf{x}}\phi(\mathbf{x}_j) \\ &\propto \int \exp\left(-\frac{\|\mathbf{x}_i-\mathbf{z}\|^2}{\sigma^2}\right) P_{\mathbf{x}}(\mathbf{z}) \exp\left(-\frac{\|\mathbf{x}_j-\mathbf{z}\|^2}{\sigma^2}\right) d\mathbf{z} \\ &\propto \sum_k \exp\left(-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2+\|\mathbf{x}_i-\mathbf{x}_k\|^2+\|\mathbf{x}_j-\mathbf{x}_k\|^2}{3\sigma^2}\right) \end{aligned}$$

Complexity Whereas the method discussed in the first subsection has virtually no effect on the complexity of the algorithm, here the computation of the full kernel matrix requires $O(n^3)$ computations instead of the usual $0(n^2)$, where n is the total number of samples (training set plus working set). However speedups could be achieved by making an approximation by only summing over the k nearest neighbors, reducing the complexity to $0(kn^2)$.

Extensions Instead of weighting the inner product with $P_{\mathbf{x}}$, one can also weigh with $P_{\mathbf{x}}^m$. Under the same conditions as above, this gives rise to a kernel

$$k_m(\mathbf{x}_i, \mathbf{x}_j) \propto \textstyle\sum_{I\in\{\{1\ldots,n\}\setminus\{i,j\}\}^m}$$
$$\exp\left(-\frac{\sum_{k,l\in I:k>l}\|\mathbf{x}_k-\mathbf{x}_l\|^2+\sum_{k\in I}\|\mathbf{x}_i-\mathbf{x}_k\|^2+\sum_{k\in I}\|\mathbf{x}_j-\mathbf{x}_k\|^2+\|\mathbf{x}_i-\mathbf{x}_j\|^2}{(m+2)\sigma^2}\right).$$

However, this goes at a significant additional computational cost.

In the next 2 sections we discuss 2 transduction settings in the traditional sense: *find the labels of the working set such that a global cost function involving all training and working set samples is optimized*. In the next section, the cost function is the inverse of the margin achieved on the training plus working set. In the last section, the cost function is the normalized cut cost on a graph that is derived from the samples.

5.2 A convex relaxation of SVM transduction

The two-class transduction problem, as it was originally formulated by Vapnik [115], involves finding a separating hyperplane for a labelled data set that is also maximally distant from a given set of unlabelled test points. In this form, the problem has exponential computational complexity in the size of the working set. So far it has been attacked by means of integer programming techniques [12] that do not scale to reasonable problem sizes, or by local search procedures [61].

In this section we present a relaxation of this task based on semi-definite programming (SDP), resulting in a convex optimization problem that has polynomial complexity in the size of the data set. (The material in this section has been published in [13].)

The results are very encouraging for mid sized data sets, however the cost is still too high for large scale problems, due to the high dimensional search space. To this end, we restrict the feasible region by introducing an approximation based on solving an eigenproblem. With this approximation, the computational cost of the algorithm is such that problems with more than 1000 points can be treated.

5.2.1 The transductive SVM

The dual formulation of the transductive SVM optimization problem can be written as a minimization of the dual SVM cost function (which is the inverse margin plus training errors) over label matrix $\mathbf{\Gamma}$ ([115], p. 437):[3]

$$\min_{\mathbf{\Gamma}} \max_{\boldsymbol{\alpha}} \quad 2\boldsymbol{\alpha}'\mathbf{1} - \boldsymbol{\alpha}'(\mathbf{K} \odot \mathbf{\Gamma})\boldsymbol{\alpha} \tag{5.1}$$

$$\text{s.t.} \quad C \geq \alpha_i \geq 0 \tag{5.2}$$

$$\mathbf{\Gamma} = \begin{pmatrix} \mathbf{y}^t \\ \mathbf{y}^w \end{pmatrix} \cdot \begin{pmatrix} \mathbf{y}^t \\ \mathbf{y}^w \end{pmatrix}' \tag{5.3}$$

$$y_i^w \in \{1, -1\}. \tag{5.4}$$

The (symmetric) matrix $\mathbf{\Gamma}$ is thus parameterized by the unknown working set label vector $\mathbf{y}^w \in \{-1, 1\}^{n_w}$ (with n_w the size of the working set). The vector $\mathbf{y}^t \in \{-1, 1\}^{n_t}$ (with n_t the number of training points) is the given fixed vector containing the known labels for the training points. The (symmetric) matrix $\mathbf{K} \in \mathbb{R}^{(n_w+n_t)\times(n_w+n_t)}$ is the entire kernel matrix on the training set together with the working set. The dual vector is denoted by $\boldsymbol{\alpha} \in \mathbb{R}^{n_w+n_t}$.

This is a combinatorial optimization problem. The computational complexity scales exponentially in the size of the working set. Relaxing this cost function to make its solution computationally feasible is the subject of this section.

[3] We do not include a bias term since this would make it much harder to relax the problem to a convex optimization problem. However this does not impair the result as is explained in [89].

Specific notation For ease of notation, the training part of the label matrix (and thus also of the kernel matrix) is always assumed to be its upper $n_t \times n_t$ block (as is assumed already in (5.3)). Furthermore, the n_{t+} positive training samples are assumed to correspond to the first entries in $\mathbf{y}^t$, the n_{t-} negative samples being at the end of this vector.

5.2.2 Relaxation to an SDP problem

In this subsection, we will gradually derive a relaxed version of the transductive SVM formulation. To start with, we replace some of the constraints by an equivalent set:

Proposition 5.1 *(5.3) and (5.4) are equivalent with the following set of constraints:*

$$[\mathbf{\Gamma}]_{i,j\in\{1:n_t,1:n_t\}} = y_i^t y_j^t \tag{5.5}$$

$$\text{diag}(\mathbf{\Gamma}) = \mathbf{1} \tag{5.6}$$

$$\text{rank}(\mathbf{\Gamma}) = 1. \tag{5.7}$$

The values of $\mathbf{\Gamma}$ will then indeed be equal to 1 or -1. It is basically the rank constraint that makes the resulting constrained optimization problem combinatorial.

Note that these constraints imply that $\mathbf{\Gamma}$ is semi positive definite (SPD): $\mathbf{\Gamma} \succeq \mathbf{0}$ (this follows trivially from (5.3), or from (5.6) together with (5.7)). Now, in literature (see eg [49]) it is observed that such a SPD rank one constraint can often be relaxed to only the SPD constraint without sacrificing too much of the performance. Furthermore:

Proposition 5.2 *If we relax the constraints by replacing (5.7) with*

$$\mathbf{\Gamma} \succeq \mathbf{0}, \tag{5.8}$$

the optimization problem becomes convex.

This follows from the fact that $\mathbf{\Gamma}$ appears linearly in the cost function, and that the constraints (5.2), (5.5), (5.6) and (5.8) consist of only linear equalities and linear (matrix) inequalities in the variables. Further on we will show that this is an SDP problem.

While this relaxation of the rank constraint makes the optimization problem convex, the result will not be a rank one matrix anymore; it will only provide an approximation for the optimal rank one matrix. Thus the values of $\mathbf{\Gamma}$ will not be equal to 1 or -1 anymore. However, it is well known that:

Lemma 5.2.1 *A principal submatrix of a SPD matrix is also SPD [54].*

By applying this lemma on all 2×2 principal submatrices of $\mathbf{\Gamma}$, it is shown that

Corollary 5.3 *From constraints (5.6) and (5.8) follows:* $-1 \leq [\mathbf{\Gamma}]_{i,j} \leq 1$.

This is the problem we will solve here: optimize (5.1) subject to (5.2), (5.5), (5.6) and (5.8). In the remainder of this section we will reformulate the optimization problem into a standard form of SDP, make further simplifications based on the problem structure, and show how to extract an approximation for the labels from the result.

Formulation as a standard SDP problem

In the derivations in this subsection the equality constraints (5.5) and (5.6) will not be stated for brevity. Their consequences will be treated further. Furthermore, in the implementation, they will be enforced explicitly by the parameterization, thus they will not appear as constraints in the optimization problem. Also the SPD constraint (5.8) is not written every time, it should be understood.

Let $2\boldsymbol{\nu} \geq \mathbf{0}$ be the Lagrange dual variables corresponding to constraints $\alpha_i \geq 0$, and $2\boldsymbol{\mu} \geq \mathbf{0}$ corresponding to constraints $\alpha_i \leq C$. Then, since the problem is convex and thus the minimization and maximization are exchangeable (strong duality, see [26] for a thorough introduction to duality), the optimization problem is equivalent with:

$$\min_{\mathbf{\Gamma},\boldsymbol{\nu}\geq\mathbf{0},\boldsymbol{\mu}\geq\mathbf{0}} \max_{\boldsymbol{\alpha}} \quad 2\boldsymbol{\alpha}'(\mathbf{1}+\boldsymbol{\nu}-\boldsymbol{\mu}) - \boldsymbol{\alpha}'(\mathbf{K}\odot\mathbf{\Gamma})\boldsymbol{\alpha} + 2C\boldsymbol{\mu}'\mathbf{1}.$$

In case $\mathbf{K}\odot\mathbf{\Gamma}$ is rank deficient, $(\mathbf{1}+\boldsymbol{\nu}-\boldsymbol{\mu})$ will be orthogonal to the null space of $\mathbf{K}\odot\mathbf{\Gamma}$ (otherwise, the object function could grow to infinity, and this while $\boldsymbol{\nu}$ and $\boldsymbol{\mu}$ on the contrary are minimizing the objective). The maximum over $\boldsymbol{\alpha}$ is then reached for $\boldsymbol{\alpha} = (\mathbf{K}\odot\mathbf{\Gamma})^{\dagger}(\mathbf{1}+\boldsymbol{\nu}-\boldsymbol{\mu})$. Substituting this in the object function gives:

$$\min_{\mathbf{\Gamma},\boldsymbol{\nu}\geq\mathbf{0},\boldsymbol{\mu}\geq\mathbf{0}}(\mathbf{1}+\boldsymbol{\nu}-\boldsymbol{\mu})'(\mathbf{K}\odot\mathbf{\Gamma})^{\dagger}(\mathbf{1}+\boldsymbol{\nu}-\boldsymbol{\mu}) + 2C\boldsymbol{\mu}'\mathbf{1}.$$

or equivalently:

$$\min_{\mathbf{\Gamma},\boldsymbol{\nu}\geq\mathbf{0},\boldsymbol{\mu}\geq\mathbf{0},t} \quad t \quad \text{s.t.} \quad t \geq (\mathbf{1}+\boldsymbol{\nu}-\boldsymbol{\mu})'(\mathbf{K}\odot\mathbf{\Gamma})^{\dagger}(\mathbf{1}+\boldsymbol{\nu}-\boldsymbol{\mu}) + 2C\boldsymbol{\mu}'\mathbf{1},$$

with as additional constraint that $(\mathbf{1}+\boldsymbol{\nu}-\boldsymbol{\mu})$ is orthogonal to the null space of $\mathbf{K}\odot\mathbf{\Gamma}$. This latter constraint and the quadratic constraint can be reformulated as one SPD constraint thanks to the following extension of the Schur complement lemma [54] (we state it without proof, here):

Lemma 5.2.2 (Extended Schur complement lemma) *For symmetric matrices* $\mathbf{A} \succeq \mathbf{0}$ *and* $\mathbf{C} \succeq \mathbf{0}$*:*

$$\left.\begin{array}{rcl} \textit{The column space of } \mathbf{B} & \perp & \textit{the null space of } \mathbf{A} \\ \mathbf{C} & \succeq & \mathbf{B}'\mathbf{A}^{\dagger}\mathbf{B} \end{array}\right\} \Leftrightarrow \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}' & \mathbf{C} \end{pmatrix} \succeq \mathbf{0}.$$

Indeed, applying this lemma to our problem with $\mathbf{A} = \mathbf{K} \odot \mathbf{\Gamma}$, $\mathbf{B} = \mathbf{1} + \boldsymbol{\nu} - \boldsymbol{\mu}$ and $\mathbf{C} = t - 2C\boldsymbol{\mu}'\mathbf{1}$, leads to the problem formulation in the standard SDP form:

$$\min_{\mathbf{\Gamma}, \boldsymbol{\nu} \geq \mathbf{0}, \boldsymbol{\mu} \geq \mathbf{0}, t} \quad t \tag{5.9}$$

$$\text{s.t.} \quad \begin{pmatrix} \mathbf{K} \odot \mathbf{\Gamma} & (\mathbf{1} + \boldsymbol{\nu} - \boldsymbol{\mu}) \\ (\mathbf{1} + \boldsymbol{\nu} - \boldsymbol{\mu})' & t - 2C\boldsymbol{\mu}'\mathbf{1} \end{pmatrix} \succeq \mathbf{0} \tag{5.10}$$

together with the constraints (5.5), (5.6) and (5.8). The relaxation for the hard margin SVM is found by following a very similar derivation, or by just equating $\boldsymbol{\mu}$ to $\mathbf{0}$.

The number of variables specifying $\mathbf{\Gamma}$, and the size of constraint (5.8) can be greatly reduced due to structure in the problem. This is subject of what follows now.

Simplifications due to the problem structure

The matrix $\mathbf{\Gamma}$ can be parameterized as $\mathbf{\Gamma} = \begin{pmatrix} \mathbf{y}^t\mathbf{y}^{t'} & \mathbf{\Gamma}^c \\ \mathbf{\Gamma}^{c\prime} & \mathbf{\Gamma}^w \end{pmatrix}$ where we have a training block $\mathbf{y}^t\mathbf{y}^{t'} \in \mathbb{R}^{n_t \times n_t}$, cross blocks $\mathbf{\Gamma}^c \in \mathbb{R}^{n_t \times n_w}$ and $\mathbf{\Gamma}^{c\prime}$, and a transduction block $\mathbf{\Gamma}^w \in \mathbb{R}^{n_w \times n_w}$, which is a symmetric matrix with diagonal entries equal to 1. We now use Lemma 5.2.1: by choosing a submatrix that contains all rows and columns corresponding to the training block, and just one row and column corresponding to the transduction part, the SPD constraint of $\mathbf{\Gamma}$ is seen to imply that

$$\begin{pmatrix} \mathbf{y}^t\mathbf{y}^{t'} & \boldsymbol{\gamma}_i^c \\ \boldsymbol{\gamma}_i^{c\prime} & 1 \end{pmatrix} \succeq \mathbf{0}$$

where $\boldsymbol{\gamma}_i^c$ represents the ith column of $\mathbf{\Gamma}^c$. Using the extended Schur complement lemma 5.2.2, it follows that $\boldsymbol{\gamma}_i^c$ is proportional to $\mathbf{y}^t$ (denoted by $\boldsymbol{\gamma}_i^c = g_i\mathbf{y}^t$), and $1 \succeq \boldsymbol{\gamma}_i^{c\prime}\left(\mathbf{y}^t\mathbf{y}^{t'}\right)^\dagger \boldsymbol{\gamma}_i^c = \boldsymbol{\gamma}_i^{c\prime}\frac{\mathbf{y}^t\mathbf{y}^{t'}}{\|\mathbf{y}^t\|^4}\boldsymbol{\gamma}_i^c$. This implies that $1 \geq g_i\mathbf{y}^{t\prime}\frac{\mathbf{y}^t\mathbf{y}^{t'}}{\|\mathbf{y}^t\|^4}\mathbf{y}^t g_i = g_i^2$ such that $-1 \leq g_i \leq 1$. (Note that this is a corollary of the SPD constraint and does not need to be imposed explicitly.) Thus, the parameterization of $\mathbf{\Gamma}$ can be reduced to:

$$\mathbf{\Gamma} \quad = \quad \begin{pmatrix} \mathbf{y}^t\mathbf{y}^{t'} & \mathbf{y}^t\mathbf{g}' \\ \mathbf{g}\mathbf{y}^{t'} & \mathbf{\Gamma}^w \end{pmatrix} \quad \text{with} \quad \mathbf{\Gamma}^w_{ii} = 1$$

where $\mathbf{g}$ is the vector with g_i as ith entry. We can now show that:

Proposition 5.4 *The constraint $\mathbf{\Gamma} \succeq \mathbf{0}$ is equivalent to (and can thus be replaced by) the following SPD constraint on a smaller matrix $\widetilde{\mathbf{\Gamma}}$:*

$$\widetilde{\mathbf{\Gamma}} = \begin{pmatrix} 1 & \mathbf{g}' \\ \mathbf{g} & \mathbf{\Gamma}^w \end{pmatrix} \succeq \mathbf{0}.$$

Since $\widetilde{\mathbf{\Gamma}}$ is a principal submatrix of $\mathbf{\Gamma}$ (assuming at least one training label is equal to 1), lemma 5.2.1 indeed shows that $\mathbf{\Gamma} \succeq \mathbf{0}$ implies $\widetilde{\mathbf{\Gamma}} \succeq \mathbf{0}$. On the other hand, note that by adding a column and corresponding row to $\widetilde{\mathbf{\Gamma}}$, the rank is not increased. Thus, an eigenvalue equal to 0 is added. Due to the interlacing property for bordered matrices [54] and the fact that $\widetilde{\mathbf{\Gamma}} \succeq \mathbf{0}$, we know this can only be the smallest eigenvalue of the resulting matrix. By induction this shows that also $\widetilde{\mathbf{\Gamma}} \succeq \mathbf{0}$ implies $\mathbf{\Gamma} \succeq \mathbf{0}$.

This is the final formulation of the problem. For the soft margin case, the number of parameters is now $1 + 2n_t + \frac{n_w^2+5n_w}{2}$. For the hard margin case, this is $1 + n_t + \frac{n_w^2+3n_w}{2}$.

Extraction of an estimate for the labels from $\mathbf{\Gamma}$

In general, the optimal $\mathbf{\Gamma}$ will of course not be rank one. We can approximate it by a rank one matrix however, by taking $\mathbf{g}$ as an approximation for the labels optimizing the unrelaxed problem. This is the approach we adopt: a thresholded value of the entries of $\mathbf{g}$ will be taken as a guess for the labels of the working set.

Note that the minimum of the relaxed problem is always smaller than or equal to the minimum of the unrelaxed problem. Furthermore, the minimum of the unrelaxed problem is smaller than or equal to the value achieved by the thresholded relaxed labels. Thus, we obtain a *lower* and an *upper bound* for the true optimal cost.

Remarks

The performance of this method is very good, as is seen on a toy problem (figure 5.1 shows an illustrative example). However, due to the (even though polynomial) complexity of SDP in combination with the quadratic dependence of the number of variables on the number of transduction points[4], problems with more than about 1000 training samples and 100 transduction samples can not practically be solved with general purpose SDP algorithms. Especially the limitation on the working set is a drawback, since the advantage of transduction becomes apparent especially for a large working set as compared to the number of training samples. This makes the applicability of this approach for large real life problems rather limited.

5.2.3 Subspace SDP formulation

However, if we would know a subspace (spanned by the d columns of a matrix $\mathbf{V} \in \mathbb{R}^{(n_t+n_w)\times d}$) in which (or close to which) the label vector lies, we can restrict the feasible region for $\mathbf{\Gamma}$, leading to a much more efficient algorithm.

[4]The worst case complexity for the problem at hand is $O((n_t + n_w^2)^2(n_t + n_w)^{2.5})$, which is of order 6.5 in the number of transduction points n_w.

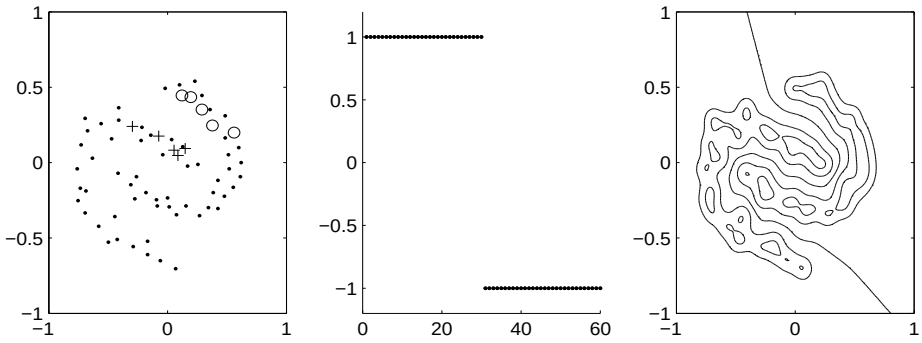


Figure 5.1: The left picture shows 10 labelled samples represented by a 'o' or a '+', depending on their class, together with 60 unlabelled samples represented by a '·'. The middle picture shows the labels for the working set as estimated using the SDP method before thresholding: all are already invisibly close to 1 or -1. The right picture shows contour lines of the classification surface obtained by training an SVM using all labels as found by the SDP method. The method clearly finds a visually good label assignment that takes cluster structure in the data into account.

One way to construct a good choice of $\mathbf{V}$ is by using the constrained spectral clustering method as explained in section 3.2: indeed the label vector will probably be close to the column space of the matrix made up by the d eigenvectors corresponding to the d smallest (except for the 0) eigenvalues of the spectral clustering eigenvalue problem. The larger d, the better this approximation will be.

Now, if we know that the true label vector $\mathbf{y}$ (approximately) lies in the column space of a matrix $\mathbf{V}$, we know the true label matrix can be written in the form $\mathbf{\Gamma} = \mathbf{VMV}'$, with $\mathbf{M}$ a symmetric matrix. The number of parameters is now only $d(d+1)/2$. Furthermore, constraint (5.8) that $\mathbf{\Gamma} \succeq 0$ is then equivalent to $\mathbf{M} \succeq 0$, which is a cheaper constraint.

