Explorations in Intelligible Classification

Michiel H. van der Ree

August 2014

Master's Thesis

Artificial Intelligence

Department of Artificial Intelligence

University of Groningen, the Netherlands

Supervisors:

Dr. M.A. Wiering

Artificial Intelligence and Cognitive Engineering, University of Groningen

Prof. dr. J.B.T.M. Roerdink

Scientific Visualization and Computer Graphics, University of Groningen

Dr. D.P. Williams

Scientific Visualization and Computer Graphics, University of Groningen

university of groningen

faculty of mathematics and natural sciences

Abstract

Machine learning (ML) methods are becoming increasingly popular in the identification of biomarkers in medical imaging data. While these methods provide superior predictive accuracy, their models can be hard to interpret. More conventional statistical methods lack the predictive accuracy of the ML models, but are able to identify regions in which the difference between a patient group and healthy controls is statistically significant. We attempted to strike a middle ground between these two extremes by employing supervised dimensionality reduction (SDR) methods in the identification of useful discriminative patterns for the differential diagnosis of various parkinsonisms based on FDG-PET scans. Additionally, a new SDR method based on margin maximization in a lower-dimensional space has been developed. Results indicate that our method performs on-par with existing methods in terms of accuracy, while at the same time providing an intelligible model through the generation of features that improve the performance of a radial kernel support vector machine. In addition, we are able to demonstrate that our method is able to learn an efficient low-dimensional representation of high-dimensional data.

Contents

Chapter 1

Introduction

Any introductory machine learning textbook will devote its first few pages to the distinction between *supervised* and *unsupervised* learning. Supervised techniques try to learn from labelled examples. A classification model might be trained to make the best guess whether a person is male or female based on their weight and hair length. A regression model might be able to predict the price of a house based on its number of rooms, the lot size and the average income in the neighborhood. Unsupervised techniques on the other hand, work with *unlabelled* data. Here, the aim is to find regularities in the input. Clustering techniques such as k-means are able to find groupings in the input space to the extent that such groupings are present. Dimensionality reduction is another form of unsupervised learning that tries to find a small number of underlying factors on the basis of which the input can be represented reasonably well.

In recent years, *semi-supervised* [51] methods have been used to solve supervised learning problems in which labelled examples are scarce while an abundance of unlabelled data is available. There the idea is that if we can gain insight into the underlying distribution of the data, we can generalize on the basis of just a few labelled examples.

The focus of this thesis is another cross-breed of supervised and unsupervised learning: *supervised dimensionality reduction* (SDR). Informally, SDR can be said to find the underlying factors in the data that best explain the class differences. It thus performs dimensionality reduction while taking class information into account. We illustrate the possible benefits of SDR here using, perhaps somewhat counterintuitively, an example of what it is not. In [36] Novembre et al. present the results of applying principal component analysis (PCA, an *unsupervised* dimensionality reduction technique) to genetic samples of 1,387 Europeans. The genetic samples consisted of about 200,000 loci on the DNA sequence. PCA finds the directions – principal components – which explain most of the variance in the input data. Finding the position of each sample along the first two directions and displaying them in a 2D scatter plot yielded the figure shown in fig. 1.1. As the figure shows, the projection onto the first two principal components

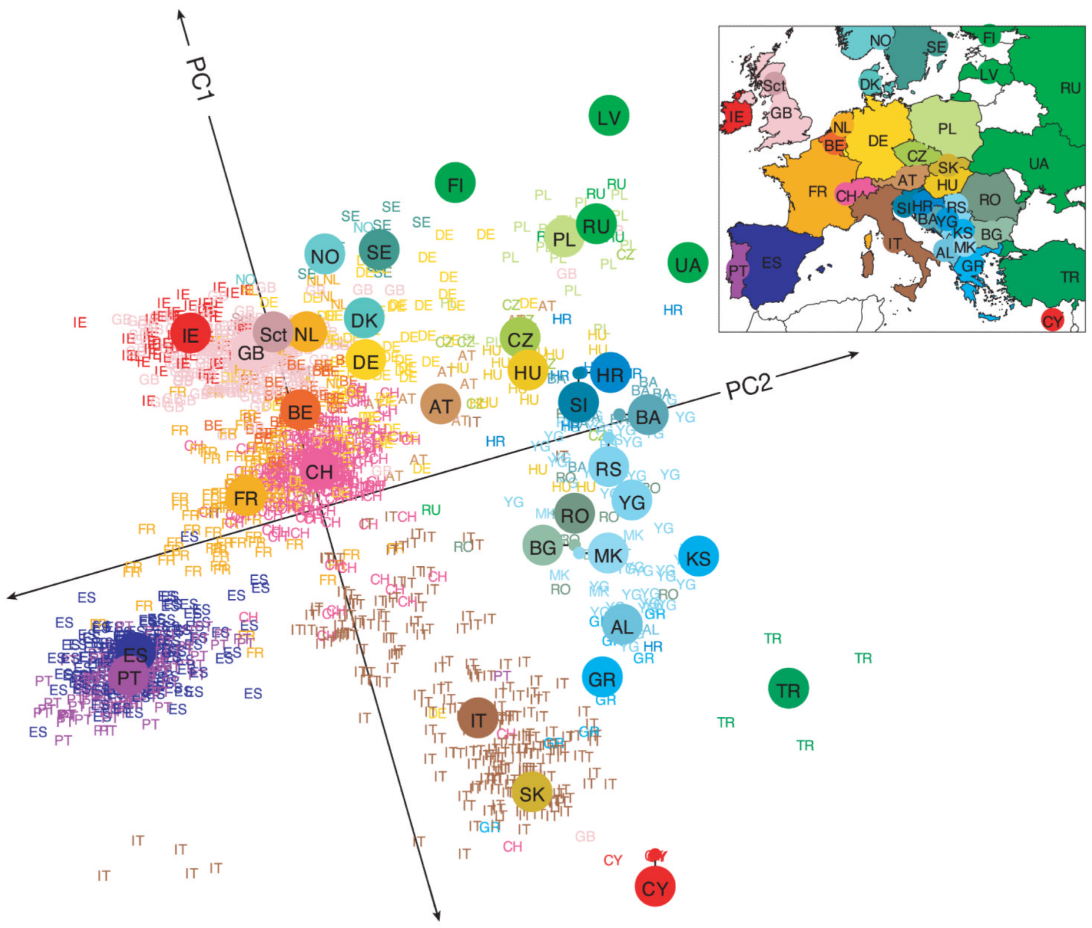


FIGURE 1.1: Projection of genetic samples of 1,387 Europeans onto the first two principal components of the dataset. Small coloured labels represent individuals and large coloured points represent median PC1 and PC2 values for each country. Figure from [36].

correlates with the geographical location of the country of origin of the subjects after rotation and scaling. Because of the way PCA works, the first two principal components now indicate at which loci the measurements covary the most across the data set. Because these components attribute relevancies to locations on the DNA sequence, they also show which loci are useful in predicting the country of origin of the person to which the sample belongs. That last property was never an explicit goal of the PCA procedure however, while it would be in SDR if we consider the countries of origin the class labels. In a way, Novembre et al. can be considered to have stumbled upon "discriminative components" – SDR tries not to leave such a discovery to chance.