Note however that in practical cases, the true label vector will not lie within but only close to the subspace spanned by the columns of $\mathbf{V}$. Then the diagonal of the label matrix $\mathbf{\Gamma}$ can not always be made exactly equal to $\mathbf{1}$ as required by (5.6). We thus relax this constraint to the requirement that the diagonal is not larger than $\mathbf{1}$. Similarly, the block in the label matrix corresponding to the training samples may not contain 1's and -1's exactly (constraint (5.5)). However, the better $\mathbf{V}$ is chosen, the better this constraint will be met. Thus we optimize (5.9) subject to (5.10) together with three constraints that replace the constraints (5.5), (5.6) and (5.8):

$$\begin{aligned} \mathbf{\Gamma} &= \mathbf{VMV}' \\ \operatorname{diag}(\mathbf{\Gamma}) &\leq \mathbf{1} \\ \mathbf{M} &\succeq \mathbf{0}. \end{aligned}$$

Thus we can approximate the relaxed transductive SVM using this reduced parameterization for $\mathbf{\Gamma}$. The number of effective variables is now only a linear function of n_w: $1 + n_t + n_w + d(d+1)/2$ for a hard margin and $1 + 2(n_t + n_w) + d(d+1)/2$ for a soft margin SVM. Furthermore, one of the SPD constraints is now a constraint on a $d \times d$ matrix instead of a potentially large $(n_w + 1) \times (n_w + 1)$ matrix. For a constant d, the worst case complexity is thus reduced to $O((n_t + n_w)^{4.5})$.

The quality of the approximation can be determined by the user: the number of components d can be chosen depending on the available computing resources, however empirical results show a good performance already for relatively small d.

5.2.4 Empirical results

To show the potential of the method, we extracted data from the USPS data set [48] to form two classes. The positive class is formed by 100 randomly chosen samples representing a number 0, and 100 representing a 1; the negative class by 100 samples representing a 2 and 100 representing a 3. Thus, we have a balanced classification problem with two classes of each 200 samples. The training set is chosen to contain only 10 samples from each of both classes, and is randomly drawn but evenly distributed over the 4 numbers. We used a hard margin SVM with an RBF kernel with $\sigma = 7$ (which is equal to the average distance of the samples to their nearest neighbors, verified to be a good value for the induction as well as for the transduction case). The average ROC-score (area under the ROC-curve) over 10 randomizations is computed, giving 0.75 ± 0.03 as average for the inductive SVM, and 0.959 ± 0.03 for the method developed in this paper (we chose $d = 4$). To illustrate the scalability of the method, and to show that a larger working set is effectively exploited, we used a similar setting (same training set size) but with 1000 samples and $d = 3$, giving an average ROC-score of 0.993 ± 0.004.

For all implementations, we used SeDuMi [107] as a Matlab optimization toolbox.

5.3 Convex transduction using the normalized cut

In this section (see Technical Report [14]) we discuss an alternative approach to transduction based on graph cut cost functions. More specifically, we focus on the normalized cut, which is the cost function of choice in many clustering applications, notably in image segmentation. Since optimizing the normalized cut cost is an NP-complete problem, much of the research attention so far has gone to relaxing the problem of normalized cut clustering to tractable problems, producing so far a spectral relaxation (as presented in Section 2.4, see also

[103]) and a more recently a tighter but computationally much tougher semi-definite programming (SDP) relaxation [127]. In this section we deliver two main contributions: first, we show how an alternative SDP relaxation yields a much more tractable optimization problem, and we show how scalability and speed can further be increased by making a principled approximation. Second, we show how it is possible to efficiently optimize the normalized cut cost in a transduction setting using our newly proposed approaches.

As compared to the previous section, our relaxation of the normalized cut cost leads to a computationally more attractive algorithm than the relaxation of the margin based transduction method. Successful empirical results are reported.

The constrained graph cut problem

The problem setting we consider is the same as where spectral clustering was discussed (in section 3.2): given is a sample $\mathcal{S}$ of size n, consisting of a (labeled) *training set* $\mathcal{S}_t$ and an (unlabeled) *working set* $\mathcal{S}_w$ of size n_t and n_w respectively. Between every pair of samples $(\mathbf{x}_i, \mathbf{x}_j)$, an affinity measure $a_{ij} = a(\mathbf{x}_i, \mathbf{x}_j)$ is given, such that we are able to make an affinity matrix $\mathbf{A}$ containing element a_{ij} at its ith row and jth column. Also here we assume that the function a is symmetric and positive, and no positive definiteness of $\mathbf{A}$ will be necessary, making the application domain larger than that of other methods such as the one discussed in the previous section on SVM transduction and other kernel based methods [29, 13].

In this setting, the problem of transduction can be approached as a constrained graph cut problem on a fully connected graph, where the nodes in the graph represent the samples, and the edges between them are assigned weights equal to the affinities, and the constraints specify that training samples with the same label can not be separated by the cut. Here, we will complement our approach of section 3.2 where we developed a spectral method that deals with this kind of transduction settings. Concretely, we will propose a tight SDP relaxation instead. Furthermore, we will show how both approaches can be combined to yield an algorithm that is both computationally tractable and a tight relaxation of the original optimization problem.

Outline of this section

To put this section in the right context, in subsection 5.3.2 we provide a short derivation of the well-known *spectral* relaxation of the NCut optimization problem, as well as of an (impractical) *SDP* relaxation that is similar to but different from the one obtained in [127] for *clustering*.

In subsection 5.3.3, we develop a practically viable SDP-based transduction algorithm that scales up to real size problems, the main goal in this section. To achieve this, we make three new contributions. First, we propose a novel SDP relaxation of the NCut optimization problem that is computationally much more tractable than the one proposed in [127]. Second, we show how this relaxation

can efficiently deal with label information to operate in a *transduction* setting. Third, we show how this *SDP* relaxation can be approximated using the (looser) *spectral* relaxation, to scale up even further. With the resulting algorithm, one can scale up to thousands of samples. We conclude this section with empirical results in subsection 5.3.4, clearly showing the scalability and accuracy of the method.

5.3.1 NCut transduction

As discussed in section 2.4, the NCut cost function for a partitioning of the sample $\mathcal{S}$ into a positive $\mathcal{P}$ and a negative $\mathcal{N}$ set is given by:

$$\frac{\text{cut}(\mathcal{P},\mathcal{N})}{\text{assoc}(\mathcal{P},\mathcal{S})} + \frac{\text{cut}(\mathcal{N},\mathcal{P})}{\text{assoc}(\mathcal{N},\mathcal{S})} = \left(\frac{1}{\text{assoc}(\mathcal{P},\mathcal{S})} + \frac{1}{\text{assoc}(\mathcal{N},\mathcal{S})}\right) \text{cut}(\mathcal{P},\mathcal{N}), \quad (5.11)$$

where $\text{cut}(\mathcal{P},\mathcal{N}) = \text{cut}(\mathcal{N},\mathcal{P}) = \sum_{i:\mathbf{x}_i\in\mathcal{P},j:\mathbf{x}_j\in\mathcal{N}} a_{ij}$ is the cut between sets $\mathcal{P}$ and $\mathcal{N}$, and $\text{assoc}(\mathcal{P},\mathcal{S}) = \sum_{i:\mathbf{x}_i\in\mathcal{P},j:\mathbf{x}_j\in\mathcal{S}} a_{ij}$ the association between sets $\mathcal{P}$ and the full sample $\mathcal{S}$.

Using the unknown label vector $\mathbf{y} \in \{-1,1\}^n$, the affinity matrix $\mathbf{A}$, the degree vector $\mathbf{d} = \mathbf{A1}$ and associated matrix $\mathbf{D} = \text{diag}(\mathbf{d})$, and shorthand notations $s_+ = \text{assoc}(\mathcal{P},\mathcal{S})$ and $s_- = \text{assoc}(\mathcal{N},\mathcal{S})$, we reformulated this cost function in algebraical terms as (we repeat Equation (2.32)):

$$\begin{aligned} \min_{\mathbf{y},s_+,s_-} \quad & \frac{1}{4}\left(\frac{1}{s_+} + \frac{1}{s_-}\right) \cdot \mathbf{y}'(\mathbf{D}-\mathbf{A})\mathbf{y} \\ \text{s.t.} \quad & \mathbf{y} \in \{-1,1\}^n \\ & \begin{cases} s_+ = \frac{1}{2}\mathbf{d}'(\mathbf{1}+\mathbf{y}) \\ s_- = \frac{1}{2}\mathbf{d}'(\mathbf{1}-\mathbf{y}) \end{cases} \Leftrightarrow \begin{cases} \mathbf{d}'\mathbf{y} = s_+ - s_- \\ \mathbf{d}'\mathbf{1} = s_+ + s_-. \end{cases} \end{aligned} \quad (5.12)$$

Minimizing this cost function with respect to additional constraints on the labels $\mathbf{y}$ as specified by the training labels is equivalent to performing transduction with this cost function.

5.3.2 A spectral and a first SDP relaxation of NCut clustering

A spectral relaxation Again, as shown in Section 2.4, Equation (5.12) can be rewritten as (see also Equation (2.33)):

$$\begin{aligned} \min_{\widetilde{\mathbf{y}},s_+,s_-} \quad & \widetilde{\mathbf{y}}'(\mathbf{D}-\mathbf{A})\widetilde{\mathbf{y}} \\ \text{s.t.} \quad & \widetilde{\mathbf{y}} \in \left\{-\sqrt{\tfrac{s_+}{s_-}}\sqrt{\tfrac{1}{s_+ s_-}}, \sqrt{\tfrac{s_-}{s_+}}\sqrt{\tfrac{1}{s_+ s_-}}\right\}^n \\ & \mathbf{d}'\widetilde{\mathbf{y}} = 0 \quad \text{and} \quad \mathbf{d}'\mathbf{1} = s_+ + s_-. \end{aligned} \quad (5.13)$$

By relaxing the combinatorial constraint on $\mathbf{y}$, the following relaxed optimization problem is obtained:

$$\textbf{Spectral}: \left\{ \begin{array}{ll} \min\limits_{\widetilde{\mathbf{y}}} & \widetilde{\mathbf{y}}'(\mathbf{D}-\mathbf{A})\widetilde{\mathbf{y}} \\ \text{s.t.} & \widetilde{\mathbf{y}}'\mathbf{D}\widetilde{\mathbf{y}}=1 \quad \text{and} \quad \mathbf{d}'\widetilde{\mathbf{y}}=0, \end{array} \right. \tag{5.14}$$

which can be solved by taking the eigenvector corresponding to the second smallest generalized eigenvalue of $(\mathbf{D}-\mathbf{A})\widetilde{\mathbf{y}}=\lambda\mathbf{D}\widetilde{\mathbf{y}}$.

Recently, several methods have been proposed to use spectral clustering relaxations of graph cut problems in a transduction setting [64, 62, 21]. Here we will later on make use of the method we proposed in Section 3.2.

An SDP relaxation Starting from (5.13), and using a matrix $\widetilde{\mathbf{\Gamma}}=\widetilde{\mathbf{y}}\widetilde{\mathbf{y}}'$, we will now show how a (tighter but more expensive) *SDP* relaxation can be obtained. First rewrite (5.13) as:

$$\begin{array}{ll} \min\limits_{\widetilde{\mathbf{\Gamma}},s_+,s_-} & \langle\widetilde{\mathbf{\Gamma}},\mathbf{D}-\mathbf{A}\rangle \\ \text{s.t.} & \widetilde{\mathbf{\Gamma}}=\widetilde{\mathbf{y}}\widetilde{\mathbf{y}}' \quad \text{with} \quad \widetilde{\mathbf{y}}\in\left\{-\sqrt{\frac{s_+}{s_-}}\sqrt{\frac{1}{s_++s_-}},\sqrt{\frac{s_-}{s_+}}\sqrt{\frac{1}{s_++s_-}}\right\}^n \\ & \widetilde{\mathbf{\Gamma}}\mathbf{d}=\mathbf{0} \quad \text{and} \quad \mathbf{d}'\mathbf{1}=s_++s_-. \end{array} \tag{5.15}$$

The pair of constraints (5.15) on $\widetilde{\mathbf{y}}$ are the hard ones. So we will relax it, by finding a constraint set that is convex while as tight as possible. By inspection, we see that for $\widetilde{\mathbf{y}}$ satisfying (5.15): $\widetilde{\mathbf{\Gamma}}\succeq\mathbf{0}$ and $\widetilde{\mathbf{\Gamma}}\geq-\frac{1}{s_++s_-}$. Furthermore, the $\mathbf{D}$-norm constraint we used in the spectral clustering relaxation translates here into $\langle\widetilde{\mathbf{\Gamma}},\mathbf{D}\rangle=1$. A last constraint that is slightly more difficult to observe is that $\mathbf{I}\succeq\mathbf{D}^{1/2}\left(\mathbf{\Gamma}+\frac{\mathbf{1}\mathbf{1}'}{s_++s_-}\right)\mathbf{D}^{1/2}$ (to see this, note that given (5.15), the matrix on the right hand side is of rank 2 with two eigenvalues equal to 1). As a result we get the relaxed problem ($\mathbf{P}$ stands for primal, the dual $\mathbf{D}$ is stated without derivation):

$$\mathbf{P}^{\textbf{clust}}_{\textbf{SDP1}}: \left\{ \begin{array}{ll} \min\limits_{\widetilde{\mathbf{\Gamma}}} & \langle\widetilde{\mathbf{\Gamma}},\mathbf{D}-\mathbf{A}\rangle \\ \text{s.t.} & \mathbf{D}-\frac{\mathbf{1}\mathbf{1}'}{s_++s_-}\succeq\widetilde{\mathbf{\Gamma}} \\ & \widetilde{\mathbf{\Gamma}}\succeq\mathbf{0} \\ & \widetilde{\mathbf{\Gamma}}\geq-\frac{\mathbf{1}\mathbf{1}'}{s_++s_-} \\ & \widetilde{\mathbf{\Gamma}}\mathbf{1}=\mathbf{0} \\ & \langle\widetilde{\mathbf{\Gamma}},\mathbf{D}\rangle=1, \end{array} \right.$$

$$\mathbf{D}^{\textbf{clust}}_{\textbf{SDP1}}: \left\{ \begin{array}{ll} \min\limits_{\mathbf{\Lambda}_1,\mathbf{\Lambda}_2,\boldsymbol{\lambda},\mu} & \langle\mathbf{\Lambda}_1,\frac{\mathbf{1}\mathbf{1}'}{s_++s_-}\rangle-\langle\mathbf{\Lambda}_2,\mathbf{D}\rangle-\frac{n}{s_++s_-}\mathbf{1}'\boldsymbol{\lambda} \\ & \qquad +\mu\left(\langle\frac{\mathbf{1}\mathbf{1}'}{s_++s_-},\mathbf{D}\rangle+1\right) \\ \text{s.t.} & \mathbf{\Lambda}_1\succeq\mathbf{0} \\ & \mathbf{\Lambda}_2\succeq\mathbf{0} \\ & (\mathbf{D}-\mathbf{A})-\mathbf{\Lambda}_1+\mathbf{\Lambda}_2-\mu\mathbf{D}-\mathbf{1}'\boldsymbol{\lambda}\geq\mathbf{0}. \end{array} \right. \tag{5.16}$$

(We want to point out that a similar —but slightly different— result is obtained in [127].) As an SDP problem, it can be solved in polynomial time. However, the time and space complexity are still prohibitively large, making these results impractical. This is due to the two large SDP constraints and the n^2 inequality constraints on the elements of $\widetilde{\mathbf{\Gamma}}$.

The label constraints —for this method to work in a transduction setting— could be imposed on $\widetilde{\mathbf{\Gamma}}$ directly: its entries $\widetilde{\mathbf{\Gamma}}_{i,j}$ with $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{P}_t$ can be constrained to be equal to each other, and similarly for the entries $\widetilde{\mathbf{\Gamma}}_{i,j}$ with $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{N}_t$. However it is not clear for this relaxation how to impose constraints reflecting the fact that two samples are from the opposite class. We will not investigate this further here however, because the basic clustering problem is already intractable in most practical cases.

5.3.3 Two tractable SDP relaxations for transduction with the NCut

We have derived the well-known spectral relaxation as well as a first SDP relaxation of NCut clustering. We will now present an alternative way to relax the NCut optimization problem and show how it can be effectively used in the transduction setting. While this approach will lead to a computationally much more tractable optimization problem already, we will also show how the method can be sped up even further by performing a principled approximation, ultimately leading to a method that easily handles thousands of samples.

A first practically useful SDP relaxation

The approach taken here starts from formulation (5.12). We introduce the notation $\mathbf{\Gamma} = \mathbf{y}\mathbf{y}'$. Then, we can write the equivalent optimization problem:

$$\begin{array}{rl} \min\limits_{\mathbf{\Gamma}, s_+, s_-} & \frac{1}{4}\left(\frac{1}{s_+} + \frac{1}{s_-}\right)\langle \mathbf{\Gamma}, \mathbf{D} - \mathbf{A}\rangle = \frac{s_+ + s_-}{4 s_+ s_-}\langle \mathbf{\Gamma}, \mathbf{D} - \mathbf{A}\rangle \\ \text{s.t.} & \mathbf{\Gamma} = \mathbf{y}\mathbf{y}' \\ & \mathbf{y} \in \{-1, 1\}^n \\ & \langle \mathbf{\Gamma}, \mathbf{d}\mathbf{d}'\rangle = (s_+ - s_-)^2 = (s_+ + s_-)^2 - 4 s_+ s_- \\ & \mathbf{d}'\mathbf{1} = s_+ + s_-. \end{array} \tag{5.17}$$

Now we can relax the combinatorial constraint by replacing it with $\mathbf{\Gamma} \succeq \mathbf{0}$ and $\mathrm{diag}(\mathbf{\Gamma}) = \mathbf{1}$ (while this is a tight relaxation, tighter relaxations are possible at higher computational cost, see [49]). If we further use the notation $p = 4 s_+ s_-$, and the shorthand notation $s = s_+ + s_- = \mathbf{d}'\mathbf{1}$, we get:

$$\begin{array}{rl} \min\limits_{\mathbf{\Gamma}, p} & \frac{s}{p}\langle \mathbf{\Gamma}, \mathbf{D} - \mathbf{A}\rangle \\ \text{s.t.} & \mathbf{\Gamma} \succeq \mathbf{0} \\ & \mathrm{diag}(\mathbf{\Gamma}) = \mathbf{1} \\ & \langle \mathbf{\Gamma}, \mathbf{d}\mathbf{d}'\rangle = s^2 - p \\ & 0 < p \leq s^2. \end{array}$$

By once again reparameterizing with $\widehat{\mathbf{\Gamma}} = \frac{\mathbf{\Gamma}}{p}$ and $q = 1/p$, we obtain an SDP problem as shown below (P) along with its dual (D) (we give the result without derivation):

$$\mathbf{P}^{\mathbf{clust}}_{\mathbf{SDP2}}: \begin{cases} \min\limits_{\widehat{\mathbf{\Gamma}},q} & s\langle \widehat{\mathbf{\Gamma}}, \mathbf{D}-\mathbf{A}\rangle \\ \text{s.t.} & \widehat{\mathbf{\Gamma}} \succeq \mathbf{0} \\ & \operatorname{diag}(\widehat{\mathbf{\Gamma}}) = q\mathbf{1} \\ & \langle \widehat{\mathbf{\Gamma}}, \mathbf{d}\mathbf{d}'\rangle = qs^2 - 1 \\ & q \geq \frac{1}{s^2}, \end{cases}$$

$$\mathbf{D}^{\mathbf{clust}}_{\mathbf{SDP2}}: \begin{cases} \max\limits_{\boldsymbol{\lambda},\mu} & \frac{1}{s^2}\mathbf{1}'\boldsymbol{\lambda} \\ \text{s.t.} & s(\mathbf{D}-\mathbf{A}) - \operatorname{diag}(\boldsymbol{\lambda}) \\ & \quad -\mu\mathbf{d}\mathbf{d}' \succeq \mathbf{0} \\ & \mu s^2 + \mathbf{1}'\boldsymbol{\lambda} \geq 0. \end{cases}$$

Importantly, this relaxation contains much less constraints than $\mathbf{P}_{\mathbf{SDP1}}$. Furthermore, the dual contains only $n+1$ variables. It is this difference that makes this relaxation much more efficiently solvable, for example by using self-dual SDP solvers like SeDuMi [107].

To impose label constraints for the transductive version, we define the label constraint matrix $\mathbf{L} = \begin{pmatrix} \mathbf{y}_t & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$, where we assume without loss of generality that the samples are ordered such that the training samples precede the unlabeled samples. $\mathbf{y}_t$ is a column vector containing the training labels. Then the label constraints can be imposed by observing that any valid $\widehat{\mathbf{\Gamma}}$ must satisfy $\widehat{\mathbf{\Gamma}} = \mathbf{L}\widehat{\mathbf{\Gamma}}_c\mathbf{L}'$. Thus the transductive NCut relaxation becomes:

$$\mathbf{P}^{\mathbf{trans}}_{\mathbf{SDP2}}: \begin{cases} \min\limits_{\widehat{\mathbf{\Gamma}}_c,q} & s\langle \widehat{\mathbf{\Gamma}}_c, \mathbf{L}'(\mathbf{D}-\mathbf{A})\mathbf{L}\rangle \\ \text{s.t.} & \widehat{\mathbf{\Gamma}}_c \succeq \mathbf{0} \\ & \operatorname{diag}(\widehat{\mathbf{\Gamma}}_c) = q\mathbf{1} \\ & \langle \widehat{\mathbf{\Gamma}}_c, \mathbf{L}'\mathbf{d}\mathbf{d}'\mathbf{L}\rangle = qs^2 - 1 \\ & q \geq \frac{1}{s^2}, \end{cases}$$

$$\mathbf{D}^{\mathbf{trans}}_{\mathbf{SDP2}}: \begin{cases} \max\limits_{\boldsymbol{\lambda},\mu} & \frac{1}{s^2}\mathbf{1}'\boldsymbol{\lambda} \\ \text{s.t.} & s\mathbf{L}'(\mathbf{D}-\mathbf{A})\mathbf{L} - \operatorname{diag}(\boldsymbol{\lambda}) - \mu\mathbf{L}'\mathbf{d}\mathbf{d}'\mathbf{L} \succeq \mathbf{0} \\ & \mu s^2 + \mathbf{1}'\boldsymbol{\lambda} \geq 0, \end{cases}$$

which is computationally even easier to solve. (Note that we can handle more general label constraints as in [21], by using the appropriate matrix $\mathbf{L}$.) This is the first main result of this section.