A discriminative component indicating which of 200,000 loci have discriminative value might be hard to interpret. However, if the dataset we are working with is comprised of images, we can visualize such a component to show which regions of an image are relevant to the classification. This idea of "intelligible classification" is the theme of this thesis, and we will consider in particular the merits of supervised dimensionality reduction techniques in the differential diagnosis of parkinsonisms based on FDG-PET scans. In brain imaging analysis, there is a big difference between a classical statistical approach and a machine learning based approach. Making somewhat of a generalization, the difference between the two approaches can be characterized in the following way: Statisticians will be interested in a model that can be traced back to reality, enabling them to interpret a parameter in terms of a corresponding quantity. They do so at the expense of predictive power of the model, which is the focus of machine learning practicioners. The last-mentioned group will provide a model whose predictions are highly accurate, but which can't be easily interpreted. The idea of intelligible classification is to strike a middle ground between these two extremes.

The problem of finding discriminative components is inherently ill-posed. One of the properties we might seek is that the classes are well separated in the projected space. Starting from this insight, we propose a new supervised dimensionality reduction method using the margin-maximizing support vector machine algorithm. We will compare our "support vector components analysis" (SVCA) to more established techniques as neighborhood components analysis [27], local Fisher discriminant analysis [46] and Limited Rank Matrix LVQ [10].

Structure of this thesis This research attempts to answer two loosely coupled research questions:

1. How does support vector components analysis compare to other supervised dimensionality reduction methods?
2. What are the merits of supervised dimensionality reduction methods in the differential diagnosis of parkinsonisms based on FDG-PET scans?

In the second chapter we will review a number of supervised dimensionality reduction methods. The SVCA algorithm will be presented in the third chapter. Chapter 4 considers all particulars to the problem we are working on: we discuss the basic idea of a PET scan, the background and symptoms of the various parkinsonisms and review earlier work. The fifth chapter presents and discusses the results we obtained using our method on both an artificial dataset, a dataset from the UCI repository and two datasets of parkinsonian patients' PET scans. We discuss our findings in the sixth, concluding chapter.

Chapter 2

Supervised Dimensionality Reduction

There exists a variety of dimensionality reduction methods that take class labels into account. In this chapter we will review on three such methods: *neighbourhood components analysis* (NCA) [27], *local Fisher discriminant analysis* (LFDA) [46] and *Limited Rank Matrix Learning Vector Quantization* (LiRaM LVQ) [10]. We consider these three algorithms in particular for the following reasons: 1) Conceptually, they share similarities with the SVCA algorithm and thus set the stage well for the remainder of this chapter; 2) two of the three methods are considered classical tools in the relatively young field of supervised dimensionality reduction [10]; and 3) for each algorithm an optimized implementation is available online, enabling us to easily compare them with SVCA in the experiments of this thesis.

Suppose we have a labeled dataset consisting of P real-valued input vectors $\mathbf{x}_1, \ldots, \mathbf{x}_P$ in $\mathbb{R}^N$ and corresponding class labels $c_1, \ldots, c_P$. A matrix $\mathbf{T} \in \mathbb{R}^{K \times N}$ defines a linear map from $\mathbb{R}^N$ to $\mathbb{R}^K$ by mapping $\mathbf{x}$ to $\mathbf{Tx}$. At the most general level, the methods in this section all try to find a transformation matrix $\mathbf{T}$ that emphasizes the class differences.

2.1 Neighborhood Components Analysis

In NCA [27], we try to find a mapping to a vector space in which a nearest-neighbor classifier would perform well. When a k-nearest-neighbor classifier is presented a novel pattern $\mathbf{x}$, it considers the classes of its k nearest neighbors and assigns the new pattern the label of the majority of those neighbors. Here, the nearest neighbor of $\mathbf{x}$ is the training pattern $\mathbf{x}_i$ to which it has the smallest distance. The most common distance measure is the *Euclidean distance*:

$$d_{\text{Euc}}(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||. \tag{2.1}$$

In NCA we measure the Euclidean distance in the transformed space that we map to using matrix $\mathbf{T}$, resulting in a measure which takes the form of a Mahalanobis distance:

$$\begin{aligned} d^{\Lambda}(\mathbf{x},\mathbf{y}) &= ||\mathbf{T}(\mathbf{x}-\mathbf{y})|| \\ &= \sqrt{(\mathbf{x}-\mathbf{y})'\mathbf{T}'\mathbf{T}(\mathbf{x}-\mathbf{y})} \qquad (2.2) \\ &= \sqrt{(\mathbf{x}-\mathbf{y})'\Lambda(\mathbf{x}-\mathbf{y})} \qquad (2.3) \end{aligned}$$

where $\Lambda = \mathbf{T}' * \mathbf{T}$. Note that we can skip taking the root for computational simplicity, since we are only interested in finding the minimal distances. Conceptually, the idea of NCA is to find a transformation matrix $\mathbf{T}$ that maximizes the leave-one-out cross validation performance of a k-nearest-neighbor classifier. Since that performance w.r.t. $\mathbf{T}$ is piecewise continuous rather than differentiable, NCA formulates an alternative objective. This alternative estimates the performance of a classifier that assigns the left-out pattern $\mathbf{x}_i$ the class of training pattern $\mathbf{x}_j$ with a probability related to the transformed distance between $\mathbf{x}_i$ and $\mathbf{x}_j$. Using a softmax over the Euclidean distances in the transformed space, the probability that $\mathbf{x}_i$ has $\mathbf{x}_j$ as its nearest neighbor is expressed by

$$p_{ij} = \frac{\exp\left(-||\mathbf{T}(\mathbf{x}_i - \mathbf{x}_j)||^2\right)}{\sum_{k\neq i} \exp\left(-||\mathbf{T}(\mathbf{x}_i - \mathbf{x}_k)||^2\right)}, \qquad (2.4)$$

where p_{ii} (the probability that left-out pattern $\mathbf{x}_i$ will be $\mathbf{x}_i$'s nearest neighbor) is set to zero. The probability that $\mathbf{x}_i$ will be correctly classified is then expressed by

$$p_i = \sum_{j:c_j=c_i} p_{ij}. \qquad (2.5)$$

In NCA, the objective being maximized is *expected number of points correctly classified*:

$$J_{\mathrm{NCA}}(\mathbf{T}) = \sum_i \sum_{j:c_j=c_i} p_{ij} = \sum_i p_i. \qquad (2.6)$$

NCA tries to maximize this objective using gradient-based methods. Working out the matrix calculus yields the following derivative w.r.t. $\mathbf{T}$:

$$\frac{\partial J_{\mathrm{NCA}}}{\partial \mathbf{T}} = -2\mathbf{T} \sum_i \sum_{j:c_j=c_i} p_{ij}(\mathbf{x}_{ij}\mathbf{x}'_{ij} - \sum_k p_{ik}\mathbf{x}_{ik}\mathbf{x}'_{ik}) \qquad (2.7)$$

2.2 Local Fisher Discriminant Analysis

LFDA builds on the classic Fisher linear discriminant analysis (FDA) [18, 22]. In this section, we start by describing two-class FDA. Then we show how the method can be used for more than two classes prior to discussing the adaptations made in LFDA.