It turns out for small n_t/n —and in particular for the unsupervised case— that the problem is often badly conditioned. To solve this, the original objective (5.12) can be slightly altered by replacing the factor $\left(\frac{1}{s_+} + \frac{1}{s_-}\right)$ with

$\left(\frac{1}{s_+ - \epsilon s} + \frac{1}{s_- - \epsilon s}\right)$. This will give rise to solutions that are slightly more biased towards balanced partitionings. The primal and dual formulations above can be changed according to this modification by simply multiplying s by $1 - 2\epsilon$. In practice, $\epsilon = 0.1$ seems to suffice, corresponding to disallowing partitions with one of s_+ and s_- smaller than $0.1s$, thus having a minor effect on the quality of the result.

Note that the methods thus obtained are fully automatic and require no parameter-tuning whatsoever. In conclusion, this approach is very suitable for transduction, and easily allows to handle data sets containing more than 500 unlabeled samples and a much more labeled samples on a 2GHz pentium computer with 0.5Gb RAM. For larger data sets further approximations will have to be made. This is the subject of the following subsection.

A fast approximation

In practice, if we assume the sample is drawn randomly from the population, we can estimate the imbalance parameter q from the training data if n_t is large enough. Then, we can fix q or equivalently $s_+ - s_-$ in (5.13) to its estimate. (Of course, one could also try several values for the imbalance, thus essentially performing a line search for this one parameter.)

If we do this, we can make the following approximation. Assuming that the spectral transduction method performs well, we know that the label vector will be close to the space spanned by the eigenvectors corresponding to the d smallest eigenvalues, stored in the columns of the matrix $\mathbf{V} \in \mathbb{R}^{n \times d}$. Then, we can approximate $\mathbf{\Gamma}$ in (5.17) by $\mathbf{\Gamma} \approx \mathbf{V}\mathbf{M}\mathbf{V}'$. Without derivational details, we state the thus obtained approximated SDP relaxation:

$$\mathbf{P}_{\mathbf{SDP3}}: \quad \begin{cases} \max\limits_{\mathbf{M}} & \langle \mathbf{M}, \mathbf{V}'\mathbf{A}\mathbf{V} \rangle \\ \text{s.t.} & \mathbf{M} \succeq \mathbf{0} \\ & \operatorname{diag}(\mathbf{V}\mathbf{M}\mathbf{V}') \leq \mathbf{1} \\ & \langle \mathbf{M}, \mathbf{V}'\mathbf{d}\mathbf{d}'\mathbf{V} \rangle \leq (s_+ - s_-)^2, \end{cases}$$

$$\mathbf{D}_{\mathbf{SDP3}}: \quad \begin{cases} \min\limits_{\boldsymbol{\lambda}, \mu} & \mathbf{1}'\boldsymbol{\lambda} + (s_+ - s_-)^2 \mu \\ \text{s.t.} & \mathbf{V}'\mathbf{A}\mathbf{V} - \mathbf{V}'\operatorname{diag}(\boldsymbol{\lambda})\mathbf{V} - \mu \mathbf{V}'\mathbf{d}\mathbf{d}'\mathbf{V} \preceq \mathbf{0} \\ & \boldsymbol{\lambda} \geq \mathbf{0}, \\ & \mu \geq 0. \end{cases}$$

Depending on whether the matrix $\mathbf{V}$ is obtained using a standard or using a transductive spectral relaxation of NCut as in [21], this allows one to do approximate NCut clustering or transduction respectively. Note that the number of primal variables as well as the size of the primal and dual constraints can be drastically reduced by taking d small enough. We will see that this is often possible in practice. Using this formulation, data sets of several thousands of samples can be dealt with. This is the second main result of this section.

5.3.4 Empirical results

All SDP optimization problems are implemented using SeDuMi [107]. The algorithms described here compute a good label matrix $\mathbf{\Gamma}$. To find an approximate label vector $\mathbf{y}$ from $\mathbf{\Gamma}$ one can use several techniques, including thresholding its dominant eigenvector, or simply picking a column corresponding to one of the training samples (which is what we will do in this paper). Other techniques are described in literature [49].

First experiment We show empirical experiments on a data set extracted from the Swiss constitution as described in Section 2.1.3. The data set contains 195 articles in each of 4 languages language (so $n = 780$), which are organized in so-called Titles. The affinity matrix used here is the bag of words kernel after stop word removal and stemming. More information can be found in [15]. To demonstrate the ability of the system to exploit partial label information, we try two different partitions: first, all English plus French articles are classified versus all German plus Italian articles; second, all articles in the largest of the 7 Titles, versus all other articles, no matter what language (note: the size of the Title is not directly reflected in the articles themselves). Figure 5.2, left panel, shows the error rates for both problems for increasing n_t with fixed n, showing that label information is effectively exploited to find the right bipartitioning of the data. Obviously, more natural splits should require less labels. Indeed, as one could expect, a small n_t already gives a good performance for the language partition, while the partitioning by Title needs much more label information.

Second experiment Here we use the test set of the USPS data set: the positive class contains all digits from 0 through 4, the negative class contains the other digits. Since the number of samples in this data set ($n = 2007$) is too large for the unapproximated relaxation to be practically solvable, we resort to the approximated method $\mathbf{P/D_{SDP3}}$. The ROC-score as a function of the dimensionality d is shown in figure 5.2, right panel, and this for three different sizes n_t of the training set. A 20-nearest neighbor affinity matrix is used (with nearest in the Euclidian sense). For comparison, the performance of the method described in [64], shown by the authors to operate well on nearest neighbor affinity matrices, is shown as well. From the figure, we can conclude that generally a better performance is achieved for larger n_t and d. Furthermore, for small n_t the method performs significantly better than the approach proposed in [64]. For larger n_t, the performance seems to be slightly worse.

5.4 Conclusions

We discussed different approaches to transduction. In a first section, we proposed two methods to modify standard kernel based induction methods to incorporate the additional knowledge captured by the working set. As such, the

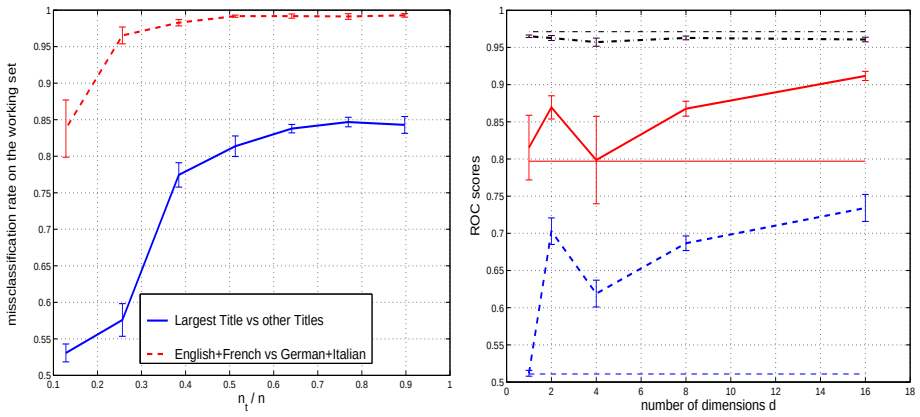


Figure 5.2: The left picture shows the average working set error over 10 randomizations for the two Swiss constitution experiments. Method $\mathbf{P/D_{SDP2}}$ was used. Clearly, the larger the training set size n_t, the better the performance. In the right picture, method $\mathbf{P/D_{SDP3}}$ was used on the USPS test set, classifying the digits up to 4 in one class, the other 5 digits in the other class. This transduction task was carried out for 3 different amounts of labeled samples n_t, respectively 1% (dashed), 5% (full) and 20% (dash-dotted) of the total sample size n. The picture shows average ROC-scores on the working set, as a function of the dimensionality d used in the approximation. For comparison, the fainter straight horizontal lines give the average scores obtained by the method described in [64].

computational complexity of transduction is not much different from induction, and the combinatorial problem is avoided.

In the second and last sections, we discussed relaxations of actual transduction formulations, that are combinatorial in nature. Two different cost functions have been dealt with: the inverse margin (as in SVM's) and the normalized cut cost. Whereas in both these cases only the transduction setting was worked out, generalizations towards side-information learning as in section 3.2 are easily conceivable.

While it is possible to show positive empirical results for each of these methods, it is quite impossible to compare them among each other. They all have their own strengths, be it of computational nature or because of a superior empirical performance.

As a guideline, we can say that methods like the ones described in the first section are preferable when the sample size is large and when computation time is an issue: true-sense transductive methods are, even after convex relaxation of the associated combinatorial problem, often computationally too hard to scale up to 5000 samples or more. On the other hand, for the smallest sample sizes, the transductive SVM is probably the best. For mid size data sets, the method presented in the last section is probably the option of choice. However, it is difficult to give general guidelines, and the choice of the method should be based on the data at hand.

Further work Recently, a kernel has been proposed [39] for clustering that makes samples more similar when they can be connected by a path of distances that are separated one from another by a small distance. The advantage of this approach is that clusters can have arbitrary shapes, as long as the samples are interconnected. Interestingly, this kernel can be computed in $O(n^2 log(n))$ time. It would be interesting to consider the use of this kernel (and other shortest-path based kernels) in a transduction setting.

Chapter 6

Convex optimization for kernel learning: a bioinformatics example

In this chapter, we will shortly review some recent theoretical developments in literature [71] concerning kernel learning (section 6.1.1), for completeness. In Section 6.2 we report the results of an application of this method in bioinformatics we have been involved in (published in [70]).

6.1 Optimizing the kernel

A nice feature of kernel based methods is their modularity: kernel methods can be decomposed into 2 disjunct steps, namely the kernel design and the algorithm. This decomposition is possible thanks to a specification of the kernel matrix as a semi positive definite (SPD) matrix, which is all that is needed for kernel based methods to operate on it.

We have seen the usefulness of SDP in the previous 2 chapters. The fact that the kernel matrix is SPD brings to light another class of applications of SDP: optimizing the kernel, subject to the constraint that it is SPD [71].

Specifically, suppose a number k of kernel matrices $\mathbf{K}_i, i = 1, \ldots, k$ are known, each reflecting a different aspect of the n samples. Then, we can construct a new kernel

$$\mathbf{K} = \sum_i \mu_i \mathbf{K}_i$$

that is a linear combination of the individual kernels, and optimize some cost function over the weights μ_i subject to the constraint $\mathbf{K} \succeq \mathbf{0}$. As long as $\mathbf{K}$ is SPD, it is a valid kernel to be used in kernel methods.

6.1.1 The optimization problem

Now, what should be the cost function? Remember that the optimal SVM object function equals the inverse margin (plus an error reflecting the training error), for a given kernel matrix $\mathbf{K}$ with a fixed trace:

$$\min_{\mathbf{w},b,\xi_i} \frac{1}{2}\mathbf{w}'\mathbf{w} + C\sum_i \xi_i \;\text{ s.t. }\; \xi_i \geq 0 \;\text{ and }\; y_i(\mathbf{w}'\mathbf{x}_i + b) \geq 1 + \xi_i$$

or the dual:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & 2\boldsymbol{\alpha}'\mathbf{1} + \boldsymbol{\alpha}'(\mathbf{K} \odot \mathbf{y}\mathbf{y}')\boldsymbol{\alpha} \\ \text{s.t.} \quad & C \geq \alpha_i \geq 0 \\ & \boldsymbol{\alpha}'\mathbf{y} = 0. \end{aligned}$$

The smaller this optimum (or equivalently the larger the margin), the better a generalization bound can be found. Therefore, we could in fact minimize this quantity with respect to the kernel parameters μ_i, *while keeping* $\mathbf{K}$ *SPD*:[1]

$$\begin{aligned} \min_{\mu_i} \quad \max_{\boldsymbol{\alpha}} \quad & 2\boldsymbol{\alpha}'\mathbf{1} + \boldsymbol{\alpha}'(\mathbf{K} \odot \mathbf{y}\mathbf{y}')\boldsymbol{\alpha} \\ \text{s.t.} \quad & C \geq \alpha_i \geq 0 \\ & \boldsymbol{\alpha}'\mathbf{y} = 0 \\ & \mathbf{K} = \sum_i \mu_i \mathbf{K}_i \\ & \mathbf{K} \succeq \mathbf{0} \\ & \mathrm{tr}(\mathbf{K}) = 1. \end{aligned}$$

This optimization problem can be reformulated as a standard SDP problem [71].

Furthermore, in order to make it computationally easier to solve, we can modify this optimization problem by constraining all weights μ_i to be positive. Since a convex combination of SPD matrices $\mathbf{K}_i$ is SPD itself, then the (rather expensive) SPD constraint can be omitted. The result is a convex quadratically constrained quadratic programming (QCQP) problem which can be solved efficiently.

6.1.2 Corollary: a convex method for tuning the soft-margin parameter

Interesting to note is that a 2-norm soft margin SVM is equivalent to a hard margin SVM where the kernel $\mathbf{K}$ is replaced by a kernel $\mathbf{K} + \gamma\mathbf{I}$. Thus, using the above methodology, in principle the value for γ optimizing a learning theory bound can be learned using convex optimization. While this is a personal contribution, we will not go into the details here, as it would distract the reader from the other ideas in this chapter. For more information we refer to a technical report [17].

[1] Here we equated $\mathrm{tr}(\mathbf{K}) = 1$, but any other constant would do.

Table 6.1: Kernel functions. The table lists the seven kernels used to compare proteins, the data on which they are defined, and the method for computing similarities. The final kernel, K_{RND}, is included as a control. All kernel matrices, along with the data from which they were generated, are available at `noble.gs.washington.edu/proj/sdp-svm`.

Kernel	Data	Similarity measure
$\mathbf{K}_{SW}$	protein sequences	Smith-Waterman
$\mathbf{K}_{B}$	protein sequences	BLAST
$\mathbf{K}_{Pfam}$	protein sequences	Pfam HMM
$\mathbf{K}_{FFT}$	hydropathy profile	FFT
$\mathbf{K}_{LI}$	protein interactions	linear kernel
$\mathbf{K}_{D}$	protein interactions	diffusion kernel
$\mathbf{K}_{E}$	gene expression	radial basis kernel
$\mathbf{K}_{RND}$	random numbers	linear kernel

6.2 A case study in bioinformatics

In this section we discuss a case study in bioinformatics that involves kernel optimization as described above. Concretely, we performed experiments using yeast genome-wide data sets, including amino acid sequences, hydropathy profiles, gene expression data and known protein-protein interactions, each giving rise to one or more of the individual kernel matrices $\mathbf{K}_i$. Two classification tasks are solved, one concerning the classification of proteins as membrane proteins versus non-membrane proteins, the other one classifying proteins as ribosomal proteins or not [70].

6.2.1 The individual kernels

As said, several points of view on the genes are offered by different information sources. In total 7 different kernels have been used, as summarized in Table 6.1. Below we will provide a description of each, and the rationale behind their design. Each of these kernels captures some aspect of the protein, and measures the similarity between proteins according to this specific aspect.

Protein sequence: Smith-Waterman, BLAST and Pfam HMM kernels. A homolog of a membrane protein is likely also to be located in the membrane, and similarly for the ribosome. Therefore, we define three kernel matrices based upon standard homology detection methods. The first two sequence-based kernel matrices ($\mathbf{K}_{SW}$ and $\mathbf{K}_{B}$) are generated using the BLAST [3] and Smith-Waterman (SW) [104] pairwise sequence comparison algorithms, as described previously [76]. Both algorithms use gap opening and extension penalties of 11 and 1, and the BLOSUM 62 matrix. Because matrices of BLAST or Smith-Waterman scores are not necessarily positive semidefinite, we represent each protein as a vector of scores against all other proteins. Defining the

similarity between proteins as the inner product between the score vectors (the so-called *empirical kernel map* [113]) leads to valid kernel matrices, one for the BLAST score and one for the SW score. Note that including in the comparison set proteins with unknown labels allows the kernel to exploit this unlabeled data. The third kernel matrix ($\mathbf{K}_{Pfam}$) is a generalization of the previous pairwise comparison-based matrices in which the pairwise comparison scores are replaced by expectation values derived from hidden Markov models in the Pfam database [105].

Protein sequence: FFT kernel. The fourth sequence-based kernel matrix ($\mathbf{K}_{FFT}$) is specific to the membrane protein recognition task. This kernel directly incorporates information about hydrophobicity patterns, which are known to be useful in identifying membrane proteins. Generally, each membrane protein passes through the membrane several times. The transmembrane regions of the amino acid sequence are typically hydrophobic, whereas the non-membrane portions are hydrophilic. This specific hydrophobicity profile of the protein allows it to anchor itself in the cell membrane. Because the hydrophobicity profile of a membrane protein is critical to its function, this profile is better conserved in evolution than the specific amino acid sequence. Therefore, classical methods for determining whether a protein $\mathbf{p}_i$ (consisting of $|\mathbf{p}_i|$ amino acids, by definition) spans a membrane [30], depend upon its *hydropathy profile* $h(\mathbf{p}_i) \in \mathbb{R}^{|\mathbf{p}_i|}$: a vector containing the hydrophobicities of the amino acids along the protein [37, 22, 53]. The FFT kernel uses hydropathy profiles generated from the Kyte-Doolittle index [69]. This kernel compares the frequency content of the hydropathy profiles of the two proteins. First, the hydropathy profiles are pre-filtered with a low-pass filter to reduce noise:

$$h_f(\mathbf{p}_i) = f \otimes h(\mathbf{p}_i)$$

where $f = \frac{1}{4}(1\ 2\ 1)$ is the impulse response of the filter and $\otimes$ denotes convolution with that filter. After pre-filtering the hydropathy profiles (and if necessary appending zeros to make them equal in length—a commonly used technique not altering the frequency content), their frequency contents are computed with the Fast Fourier Transform (FFT) algorithm:

$$H_f(\mathbf{p}_i) = FFT(h_f(\mathbf{p}_i)).$$

The FFT kernel between proteins $\mathbf{p}_i$ and $\mathbf{p}_j$ is then obtained by applying a Gaussian kernel function to the frequency contents of their hydropathy profiles:

$$\mathbf{K}_{FFT}(\mathbf{p}_i, \mathbf{p}_j) = \exp(-||H_f(\mathbf{p}_i) - H_f(\mathbf{p}_j)||^2/2\sigma)$$

with width $\sigma = 10$. This kernel detects periodicities in the hydropathy profile, a feature that is relevant to the identification of membrane proteins and complementary to the previous, homology-based kernels.

Protein interactions: linear and diffusion kernels. For the recognition of ribosomal proteins, protein-protein interactions are clearly informative, since all ribosomal proteins interact with other ribosomal proteins. For membrane protein recognition, we expect information about protein-protein interactions to be informative for two reasons. First, hydrophobic molecules or regions of molecules are probably more likely to interact with each other than with hydrophilic molecules or regions. Second, transmembrane proteins are often involved in signaling pathways, and therefore different membrane proteins are likely to interact with a similar class of molecules upstream and downstream in these pathways (e.g., hormones upstream or kinases downstream). The two protein interaction kernels are generated using medium- and high-confidence interactions from a database of known interactions [119]. These interactions can be represented as an interaction matrix, in which rows and columns correspond to proteins, and binary entries indicate whether the two proteins interact.

The first interaction kernel matrix ($\mathbf{K}_{LI}$) is comprised of linear interactions, i.e., inner products of rows and columns from the centered, binary interaction matrix. The more similar the interaction pattern (corresponding to a row or column from the interaction matrix) is for a pair of proteins, the larger the inner product will be.

An alternative way to represent the same interaction data is to consider the proteins as nodes in a large graph. In this graph, two proteins are linked when they interact and otherwise not. In [67] one proposes a general method for establishing similarities between the nodes of a graph, based on a random walk on the graph. This method efficiently accounts for all possible paths connecting two nodes, and for the lengths of those paths. Nodes that are connected by shorter paths or by many paths are considered more similar. The resulting *diffusion kernel* generates the second interaction kernel matrix ($\mathbf{K}_D$).

An appealing characteristic of the diffusion kernel is its ability, like the empirical kernel map, to exploit unlabeled data. In order to compute the diffusion kernel, a graph is constructed using all known protein-protein interactions, including interactions involving proteins whose subcellular locations are unknown. Therefore, the diffusion process includes interactions involving unlabeled proteins, even though the kernel matrix only contains entries for labeled proteins. This allows two labeled proteins to be considered close to one another if they both interact with an unlabeled protein.

Gene expression: radial basis kernel. Finally, we also include a kernel constructed entirely from microarray gene expression measurements. A collection of 441 distinct experiments was downloaded from the Stanford Microarray Database (`genome-www.stanford.edu/microarray`). This data provides us with a 441-element expression vector characterizing each gene. A Gaussian kernel matrix ($\mathbf{K}_E$) is computed from these vectors by applying a Gaussian kernel function with width $\sigma = 100$ to each pair of 441-element vectors, characterizing a pair of genes. Gene expression data is expected to be useful for recognizing ribosomal proteins, since their expression signatures are known to

be highly correlated with one another. We do not expect that gene expression will be particularly useful for the membrane classification task. We do not need to eliminate the kernel *a priori*, however; as explained in the following section, our method is able to provide an *a posteriori* measure of how useful a data source is relative to the other sources of data.

6.2.2 Experimental Design

In order to test the kernel-based approach (further referred to as *SDP/SVM*) in the setting of yeast protein classification, we use as a gold standard the annotations provided by the MIPS Comprehensive Yeast Genome Database (CYGD) [81]. The CYGD assigns 1125 yeast proteins to particular complexes, of which 138 participate in the ribosome. The remaining approximately 5000 yeast proteins are unlabeled. Similarly, CYGD assigns subcellular locations to 2318 yeast proteins, of which 497 belong to various membrane protein classes, leaving approximately 4000 yeast proteins with uncertain location.

The primary input to the classification algorithm is a collection of kernel matrices from Table 6.1. For membrane protein classification, for comparison with the SDP/SVM learning algorithm, we consider several classical biological methods that are commonly used to determine whether a Kyte-Doolittle plot corresponds to a membrane protein, as well as a state-of-the-art technique using hidden Markov models (HMMs) to predict transmembrane helices in proteins [68]. The first method relies on the observation that the average hydrophobicity of membrane proteins tends to be higher than that of non-membrane proteins, because the transmembrane regions are more hydrophobic. We therefore define f_1 as the average hydrophobicity, normalized by the length of the protein. We will compare the classification performance of our statistical learning algorithm with this metric.

Clearly, however, f_1 is too simplistic. For example, protein regions that are not transmembrane only induce noise in f_1. Therefore, an alternative metric filters the hydrophobicity plot with a low-pass filter and then computes the number, the height and the width of those peaks above a certain threshold [30]. The filter is intended to smooth out periodic effects. We implement two such filters, choosing values for the filter order and the threshold based on [30]. In particular, we define f_2 as the area under the 7th-order low-pass filtered Kyte-Doolittle plot and above a threshold value 2, normalized by the length of the protein. Similarly, f_3 is the corresponding area using a 20th-order filter and a threshold of 1.6.

Finally, the Transmembrane HMM (TMHMM) web server (`www.cbs.dtu.dk/services/TMHMM`) is used to make predictions for each protein. In [68], transmembrane proteins are identified by TMHMM using three different metrics: the expected number of amino acids in transmembrane helices, the number of transmembrane helices predicted by the N-best algorithm, and the expected number of transmembrane helices. Only the first two of these metrics are provided in the TMHMM output. Accordingly, we produce two lists of proteins, ranked

by the number of predicted transmembrane helices (T_{PH}) and by the expected number of residues in transmembrane helices (T_{ENR}).

Each algorithm's performance is measured by randomly splitting the data (without stratifying) into a training and test set in a ratio of 80/20. We report the receiver operating characteristic (ROC) score, which is the area under a curve that plots true positive rate as a function of false positive rate for differing classification thresholds [46, 44]. The ROC score measures the overall quality of the ranking induced by the classifier, rather than the quality of a single point in that ranking. An ROC score of 0.5 corresponds to random guessing, and an ROC score of 1.0 implies that the algorithm succeeded in putting all of the positive examples before all of the negatives. In addition, we select the point on the ROC curve that yields a 1% false positive rate, and we report the rate of true positives at this point (TP1FP). Each experiment is repeated 30 times with different random splits in order to estimate the variance of the performance values.

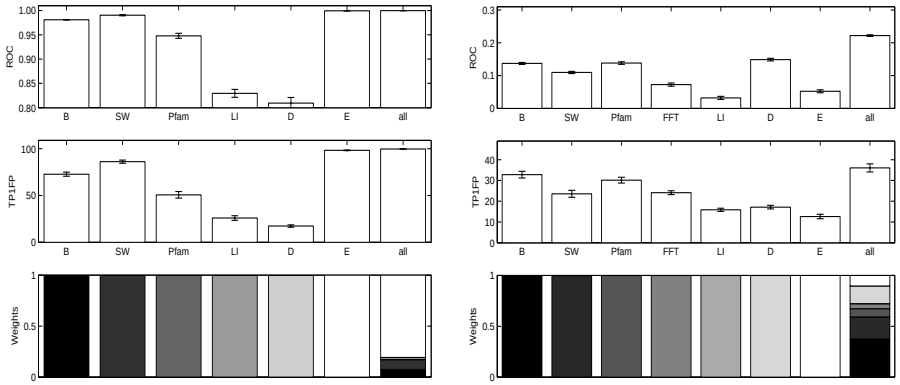
6.2.3 Results

We performed computational experiments that study the performance of the SDP/SVM approach as a function of the number of data sources, compare the approach to a simpler approach using an unweighted combination of kernels, study the robustness of the method to the presence of noise, and for membrane protein classification, compare the performance of the method to classical biological methods and state-of-the-art techniques for membrane protein classification.

Ribosomal Protein Classification

Figure 6.1(A) shows the results of training an SVM to recognize the cytoplasmic ribosomal proteins, using various kernel functions. Very good recognition performance can be achieved using several types of data individually: the Smith-Waterman kernel yields an ROC of 0.9903 and a TP1FP of 86.23%, and the gene expression kernel yields corresponding values of 0.9995 and 98.31%. However, combining all six kernels using SDP provides still better performance (ROC of 0.9998 and TP1FP of 99.71%). These differences, though small, are statistically significant according to a Bonferroni corrected Wilcoxon signed rank test.

For this task, the SDP approach performs no better than the naive approach of combining all six kernel matrices in an unweighted fashion. Note, however, that the SDP solution also provides an additional explanatory result, in the form of the weights assigned to the kernels. These weights are illustrated in Figure 6.1(A) and suggest that, as expected, the cytoplasmic ribosomal proteins are best defined by their expression profiles and, secondarily, by their sequences. An additional benefit offered by SDP over the naive approach is its robustness in the presence of noise. In order to illustrate this effect, we omit the expression kernel from the combination and add six kernels generated from Gaussian noise ($\mathbf{K}_{R1\ldots R6}$). This set of kernels degrades the performance of the



(A) Ribosomal proteins (B) Membrane proteins

Figure 6.1: Combining data sets yields better classification performance. The height of the bars in the upper two plots are proportional to the ROC score (top) and the percentage of true positives at one percent false positives (middle), for the SDP/SVM method using the given kernel. Error bars indicate standard error across 30 random train/test splits. In the lower plots, the heights of the colored bars indicate the relative weights of the different kernel matrices in the optimal linear combination.

naive combination, but has no effect on the SDP/SVM. With six additional random kernels ($\mathbf{K}_{R7...R12}$) the benefit of optimizing the weights is even more apparent (see Table 6.2 and the online supplement).

Among the 30 train/test splits, seven proteins are consistently mislabeled by SDP/SVM (see online supplement). These include one, YLR406C (RPL31B), that was previously misclassified as non-ribosomal in an SVM-based study using a smaller microarray expression data set [28]. In order to better understand the seven false negatives, we separated out the kernel-specific components of the SVM discriminant score. In nearly every case, the component corresponding to the gene expression kernel is the only one that is negative (data not shown). In other words, these seven proteins show atypical expression profiles, relative to the rest of the ribosome, which explains their misclassification by the SVM. Visual inspection of the expression matrix (online supplement) verifies these differences.

Finally, the trained SVM was applied to the set of approximately 5000 proteins that are not annotated in CYGD as participating in any protein complex. Among these, the SVM predicts that 14 belong in the cytoplasmic ribosomal class (see online supplement). However, nine of these predictions correspond to questionable ORFs, each of which lies directly opposite a gene that encodes a ribosomal protein. In these cases, the microarray expression data for the questionable ORFs undoubtedly reflect the strong pattern of expression from

Table 6.2: Classification performance on the cytoplasmic ribosomal class, in the presence of noise or improper weighting. The table lists the percentage of true positives at one percent false positives (TP1FP) and the ROC score for several combinations of kernels. The first three lines of results were obtained using SDP-SVM, and the last three lines by setting the weights uniformly. Columns 1 through 5 report the average weights for the potentially informative kernels (averaged over the training/test splits), column 6 contains the average weight for a first set of 6 random kernels (averaged over the 6 kernels and the training/test splits) and column 7 similarly for an additional set of 6 random kernels. Each random kernel was generated by computing inner products on randomly generated 400-element vectors, in which each vector component was sampled independently from a standard normal distribution. In the table, a hyphen indicates that the corresponding kernel is not considered in the combination.

$\mathbf{K}_{SW}$	$\mathbf{K}_{PF}$	$\mathbf{K}_{LI}$	$\mathbf{K}_{B}$	$\mathbf{K}_{D}$	$\mathbf{K}_{R1-6}$	$\mathbf{K}_{R7-12}$	TP1FP	ROC
5.08	0.31	0.22	0.39	0.00	–	–	$88.21 \pm 1.73\%$	0.9933 ± 0.0011
5.07	0.31	0.22	0.39	0.00	0.01	–	$88.19 \pm 1.60\%$	0.9932 ± 0.0011
5.06	0.30	0.22	0.38	0.01	0.02	0.01	$88.08 \pm 1.65\%$	0.9932 ± 0.0010
1.00	1.00	1.00	1.00	1.00	–	–	$75.20 \pm 2.38\%$	0.9906 ± 0.0012
1.00	1.00	1.00	1.00	1.00	1.00	–	$59.66 \pm 3.03\%$	0.9791 ± 0.0017
1.00	1.00	1.00	1.00	1.00	1.00	1.00	$42.87 \pm 2.59\%$	0.9620 ± 0.0027

the corresponding ribosomal genes. Among the remaining five proteins, two (YNL119W and YKL056C) were predicted to be ribosomal proteins in a previous SVM-based study [28]. YKL056C is particularly interesting: it is a highly conserved, ubiqitous protein homologous to the mammalian translationally controlled tumor protein [45] and to human IgE-dependent histamine-releasing factor.

Membrane Protein Classification

The results of the first membrane protein classification experiment are summarized in Figure 6.1(B). The plot illustrates that SDP/SVM learns significantly better from the heterogeneous data than from any single data type. The mean ROC score using all seven kernel matrices (0.9219 ± 0.0024) is significantly higher than the best ROC score using only one matrix (0.8487 ± 0.0039 using the diffusion kernel). This improvement corresponds to a change in TP1FP of 18.91%, from 17.15% to 36.06% and a change in test set accuracy of 7.36%, from 81.30% to 88.66%.

As expected, the sequence-based kernels yield good individual performance. The value of these kernels is evidenced by their corresponding ROC scores and by the relatively large weights assigned to the sequence-based kernels by the SDP. These weights are as follows: $\mu_B = 2.62$, $\mu_{SW} = 1.52$, $\mu_{Pfam} = 0.57$, $\mu_{FFT} = 0.35$, $\mu_{LI} = 0.01$, $\mu_D = 1.21$ and $\mu_E = 0.73$.[2] Thus, two of the three kernel matrices that receive weights larger than 1 are derived from the amino acid sequence.

The results also show that the interaction-based diffusion kernel is more informative than the expression kernel. The diffusion kernel yields an individual ROC score which is significantly higher than the expression kernel, and the SDP also assigns a larger weight to the diffusion kernel (1.21) than to the expression kernel (0.73). Accordingly, removing the diffusion kernel reduces the percentage true positives at one percent false positives from 36.06% to 34.52%, whereas removing the expression kernel has a smaller effect, leading to a TP1FP of 35.88%. Further description of the results obtained when various subsets of kernels are used is provided in the online supplement.

In order to test the robustness of our approach, we performed a second experiment using four real kernels—$\mathbf{K}_B, K_{SW}, K_D$, and $\mathbf{K}_E$—and four Gaussian noise kernels $\mathbf{K}_{R1\ldots R4}$. Using all eight kernels, SDP assigns values to the random kernels weights that are close to zero. Therefore, the overall performance, as measured by TP1FP or ROC score, remains virtually unchanged. In contrast, the performance of the uniformly weighted kernel combination, which was previously competitive with the SDP combination, degrades significantly in the presence of noise, from TP1FP of 33.87% down to 26.24%. Thus, the SDP approach provides a kind of insurance against the inclusion of noisy or irrelevant kernels.

[2] For ease of interpretation, we scale the weights such that their sum is equal to the number m of kernel matrices.

Table 6.3: Classification performance on the membrane proteins, in the presence of noise or improper weighting. The table lists the percentage true positives at one percent false positives (TP1FP) and the ROC score for several combinations of kernels. The first two lines of results were obtained using SDP-SVM, and the last two lines were obtained using a uniform kernel weighting. Columns 1 through 8 report the average weights for the respective kernels (averaged over the training/test splits). A hyphen indicates that the corresponding kernel is not considered in the combination.

$\mathbf{K}_B$	$\mathbf{K}_{SW}$	$\mathbf{K}_D$	$\mathbf{K}_E$	$\mathbf{K}_{R1}$	$\mathbf{K}_{R2}$	$\mathbf{K}_{R3}$	$\mathbf{K}_{R4}$	TP1FP	ROC
1.81	1.05	0.73	0.42	–	–	–	–	$35.71 \pm 2.13\%$	0.9196 ± 0.0023
3.30	1.98	1.31	0.79	0.08	0.17	0.21	0.17	$34.14 \pm 2.09\%$	0.9145 ± 0.0026
1.00	1.00	1.00	1.00	–	–	–	–	$33.87 \pm 2.20\%$	0.9180 ± 0.0026
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	$26.24 \pm 1.39\%$	0.8627 ± 0.0033

Table 6.4: Comparison of membrane protein recognition methods. Each row in the table corresponds to one of the membrane protein recognition methods described in the text: three methods that apply filters directly to the hydrophobicity profile, two methods based upon the TMHMM model, and the SDP/SVM approach. For each method, the ROC and TP1FP are reported.

Method	ROC	TP1FP
f_1	0.7345	16.70%
f_2	0.7504	13.48%
f_3	0.7879	21.93%
T_{PH}	0.7362	30.02%
T_{ENR}	0.8018	31.38%
SDP/SVM	0.9219	36.06%

We also compared the membrane protein classification performance of the SDP/SVM method with that of several other techniques for membrane protein classification. The ROC and TP1FP for these methods are listed in Table 6.4. The results indicate that using learning in this context dramatically improves the results relative to the simple hydropathy profile approach. The SDP/SVM method also improves, though to a lesser degree, upon the performance of the state-of-the-art TMHMM model. However, the comparison to TMHMM is somewhat problematic, for several reasons. First, TMHMM is provided as a pre-trained model. As such, a cross-validated comparison with the SDP/SVM is not possible. In particular, some members of the cross-validation test sets were almost certainly used in training TMHMM, making its performance estimate too optimistic. On the other hand, TMHMM aims to predict membrane protein topology across many different genomes, rather than in a yeast-specific fashion. Despite these difficulties, the results in Table 6.4 are interesting because they suggest that an approach that exploits multiple genome-wide data sets may provide better membrane protein recognition performance than a sequence-specific approach.

6.2.4 Discussion

In this section we have described a general method for combining heterogeneous genome-wide data sets in the setting of kernel-based statistical learning algorithms, and we have demonstrated an application of this method to the problems of classifying yeast ribosomal and membrane proteins. The performance of the resulting SDP/SVM algorithm improves upon the SVM trained on any single data set or trained using a naive combination of kernels. Moreover, the SDP/SVM algorithm's performance consistently improves as additional genome-wide data sets are added to the kernel representation and is robust in the presence of noise.

In [128] a kernel-based approach to data fusion that is complementary to the SDP/SVM method is presented. In their approach, canonical correlation analysis (CCA) (see Section 2.2.2 of this thesis) is used to select features from the space defined by a second kernel, and can be generalized to operate with more than two kernels. Thus, whereas the SDP approach combines different sources into a joint representation, kernel CCA separates components of a single kernel matrix, identifying the most relevant ones.

SDP is viewed as a tractable instance of general convex programming, because it is known to be solvable in polynomial time, whereas general convex programs need not be [84]. In practice, however, there are important computational issues that must be faced in any implementation. In particular, our application requires the formation and manipulation of $n \times n$ kernel matrices. For genome-scale data, such matrices are large, and naive implementation can create serious demands on memory resources. However, kernel matrices often have special properties that can be exploited by more sophisticated implementations. In particular, it is possible to prove that certain kernels necessarily

lead to low-rank kernel matrices, and indeed low-rank matrices are also often encountered in practice [123]. Methods such as incomplete Cholesky decomposition can be used to find low-rank approximations of such matrices, without even forming the full kernel matrix, and these methods have been used successfully in implementations of other kernel methods [6, 38]. Time complexity is another concern. The worst-case complexity of the SDP approach (where weights are allowed to be negative) is $O(n^{4.5})$ [71], although it can be solved in $O(n^3)$, as a QCQP, constraining the weights to be positive. In practice, however, this complexity bound is not necessarily reached by any given class of problem, and indeed time complexity has been less of a concern than space complexity in our work thus far. Moreover, the low-rank approximation tools may also provide some help with regards to time complexity. Nonetheless, running time issues are a concern for deployment of our approach with higher eukaryotic genomes, and new implementational strategies may be needed.

Kernel-based statistical learning methods have a number of general virtues as tools for biological data analysis. First, the kernel framework accommodates not only the vectorial and matrix data that are familiar in classical statistical analysis, but also more exotic data types such as strings, trees, graphs and text. The ability to handle such data is clearly essential in the biological domain. Second, kernels provide significant opportunities for the incorporation of more specific biological knowledge, as we have seen with the FFT kernel and the Pfam kernel. Third, the growing suite of kernel-based data analysis algorithms require only that data be reduced to a kernel matrix; this creates opportunities for standardization. Finally, as we have shown here, the reduction of heterogeneous data types to the common format of kernel matrices allows the development of general tools for combining multiple data types. Kernel matrices are required only to respect the constraint of positive semidefiniteness, and thus the powerful technique of semidefinite programming can be exploited to derive general procedures for combining data of heterogeneous format and origin.

We thus envision the development of general libraries of kernel matrices for biological data, such as those that we have provided at `noble.gs.washington.edu/proj/sdp-svm`, that summarize the statistically-relevant features of primary data, encapsulate biological knowledge, and serve as inputs to a wide variety of subsequent data analyses. Indeed, given the appropriate kernel matrices, the methods that we have described here are applicable to problems such as the prediction of protein metabolic, regulatory and other functional classes, the prediction of protein subcellular locations, and the prediction of protein-protein interactions.

Finally, while we have focused on the binary classification problem in the current paper, there are many possible extensions of our work to other statistical learning problems. One notable example is the problem of *transduction*, in which the classifier is told *a priori* the identity of the points that are in the test set (but not their labels). This approach can deliver superior predictive performance [115], and would seem particularly appropriate in gene or protein classification problems, where the entities to be classified are often known *a priori*.

6.3 Conclusions

In this chapter, we have demonstrated the power of convex optimization in real life problems, and specifically in a genome wide bioinformatics classification problem. Apart from this demonstration, we reported a theoretical result in the field of kernel optimization, namely the use of the kernel learning approach for the automatic tuning of the regularization parameter. Further work remains to show how well this approach performs in practice.

Part III

Further topics

Chapter 7

On regularization, canonical correlation analysis, and beyond

In this chapter (published in [20]), we present a unifying approach to canonical correlation analysis (CCA) [58] and RR [51] from the viewpoint of estimation, and from the viewpoint of regularization in order to deal with noise. The goal of the research that led to this chapter was purely theoretical, to gain insight in the type of regularization commonly used for CCA.

Since the chapter is different in nature from the previous chapters (the current chapter representing a theoretical study of an existing algorithm, while the emphasis in the previous chapters is on novel algorithmic results), it is put right here at the end of this thesis.

7.1 Regularization: what and why?

Regularization can be seen in different ways, one of which is to deal with numerical problems in inverse problems, that often originate due to finiteness of sample sizes, thus leading to inaccurate estimates of the process parameters used for the estimation problem.

Another point of view, and probably more relevant, focuses on the fact that learning is subject to overfitting, if the number of degrees of freedom is too large. A learned hypothesis can only generalize towards new examples, if the hypothesis space is small enough, so that each hypothesis can be falsified easily enough. This is often achieved by imposing a Bayesian prior on the solution, thus reducing the effective number of degrees of freedom.

A third and last way to understand regularization, is by considering the observed data as data corrupted by noise, and performing the estimation problem in a robust way, so that the influence of noise is as small as possible in a specific way. In a first subsection, we will show how this viewpoint gives rise to a natural interpretation of the regularization of LSR towards RR.

Given this interpretation of RR, elucidating the parallel between LSR and CCA will lead to a new interpretation of RCCA as described in [117], [87] and [6]. This is done in a following subsection.

A fundamental property of the approach we adopt, is the fact that we assume an underlying generative model for the data. However, this does not restrict the applicability of CCA, but rather gives an alternative interpretation, complementary to the usual interpretation in geometrical terms like correlation properties.

7.2 From least squares regression to ridge regression

Based on a very simple linear model, with two noise sources, we will review how it is possible to derive the least squares estimator as a maximum likelihood estimator (and thus the maximizer of the log likelihood). Subsequently, we will show how RR [51] naturally follows as a maximizer of the *expected* log likelihood. The expectation is carried out over all possible values of a noise source. An interpretation of the results will be provided.

Specific notation $\mathbf{X}, \mathbf{D} \in \mathbb{R}^{n \times d}$ contain n d-dimensional samples $\mathbf{x}_i, \mathbf{d}_i$ as its rows. The column vectors $\mathbf{y}, \mathbf{n} \in \mathbb{R}^n$ contain n samples of the scalars y_i, n_i. The (column) vector $\mathbf{w} \in \mathbb{R}^d$ is a d-dimensional weight vector. The sample covariance matrices $\mathbf{C_{XX}}$ and $\mathbf{C_{Xy}}$ are defined as $\mathbf{C_{XX}} = \frac{\mathbf{X'X}}{n}$ and $\mathbf{C_{Xy}} = \frac{\mathbf{X'y}}{n}$.