In two-class FDA, we project all patterns $\mathbf{x}$ onto a line with direction vector $\mathbf{t}$ by taking the scalar dot product

$$y = \mathbf{t}'\mathbf{x}. \tag{2.8}$$

We are interested in finding the vector $\mathbf{t}$ such that the projections y show great separation between the two classes. Such a $\mathbf{t}$ is found by maximizing the following objective function:

$$J_{\text{FDA-2}} = \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2} \tag{2.9}$$

where $\tilde{m}_\ell$ is the mean of the projected points belonging to class ℓ

$$\tilde{m}_\ell = \frac{1}{n_\ell} \sum_{i:c_i=\ell} \mathbf{t}'\mathbf{x}_i = \mathbf{t}'\mathbf{m}_\ell \tag{2.10}$$

with n_ℓ the number of patterns with label ℓ and $\mathbf{m}_\ell$ the mean of patterns with label ℓ in the input space. In equation 2.9, the $\tilde{s}_\ell^2$'s refer to the *scatter* of patterns with label ℓ:

$$\tilde{s}_\ell^2 = \sum_{i:c_i=\ell} (y_i - \tilde{m}_\ell)^2. \tag{2.11}$$

where y_i denotes the projection of $\mathbf{x}_i$ according to equation 2.8. Maximizing equation 2.9 means we seek a linear function $\mathbf{t}'\mathbf{x}$ for which the difference *between* class means is large while the scatter *within* a class is small. In other words, we want tight clusters that are far apart. Rewriting equation 2.9 in terms of scatter matrices in the input space we obtain

$$J_{\text{FDA-2}}(\mathbf{t}) = \frac{\mathbf{t}'\mathbf{S}_{\text{B}}\mathbf{t}}{\mathbf{t}'\mathbf{S}_{\text{W}}\mathbf{t}} \tag{2.12}$$

with *between-class scatter matrix*

$$\mathbf{S}_{\text{B}} = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)' \tag{2.13}$$

and *within-class scatter matrix*

$$\mathbf{S}_{\text{W}} = \sum_\ell \sum_{i:c_i=\ell} (\mathbf{x}_i - \mathbf{m}_\ell)(\mathbf{x}_i - \mathbf{m}_\ell)'. \tag{2.14}$$

The derivative of 2.12 w.r.t. $\mathbf{t}$ is given by

$$\frac{\partial J_{\text{FDA-2}}}{\partial \mathbf{t}} = \frac{2(\mathbf{S}_{\text{B}}\mathbf{t} \cdot \mathbf{t}'\mathbf{S}_{\text{W}}\mathbf{t} - \mathbf{S}_{\text{W}}\mathbf{t} \cdot \mathbf{t}'\mathbf{S}_{\text{B}}\mathbf{t})}{(\mathbf{t}'\mathbf{S}_{\text{W}}\mathbf{t})^2} \tag{2.15}$$

so we know that the $\mathbf{t}$ that maximizes equation 2.12 must satisfy

$$\begin{aligned} \mathbf{S}_{\text{B}}\mathbf{t} &= \frac{\mathbf{t}'\mathbf{S}_{\text{B}}\mathbf{t}}{\mathbf{t}'\mathbf{S}_{\text{W}}\mathbf{t}}\mathbf{S}_{\text{W}}\mathbf{t} \\ &= \lambda \mathbf{S}_{\text{W}}\mathbf{t} \end{aligned} \tag{2.16}$$

with λ a positive scalar. Equation 2.16 takes the form of a generalized eigenvalue problem, and can be simplified to a normal eigenvalue problem if $\mathbf{S}_{\mathrm{W}}$ is singular by pre-multiplying both sides with $\mathbf{S}_{\mathrm{W}}^{-1}$:

$$\mathbf{S}_{\mathrm{W}}^{-1}\mathbf{S}_{\mathrm{B}}\mathbf{t} = \lambda\mathbf{t}. \tag{2.17}$$

From equation 2.17 we can see that the transformation vector $\mathbf{t}$ we are looking for is an eigenvector of the matrix $\mathbf{S}_{\mathrm{W}}^{-1}\mathbf{S}_{\mathrm{B}}$. Since $\mathbf{S}_{\mathrm{B}}$ as in equation 2.13 we can reason that $\mathbf{m}_1 - \mathbf{m}_2$ has to be $\mathbf{S}_{\mathrm{B}}$'s only eigenvector, and hence $\mathbf{S}_{\mathrm{B}}\mathbf{t}$ will lie in the same direction as $\mathbf{m}_1 - \mathbf{m}_2$. Additionally, since we do not care about the scale factor of $\mathbf{t}$ but only about its direction, we can drop the λ and obtain the following expression maximizing equation 2.12:

$$\mathbf{t} = \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_1 - \mathbf{m}_2). \tag{2.18}$$

When the number of classes M is larger than 2, the within-class scatter matrix is still defined according to 2.14 [18]. The multi-class generalization of between-class scatter matrix $\mathbf{S}_{\mathrm{B}}$ is not as obvious:

$$\mathbf{S}_{\mathrm{B}} = \sum_{\ell} n_{\ell}(\mathbf{m}_{\ell} - \mathbf{m})(\mathbf{m}_{\ell} - \mathbf{m})' \tag{2.19}$$

where $\mathbf{m}$ is the mean of all patterns, regardless of class. The multi-class equivalent of 2.12 is defined as

$$J_{\mathrm{FDA}}(\mathbf{T}) = \frac{|\mathbf{T}\mathbf{S}_{\mathrm{B}}\mathbf{T}'|}{|\mathbf{T}\mathbf{S}_{\mathrm{W}}\mathbf{T}'|} \tag{2.20}$$

where $\mathbf{T} \in \mathbb{R}^{(M-1)\times N}$. Here, the determinant of the scatter matrices in the transformed space serves as a scalar measure of the scatter. An alternative definition can be encountered in the literature as well and yields the same solution for the optimal value of $\mathbf{T}$ [46]:

$$J'_{\mathrm{FDA}}(\mathbf{T}) = \mathrm{tr}\left((\mathbf{T}\mathbf{S}_{\mathrm{W}}\mathbf{T}')^{-1}\mathbf{T}\mathbf{S}_{\mathrm{B}}\mathbf{T}'\right) \tag{2.21}$$

Generalizing the two class case, an optimal $\mathbf{T}$ is formed by the generalized eigenvectors that correspond to the largest eigenvalues in

$$\mathbf{S}_{\mathrm{B}}\mathbf{t}_i = \lambda_i\mathbf{S}_{\mathrm{W}}\mathbf{t}_i \tag{2.22}$$

where the $\mathbf{t}_i'$'s form the rows of $\mathbf{T}$.