7.2.1 Least squares as a maximum likelihood estimator

We will briefly restate the following well known result on linear regression:

Theorem 7.1 *The maximum likelihood estimator of* $\mathbf{w}$ *given* $\mathbf{X}$ *and* $\mathbf{y}$ *and the model*

$$\mathbf{y} = \mathbf{Xw} + \mathbf{n} \tag{7.1}$$

where $\mathbf{n}$ *is iid gaussian noise with zero mean and variance* $\mathcal{E}\{\frac{\mathbf{nn}'}{n}\} = \sigma^2\mathbf{I}$*, is given by the least squares estimator*

$$\mathbf{w}_{LS} = \mathbf{C_{XX}}^{-1}\mathbf{C_{Xy}} \tag{7.2}$$

Proof The probability to observe the data, given $\mathbf{w}$, is

$$\begin{aligned} P(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2) &\propto P(\mathbf{n}|\sigma^2) \\ &\propto \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(\mathbf{y}-\mathbf{X}\mathbf{w})'(\mathbf{y}-\mathbf{X}\mathbf{w})}{2\sigma^2}\right] \end{aligned}$$

In order to maximize this with respect to $\mathbf{w}$, we can minimize minus two times the log likelihood as well. Differentiating this with respect to $\mathbf{w}$ and equating to zero leads the optimality conditions on $\mathbf{w}$ (often called normal equations). Since $\mathbf{X}'\mathbf{X}$ is positive definite, the optimum $\mathbf{w}_{\mathrm{LS}}$ corresponds to the maximum of the likelihood:

$$2\mathbf{X}'\mathbf{X}\mathbf{w}_{\mathrm{LS}} - 2\mathbf{X}'\mathbf{y} = \mathbf{0}$$

and thus

$$\mathbf{w}_{\mathrm{LS}} = (\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{y}) = \mathbf{C}_{\mathbf{X}\mathbf{X}}{}^{-1}\mathbf{C}_{\mathbf{X}\mathbf{y}}.$$

This completes the proof. ∎

7.2.2 Ridge regression as the maximizer of the expected log likelihood

In the model underlying the LSR solution, we included noise on $\mathbf{y}$. But of course, in practice, the measurements $\mathbf{X}$ are not noise free either. If we do include it, the model becomes

$$\mathbf{y} = (\mathbf{X} - \mathbf{D})\mathbf{w} + \mathbf{n}, \tag{7.3}$$

where $\mathbf{D}$ is the noise on the measurements $\mathbf{X}$. Suppose we know however the covariance matrix of $\mathbf{D}$, assuming furthermore that the rows of $\mathbf{D}$ are drawn iid from an isotropic gaussian distribution (extension towards non-isotropic gaussian distributions is straightforward):

$$\begin{aligned} \frac{\mathcal{E}\{\mathbf{D}'\mathbf{D}\}}{k} &= \lambda^2\mathbf{I} \\ P(\mathbf{D}) &= \frac{1}{\sqrt{2\pi\lambda^2}} \exp\left[-\mathrm{tr}\left(\frac{\mathbf{D}'\mathbf{D}}{2\lambda^2}\right)\right]. \end{aligned}$$

Taking minus two times the log likelihood gives (up to a constant term):

$$\begin{aligned} -2l(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2, \mathbf{D}) &= -2\log(P(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2, \mathbf{D})) \\ &= \frac{(\mathbf{y}-(\mathbf{X}-\mathbf{D})\mathbf{w})'(\mathbf{y}-(\mathbf{X}-\mathbf{D})\mathbf{w})}{2\sigma^2}. \end{aligned}$$

The maximizer of the likelihood is equal to the minimizer of this quantity. Note that $-2l(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2, \mathbf{D})$ is equal to the square loss corresponding to the weight vector $\mathbf{w}$.

Since we do not know the noise $\mathbf{D}$, this minimization of the squared loss can not be carried out in an exact way. However, we are able to average the log likelihood with respect to $\mathbf{D}$. This leads to

$$\begin{aligned} & L(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2) \\ = \; & \mathcal{E}_{\mathbf{D}}\{-2l(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2, \mathbf{D})\} \\ = \; & \int_{\mathbf{D}} \frac{(\mathbf{y} - (\mathbf{X} - \mathbf{D})\mathbf{w})'(\mathbf{y} - (\mathbf{X} - \mathbf{D})\mathbf{w})}{2\sigma^2} P(\mathbf{D}) d\mathbf{D} \\ \propto \; & \mathbf{y}'\mathbf{y} + \mathbf{w}'\mathbf{X}'\mathbf{X}\mathbf{w} + \lambda^2 \mathbf{w}'\mathbf{w} - 2\mathbf{y}'\mathbf{X}\mathbf{w} \\ \propto \; & \mathbf{C}_{\mathbf{yy}} - 2\mathbf{C}_{\mathbf{yX}}\mathbf{w} + \mathbf{w}'(\mathbf{C}_{\mathbf{XX}} + \lambda^2 \mathbf{I})\mathbf{w}. \end{aligned} \tag{7.4}$$

Differentiating this equation with respect to $\mathbf{w}$ and equating to zero, leads to the optimal value for $\mathbf{w}_{\mathrm{RR}}$

$$\mathbf{w}_{\mathrm{RR}} = (\mathbf{C}_{\mathbf{XX}} + \lambda^2 \mathbf{I})^{-1} \mathbf{C}_{\mathbf{Xy}}. \tag{7.5}$$

Note again that $\mathbf{C}_{\mathbf{XX}} + \lambda^2 \mathbf{I}$ is positive definite, so, the optimality conditions correspond to a minimum of $L(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2)$. We have thus proven the following:

Theorem 7.2 *The estimator of* $\mathbf{w}$ *that maximizes the expected log likelihood of the data, given model (7.3), is given by the ridge regression estimator* $\mathbf{w}_{RR} = (\mathbf{C}_{\mathbf{XX}} + \lambda^2 \mathbf{I})^{-1} \mathbf{C}_{\mathbf{Xy}}$.

7.2.3 Interpretation

As a first remark, we note that it is well known that by application of Jensen's inequality [32], the expected log likelihood is proved to be never larger than the log likelihood of the expectation:

$$\begin{aligned} L(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2) &\leq \log\left(E_{\mathbf{D}}\{P(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2, \mathbf{D})\}\right) \\ &= \log\left(P(\mathbf{X}, \mathbf{y}|\mathbf{w}, \sigma^2)\right). \end{aligned}$$

Optimizing this last quantity would lead to the maximum likelihood estimator of $\mathbf{w}$ in the presence of noise. Thus in fact, we are *optimizing a lower bound on the log likelihood*. However, it is clear that it is not guaranteed that the maximum likelihood estimator will yield the best performance, where performance is measured in terms of expected squared error (the quantity optimized by the ridge regression estimator).

Therefore, as a second remark, note that the sample expectation over the $(\mathbf{X}, \mathbf{y})$ samples and the expectation over $\mathbf{D}$ (as given in equation (7.4)) of the squared error is the best estimate for the expected squared error over the distributions over $\mathbf{X}$, $\mathbf{y}$ and $\mathbf{D}$. (This is due to a standard property of the sample mean of a random variable, namely that it is the best estimate of the population mean. Furthermore, the law of large numbers guarantees convergence. Note however we assume that $\mathbf{X}$, $\mathbf{y}$ and $\mathbf{D}$ are iid distributed and that there

is no prior on the distributions of $\mathbf{X}$ and $\mathbf{y}$.) Therefore, the RR estimator is the *minimum least squares error estimator* for this type of linear systems with noise.

As a third remark, we would like to point out that it is a standard result that maximum likelihood estimators are unbiased. Since the RR estimator is not a maximum likelihood estimator, we can conclude that it is *a biased estimator*.

7.2.4 Practical aspects

In general, λ will not be known. The standard approach to estimating this *hyperparameter* is applying cross validation. The cross validation score is then the squared error of the test set, evaluated for a w obtained using a training set.

So far our treatment of RR. Note that these ideas are no more than a reformulation of ideas existing in literature. We provided them for reference, and in order to motivate the development of thoughts in the next subsection on CCA.

7.3 From canonical correlation analysis to its regularized version

In contrast to regression, CCA should be catalogued under the denominator of multivariate statistics. While in regression, one searches for a direction in the $\mathbf{X}$ space that predicts the $\mathbf{y}$ space as well as possible, in doing CCA, one searches for a direction in the $\mathbf{X}$ space and one in the $\mathbf{Y}$ space (that is multidimensional as well) so that order the correlation between the projections on these directions is maximal. One can still go further, and look for directions uncorrelated with the previous ones, that, under this additional constraint, maximize the correlation between the $\mathbf{X}$ and the $\mathbf{Y}$ space.

The classical approach to CCA is using these geometrical arguments [58]. However, in this paper, we will approach CCA as an estimator of a linear system underlying the data $\mathbf{X}$ and $\mathbf{Y}$.

For conciseness and ease of notation, we will assume that the dimensions of $\mathbf{X}$ and $\mathbf{Y}$ are both equal to $\mathbf{d}$. However, most of the results are easy to extend towards different dimensions for $\mathbf{X}$ and $\mathbf{Y}$.

Specific notation In contrast to the previous subsection, we will not work with a one-dimensional $\mathbf{y}$, but with $\mathbf{Y} \in \mathbb{R}^{n \times d}$. Also, $\mathbf{C} \in \mathbb{R}^{n \times d}$. The sample covariance matrices $\mathbf{C_{YY}}$ and $\mathbf{C_{XY}} = \mathbf{C_{YX}}'$ are defined as $\mathbf{C_{YY}} = \frac{\mathbf{Y'Y}}{n}$ and $\mathbf{C_{XY}} = \frac{\mathbf{X'Y}}{n}$. $\mathbf{M_X}, \mathbf{M_Y} \in \mathbf{R}^{d \times d}$ are square mixing matrices. Furthermore, $\mathbf{W'_X} = \mathbf{M_X^{-1}}$ and $\mathbf{W'_Y} = \mathbf{M_Y^{-1}}$. These matrices contain d-dimensional weight vectors $\mathbf{w_{X},}_i$ and $\mathbf{w_{Y},}_i$ in their rows. Analogously, $\mathbf{v_{X},}_i$ and $\mathbf{v_{Y},}_i$ are the d-dimensional rows of $\mathbf{V_X}, \mathbf{V_Y} \in \mathbb{R}^{n \times d}$. The diagonal elements of the diagonal matrices $\mathbf{\Xi}, \mathbf{\Sigma}, \mathbf{\Lambda}_X, \mathbf{\Lambda}_Y \in \mathbf{R}^{d \times d}$ are $\xi_i, \sigma_i, \lambda_{X,i}$ and $\lambda_{Y,i}$.

Most of the results can be generalized towards different dimensions $n \times d_{\mathbf{X}}$ and $n \times d_{\mathbf{Y}}$ for $\mathbf{X}$ and $\mathbf{Y}$.

7.3.1 Standard geometrical approach to canonical correlation analysis

CCA solves the following optimization problem:

$$\begin{aligned} \xi_i = \quad & \max_{\mathbf{v}_{\mathbf{X},i},\mathbf{v}_{\mathbf{Y},j},\forall i,j} \quad \frac{\mathbf{v}'_{\mathbf{X},i}\mathbf{X}'\mathbf{Y}\mathbf{v}_{\mathbf{Y},i}}{\sqrt{\mathbf{v}'_{\mathbf{X},i}\mathbf{X}'\mathbf{X}\mathbf{v}_{\mathbf{X},i}\mathbf{v}'_{\mathbf{Y},i}\mathbf{Y}'\mathbf{Y}\mathbf{v}_{\mathbf{Y},i}}} \\ & \text{s.t.} \qquad \mathbf{v}_{\mathbf{X},i}\mathbf{X}'\mathbf{X}\mathbf{v}_{\mathbf{X},j} = 0 \\ & \qquad\quad \mathbf{v}'_{\mathbf{Y},j}\mathbf{Y}'\mathbf{Y}\mathbf{v}_{\mathbf{Y},i} = 0 \end{aligned} \tag{7.6}$$

where $\mathbf{v}_{\mathbf{X},i}, \mathbf{v}_{\mathbf{Y},i} \in \mathbb{R}^d$ and $i = 1, \ldots, d$.

One can show this problem reduces to solving the following generalized eigenvalue problem:

$$\begin{pmatrix} 0 & \mathbf{C}_{\mathbf{XY}} \\ \mathbf{C}_{\mathbf{YX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{v}_{\mathbf{X},i} \\ \mathbf{v}_{\mathbf{Y},i} \end{pmatrix} = \xi_i \begin{pmatrix} \mathbf{C}_{\mathbf{XX}} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{\mathbf{YY}} \end{pmatrix} \begin{pmatrix} \mathbf{v}_{\mathbf{X},i} \\ \mathbf{v}_{\mathbf{Y},i} \end{pmatrix}. \tag{7.7}$$

The generalized eigenvectors $\mathbf{v}_{\mathbf{X},i}$ with $\mathbf{v}_{\mathbf{Y},i}$ are the corresponding canonical components in both spaces, and ξ_i is the canonical correlation corresponding to $\mathbf{v}_{\mathbf{X},i}$ and $\mathbf{v}_{\mathbf{Y},i}$.

7.3.2 Canonical correlation analysis as a maximum likelihood estimator

Now, assume the following model underlying the data

$$\begin{aligned} \mathbf{X} &= (\mathbf{C} + \mathbf{N}_{\mathbf{X}})\mathbf{M}'_{\mathbf{X}} \\ \mathbf{Y} &= (\mathbf{C} + \mathbf{N}_{\mathbf{Y}})\mathbf{M}'_{\mathbf{Y}} \end{aligned} \tag{7.8}$$

and thus

$$\begin{aligned} \mathbf{X}\mathbf{W}_{\mathbf{X}} &= \mathbf{C} + \mathbf{N}_{\mathbf{X}} \\ \mathbf{Y}\mathbf{W}_{\mathbf{Y}} &= \mathbf{C} + \mathbf{N}_{\mathbf{Y}}, \end{aligned} \tag{7.9}$$

where we assume that $\mathbf{N}_{\mathbf{X}}$ and $\mathbf{N}_{\mathbf{Y}}$ are iid gaussianly distributed with covariance matrices

$$\frac{E\{\mathbf{N}'_{\mathbf{X}}\mathbf{N}_{\mathbf{X}}\}}{k} = \quad \mathbf{\Sigma}^2 \quad = \frac{E\{\mathbf{N}'_{\mathbf{Y}}\mathbf{N}_{\mathbf{Y}}\}}{k},$$

where for each $i < j$, $\sigma_i > \sigma_j > 0$. Furthermore, without loss of generality, we assume that the covariance matrix of $\mathbf{C}$ is equal to the identity. This model is much alike the models assumed by ICA algorithms (see e.g. [73]). Indeed, there it is assumed that some independent components $\mathbf{C}$ are underlying the signal $\mathbf{X}$ in a linear way, and corrupted by some noise $\mathbf{N}_{\mathbf{X}}$. The important difference here is that we assume the weaker assumption that the components of $\mathbf{C}$ are

uncorrelated instead of independent, and, importantly, $\mathbf{C}$ is expressed in two signals here, $\mathbf{X}$ and $\mathbf{Y}$ (which makes uncorrelatedness sufficient to reconstruct $\mathbf{C}$).

We are now ready to state the following theorem:

Theorem 7.3 *The generalized eigenvectors $\mathbf{V_X}$ and $\mathbf{V_Y}$ given by the generalized eigenvalue problem solved by CCA (if properly normalized) and $(\mathbf{\Xi}^{-1} - 1)$ where ξ_i are the canonical correlations, correspond to a stationary point of the likelihood function with variables $\mathbf{W_X}$, $\mathbf{W_Y}$ and $\mathbf{\Sigma}^2$ respectively, parameters in the model (7.8), given the data $\mathbf{X}$ and $\mathbf{Y}$.*

That this stationary point corresponds to a maximum, i.e. that the CCA solution is a maximum likelihood estimator of the parameters of (7.8), will be left as a conjecture here.

Proof The probability of $\mathbf{X}$ and $\mathbf{Y}$ given $\mathbf{W_X}$, $\mathbf{W_Y}$, $\mathbf{\Sigma}^2$ and $\mathbf{C}$ is given by

$$P(\mathbf{X}, \mathbf{Y}|\mathbf{W_X}, \mathbf{W_Y}, \mathbf{\Sigma}^2, \mathbf{C}) \quad = \quad P(\mathbf{X}|\mathbf{W_X}, \mathbf{\Sigma}^2, \mathbf{C}) P(\mathbf{Y}|\mathbf{W_Y}, \mathbf{\Sigma}^2, \mathbf{C}),$$

where

$$\begin{aligned} & P(\mathbf{X}|\mathbf{W_X}, \mathbf{\Sigma}^2, \mathbf{C}) \qquad (7.10) \\ = \quad & \frac{\exp\Big(-\frac{1}{2}\mathrm{tr}\big[(\mathbf{X}\mathbf{W_X} - \mathbf{C})\mathbf{\Sigma}^{-2}(\mathbf{X}\mathbf{W_X} - \mathbf{C})'\big]\Big)}{\sqrt{2\pi \det(\mathbf{W_X}\mathbf{\Sigma}^{-2}\mathbf{W_X}')^{-1}}} \\ \text{and} \quad & P(\mathbf{Y}|\mathbf{W_Y}, \mathbf{\Sigma}^2, \mathbf{C}) \qquad (7.11) \\ = \quad & \frac{\exp\Big(-\frac{1}{2}\mathrm{tr}\big[(\mathbf{Y}\mathbf{W_Y} - \mathbf{C})\mathbf{\Sigma}^{-2}(\mathbf{Y}\mathbf{W_Y} - \mathbf{C})'\big]\Big)}{\sqrt{2\pi \det(\mathbf{W_Y}\mathbf{\Sigma}^{-2}\mathbf{W_Y}')^{-1}}}. \end{aligned}$$

After some tedious but straightforward calculations, one can thus show that the evidence $P(\mathbf{X}, \mathbf{Y}|\mathbf{W_X}, \mathbf{W_Y}, \mathbf{\Sigma}^2)$ is equal to

$$\begin{aligned} & P(\mathbf{X}, \mathbf{Y}|\mathbf{W_X}, \mathbf{W_Y}, \mathbf{\Sigma}^2) \qquad (7.12) \\ = \quad & \int_C P(\mathbf{X}, \mathbf{Y}|\mathbf{W_X}, \mathbf{W_Y}, \mathbf{\Sigma}^2, \mathbf{C}) P(\mathbf{C}) d\mathbf{C} \\ \propto \quad & \sqrt{\det(\mathbf{I} + 2\mathbf{\Sigma}^{-2})} \cdot \sqrt{\det((\mathbf{W_X}\mathbf{\Sigma}^{-2}\mathbf{W_X}')(\mathbf{W_Y}\mathbf{\Sigma}^{-2}\mathbf{W_Y}'))} \cdot \\ & \exp\Bigg(-\frac{1}{2}\mathrm{tr}\Big[(\mathbf{X}\mathbf{W_X})\mathbf{\Sigma}^{-2}(\mathbf{X}\mathbf{W_X})' + (\mathbf{Y}\mathbf{W_Y})\mathbf{\Sigma}^{-2}(\mathbf{Y}\mathbf{W_Y})' \\ & -(\mathbf{X}\mathbf{W_X} + \mathbf{Y}\mathbf{W_Y}) \cdot \mathbf{\Sigma}^{-2}(\mathbf{I} + 2\mathbf{\Sigma}^{-2})^{-1}\mathbf{\Sigma}^{-2} \cdot (\mathbf{X}\mathbf{W_X} + \mathbf{Y}\mathbf{W_Y})'\Big]\Bigg). \end{aligned}$$

In order to maximize the likelihood, we can as well minimize minus 2 times the log likelihood:

$$\begin{aligned} & -2l(\mathbf{X},\mathbf{Y}|\mathbf{W_X},\mathbf{W_Y},\mathbf{\Sigma}^2) \\ = & -\log\det(\mathbf{I}+2\mathbf{\Sigma}^{-2}) - \log\det((\mathbf{W_X}\mathbf{\Sigma}^{-2}\mathbf{W_X'})(\mathbf{W_Y}\mathbf{\Sigma}^{-2}\mathbf{W_Y'})) \\ & +\mathrm{tr}\Big[(\mathbf{XW_X})\mathbf{\Sigma}^{-2}(\mathbf{XW_X})' + (\mathbf{YW_Y})\mathbf{\Sigma}^{-2}(\mathbf{YW_Y})' \\ & -(\mathbf{XW_X}+\mathbf{YW_Y})\cdot\mathbf{\Sigma}^{-2}(\mathbf{I}+2\mathbf{\Sigma}^{-2})^{-1}\mathbf{\Sigma}^{-2}\cdot(\mathbf{XW_X}+\mathbf{YW_Y})'\Big] \\ & +\text{a constant.} \end{aligned} \tag{7.13}$$

Differentiating this with respect to the matrix $\mathbf{W_X}$ and equating to zero, gives

$$\begin{aligned} & -2\mathbf{W_X}^{-1} + 2[\mathbf{\Sigma}^{-2} - \mathbf{\Sigma}^{-2}(I+2\mathbf{\Sigma}^{-2})^{-1}\mathbf{\Sigma}^{-2}]\mathbf{W_X'C_{XX}} \\ & -2[\mathbf{\Sigma}^{-2}(I+2\mathbf{\Sigma}^{-2})^{-1}\mathbf{\Sigma}^{-2}]\mathbf{W_Y'C_{YX}} = \mathbf{0}. \end{aligned}$$