Local Fisher Discriminant Analysis redefines the scatter matrices in the following way:

$$\mathbf{S}_{\mathrm{W}} = \frac{1}{2}\sum_{i,j}\mathbf{W}_{\mathrm{W}}^{(i,j)}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)' \tag{2.23}$$

$$\mathbf{S}_{\mathrm{B}} = \frac{1}{2}\sum_{i,j}\mathbf{W}_{\mathrm{B}}^{(i,j)}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)' \tag{2.24}$$

where

$$\mathbf{W}_{\mathrm{W}}^{(i,j)} = \begin{cases} \mathbf{A}^{(i,j)}/n_\ell & \text{if } c_i = c_j = \ell \\ 0 & \text{if } c_i \neq c_j \end{cases} \tag{2.25}$$

$$\mathbf{W}_{\mathrm{B}}^{(i,j)} = \begin{cases} \mathbf{A}^{(i,j)}\left(\frac{1}{n} - \frac{1}{n_\ell}\right) & \text{if } c_i = c_j = \ell \\ 1/n & \text{if } c_i \neq c_j \end{cases} . \tag{2.26}$$

Here, $\mathbf{A} \in [0,1]^{P\times P}$ is the *affinity matrix*, where $\mathbf{A}^{(i,j)}$ expresses the affinity between $\mathbf{x}_i$ and $\mathbf{x}_j$. $\mathbf{A}^{(i,j)}$ is large if $\mathbf{x}_i$ and $\mathbf{x}_j$ are close to each other in the input space, and small if they are far apart. Several definitions of $\mathbf{A}$ are possible.

2.3 Limited Rank Matrix LVQ

Limited Rank Matrix LVQ [10] builds on Generalized Matrix LVQ [42], which in turn is based on learning vector quantization [30]. The idea of LVQ is fairly simple: initialize a number of prototypes, each associated with a class label (it is possible to have multiple prototypes per label), then (stochastically) iterate through all training patterns updating the nearest prototype $\mathbf{m}_\nu$ using the following procedure when visiting pattern $\mathbf{x}_i$

$$\mathbf{m}_\nu = \mathbf{m}_\nu + \Delta\mathbf{m}_\nu \tag{2.27}$$

where

$$\Delta\mathbf{m}_\nu = \begin{cases} \eta(\mathbf{x}_i - \mathbf{m}_\nu) & \text{if } \mathbf{x}_i \text{ and } \mathbf{m}_\nu \text{ have the same label} \\ -\eta(\mathbf{x}_i - \mathbf{m}_\nu) & \text{otherwise} \end{cases} \tag{2.28}$$

In other words: prototypes are locally pushed towards patterns with which they share a label and pushed away from patterns with a different label, the 'force' of this push being proportional to the distance between the pattern and the prototype. In [41] another, more flexible approach is introduced in which training is guided by minimization of a cost function

$$J_{\mathrm{GLVQ}} = \sum_i \Phi(\mu_i) \quad \text{where } \mu_i = \frac{d_J^{\mathbf{T}} - d_K^{\mathbf{T}}}{d_J^{\mathbf{T}} + d_K^{\mathbf{T}}} \tag{2.29}$$

where the quantity $d_J^{\mathbf{T}}$ ($d_K^{\mathbf{T}}$) corresponds to the distance of pattern $\mathbf{x}_i$ to its nearest *correct* (*wrong*) prototype. In equation 2.29, Φ is a monotonic function. In [10] and the following, Φ is simply the identity function $\Phi(x) = x$. As in NCA, the distance function takes the form of a Mahalanobis distance defined in equation 2.3.

The aim of the GMLVQ algorithm is to minimize equation 2.29 both with respect to the prototype configuration and the transformation matrix $\mathbf{T}$ in equation 2.3. In GMLVQ, $\mathbf{T}$ lies in $\mathbb{R}^{N\times N}$. LiRaM LVQ adapts this procedure by searching for $\mathbf{T}$ in $\mathbb{R}^{K\times N}$, with $K < N$.

During training, LiRaM LVQ minimizes equation 2.29 through stochastic gradient descent. Relevant derivatives are

$$\frac{\partial d_L^{\mathbf{T}}}{\partial \mathbf{m}_L} = -2\mathbf{T}'\mathbf{T}(\mathbf{x}-\mathbf{m}_L), \tag{2.30}$$

$$\gamma^+ = \frac{\partial \mu}{\partial d_J^{\mathbf{T}}} = \frac{2d_K^{\mathbf{T}}}{(d_J^{\mathbf{T}}+d_K^{\mathbf{T}})^2}, \tag{2.31}$$

and

$$\gamma^- = \frac{\partial \mu}{\partial d_K^{\mathbf{T}}} = \frac{-2d_J^{\mathbf{T}}}{(d_J^{\mathbf{T}}+d_K^{\mathbf{T}})^2}. \tag{2.32}$$

Here, we use the same notation as in [10] by having $L \in \{J, K\}$ and J (K) denotes the index of the closest correct (wrong) prototype. Prototypes $\mathbf{m}_J, \mathbf{m}_K$ are then updated according to

$$\mathbf{m}_J \leftarrow \mathbf{m}_J + \lambda_1 \cdot \gamma^+ \cdot \mathbf{T}'\mathbf{T}(\mathbf{x}-\mathbf{m}_J) \tag{2.33}$$

and

$$\mathbf{m}_K \leftarrow \mathbf{m}_K + \lambda_1 \cdot \gamma^- \cdot \mathbf{T}'\mathbf{T}(\mathbf{x}-\mathbf{m}_K) \tag{2.34}$$

The corresponding matrix update is given by

$$\mathbf{T} \leftarrow \mathbf{T} - \lambda_2 \cdot \frac{\partial \mu}{\partial \mathbf{T}} \tag{2.35}$$

with

$$\frac{\partial \mu}{\partial \mathbf{T}} = \gamma^+ \frac{\partial d_J^{\mathbf{T}}}{\partial \mathbf{T}} + \gamma^- \frac{\partial d_K^{\mathbf{T}}}{\partial \mathbf{T}} \tag{2.36}$$

and

$$\frac{\partial d_L^{\mathbf{T}}}{\partial \mathbf{T}} = 2\mathbf{T}(\mathbf{x}-\mathbf{m}_L)'(\mathbf{x}-\mathbf{m}_L). \tag{2.37}$$

After updates according to (2.35), $\mathbf{T}$ is normalized such that the sum of its elements squared equals 1.

Concluding remarks In this chapter we have reviewed three supervised dimensionality reduction algorithms. NCA tries to optimize the performance of a nearest neighbor classifier in the transformed space. LFDA makes the classic Fisher's discriminant analysis non-linear through the introduction of an affinity matrix. LiRaM LVQ is a prototype based method which minimizes an error function both through optimization of the prototype configuration and the transformation matrix. Both NCA and LiRaM LVQ are iterative algorithms without the theoretical guarantee that they will converge to a global optimum, so the initial $\mathbf{T}$ will influence their final performance. When using LFDA an initial $\mathbf{T}$ does not have to be provided, but in that case the definition of the affinity matrix is important. The next chapter will introduce a novel supervised dimensionality reduction algorithm called Support Vector Component Analysis.

Chapter 3

Support Vector Components Analysis

In this chapter we will introduce Support Vector Components Analysis (SVCA). Since SVCA builds uses the same concept as the support vector machine (SVM), we will start by describing the main theory behind SVMs.

3.1 The Support Vector Machine

In this section we treat the basic derivation of the objective function of the support vector machine algorithm [11, 43, 49]. Then we show how the decision surfaces of the algorithm can be made non-linear through the use of a kernel function.

Again consider training patterns $\mathbf{x}_i \in \mathbb{R}^N$ now with corresponding binary class labels $y_i \in \{\pm 1\}$. Support vector machines are firmly rooted in statistical learning or VC (Vapnik-Chervonenkis) theory [49]. VC theory considers the properties of learning machines whose parameters $\boldsymbol{\alpha}$ are optimized during training. Such a machine can be characterized by a function f assigning a label to a pattern $\mathbf{x}$, i.e. $f(\mathbf{x}, \boldsymbol{\alpha}) \in \{\pm 1\}$. The *risk* or *test error* of a machine defined by $\boldsymbol{\alpha}$ is given by

$$R(\boldsymbol{\alpha}) = \int \frac{1}{2}|f(\mathbf{x}, \boldsymbol{\alpha}) - y| dP(\mathbf{x}, y). \tag{3.1}$$

where $P(\mathbf{x}, y)$ is the joint distribution function of $\mathbf{x}$ and y. As long as we don't have an estimate of $P(\mathbf{x}, y)$ this expression is not very useful. A measure we *can* obtain is the *empirical risk* or *training error*:

$$R_{\text{emp}}(\boldsymbol{\alpha}) = \frac{1}{2P}\sum_{i=1}^{P} |f(\mathbf{x}_i, \boldsymbol{\alpha}) - y_i| \tag{3.2}$$

where P is the total number of training patterns. VC theory shows that the following bound on the test error holds with probability $(1-\delta)$ [49]:

$$R(\boldsymbol{\alpha}) \leq R_{\text{emp}}(\boldsymbol{\alpha}) + \phi(h, P, \delta) \tag{3.3}$$

with VC confidence ϕ defined as

$$\phi(h, P, \delta) = \sqrt{\frac{1}{P}\left(h\left(\log\frac{2P}{h}+1\right)+\log\frac{4}{\delta}\right)}. \tag{3.4}$$

and where h is the so-called *VC dimension* that measures the ability of the system to learn any training set without error. Intuitively, we can understand the role of h in the upper bound in (3.3) in the following way: if our system is able to explain any structure it encounters in the training data perfectly, it is very sensitive to small changes in that training data and thus has a high *variance*. A low training error should not lead us to think the system will generalize very well in that case and indeed, for high values of h the upper bound on the test error is large, regardless of the training error. Classifiers with a low VC dimension will have a lower VC confidence, but if a classifier is less able to explain any training set without error we can expect its training error to increase. In training the system, we want to obtain the optimal trade-off between low training error and low VC confidence. Vapnik has shown that the parameters $\boldsymbol{\alpha}$ of such an optimal classifier maximize the *margin* of that machine [49]. The margin is defined as the distance between the decision boundary and the instance(s) closest to it, as illustrated in figure 3.1.

3.1.1 The Support Vector Objective Function

Following the example of other SVM tutorials [11, 43], we start by explaining the training of a support vector machine that tries to linearly discriminate one class from another. The support vector machine algorithm realizes a decision function through the construction of a decision hyperplane

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{3.5}$$

corresponding to a decision function

$$y(\mathbf{x}) = \operatorname{sgn}(\mathbf{w} \cdot \mathbf{x} + b). \tag{3.6}$$

As explained in the introduction of this section, we aim to maximize the margin between the two classes in support vector classification. It can be shown that the problem of maximizing the margin is the same as that of minimizing

$$\frac{1}{2}||\mathbf{w}||^2 \tag{3.7}$$

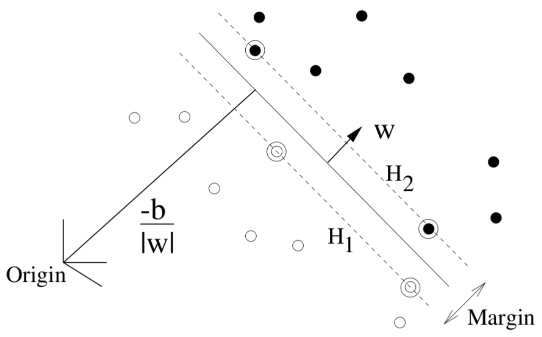


FIGURE 3.1: Illustration of the separating hyperplane and the margin of a support vector machine. The circled points in the scatter plot are support vectors. Image from [11].

subject to constraints

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1. \tag{3.8}$$

where y_i is the predicted label of $\mathbf{x}_i$ according to (3.6). So far we have assumed that the two classes can be completely separated by the hyperplane defined by equation 3.5. Of course, this might not be the case. We want to decrease the influence outliers might have on the algorithm by relaxing the constraints in (3.8). This can be done by introducing the slack variables $\boldsymbol{\xi}$. The problem now consists of finding

$$\min_{\mathbf{w},\boldsymbol{\xi},b} \frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi_i \tag{3.9}$$

subject to constraints

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i \quad \text{and} \quad \xi_i \geq 0 \tag{3.10}$$

with C a constant determining the trade-off between margin maximization and training error minimization. The problem defined by equations 3.9 and 3.10 is a constrained optimization problem. Such a problem can be rewritten through introduction of *Lagrange multipliers* $\boldsymbol{\alpha} \geq 0$ and a *Lagrangian*

$$\min_{\mathbf{w},\boldsymbol{\xi},b} \max_{\boldsymbol{\alpha},\boldsymbol{\eta}} \mathcal{L}(\mathbf{w},\boldsymbol{\xi},b,\boldsymbol{\alpha},\boldsymbol{\eta}) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi_i - \sum_i \eta_i\xi_i - \sum_i \alpha_i(y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i) \tag{3.11}$$

As we can see from equation 3.11, $\mathcal{L}$ has to be minimized w.r.t. *primal variables* $\mathbf{w}$, $\boldsymbol{\xi}$ and b and maximized w.r.t. the *dual variables* $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$. That means we are looking for a saddle point of $\mathcal{L}$.