After multiplication on the left with $\mathbf{\Sigma}^4(I+2\mathbf{\Sigma}^{-2})(I+\mathbf{\Sigma}^2)^{-1}$ and on the right with $\mathbf{W_X}$, this leads to

$$\mathbf{W_X'C_{XX}W_X} - (\mathbf{I}+\mathbf{\Sigma}^2)^{-1}\mathbf{W_Y'C_{YX}W_X} = (\mathbf{I}+\mathbf{\Sigma}^2) - (\mathbf{I}+\mathbf{\Sigma}^2)^{-1} \tag{7.14}$$

Similarly, we can derive an analogous equation by equating the derivative with respect to W_Y to zero.

$$\mathbf{W}_Y'\mathbf{C_{YY}W_Y} - (\mathbf{I}+\mathbf{\Sigma}^2)^{-1}\mathbf{W_X C_{XY} W_Y} = (\mathbf{I}+\mathbf{\Sigma}^2) - (\mathbf{I}+\mathbf{\Sigma}^2)^{-1} \tag{7.15}$$

Furthermore, differentiating this with respect to $\mathbf{\Sigma}^{-2}$ (taking the diagonality of $\mathbf{\Sigma}$ into account), leads to

$$\begin{aligned} & -2(\mathbf{I}+2\mathbf{\Sigma}^{-2})^{-1} - 2\mathbf{\Sigma}^2 \\ & +\mathrm{diag}[\mathbf{W_X'C_{XX}W_X} + \mathbf{W_Y'C_{YY}W_Y} \\ & -2\mathbf{\Sigma}^{-2}(\mathbf{I}+\mathbf{\Sigma}^{-2})(\mathbf{I}+2\mathbf{\Sigma}^{-2})^{-2}\cdot \\ & (\mathbf{W_X'C_{XX}W_X} + \mathbf{W_Y'C_{YY}W_Y} + \mathbf{W_X'C_{XY}W_Y} + \mathbf{W_Y'C_{YX}W_X})] = \mathbf{0}. \end{aligned} \tag{7.16}$$

Now, one can see that these equations (7.14), (7.15) and (7.16) hold for the CCA solution, if the columns of $\mathbf{V_X}$ and $\mathbf{V_Y}$ are properly normalized, so that $\mathbf{V_X'C_{XX}V_X} = \mathbf{I}+\mathbf{\Sigma}^2$, $\mathbf{V_Y'C_{YY}V_Y} = \mathbf{I}+\mathbf{\Sigma}^2$ and with $\mathbf{\Xi} = (\mathbf{I}+\mathbf{\Sigma}^2)^{-1}$. We can see this since, if we multiply the CCA generalized eigenvalue equation by $\begin{pmatrix} \mathbf{V_X'} & \mathbf{V_Y'} \end{pmatrix}$ on the left hand side, we obtain:

$$\begin{aligned} \mathbf{V_X'C_{XY}V_Y} &= \mathbf{V_X'C_{XX}V_X\Xi} \\ \mathbf{V_Y'C_{YX}V_X} &= \mathbf{V_Y'C_{YY}V_Y\Xi}. \end{aligned}$$

Filling everything out completes the proof. ∎

7.3.3 Regularized CCA as the maximizer of the expected log likelihood

In general, however, we will not only encounter noise on the latent variables $\mathbf{C}$, but there will be measurement noise on $\mathbf{X}$ and on $\mathbf{Y}$ as well. Therefore, we adopt the following model

$$\begin{aligned}(\mathbf{X}-\mathbf{D_X})\mathbf{W_X} &= \mathbf{C}-\mathbf{N_X} \\ (\mathbf{Y}-\mathbf{D_Y})\mathbf{W_Y} &= \mathbf{C}-\mathbf{N_Y}.\end{aligned} \tag{7.17}$$

We will assume the noise terms $\mathbf{D_X}$ and $\mathbf{D_Y}$ are gaussianly distributed, with covariance matrices equal to $\mathbf{\Lambda_X^2}$ and $\mathbf{\Lambda_Y^2}$.

We are now ready for the main theorem of the paper:

Theorem 7.4 *Properly normalized* $\mathbf{V_X}$ *and* $\mathbf{V_Y}$*, and* $(\mathbf{\Xi}^{-1}-\mathbf{I})$*, given by the regularized canonical correlation (RCCA) estimate as defined by the following generalized eigenvalue problem*

$$\begin{pmatrix} 0 & \mathbf{C_{XY}} \\ \mathbf{C_{YX}} & \mathbf{0} \end{pmatrix}\begin{pmatrix} \mathbf{v_{X},}_i \\ \mathbf{v_{Y},}_i \end{pmatrix} = \xi_i \begin{pmatrix} \mathbf{C_{XX}}+\mathbf{\Lambda_X^2} & \mathbf{0} \\ \mathbf{0} & \mathbf{C_{YY}}+\mathbf{\Lambda_Y^2} \end{pmatrix}\begin{pmatrix} \mathbf{v_{X},}_i \\ \mathbf{v_{Y},}_i \end{pmatrix}$$

correspond to a stationary point of the expected log likelihood with variables $\mathbf{W_X}$*,* $\mathbf{W_Y}$ *and* $\mathbf{\Sigma}^2$ *respectively, parameters in the generative model (7.17). The expectation is carried out over the distributions of* $\mathbf{D_X}$ *and* $\mathbf{D_Y}$*.*

That this stationary point corresponds to a maximum, will be left as a conjecture in this thesis.

Proof The outline of the proof is clear, given the maximum likelihood derivation of ordinary CCA, and the RR derivation. We will thus only state some intermediate results of the proof, for conciseness.

The probability of $\mathbf{X}$ and $\mathbf{Y}$ given $\mathbf{W_X}$, $\mathbf{W_Y}$, $\mathbf{\Sigma}^2$, $\mathbf{C}$ and $\mathbf{D_X}$ and $\mathbf{D_Y}$ is given by

$$\begin{aligned} & P(\mathbf{X},\mathbf{Y}|\mathbf{W_X},\mathbf{W_Y},\mathbf{\Sigma}^2,\mathbf{C},\mathbf{D_X},\mathbf{D_Y}) \\ = \; & P(\mathbf{X}|\mathbf{W_X},\mathbf{\Sigma}^2,\mathbf{C},\mathbf{D_X})P(\mathbf{Y}|\mathbf{W_Y},\mathbf{\Sigma}^2,\mathbf{C},\mathbf{D_Y}) \\ = \; & P(\mathbf{X}-\mathbf{D_X}|\mathbf{W_X},\mathbf{\Sigma}^2,\mathbf{C},\mathbf{D_X})P(\mathbf{Y}-\mathbf{D_Y}|\mathbf{W_Y},\mathbf{\Sigma}^2,\mathbf{C},\mathbf{D_Y}).\end{aligned}$$

Each of these factors can be expressed in the same way as equations (7.10) and (7.11), where X and Y have to be replaced by $\mathbf{X}-\mathbf{D_X}$ and $\mathbf{Y}-\mathbf{D_Y}$.

We can again take the expectation of (7.18) over $\mathbf{C}$, leading to the analogon to equation (7.12), with the same replacements. Taking the logarithm multiplied by minus two, and subsequently averaging with respect to $\mathbf{D_X}$ and $\mathbf{D_Y}$, leads to minus two times the average log likelihood

$$-2L(\mathbf{X},\mathbf{Y}|\mathbf{W_X},\mathbf{W_Y},\mathbf{\Sigma}^2) = \mathcal{E}_{\mathbf{D_X},\mathbf{D_Y}}\{l(\mathbf{X},\mathbf{Y}|\mathbf{W_X},\mathbf{W_Y},\mathbf{\Sigma}^2,\mathbf{D_X},\mathbf{D_Y})\}$$

to be equal to a sum of two terms, the first one of which is equal equation (7.13), and the second one of which is

$$\begin{aligned} \text{tr} \quad & \Big[\mathbf{\Sigma}^{-1}\mathbf{W}_{\mathbf{X}}'\mathbf{\Lambda}_{\mathbf{X}}^2\mathbf{W}_{\mathbf{X}}\mathbf{\Sigma}^{-1} + \mathbf{\Sigma}^{-1}\mathbf{W}_{\mathbf{Y}}'\mathbf{\Lambda}_{\mathbf{Y}}^2\mathbf{W}_{\mathbf{Y}}\mathbf{\Sigma}^{-1} \\ - \quad & (\mathbf{I}+2\mathbf{\Sigma}^{-2})^{-1}\mathbf{\Sigma}^{-2}\mathbf{W}_{\mathbf{X}}'\mathbf{\Lambda}_{\mathbf{X}}^2\mathbf{W}_{\mathbf{X}}\mathbf{\Sigma}^{-2}(\mathbf{I}+2\mathbf{\Sigma}^{-2})^{-1} \\ - \quad & (\mathbf{I}+2\mathbf{\Sigma}^{-2})^{-1}\mathbf{\Sigma}^{-2}\mathbf{W}_{\mathbf{Y}}'\mathbf{\Lambda}_{\mathbf{Y}}^2\mathbf{W}_{\mathbf{Y}}\mathbf{\Sigma}^{-2}(\mathbf{I}+2\mathbf{\Sigma}^{-2})^{-1}\Big]. \end{aligned}$$

By simply writing out the equations, this can be shown to be equal to (7.13), with every $\mathbf{C_{XX}} = \frac{\mathbf{X}'\mathbf{X}}{n}$ and $\mathbf{C_{YY}} = \frac{\mathbf{Y}'\mathbf{Y}}{n}$ replaced by $\frac{\mathbf{X}'\mathbf{X}}{n}+\mathbf{\Lambda}_{\mathbf{X}}^2$ and $\frac{\mathbf{Y}'\mathbf{Y}}{n}+\mathbf{\Lambda}_{\mathbf{Y}}^2$. The cross product terms in $\mathbf{C_{XY}} = \frac{\mathbf{X}'\mathbf{Y}}{n} = \mathbf{C_{YX}}'$ remain unchanged.

Now, we can differentiate this cost function with respect to $\mathbf{W_X}$ and $\mathbf{W_Y}$ again, leading to optimality conditions that are fulfilled by the regularized CCA solution, given that $\mathbf{W_X}$ and $\mathbf{W_Y}$ are normalized so that $\mathbf{W}_{\mathbf{X}}'(\mathbf{C_{XX}} + \mathbf{\Lambda}_{\mathbf{X}}^2)\mathbf{W_X} = \mathbf{I}+\mathbf{\Sigma}^2$, $\mathbf{W}_{\mathbf{Y}}'(\mathbf{C_{YY}} + \mathbf{\Lambda}_{\mathbf{Y}}^2)\mathbf{W_Y} = \mathbf{I}+\mathbf{\Sigma}^2$ and with $\mathbf{\Xi} = (\mathbf{I}+\mathbf{\Sigma}^2)^{-1}$. This completes the outline of the proof. ∎

Note that it is straightforward to extend the theorem to general covariance matrices for $\mathbf{D_X}$ and $\mathbf{D_Y}$.

7.3.4 Interpretation

We can interpret this in a similar way as RR. The expected log likelihood is a lower bound of the log of the expectation. Therefore, the maximum of the expected log likelihood is a lower bound on the likelihood, $\mathbf{D_X}$ and $\mathbf{D_Y}$ taken into account.

Again, this does not represent an unbiased estimator, since it is not the maximum likelihood estimator. Neither is it the least squares estimator, as it was in the RR case. However, instead of least squares, another measure is appropriate, namely the log likelihood (in the RR case, least squares was equivalent with log likelihood).

7.3.5 Practical aspects

Even though we only intended to perform a theoretical study of the motivation behind this particular type of regularization of CCA, there are some practical implications.

In general, the noise covariance matrices $\mathbf{\Lambda_X}$ and $\mathbf{\Lambda_Y}$ are not known. Thus, they have to be estimated using cross validation, or by some other method. If we use cross validation, we need a cost function to be optimized. For this, based on the analogy with RR, it has now been made acceptable to use the log likelihood given the data (equation (7.13)). Thus, for each of the regularization parameters, we solve the generalized eigenvalue problem making use of the training set. The regularization parameter for which the log likelihood of the test set is maximal, can be taken as the estimate for the noise covariance.

7.4 Conclusions

LSR and RR are well established regression methods. By analogy to a derivation of LSR, we propose an interpretation of CCA in terms of a maximum likelihood estimator. Extending the result along the same lines as the extension of LSR towards RR, we derived a regularized version CCA, that is around for quite some time already, however, its interpretation was subject of debate.

Apart from these theoretical results concerning the regularization of CCA, we made clear how to train the regularization parameter using cross validation. This was not known before, and at hoc techniques were applied.

Importantly, the model underlying CCA shows tight similarities with the independent component analysis (ICA) model (for an introduction to ICA, see [60]). Where the identification for ICA is possible thanks to supposed independencies among the components of C (and this is basically exploited using higher order information), CCA only uses second order information to identify essentially the same model.

Chapter 8

Integrating heterogeneous data: a database approach

This chapter is motivated by a concrete practical problem that is currently of high relevance in the bioinformatics research domain, and therefore distinguishes itself from the previous chapters in this thesis in that the method developed here is motivated by the application and not so much by our curiosity for novel machine learning problems and techniques. Indeed, the algorithm developed in this chapter is purely a *database technique*, as opposed to the *machine learning techniques* from the previous chapters. Therefore, this chapter might be seen as a (quite obvious) negative answer to the question whether kernel methods are appropriate to solve all types of pattern discovery problems: however powerful they are, sometimes other techniques such as the one presented in this chapter can be more appropriate.

Because of the fact that this chapter is driven by a bioinformatics problem, it contains a number of concepts from bioinformatics that can not be explained in this thesis. However, the algorithmic concepts should be understandable if abstraction is made of the bioinformatics terminology.

8.1 Introduction

Nowadays, data representative of different cellular processes are being generated at large scale. Based on these *omics* data sources (genomics, proteomics, metabolomics,...), the action of the regulatory network that underlies the organism's behavior can be observed.

Whereas until recently bioinformatics research was driven by the development of methods that deal with each of these data sources separately, the focus is now shifting drastically towards integrative approaches dealing with several data sources simultaneously. Indeed, technological and biological noise in the individ-

ual data sources is often so prohibitive and unavoidable that standard methods are bound to fail. Then only a combined use of *heterogeneous* and *independently acquired* information sources can help to solve the problem. Furthermore, these different points of view on the biological system allow gaining a holistic insight into the network studied. Therefore, the integration of heterogeneous data is an important, though non-trivial, challenge of current bioinformatics research.

8.2 Situation

In this study (which is published as [19]) we focus on 3 types of omics data that give independent information on the composition of transcriptional modules, the basic building blocks of transcriptional networks in the cell: *ChIPchip* data (*chromatin immunoprecipitation on arrays*) provides information on the direct physical interaction between a regulator and the upstream regions of its target genes; *motif* information as obtained by phylogenetic shadowing describes the DNA recognition sites of these regulators; and *gene expression profiles* obtained using microarray experiments describe the expression behavior in the conditions tested. By integrating these three data sources, we aim at identifying the concerted action of regulators that elicit a characteristic expression profile in the conditions tested, the target genes of these regulators, and the DNA binding sites recognized by these regulators, thus fully specifying the relevant regulatory modules.

A previous successful approach to integrative analysis in bioinformatics can be found in the class methods based on *graphical models* [83, 97, 96, 47]. Unfortunately, most of these and related methods exploit the availability of heterogeneous data sources in a *sequential* or an *iterative* way (see e.g. [72] for simultaneous detection of motifs and clustering of expression data, e.g. [7] for an iterative approach using ChIPchip and expression data, and e.g. [77] for simultaneous motif detection and analysis of ChIPchip data). Obviously, this is not beneficial for the interpretability of the results.

As presented in the previous chapters of this thesis, other approaches that elegantly deal with heterogeneous information can be found among the *kernel methods* [116, 70]: CCA makes it possible to find dimensions in the different data sources that show a strong (possibly nonlinear) correlation. Therefore, CCA allows to extract information that is in common to the different sources of information. On the other hand, the kernel combination formalism allows to complement kernels with each other, weighing the kernels according to their informativeness for the problem at hand. However, in all cases, kernel methods are developed for problems such as classification and regression, ranking and clustering,... but to date kernel methods have rarely been applied to identifying discrete patterns in data such as regulatory modules considered here.

Thus, to our knowledge, no successful attempts to solve the problem of module inference exploiting all 3 independently acquired ChIPchip, motif and expression data have been made so far. In this last chapter of this thesis, we present an approach for this problem, that is different in spirit from previous

methods that deal with heterogeneous information, taking the different data sources into account in a highly concurrent way. Furthermore, the approach adopted here is radically different from all other algorithms studied in this thesis. Doing this, it is our intention to point the reader to the limitations of kernel methods and standard continuous optimization methods, and to give suggestions for alternatives.

The performance of the algorithm was demonstrated using the Spellman dataset [106] as a benchmark.

8.3 Materials and Algorithms

8.3.1 Data sources

As microarray benchmark set the Spellman dataset was used [106], which contains 77 experiments describing the dynamic changes of 6178 genes during the yeast cell cycle. The profiles were normalized (subtracting the mean of each profile and dividing by the standard deviation across the time points) and stored in a gene expression data matrix further denoted by **A** with a row for each gene expression profile and a column for each condition.

Genome-wide location data performed by Lee et al. [74] were downloaded from `http://web.wi.mit.edu/young/regulator_network`. These contain data on the binding of 106 regulators to their respective target genes in rich medium. The ChIPchip data matrix (further denoted by **R**) used in our study consists of one minus the p-values obtained from combined ratio's between immunoprecipitated and control DNA (see [74]). Thus, a large value (close to one) indicates that the regulator is probably present.

The motif data used in this study were obtained from a comparative genome analysis between distinct yeast species (phylogenetic shadowing) performed by Kellis et al. [66]. The authors describe the detection of 72 putative regulatory motifs in yeast. These motifs, available online as regular expressions, were transformed into the corresponding probabilistic representation (weight matrix): for each motif, the 20 *Saccharomyces cerevisiae* genes in which the motif was most reliably detected according to the scoring scheme of Kellis et al. [66] were selected. The intergenic sequences of these genes were subjected to motif detection based on Gibbs sampling [112, MotifSampler]. If the statistically overrepresented motif in this set of putatively coexpressed genes corresponded to the motif that was detected by the comparative motif search of [66] the motif model was retained. As such 53 of the 71 motifs could be converted into a weight matrix. This weight matrix was subsequently used to screen all intergenic sequences for the presence of the respective regulatory motifs using MotifLocator [79]. Absolute scores were normalized [79]. As the score distribution of the motif hits depends on the motif length and the degree of conservation of the motif, the distribution of the normalized scores differs between motifs. Therefore, normalized scores were converted into percentile values. This allows for an unbiased choice of the thresholds on the motif quality parameter in the algorithm. The

matrix containing these percentile values is the motif data matrix $\mathbf{M}$ that will be used in this work.

8.3.2 Module construction algorithm

The aim of the method is to find regulatory modules, based on the gene expression, ChIPchip, and Motif data matrices as specified above. A module is fully specified by the set of genes it regulates (denoted by an index set g, pointing to the relevant set of rows of $\mathbf{R}$, $\mathbf{M}$ and $\mathbf{A}$), in addition to the set of regulators (corresponding to the columns with indices in a set called r in the ChIPchip matrix R) and motifs (corresponding to the columns with indices m in the Motif matrix M) that are responsible for the regulation of these genes. The goal of our method is to come up with regulatory modules specified in this way, by fully exploiting the heterogeneous data sources available. We note that the principles behind the method developed here are based on ideas similar to those that laid the foundations for the Apriori algorithm, originally developed in the database community [1].

Seed construction

This is the main step of the algorithm, and allows the construction of a good guess (or *seed*) of the modules. The idealized goal of this step is to find a set of genes $\mathbf{g}$, that have the same expression profile, and such that there exist sufficiently large sets of regulators $\mathbf{r}$ and of motifs $\mathbf{m}$ that are entirely present in all these genes. Since in practice it is not known exactly in which intergenic regions a certain motif occurs or where a regulator binds, we have to resort to the score matrices $\mathbf{R}$ and $\mathbf{M}$. Furthermore, the expression profiles $\mathbf{A}$ of genes in a module will only be approximately equal, and possibly only in a set of conditions, so we relax this constraint to requiring a strong correlation instead of equality between them. Formally, then the task to solve is:

> Find all maximal gene sets $\mathbf{g}$ for which there exist an $\mathbf{r}$ of size $|\mathbf{r}| = r_{\min}$ and a set $\mathbf{m}$ of size $|\mathbf{m}| = m_{\min}$, such that the following 3 constraints are satisfied:
>
> 1. $\mathbf{R}(i,j) > t_r$ for all $i \in \mathbf{g}$ and $j \in \mathbf{r}$,
> 2. $\mathbf{M}(i,j) > t_m$ for all $i \in \mathbf{g}$ and $i \in \mathbf{m}$,
> 3. $\mathrm{corr}(\mathbf{A}(i,:), \mathbf{A}(j,:)) > t_a$ for all $i \in \mathbf{g}$,
>
> where $r_{\min}$, $m_{\min}$ and thresholds t_r , t_m and t_a are parameters of the method.

Here, a *maximal set* $\mathbf{g}$ is defined as a set that cannot be extended with another gene without violating one or more of these constraints. In the following, we will use the term *valid set* for a gene set $\mathbf{g}$ that satisfies these constraints. Clearly it is computationally impossible to tackle this problem with a naive

approach: the number of gene sets is exponentially large in the number of genes in the dataset, which is prohibitive even for the smallest genomes. However, it is trivial to verify that:

Observation 1: When a gene set does not satisfy the constraints, none of its supersets satisfy the constraints.