The following might provide an intuition as to why a saddle point provides a solution to the problem [43]: Suppose a certain training pattern $\mathbf{x}_i$ violates the constraint that $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i$. Then $\mathcal{L}$ can be increased by increasing the corresponding α_i. Simultaneously, $\mathbf{w}$ and b will have to change such that $\mathcal{L}$ decreases. To prevent $\alpha_i(y_i(\mathbf{w} \cdot \mathbf{x}_i + b) + 1 - \xi_i)$ from becoming an arbitrarily large negative number, the change in $\mathbf{w}$ and b will be such that the constraint will eventually be satisfied. Additionally, one can see from equation 3.11 that for training samples which don't *exactly* meet the constraint, i.e. $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) > 1 - \xi_i$, the corresponding α_i must be 0, since that is the value by which $\mathcal{J}$ is maximized in that case.

Since we are looking for a saddle point, we are looking for a point at which the derivatives of equation 3.11 with respect to the primal variables vanish,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \frac{\partial \mathcal{L}}{\partial \xi_i} = \frac{\partial \mathcal{L}}{\partial b} = 0 \tag{3.12}$$

which means that at the saddle point

$$\eta_i = C - \alpha_i, \quad \sum_i \alpha_i y_i = 0 \quad \text{and} \quad \mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i. \tag{3.13}$$

Substituting the expression for $\mathbf{w}$ into equation 3.6 we can rewrite our decision function as a linear combination of a subset of the training patterns $\mathbf{x}_i$:

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_i y_i \alpha_i (\mathbf{x} \cdot \mathbf{x}_i) + b\right). \tag{3.14}$$

Equation 3.14 is the so-called Support Vector expansion, with training patterns corresponding to a non-zero α value being dubbed the support vectors. Substituting the expression in 3.13 into equation 3.11 we eliminate primal variables $\mathbf{w}$, $\boldsymbol{\xi}$ and b and arrive at the dual optimization problem, which is the problem that is usually solved in practice:

$$\max_{\boldsymbol{\alpha}} \mathcal{Q}(\boldsymbol{\alpha}) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \tag{3.15}$$

subject to constraints

$$0 \le \alpha_i \le C \quad \text{and} \quad \sum_i \alpha_i y_i = 0. \tag{3.16}$$

3.1.2 Non-Linearity in Support Vector Machines

Barring outliers, we have thus far assumed that the two classes can be separated by a hyperplane. One way to make the SVM have a non-linear decision boundary is to preprocess the training patterns by some map Φ and then applying the support vector algorithm on the mapped patterns. For instance, training an SVM on patterns mapped by $\Phi : \mathbb{R}^2 \to \mathbb{R}^3$ with $\Phi(\mathbf{x}) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$ will correspond to a quadratic decision boundary in the original space. While this approach seems reasonable in this particular example, it can easily become computationally infeasible for both polynomial features of higher order and higher dimensionality [43].

A key observation is that SVM training only depends on labels and inner products between patterns. Rather than explicity defining some mapping to a feature space, we can implicitly map to another space by plugging in a similarity measure different from the inner product. Such a similarity measure $K(\mathbf{x}_i, \mathbf{x}_j)$ between patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ is called a kernel function. Mercer's theorem states that for every continuous positive definite function $K(\mathbf{x}_i, \mathbf{x}_j)$ there exists a corresponding mapping Φ such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)'\Phi(\mathbf{x}_j)$. The most popular kernels are the polynomial kernel and the radial basis function kernel.

3.1.3 Extension to Multiclass Problems

By definition, support vector machines solve binary classification problems. There are multiple ways to extend them to problems where the number of classes M is greater than two [43].

One-versus-rest The most straightforward way is to train M classifiers $f_1, \ldots, f_M$, each trained to distinguish one class from the rest. The actual classification is then performed by assigning a new pattern $\mathbf{x}$ the label of the system with the highest output, i.e.

$$\arg\max_{\ell} g_\ell(\mathbf{x}), \text{ where } g_\ell(\mathbf{x}) = \sum_i y_i^\ell \alpha_i^\ell K(\mathbf{x}, \mathbf{x}_i) + b^\ell \tag{3.17}$$

with y_i^ℓ being 1 if $c_i = \ell$ and -1 otherwise. The drawback of this method is that it's somewhat heuristic and supposes that the outputs of the separate classifiers are of a comparable magnitude. If that assumption does not hold, the system might be biased towards the class whose associated classifier has a larger output overall.

Pairwise classification In pairwise classification $(M-1)M/2$ binary classifiers are trained, one for each of the possible combinations of two of the classes. For example, a three-class problem will be learned through use of a 1 vs. 2, a 1 vs. 3 and a 2 vs. 3 classifier. Classification consists of each of the separate systems taking a vote on the label of the presented pattern, and the majority vote determines the predicted label.

Error-correcting output coding The basic idea of error-correcting output coding [4, 16] is to define a decoding matrix $D \in \{\pm 1\}^{M \times L}$ that assigns each class a unique vector in $\{\pm 1\}^L$. Then, we train L binary classifiers with training set labels defined by the columns of D. During classification, the concatenated outputs of all L classifiers can be considered to be a code. The class associated with the column of D to which this output code has the closest Hamming distance is the predicted class. In [16] a few heuristics for defining D are suggested. Additionally, domain knowledge might be useful in defining the class-related output codes.

Multi-class objective functions The most elegant solution to the multiclass problem is to modify the SVM objective in such a way that it simulataneously allows the computation of a multi-class classifier. In [13] the notion of the margin is generalized to multi-class probems.

In [43] it is noted that there is no multi-class approach that generally outperforms the others. In most cases, the simple one-against-rest approach produces acceptable results.

3.2 Support Vector Components Analysis

The basic idea of SVCA is to learn a transformation matrix **T** such that the margin between classes in the transformed space is maximized. If we train a one-versus-rest SVM for each class in the transformed space, the primal objective of each machine is equal to that of the normal SVM:

$$\min J^{\ell}(\mathbf{w}, \boldsymbol{\xi}) = \left[\frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi_i\right] \tag{3.18}$$

now subject to constraints

$$y_i^{\ell}(\mathbf{w} \cdot \mathbf{Tx} + b) \geq 1 - \xi_i \quad \text{and} \quad \xi_i \geq 0. \tag{3.19}$$

Correspondingly, the new 'dual' objective is defined as:

$$\min_{\mathbf{T}} \max_{\boldsymbol{\alpha}} \mathcal{Q}^{\ell}(\boldsymbol{\alpha}; \mathbf{T}) = \left[\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i^{\ell} y_j^{\ell} \left(\mathbf{Tx}_i \cdot \mathbf{Tx}_j\right)\right] \tag{3.20}$$

subject to constraints

$$0 \leq \alpha_i \leq C \quad \text{and} \quad \sum_i \alpha_i y_i^{\ell} = 0. \tag{3.21}$$

Finally, the kernelized counterpart of (3.20) is given by

$$\min_{\mathbf{T}} \max_{\boldsymbol{\alpha}} \mathcal{Q}^{\ell}(\boldsymbol{\alpha}; \mathbf{T}) = \left[\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i^{\ell} y_j^{\ell} K(\mathbf{Tx}_i, \mathbf{Tx}_j)\right] \tag{3.22}$$

subject to the constraints in (3.21) as well.

3.2.1 Training Procedure

We use the following procedure to find the "support vector components": First, we solve the quadratic programming subproblem of finding the $\boldsymbol{\alpha}^{\ell}$ that maximizes the expression in (3.22) for each of the one-versus-rest support vector machines. Since that expression is equal to the objective being maximized in normal support vector machine training, we

Kernel	Kernel function $K(\mathbf{Tx}_i, \mathbf{Tx}_j)$	$\partial K/\partial \mathbf{T}$
linear	$\mathbf{x}_i'\mathbf{T}'\mathbf{T}\mathbf{x}_j$	$\mathbf{T}(\mathbf{x}_i\mathbf{x}_j' + \mathbf{x}_j\mathbf{x}_i')$
polynomial	$(\mathbf{x}_i'\mathbf{T}'\mathbf{T}\mathbf{x}_j + 1)^d$	$d(\mathbf{x}_i'\mathbf{T}'\mathbf{T}\mathbf{x}_j + 1)^{d-1}\mathbf{T}(\mathbf{x}_i\mathbf{x}_j' + \mathbf{x}_j\mathbf{x}_i')$
RBF	$\exp(-\gamma\lvert\lvert\mathbf{Tx}_i - \mathbf{Tx}_j\rvert\rvert^2)$	$-\gamma\exp(-\gamma\lvert\lvert\mathbf{Tx}_i - \mathbf{Tx}_j\rvert\rvert^2)\mathbf{T}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)'$

TABLE 3.1: Linear, polynomial and radial basis function SVM kernels and their derivatives in SVCA.

can use tried and tested optimization methods such as sequential minimal optimization (SMO) [38] to do so.

Then, for the optimized $\boldsymbol{\alpha}^\ell$'s, we can try to minimize the sums of all dual-objectives w.r.t. $\mathbf{T}$ using stochastic gradient descent. In stochastic gradient descent, we minimize $\sum_\ell \mathcal{Q}^\ell$ by minimizing the gradients of single examples. Writing the expression in (3.22) as

$$\mathcal{Q}^\ell = \sum_i q_i^\ell \quad \text{with single example terms} \quad q_i^\ell = \alpha_i^\ell - \frac{1}{2}\alpha_i^\ell y_i^\ell \sum_j \alpha_j^\ell y_j^\ell K(\mathbf{Tx}_i, \mathbf{Tx}_j) \tag{3.23}$$

we find the following derivative of q_i^ℓ w.r.t. $\mathbf{T}$:

$$\frac{\partial q_i^\ell}{\partial \mathbf{T}} = -\frac{1}{2}\alpha_i^\ell y_i^\ell \sum_j \alpha_j^\ell y_j^\ell \frac{\partial K(\mathbf{Tx}_i, \mathbf{Tx}_j)}{\partial \mathbf{T}}. \tag{3.24}$$

Table 3.1 lists the derivatives of the three most popular kernels.

We alternate between optimizing all $\boldsymbol{\alpha}^\ell$'s and $\mathbf{T}$ a preset number of times. The entire training scheme is expressed in algorithm 3.1. Alternatively, we can use batch gradient descent and adjust $\mathbf{T}$ using $\sum_\ell \partial\mathcal{Q}^\ell/\partial\mathbf{T}$ instead of its single example based estimate.

A quick glance at table 3.1 reveals a problem for the SVCA algorithm: the sizes of the updates are proportional to the magnitudes of the transformed vectors. Since the magnitudes of transformed vectors tend to grow during training, this results in a positive

Algorithm 3.1 Support Vector Components Analysis – Stochastic

```
initialize $\mathbf{T}$ and SVM parameters
**repeat**
  For each one-versus-rest support vector machine, solve the QP subproblem:
  $\begin{cases} \boldsymbol{\alpha}^\ell \leftarrow \arg\max_{\boldsymbol{\alpha}'} \left[\sum_i \alpha_i' - \frac{1}{2}\sum_{i,j}\alpha_i'\alpha_j' y_i y_j K(\mathbf{Tx}_i, \mathbf{Tx}_j)\right] \\ 0 \le \alpha_i \le C \\ \sum_i \alpha_i y_i = 0 \end{cases}$
  **repeat**
    Randomly draw $\nu$ from $\{1, \dots, P\}$
    $\mathbf{T} \leftarrow \mathbf{T} - \lambda \sum_\ell \partial q_\nu^\ell / \partial \mathbf{T}$
  **until** executed $n_\mathbf{T}$ times
**until** executed $n_{\max}$ times
```

feedback loop causing erratic learning behavior. We have considered two solutions for this problem:

1. Divide the update steps by the norm of the transformation matrix, $\|\mathbf{T}\|$. The norm of the transformation matrix provides an upper bound on magnitudes of transformed vectors, so this division compensates for the increasing magnitudes.