This means that we can build up the maximal sets incrementally, starting with valid sets of size one, and gradually expanding them. Concretely, the (already less naive) algorithm would then look like[1]:

- For all single genes, check if they satisfy constraints 1 and 2 (constraint 3 is trivially satisfied for singleton gene sets). Make a list L1 of all singleton gene sets that contain such a valid gene.
- Set i = 2.
- While size($\mathbf{L}^i - 1) \geq 0$
  - For $k = 1 :$ size($\mathbf{L}^{i-1}$), expand set $\mathbf{g}_k^{i-1} = \{g_k(1), g_k(2), \ldots, g_k(i-1)\} \in \mathbf{L}^{i-1}$ once for each gene g that is not yet contained in g_k^{i-1}. Put the thus expanded sets $\{g_k(1), g_k(2), \ldots, g_k(i-1), g\}$ that satisfy the 3 constraints (to be verified in $\mathbf{R}$, $\mathbf{M}$ and $\mathbf{A}$), in a list $\mathbf{L}^i$. Set $i = i + 1$.

Notice that following this strategy, a gene set can be constructed in different ways, by adding the genes to it in a different ordering (i.e. in different iterations i). This can be avoided by adding a gene to a gene set g_k^{i-1} only whenever its row number g is larger than that of all other genes already in g_k^{i-1}. Thus for every $\mathbf{g}_k^i = \{g_k(1), g_k(2), \ldots, g_k(i)\} \in \mathbf{L}^i$ we always have that $g_k(x) < g_k(y)$ for $x < y$.

Additionally, in this way we can easily keep the list $\mathbf{L}^i$ of gene sets $\mathbf{g}_k^i$ sorted as well, where the sorting is carried out first according to the first added gene and last according to the last added gene. More formally: $\mathbf{g}_k^i$ preceeds $\mathbf{g}_l^i$ in $\mathbf{L}^i$ if and only if $g_k(\text{argmin}_x(g_k(x) \neq g_l(x))) < g_l(\text{argmin}_x(g_k(x) \neq g_l(x)))$ (this ordering of the list $\mathbf{L}^i$ is indeed a total ordering relation.) Still the number of expanded gene sets can be huge in every iteration: each of the gene sets $\mathbf{g}_k^{i-1}$ in $\mathbf{L}^{i-1}$ must be expanded by all genes $g > g_k(i-1)$, after which the validity has to be checked by looking at the matrices $\mathbf{R}$, $\mathbf{M}$ and $\mathbf{A}$. This can still be too expensive. However, we can exploit the converse of Observation 1:

[1] Notationally, we will use $\mathbf{L}_i$ to denote the list containing all valid gene sets with i genes. For an individual valid gene set we will use a bold face g_k^i, with a superscript i to specify that it is an element of $\mathbf{L}_i$ and thus contains i genes, and with a subscript k to distinguish it from the other gene sets in $\mathbf{L}_i$. The x-th gene in this gene set is denoted as $g_k(x)$, for brevity without superscript.

Observation 2: Whenever a gene set satisfies the constraints, all of its subsets satisfy the constraints.

Using this so-called *hereditary property* of the constraint set, in some cases we can conclude a priori —i.e. without checking in **R**, **M** and **A**— if an extended gene set of size i can possibly be valid or not: we simply have to check if all of its size $i-1$ subsets belong to $\mathbf{L}^{i-1}$. Only if this is the case, we still have to access the data in **R**, **M** and **A**; if it is not the case, we know without further investigation that the extended subset is invalid.

Specifically, assume we expand the gene set $\mathbf{g}_k^{i-1} = \{g_k(1), g_k(2), \ldots, g_k(i-2), g_k(i-1)\} \in \mathbf{L}^{i-1}$ with g, leading to $\{g_k(1), g_k(2), \ldots, g_k(i-2), g_k(i-1), g\}$. Then, since for a valid size i set each of its size $i-1$ subsets must be contained in $\mathbf{L}^{i-1}$, also $\{g_k(1), g_k(2), \ldots, g_k(i-2), g\}$ must be contained in $\mathbf{L}^{i-1}$. In other words: there has to be a $\mathbf{g}_l^{i-1} = \{g_l(1), g_l(2), \ldots, g_l(i-2), g_l(i-1)\} \in \mathbf{L}^{i-1}$ for which $g_k(x) = g_l(x)$ for $x \leq i-2$, and $g = g_l(i-1)$.

This can efficiently be ensured constructively, by exploiting the fact that the list $\mathbf{L}^{i-1}$, and all $\mathbf{g}_k^{i-1}$ themselves are sorted. Indeed, thanks to this, all gene sets $\mathbf{g}_k^{i-1}$ that have the first $i-2$ genes in common occur consecutively in $\mathbf{L}^{i-1}$. Therefore, to expand $\mathbf{g}_k^{i-1}$ with an additional gene, we only have to screen the list $\mathbf{L}^{i-1}$ starting at $\mathbf{g}_{k+1}^{i-1}$ and move forward in $\mathbf{L}^{i-1}$ for as long as the first $i-2$ genes are equal to $g_k(1), g_k(2), \ldots$ and $g_k(i-2)$. For every gene set $\mathbf{g}_l^{i-1}$ screened in this way, read the last gene $\mathbf{g}_l(i-1)$ and append it to $\mathbf{g}_k^{i-1}$, thus resulting in a candidate gene set of size i, potentially to be appended to $\mathbf{L}_i$.

To find out whether this candidate gene set is valid indeed, one still has to check the constraints explicitly. However, thus constructively exploiting the hereditary property, the number of queries to **R**, **M** and **A** is drastically reduced. Note that this strategy also ensures that $\mathbf{L}_i$ is sorted automatically.

Module validation

In some cases the first step described above is not sufficient for adequate module inference. There are three reasons for this:

First, the seed construction method can be rather conservative in recruiting genes, since each of the genes in the module has to satisfy all 3 of the constraints. Therefore, in a second step, we calculate the mean of the expression profiles of the seed modules found in the first step, further called the *seed profile*. Then we can additionally recruit all genes with a high correlation with the seed profile to be incorporated in the module. In order to determine an optimal threshold value for this correlation, we compute the enrichment of each of the motifs and regulators in the genes that have an expression profile that achieves this threshold correlation with the seed profile. The logarithm of the p-value of the enrichment is then plotted as a function of this threshold (Figure 8.1), and the threshold can be chosen such that this value is minimal.

Second, sometimes it is undesirable to a priori decide how many motifs and regulators we want in the module, or it may be difficult to choose the thresholds

t_r, t_m and t_a (even though experiments show little dependence on these). Then one can first use the seed construction algorithm requiring only 1 regulator and motif, and with stringent thresholds, after which again the enrichment of all motifs and regulators can be plotted as a function of the correlation threshold with the mean profile of the seed module. For each such seed profile, the corresponding enrichment plot will visually hint at the number of motifs and regulators (namely the number of significantly enriched motifs and regulators).

Third, similarly, the enrichment plot allows excluding false positive motifs or regulators: when they are selected in step 1, but appear not to be enriched in the validation step, they are considered as a false positives and discarded. To calculate the enrichment, we first calculate the mean score of the module for the particular motif or regulator. Note that the mean score of a module by random gene selection is approximately Gaussianly distributed (central limit theorem), with mean equal to the mean over all genes, and variance equal to the overall variance divided by the size of the module. Thus, we can calculate the enrichment as the logarithm of the p-value based on a Gaussian approximation. Note that the p-values have been computed based on profiles that have been obtained from the data, such that they do not have a rigorous probabilistic interpretation here. Hence, we can only use them as explained above.

8.3.3 Calculating overrepresentation of functional classes

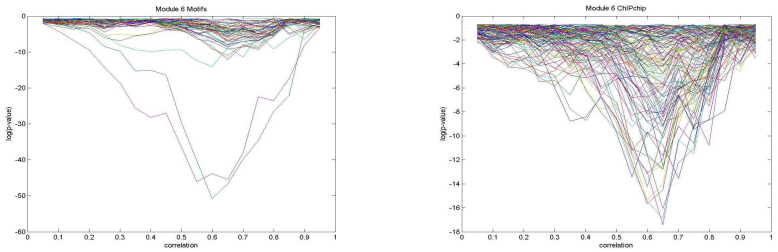
Functional categories for each gene were obtained from MIPS [80]. Functional enrichment of the modules was calculated using the hypergeometric distribution [111], which assigns to each functional class a p-value.

8.4 Results

8.4.1 Cell cycle related modules

To test the reliability of our method, we used the well-studied Spellman dataset as benchmark. The analysis we performed using our two-step algorithm is illustrated by elaborating on the detection of the cell cycle related module 1. Using the seed detection step we searched for modules of genes having at least 1 common motif (1M) in their intergenic sequences and 1 common regulator (1R) showing a small p-value in the ChIPchip data, and of which the expression profiles were mutually correlated with a minimal correlation of 0.7. This seed identification step then predicts several potential modules, and for each of them a seed profile can be calculated. For each of these modules we performed the module validation step. Fig. (8.1A) (right figure) shows how this validation step allows one to visually detect that the regulator associated with this module is probably a false positive. In Fig (8.1B), using the parameter settings of 1M/1R, we identified a potential seed module containing regulator 98 (Swi4) and motif M_11 (known as a Swi4 motif). Calculating the statistical overrepresentation of all motifs and regulators in genes correlated with the seed profile of

A



B

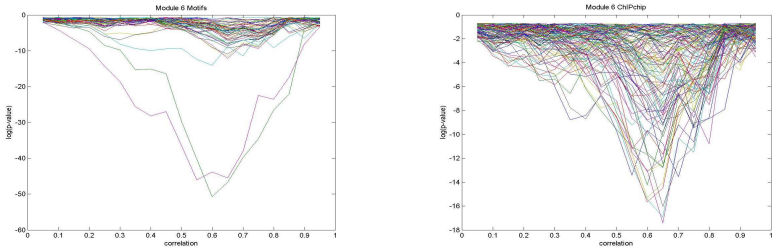


Figure 8.1: Two examples (A and B) of the module validation step for two seed profiles: on the left, the logarithms of the p-values are plotted for all motifs as a function of the correlation threshold, on the right similarly for the regulators. Panel A shows the results for a false positive prediction (module 6): the regulators (right figure) of the identified seed module turn out not to be significantly overrepresented in genes correlated with the seed profile. In panel B the results are displayed for the positive example described in the text.

this putative module showed that in this subset of genes indeed M_11 and Swi4 were overrepresented. The identified module seed thus is likely to be biologically relevant. These results also show that besides Swi4 and M_11, 3 additional motifs and regulators were overrepresented in subsets of genes correlated with the module seed profile, indicating the probable underestimation of the real module size. To verify whether these other regulators/motifs co-occur in the same subsets of genes and therefore comprise a larger module, we repeated the seed identification step using additional parameter settings (see Table 1 in the online supplement). From this result it appeared that we could recover a complete module consisting of the 3 overrepresented regulators (Mbp1, Swi4, Swi6) and 2 motifs (M_16, M_10) and that this module is present in genes displaying an expression profile that shows a correlation of at least 0.7 with the average seed profile. Checking the identities of the regulators and the motifs (regulators Mbp1, Swi4, Stb1 combined with the regulatory motifs Mbp1 (M_18, M_12) and Swi4 (M_11 and M67)) showed that we identified a previously extensively described regulatory module of the yeast cell cycle. Besides this first module, 3 additional related cell cycle (Table 8.1) modules could be retrieved. Additional information on each of the separate modules can be found in the online supplement. Genes in the different modules showed peak expressions shifted in time relative to each other, as shown in Figure 1 of the online supplement. All of the predicted modules are conform the previously described knowledge on the cell cycle [74, 31, 7].

8.4.2 Non cell cycle related modules

Besides the modules primarily involved in cell cycle, other modules could be identified in the Spellman dataset (see Table 8.2). Module 5, consisting of Fhl1, Rap1 and Yap5, involved in the regulation of ribosomal proteins was previously also identified [74]. Note that it was identified from a noise profile (i.e. a profile that does not change significantly and consistently with the cell cycles over the different time points) in this cell cycle dataset, indicating that even biological noise contains important information on regulatory networks. By our analysis we could pinpoint motif M_54 [66], as the regulatory motif correlated with this regulatory module. A second non cell cycle related module consisted of the genes regulated by the motifs M_7 and M_3 (identified as ESR1 and ESR2 [66]). For this module, related to transcription and ribosomal RNA processing only the motifs seemed informative (see module 6 in Table 8.2, and Figure 1A).

8.5 Discussion

We described a methodology combining ChIPchip, motif and expression data to infer complete descriptions of transcriptional modules. Our methodology consists of 2 steps. The seed construction step predicts the putative modules consisting of regulators, their corresponding motifs and the elicited expression profile. The validation step filters false positive predictions and gives further

Table 8.1: Cell cycle related modules. Column 'Reg.' contains the regulators, column 'Motif' the motifs, the column 'Functional Class: p-value' contains p-values for several functional classes, and the 'Seed Profile' column contains a plot with the expression profiles of the genes regulated by the module.

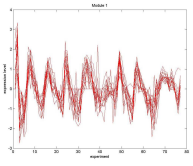
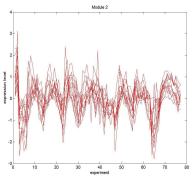
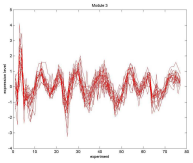
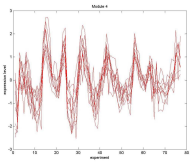
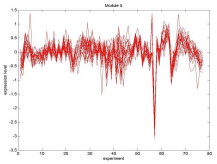
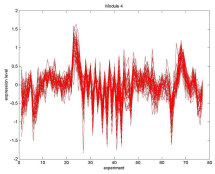
	Reg.	Motif	Functional class: p-value	Seed profile
M1	Mbp1 Swi6 Swi4 Stb1	M_18 (Mbp1) M_12 (Mbp1) M_11 (Swi4) M_67 (Swi4)	10 CELL CYCLE AND DNA PROCESSING: 0 10.03 cell cycle: 2.7e-5 10.01 DNA processing: 1.3e-4 42.04 cytoskeleton: 4.2e-3	
M2	Swi4 Mbp1 Swi6 FKH2	M_18 (Mbp1) M_12 (Mbp1) M_11 (Swi4) M_8 (Mcm)	40 CELL FATE : 5.2e-4 40.01 cell growth / morphogenesis: 2.6e-3 43 CELL TYPE DIFFERENTIATION: 5.2e-3 43.01 fungal/micro-organismic cell type differentiation: 5.2e-3 34.11 cellular sensing and response: 5.3e-3 01.05.01 C-compound and carbohydrate utilization: 6.8e-3 10.03.04.03 chromosome condensation: 9.4e-3	
M3	NDD1 FKH2 Mcm1	M_8 (Mcm) M_30 (Mcm)	43 CELL TYPE DIFFERENTIATION: 3.6e-3 43.01 fungal/micro-organismic cell type differentiation: 3.6e-3 10.03.03 cytokinesis (cell division) / septum formation: 4.8e-3	
M4	Swi5 (Ace2)	M_8 (Mcm)	32.01 stress response: 3.2e-3 10.03 cell cycle: 8.7e-3	

Table 8.2: Non cell cycle related modules.

	Reg.	Motif	Functional class: p-value	Seed profile
M5	FKL1 Yap5 Rap1	M_54	12 PROTEIN SYNTHESIS: 0 12.01 ribosome biogenesis: 0	
M6	/	M_3 (ESR1) M_7 (ESR2)	11 TRANSCRIPTION: 2e-6 11.04 RNA processing: 0 11.04.01 rRNA processing: 0	

insight into the module size. The problem is attacked in a very direct way: the integration of the data sources is achieved in a one-shot-algorithm, and requires no iteration over the different data sources. While the running time was very reasonable for all experiments carried out for this paper, it heavily depends on the parameters. The more stringent they are set, the smaller the lists Li will be and the faster the algorithm will run. Further speed-ups are possible, but not needed for the experiments reported in this paper. Therefore we will not go into these here.

The Spellman dataset was used as a benchmark to test the performance of our method. Since this dataset and the yeast cell cycle have extensively been studied before [7, 74], it is ideally suited for testing the reliability and biological relevance of the predictions. We were able to reconstruct 4 important modules known to be involved in cell cycle and also 2 non cell cycle related modules without using any prior biological knowledge or prior data reduction. These results indicate that predictions passing the module validation step are likely to be biologically relevant (no false positives present).

8.6 Conclusion

The 3 data types mutually agreeing with each other on the prediction of a module not only results in the most reliable predictions (as was the case for the cell cycle related modules), but also allows correlating a set of regulators with their corresponding regulatory motifs and elicited profiles in a very natural and

direct way. On the other hand, because of the restricted number of experimental data yet available (chip data not known for all regulators and tested in a limited set of conditions, expression data for specific conditions not available), and the questionable quality of the motif models, the presence of a signal in 1 data type can compensate for the lack of it in another data type, allowing still to retrieve the module.

While to our knowledge this is the first time these 3 independently acquired data sources are exploited in such a concurrent way for module identification, the approach is further extendible towards any number of information sources, and in principle towards the use of other data types. The only condition for an efficient method to exist is that the constraints the gene sets have to satisfy must be hereditary. This extension will be the subject of future work.

General conclusions

To conclude, we want to summarize our personal view on the field(s) covered by the research that led to this thesis. Specifically, we will try to answer the questions of what we think will remain and what will be used by the research community or beyond. We have always tried to keep this question in mind when choosing our research topics, even though sometimes mere scientific curiosity may draw one's attention from this path.

What about the new learning settings, transduction and more generally learning from side-information, learning from heterogeneous information sources? As should be clear by now, applications of these settings abound, in the broad domain of bioinformatics (mainly learning the heterogeneous information), machine vision (mainly learning from side-information, for image and video segmentation), many generic classification problems (where often the transduction setting can be used instead) and more.

What about the use of convex optimization and eigenvalue problems in machine learning? It is our belief that the immense advances made in the numerical and optimization literature will find their way to modern machine learning algorithms. Furthermore, even though to date the most sophisticated optimization methods such as SDP are still too time consuming for industrial applications, we spent time exploring their potential use, secretly hoping that one day (as happened for linear programming) computing power and algorithmic ingenuity will make it possible to solve them for large size problems. Surely it is the interaction between different levels of research –theoretical, numerical, algorithmic and applications– that drives research to a higher level.

When we look at our thesis from a distance, we have to admit it seems small in the immense realm of artificial intelligence or pattern recognition techniques in the broad sense. We concentrated on a specific branch of artificial intelligence, namely machine learning—the art of learning from examples—and mostly even on the subdomain of kernel methods and graph cut algorithms in machine learning. Still, while this branch of artificial intelligence is only a relatively small part of it, a surprisingly large variety of problems can be solved by it, including clustering, dimensionality reduction and visualization, classification, regression, ranking,... and, as thoroughly studied in this thesis, transduction,

learning from side-information and from heterogeneous data sources. However, Chapter 8 about a problem that seems best solved by a different type of methodologies shows that the application domain of kernel methods in machine learning is large but limited. Therefore, in order to get the right appreciation of an algorithm, practitioners should be on their guard: before using a particular method, one should make sure it is suited for the particular problem.

Bibliography

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proc. of the 20th International Conference on Very Large Databases (VLDB94)*, pages 487–499, 1994.

[2] S. Akaho. A kernel method for canonical correlation analysis. In *Proc. of the International Meeting of the Psychometric Society (IMPS01)*, page 100 (abstract), 2001.

[3] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[4] T.W. Anderson. *An Introduction to Support Vector Machines*. An Introduction to Multivariate Analysis, Wiley, New york, 1984.

[5] Y. Azar, A. Fiat, A. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In *Proc. of the thirty-third annual ACM symposium on Theory of computing (STOC01)*, pages 619–626, 2001.

[6] F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

[7] Z. Bar-Joseph, G. Gerber, T. Lee, N. Rinaldi, J. Yoo, F. Robert, D. Gordon, E. Fraenkel, T. Jaakkola, R. Young, and D. Gifford. Computational discovery of gene modules and regulatory networks. *Nature Biotechnology*, 21(11):1337–42, 2003.

[8] M. Barker and W.S. Rayens. Partial least squares for discrimination. *Journal of Chemometrics*, 17:166–173, 2003.

[9] M. Barker and W.S. Rayens. Partial least squares for discrimination. *Journal of Chemometrics*, 17:166–173, 2003.

[10] M. S. Bartlett. Further aspects of the theory of multiple regression. *Proc. Camb. Philos. Soc.*, 34:33–40, 1938.

[11] Y. Bengio, P. Vincent, and J.F. Paiement. Learning eigenfunctions of similarity: Linking spectral clustering and kernel pca. Technical Report 1232, Département d'informatique et recherche opérationnelle, Université de Montréal, 2003.

[12] K. Bennett and A. Demiriz. Semi-supervised support vector machines. In *Advances in Neural Information Processing Systems 11 (NIPS98)*, pages 368–374, 1999.

[13] T. De Bie and N. Cristianini. Convex methods for transduction. In *Advances in Neural Information Processing Systems 16 (NIPS03)*, pages 73–80, 2004.

[14] T. De Bie and N. Cristianini. Convex transduction with the normalized cut. Internal Report 04-128, ESAT-SISTA, K.U.Leuven (Leuven, Belgium), 2004.

[15] T. De Bie and N. Cristianini. Kernel methods for exploratory data analysis: a demonstration on text data. In *Proc. of the International Workshop on Statistical Pattern Recognition (SPR04)*, pages 16–29, 2004.

[16] T. De Bie, N. Cristianini, and R. Rosipal. Eigenproblems in pattern recognition. In E. Bayro-Corrochano, editor, *Handbook of Computational Geometry for Pattern Recognition, Computer Vision, Neurocomputing and Robotics*. Springer-Verlag, 2004.

[17] T. De Bie, G. Lanckriet, and N. Cristianini. Convex tuning of the soft margin parameter. Internal Report 04-127, ESAT-SISTA, K.U.Leuven (Leuven, Belgium), 2003.

[18] T. De Bie, M. Momma, and N. Cristianini. Efficiently learning the metric with side-information. In *Proc. of the 14th International Conference on Algorithmic Learning Theory (ALT03)*, pages 175–189, 2003.

[19] T. De Bie, P. Monsieurs, K. Engelen, B. De Moor, N. Cristianini, and K. Marchal. Discovering transcriptional modules from motif, ChIP-chip and microarray data. In *Proc. of the Pacific Symposium on Biocomputing (PSB05)*, pages 483–494, 2005.

[20] T. De Bie and B. De Moor. On the regularization of canonical correlation analysis. In *Proc. of the International Conference on Independent Component Analysis and Blind Source Separation (ICA03)*, pages 785–790, 2003.

[21] T. De Bie, J. Suykens, and De Moor B. Learning from general label constraints. In *Proc. of IAPR International Workshop on Statistical Pattern Recognition (SPR04)*, pages 671–679. 2004.

[22] S. D. Black and D. R. Mould. Development of hydrophobicity parameters to analyze proteins which bear post- or cotranslational modifications. *Anal. Biochem.*, 193:72–82, 1991.

[23] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proc. of the 18th International Conf. on Machine Learning (ICML01)*, pages 19–26, 2001.

[24] M. Borga, T. Landelius, and H. Knutsson. A Unified Approach to PCA, PLS, MLR and CCA. Report LiTH-ISY-R-1992, ISY, SE-581 83 Linköping, Sweden, November 1997.

[25] B. E. Boser, I. M. Guyon, , and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *5th Annual ACM Workshop on Computational Learning Theory (COLT92)*, pages 144–152, 1992.

[26] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, U.K., 2003.

[27] P. Bradley, K. Bennett, and Ayhan Demiriz. Constrained K-means clustering. Technical Report MSR-TR-2000-65, Microsoft Research, 2000.

[28] M. P. S. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. Sugnet, T. S. Furey, Jr. M. Ares, and D. Haussler. Knowledge-based analysis of microarray gene expression data using support vector machines. *Proceedings of the National Academy of Sciences of the United States of America*, 97(1):262–267, 2000.

[29] O. Chapelle, J. Weston, and B. Schlkopf. Cluster kernels for semi-supervised learning. In *Advances in Neural Information Processing Systems 15 (NIPS02)*, pages 585–592, 2003.

[30] C.P. Chen and B. Rost. State-of-the-art in membrane protein prediction. *Applied Bioinformatics*, 1(1):21–35, 2002.

[31] M. Costanzo, J. Hogan, M. Cusick, B. Davis, A. Fancher, P. Hodges, P. Kondu, C. Lengieza, J. Lew-Smith, C. Lingner, K. Roberg-Perez, M. Tillberg, J. Brooks, and J. Garrels. The yeast proteome database (ypd) and caenorhabditis elegans proteome database (wormpd): comprehensive resources for the organization and comparison of model organism protein information. *Nucleic Acids Research*, 28(1):73–76, 2000.

[32] T.M. Cover and J.A. Thomas. *Elements of information theory*. Wiley-interscience, New York, 1991.

[33] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, U.K., 2000.

[34] N. Cristianini, J. Shawe-Taylor, and J. Kandola. Spectral kernel methods for clustering. In *Advances in Neural Information Processing Systems 14 (NIPS01)*, pages 649–655, 2002.

[35] P. Derbeko, R. El-Yaniv, and R. Meir. Explicit learning curves for transduction and application to clustering and compression algorithms. *Journal of Artificial Intelligence Research*, 22:143–174, 2004.

[36] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, New York, 2nd edition, 2001.

[37] D. M. Engleman, T. A. Steitz, and A. Goldman. Identifying nonpolar transbilayer helices in amino acid sequences of membrane proteins. *Ann. Rev. Biophys. Biophys. Chem.*, 15:321–353, 1986.

[38] S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

[39] B. Fischer, V. Roth, and J. M. Buhmann. Clustering with the connectivity kernel. In *Advances in Neural Information Processing Systems 16 (NIPS03)*, pages 89–96, 2004.

[40] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7, Part II:179–188, 1936.

[41] C. Fyfe and P. L. Lai. ICA using kernel canonical correlation analysis. In *International workshop on Independent Component Analysis and Blind Signal Separation (ICA00)*, pages 279–284, 2000.

[42] T. Van Gestel, J. Suykens, J. De Brabanter, B. De Moor, and J. Vandewalle. Kernel canonical correlation analysis and least squares support vector machines. In *Proc. of the International Conference on Artificial Neural Networks (ICANN01)*, pages 381–386, 2001.

[43] G. Golub and Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, 3rd edition, 1996.

[44] M. Gribskov and N. L. Robinson. Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. *Computers and Chemistry*, 20(1):25–33, 1996.

[45] G. Gross, M. Gaestel, H. Bohm, and H. Bielka. cDNA sequence coding for a translationally controlled human tumor protein. *NAR*, 17(20):8367, 1989.

[46] J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143:29–36, 1982.

[47] A. Hartemink, D. Gifford, T. Jaakkola, and R. Young. Combining location and expression data for principled discovery of genetic regulatory network models. In *Pacific Symposium for Biocomputing (PSB02)*, pages 437–449, 2002.

[48] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

[49] C. Helmberg. Semidefinite programming for combinatorial optimization. Habilitationsschrift ZIB-Report ZR-00-34, TU Berlin, Konrad-Zuse-Zentrum Berlin, 2000.

[50] L. Hoegaerts, J. Suykens, J. Vandewalle, and B. De Moor. Primal space sparse kernel partial least squares regression for large scale problems. In *IEEE Proc. of the International Joint Conference on Neural Networks (IJCNN04)*, pages 561–566, 2004.

[51] A.E. Hoerl and R.W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(3):55–67, 1970.

[52] T. Hofmann. Learning what people (don't) want. In *Proc. of the European Conference on Machine Learning (ECML02)*, pages 214–225, 2002.

[53] T. P. Hopp and K. R. Woods. Prediction of protein antigenic determinants from amino acid sequences. *Proc. Natl. Acad. Sci. USA*, 78:3824–3828, 1981.

[54] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1985.

[55] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, Cambridge, 1991.

[56] A. Höskuldsson. PLS regression methods. *Journal of Chemometrics*, 2:211–228, 1988.

[57] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Education Psychology*, 24:417, 1933.

[58] H. Hotelling. Relations between two sets of variables. *Biometrika*, 28:321–377, 1936.

[59] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193218, 1985.

[60] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wileyinterscience, New York, 2001.

[61] T. Joachims. Transductive inference for text classification using support vector machines. In *Proc. of the International Conference on Machine Learning (ICML99)*, pages 200–209, 1999.

[62] T. Joachims. Transductive learning via spectral graph partitioning. In *Proc. of the International Conference on Machine Learning (ICML03)*, pages 290–297, 2003.

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

[64] S. D. Kamvar, D. Klein, and C. D. Manning. Spectral learning. In *Proc. of the International Joint Conferences on Artificial Intelligence (IJCAI03)*, pages 561–566, 2003.

[65] R. Kannan, S. Vempala, and A. Vetta. On clusterings: good, bad and spectral. In *Proc. of the 41st Foundations of Computer Science (FOCS00)*, pages 367–380, 2000.

[66] M. Kellis, N. Patterson, M. Endrizzi, B. Birren, and E. Lander. Sequencing and comparison of yeast species to identify genes and regulatory elements. *Nature*, 423(6937):241–254, 2003.

[67] R. I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proc. of the International Conference on Machine Learning (ICML02)*, pages 315–322, 2002.

[68] A. Krogh, B. Larsson, G. von Heijne, and E. L. L. Sonnhammer. Predicting transmembrane protein topology with a hidden Markov model: Application to complete genomes. *Journal of Molecular Biology*, 305(3):567–580, 2001.

[69] J. Kyte and R. F. Doolittle. A simple method for displaying the hydropathic character of a protein. *Journal of Molecular Biology*, 157:105–132, 1982.

[70] G. Lanckriet, T. De Bie, N. Cristianini, M. Jordan, and W. Stafford Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20(16):2626–2635, 2004.

[71] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[72] M. Lapidot and Y. Pilpel. Comprehensive quantitative analyses of the effects of promoter sequence elements on mrna transcription. *Nucleic Acids Research*, 31(13):3824–8, 2003.

[73] L. De Lathauwer. *Signal Processing based on Multilinear Algebra*. Phd thesis, K.U.Leuven (Leuven, Belgium), Faculty of Engineering, Sep. 1997.

[74] T. Lee, N. Rinaldi, F. Robert, D. Odom, Z. Bar-Joseph, G. Gerber, N. Hannett, C. Harbison, C. Thompson, I. Simon, J. Zeitlinger, E. Jennings, H. Murray, D. Gordon, B. Ren, J. Wyrick, J. Tagne, T. Volkert, E. Fraenkel, D. Gifford, and R. Young. Transcriptional regulatory networks in saccharomyces cerevisiae. *Science*, 298(5594):799–804, 2002.

[75] C. Leslie and R. Kuang. Fast kernels for inexact string matching. In *Conference on Learning Theory and Kernel Workshop (COLT03)*, pages 114–128, 2003.

[76] L. Liao and W. S. Noble. Combining pairwise sequence similarity and support vector machines for remote protein homology detection. In *Proc. of the Sixth Annual International Conference on Computational Molecular Biology (RECOMB02)*, pages 225–232, 2002.

[77] X. Liu, D. Brutlag, and J. Liu. An algorithm for finding protein-dna binding sites with applications to chromatin-immunoprecipitation microarray experiments. *Nature Biotechnology*, 20(8):835–9, 2002.

[78] J.B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[79] K. Marchal, S. De Keersmaecker, P. Monsieurs, N. van Boxel, K. Lemmens, G. Thijs, J. Vanderleyden, and B. De Moor. In silico identification and experimental validation of pmrab targets in salmonella typhimurium by regulatory motif detection. *Genome Biology*, 5(2):R9, 2004.

[80] H. Mewes, C. Amid, R. Arnold, D. Frishman, U. Guldener, G. Mannhaupt, M. Munsterkotter, P. Pagel, N. Strack, V. Stumpflen, J. Warfsmann, and A. Ruepp. Mips: analysis and annotation of proteins from whole genomes. *Nucleic Acids Research*, 32 Database issue:D41–44, 2004.

[81] H. W. Mewes, D. Frishman, C. Gruber, B. Geier, D. Haase, A. Kaps, K. Lemcke, G. Mannhaupt, F. Pfeiffer, C Schüller, S. Stocker, and B. Weil. MIPS: a database for genomes and protein sequences. *Nucleic Acids Research*, 28(1):37–40, 2000.

[82] T. Mitchell. *Machine Learning*. McGraw Hill, New York, 1997.

[83] N. Nariai, S. Kim, S. Imoto, and S. Miyano. Using protein-protein interactions for refining gene networks estimated from microarray data by bayesian networks. In *Pacific Symposium on Biocomputing (PSB03)*, pages 336–347, 2003.

[84] Y. Nesterov and A. Nemirovsky. *Interior Point Polynomial Methods in Convex Programming: Theory and Applications*. SIAM, Philadelphia, PA, 1994.

[85] A. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14 (NIPS01)*, 2002.

[86] A. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14 (NIPS01)*, pages 849–856, 2002.

[87] F.A. Nielsen, L.K. Hansen, and S.C. Strother. Canonical ridge analysis with ridge parameter optimization. *NeuroImage*, 7:S758, 1998.

[88] B. Parlett. *The symmetric eigenvalue problem*. Prentice Hall, Englewood Cliffs, NJ, 1980.

[89] T. Poggio, S. Mukherjee, R. Rifkin, A. Rakhlin, and A. Verri. b. In *Proc. of the Conference on Uncertainty in Geometric Computations*, pages 22–28, 2001.

[90] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

[91] R. Rosipal and L. Trejo. Kernel partial least squares regression in reproducing kernel hilbert space. *Journal of Machine Learning Research (JMLR)*, 2:2:97–123, 2002.

[92] R. Rosipal, L.J. Trejo, and B. Matthews. Kernel PLS-SVC for linear and nonlinear classification. In *Proc. of the Twentieth International Conference on Machine Learning (ICML03)*, pages 640–647, 2003.

[93] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[94] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[95] G. Scott and H. Longuet-Higgins. An algorithm for associating the features of two patterns. In *Proc. of the Royal Society London*, volume B244, pages 21–26, 1991.

[96] E. Segal, M. Shapira, A. Regev, D. Pe'er, D. Botstein, D. Koller, and N. Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature Genetics*, 34(2):166–76, 2003.

[97] E. Segal, H. Wang, and D. Koller. Discovering molecular pathways from protein interaction and gene expression data. *Bioinformatics*, 19 Suppl 1:i264–71, 2003.

[98] J. Shawe-Taylor and N. Cristianini. *Kernel methods for Pattern Analysis*. Cambridge University Press, Cambridge, U.K., 2004.

[99] J. Shawe-Taylor, N. Cristianini, and J. Kandola. On the concentration of spectral properties. In *Advances in Neural Information Processing Systems 14 (NIPS01)*, pages 511–517, 2002.

[100] J. Shawe-Taylor, C. Williams, N. Cristianini, and J. S. Kandola. On the eigenspectrum of the gram matrix and its relationship to the operator eigenspectrum. In *Proc. of the 13th International Conference on Algorithmic Learning Theory (ALT02)*, pages 23–40, 2002.

[101] N. Shental, A. Bar-Hillel, T. Hertz, and D. Weinshall. Computing gaussian mixture models with em using equivalence constraints. In *Advances in Neural Information Processing Systems 16 (NIPS03)*, pages 465–472, 2004.

[102] N. Shental, T. Hertz, D. Weinshall, and M. Pavel. Adjustment learning and relevant component analysis. In *Proc. of the 7th European Conference of Computer Vision (ECCV02)*, pages 776–792, May, 2002.

[103] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[104] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, 1981.

[105] E. Sonnhammer, S. Eddy, and R. Durbin. Pfam: a comprehensive database of protein domain families based on seed alignments. *Proteins*, 28(3):405–420, 1997.

[106] P. Spellman, G. Sherlock, M. Zhang, V. Iyer, K. Anders, M. Eisen, P. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, 9:3273–3297, 1998.

[107] J.F. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software, Special issue on Interior Point Methods (CD supplement with software)*, 11-12:625–653, 1999.

[108] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific Publishing Co., Pte, Ltd., Singapore, 2002.

[109] J. A. K. Suykens, T. Van Gestel, J. Vandewalle, and B. De Moor. A support vector machine formulation to PCA analysis and its kernel version. *IEEE Transactions on Neural Networks*, 14(2):447–450, 2003.

[110] J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

[111] S. Tavazoie, J. Hughes, M. Campbell, R. Cho, and G. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22(3):281–5, 1999.

[112] G. Thijs, K. Marchal, M. Lescot, S. Rombauts, B. De Moor, P. Rouze, and Y. Moreau. A gibbs sampling method to detect overrepresented motifs in the upstream regions of coexpressed genes. *Journal of Computational Biology*, 9(2):447–64, 2003.

[113] K. Tsuda. Support vector classification with asymmetric kernel function. In *Proc. of the European Symposium on Neural Networks (ESANN99)*, pages 183–188, 1999.

[114] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, Berlin, 1995.

[115] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, 2nd edition, 1999.

[116] J.-P. Vert and M. Kanehisa. Graph-driven features extraction from microarray data using diffusion kernels and kernel CCA. In *Advances in Neural Information Processing Systems 15 (NIPS02)*, pages 1425–1432, 2003.

[117] H.D. Vinod. Canonical ridge and econometrics of joint production. *J. Econometrics*, 4:147:166, 1976.

[118] A. Vinokourov, N. Cristianini, and J. Shawe-Taylor. Inferring a semantic representation of text via cross-language correlation analysis. In *Advances in Neural Information Processing Systems 15 (NIPS02)*, pages 1473–1480, 2003.

[119] C. von Mering, R. Krause, B. Snel, M. Cornell, S. G. Olivier, S. Fields, and P. Bork. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417:399–403, 2002.

[120] Y. Weiss. Segmentation using eigenvectors: A unifying view. In *Proc. of the International Conference on Computer Vision (ICCV99)*, pages 975–982, 1999.

[121] J. Weston, C. Leslie, D. Zhou, A. Elisseeff, and W. Noble. Semi-supervised protein classification using cluster kernels. In *Advances in Neural Information Processing Systems 16 (NIPS03)*, pages 595–602, 2004.

[122] J.H. Wilkinson. *The algebraic eigenvalue problem*. Oxford University Press, New York, USA, 1988.

[123] C. K. I. Williams and M. Seeger. Effect of the input density distribution on kernel-based classifiers. In *Proc. of Seventeenth International Conference on Machine Learning (ICML00)*, pages 1159–1166, 2000.

[124] H. Wold. Path models with latent variables: The NIPALS approach. In H.M. Blalock et al., editor, *Quantitative Sociology: International perspectives on mathematical and statistical model building*, pages 307–357. Academic Press, New York, 1975.

[125] H. Wold. Partial least squares. In S. Kotz and N.L. Johnson, editors, *Encyclopedia of the Statistical Sciences*, volume 6, pages 581–591. John Wiley & Sons, 1985.

[126] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15 (NIPS02)*, pages 505–512, 2003.

[127] E.P. Xing and M.I Jordan. On semidefinite relaxation for normalized k-cut and connections to spectral clustering. Technical Report CSD-03-1265, Division of Computer Science, University of California, Berkeley, 2003.

[128] Y. Yamanishi, J.-P. Vert, A. Nakaya, and M. Kanehisa. Extraction of correlated gene clusters from multiple genomic data by generalized kernel canonical correlation analysis. *Bioinformatics*, 19:323i–330i, 2003.

[129] Stella X. Yu and Jianbo Shi. Grouping with bias. In *Advances in Neural Information Processing Systems 14 (NIPS01)*, pages 1327–1334, 2002.

[130] H. Zha, C. Ding, M. Gu, X. He, and H. Simon. Spectral relaxation for k-means clustering. In *Advances in Neural Information Processing Systems 14 (NIPS01)*, pages 1057–1064, 2002.

Publication list

- T. De Bie, P. Monsieurs, K. Engelen, B. De Moor, N. Cristianini, and K. Marchal. Discovering transcriptional modules from motif, ChIP-chip and microarray data. In *Proc. of the Pacific Symposium on Biocomputing (PSB05)*, pages 483–494, 2005.
- T. De Bie, J. Suykens, and De Moor B. Learning from general label constraints. In *Proc. of IAPR International Workshop on Statistical Pattern Recognition (SPR04)*, pages 671–679. 2004.
- T. De Bie and N. Cristianini. Kernel methods for exploratory data analysis: a demonstration on text data. In *Proc. of the International Workshop on Statistical Pattern Recognition (SPR04)*, pages 16–29, 2004.
- T. De Bie, N. Cristianini, and R. Rosipal. Eigenproblems in pattern recognition. In E. Bayro-Corrochano, editor, *Handbook of Computational Geometry for Pattern Recognition, Computer Vision, Neurocomputing and Robotics*. Springer-Verlag, 2004.
- G. Lanckriet, T. De Bie, N. Cristianini, M. Jordan, W. Stafford Noble. A Statistical Framework for Genomic Data Fusion. *Bioinformatics*, 20(16): 2626–2635, 2004.
- T. De Bie and N. Cristianini. Convex methods for transduction. In *Advances in Neural Information Processing Systems 16 (NIPS03)*, pages 73–80, 2004.
- T. De Bie, M. Momma, and N. Cristianini. Efficiently learning the metric with side-information. In *Proc. of the 14th International Conference on Algorithmic Learning Theory (ALT03)*, pages 175–189, 2003.
- T. De Bie and B. De Moor. On the regularization of canonical correlation analysis. In *Proc. of the International Conference on Independent Component Analysis and Blind Source Separation (ICA03)*, pages 785–790, 2003.
- T. De Bie, B. De Moor. On two classes of alternatives to Canonical Correlation Analysis, using mutual information and oblique projections. In *Proc. of the 23rd symposium on information theory in the Benelux (ITB02)*, 2002.
- N. Cristianini, T. De Bie. Artificial Intelligence, Data Mining in Medicine, Expert Systems, Machine Learning, Support Vector Machines, and Neural Networks. In B. S. Everitt and C. Palmer (Eds.) *Encyclopaedic Companion to Medical Statistics*, Hodder Arnold, to appear.

Scientific curriculum vitae

Tijl De Bie was born in Aalst, Belgium, on February 17th, 1978. After attending primary school in Denderhoutem (Sint-Aloysius College), he went to the Sint-Martinus Instituut in Aalst, to study Latin-Mathematics. In September 1995, he went to the K.U.Leuven to pursue an education in applied sciences, where he received the Candidacy diploma in Applied Sciences in 1997 (summa cum laude with the congratulations of the board of examiners), and the Masters diploma in Electrical Engineering in 2000 (summa cum laude), graduating with a thesis on quantum information theory and quantum computation. Starting October 2000 he pursued his PhD research as a Research Assistant of the Fund for Scientific Research – Flanders (F.W.O.–Vlaanderen) in the research group ESAT-SCD, with Prof. Bart De Moor as his promotor. During the next 4 years he studied and performed research in a variety of domains (including quantum information theory, bioinformatics, to finally focus on machine learning in the last two years), and in a variety of places on one E.U. and three F.W.O. travel grants (two weeks in Imperial College London with Prof. Martin Plenio, six months in U.C.Berkeley with Prof. Laurent El Ghaoui and Prof. Michael Jordan, and four months in U.C.Davis with Prof. Nello Cristianini).