2. Only use the sign of the gradients. Resilient backprop (RPROP, [39]) is an optimization technique originally developed for training multi-layer perceptrons. It defines update-values Δ_{ij} for each of the elements of our transformation matrix. During training, these update-values themselves are adapted in the following way:

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^{+} \cdot \Delta_{ij}^{(t-1)} & \text{if } \frac{\partial \sum \mathcal{Q}^{\ell}}{\partial \mathbf{T}_{ij}}^{(t-1)} \cdot \frac{\partial \sum \mathcal{Q}^{\ell}}{\partial \mathbf{T}_{ij}}^{(t)} > 0 \\ \eta^{-} \cdot \Delta_{ij}^{(t-1)} & \text{if } \frac{\partial \sum \mathcal{Q}^{\ell}}{\partial \mathbf{T}_{ij}}^{(t-1)} \cdot \frac{\partial \sum \mathcal{Q}^{\ell}}{\partial \mathbf{T}_{ij}}^{(t)} < 0 \\ \Delta_{ij}^{(t-1)} & \text{otherwise} \end{cases} \tag{3.25}$$

   where $0 < \eta^{-} < 1 < \eta^{+}$. Elements of the transformation matrix are then adjusted according to

$$\mathbf{T}_{ij} \leftarrow \mathbf{T}_{ij} - \operatorname{sgn}(\partial \textstyle\sum \mathcal{Q}^{\ell} / \partial \mathbf{T}_{ij}) \cdot \Delta_{ij} \tag{3.26}$$

   In the case that the partial derivative w.r.t. $\mathbf{T}_{ij}$ changes sign, the last change to $\mathbf{T}_{ij}$ is made undone.

Both solutions have been extensively used during explorative experiments. In the experimental section of this thesis, we have used the RPROP algorithm since that has an additional benefit: During the first few update steps, the number of support vectors will strongly depend on the value of γ if using a radial kernel. The value of $\sum \mathcal{Q}_{\ell}$ will be large with a lot of support vectors, which means the initial update steps will be large when using the value of the gradient. This introduces a strong dependency between γ and the learning rate if using the value of the gradients. Similarly, the value of $\sum \mathcal{Q}_{\ell}$ will depend on C. RPROP's update sizes do not depend on the value of $\sum \mathcal{Q}_{\ell}$, making it much easier to optimize parameters C and γ. Note that RPROP is a batch-based algorithm.

3.2.2 Comparison with Other Methods

In this subsection, we show that the objective maximized in neighborhood components analysis is closely related to the SVCA objective using an RBF kernel. In addition, we review similar work on learning SVM kernels.

Similarity to the NCA objective If we rearrange terms we can obtain the following expression for the multi-class dual objective:

$$J_{\text{SVC}} = \sum_i \left[\sum_\ell \alpha_i^\ell - \frac{1}{2} \sum_\ell \sum_i \alpha_i^\ell \alpha_j^\ell y_i^\ell y_j^\ell K(\mathbf{T}\mathbf{x}_i, \mathbf{T}\mathbf{x}_j) \right] \tag{3.27}$$

where the scalar product $y_i^\ell y_j^\ell$ can only take the following two values:

$$y_i^\ell y_j^\ell = \begin{cases} 1 & \text{if } (\ell = c_i \wedge \ell = c_j) \vee (\ell \neq c_i \wedge \ell \neq c_j) \\ -1 & \text{if } (\ell = c_i \wedge \ell \neq c_j) \vee (\ell \neq c_i \wedge \ell = c_j) \end{cases} \tag{3.28}$$

Plugging these values into (3.27) we obtain the following expression:

$$\begin{aligned} J_{\text{SVC}} = &\sum_i \left[\sum_\ell \alpha_i^\ell - \frac{1}{2} \sum_{j:c_j = c_i} \alpha_i^{c_i} \alpha_j^{c_i} K(\mathbf{T}\mathbf{x}_i, \mathbf{T}\mathbf{x}_j) + \frac{1}{2} \sum_{j:c_j \neq c_i} \alpha_i^{c_i} \alpha_j^{c_i} K(\mathbf{T}\mathbf{x}_i, \mathbf{T}\mathbf{x}_j) \right. \\ &\left. + \frac{1}{2} \sum_{\ell \neq c_i} \sum_{j:c_j = c_i} \alpha_i^\ell \alpha_j^\ell K(\mathbf{T}\mathbf{x}_i, \mathbf{T}\mathbf{x}_j) - \frac{1}{2} \sum_{\ell \neq c_i} \sum_{j:c_j \neq c_i} \alpha_i^\ell \alpha_j^\ell K(\mathbf{T}\mathbf{x}_i, \mathbf{T}\mathbf{x}_j) \right]. \end{aligned} \tag{3.29}$$

To show the similarity of our method to NCA, we fix the α coefficients to the following values:

$$\alpha_i^\ell = \begin{cases} 2A & \text{if } c_i = \ell \\ 0 & \text{otherwise} \end{cases} \tag{3.30}$$

which yields the following expression:

$$J'_{\text{SVC}} = 2A \sum_i \left[n_{c_i} - A \sum_{j:c_j = c_i} K(\mathbf{T}\mathbf{x}_i, \mathbf{T}\mathbf{x}_j) \right]. \tag{3.31}$$

If we would use the RBF kernel with $\gamma = 1$ we would see that the minimization of (3.31) is very similar to the maximization of (2.6), save for the normalizing denominators in the NCA objective. In fact, we should be able to choose individual scaling factors A_i such that they perform the same role as those normalizing constants. The difference between our method and NCA is that rather than fixing these values beforehand, we optimize them during training using the support vector machine algorithm. In addition, by being able to plug in a different kernel than the radial basis function, we are able to control the complexity of the hypotheses proposed by the system in the transformed space.

Related work in SVM kernel learning In [37], Pietersma et al. suggest to learn an RBF kernel by minimizing the objective with respect to feature-scaling parameters. This is equal to optimizing a square transformation matrix $\mathbf{T}$ with non-zero elements on only the diagonal. While this “weighted RBF” (WRBF) kernel learning is able to obtain significantly higher accuracies than an RBF SVM on a noisy dataset, it does not learn a mapping to a lower-dimensional space since the corresponding transformation matrix **T** is square. At best, it can perform some sort of dimensionality reduction by only using those features with high weights.

Wiering et al. [50] propose a "neural support vector machine" that uses an ensemble of neural networks to learn a non-linear mapping that minimizes an SVM's objective. Each neural network learns a single feature node, and its weights are adjusted such that the learned representation minimizes the objective. While the system performs on par with state-of-the-art machine learning methods, it is less suited to perform supervised dimensionality reduction since it runs the risk of learning strongly correlated feature nodes. The outer products in the update step of SVCA learning (see tbl. 3.1) make SVCA less prone to this pitfall.

SVCA shares many of its keywords with the large margin nearest neighbor algorithm (LaMaNNA) [17]. LaMaNNA's goals are quite different however. It trains an SVM on a dataset, and when it is presented a new pattern finds the support vector which in the input space is that pattern's nearest neighbor. Then, it supposes that the direction through that support vector, perpendicular to the decision boundary is the most relevant to the class difference and uses that direction to estimate feature relevances specific to the test pattern. It then uses the obtained relevances to perform k-nearest-neighbor classification. Unlike SVCA, LaMaNNA calculates a transformation matrix **T** for each new test pattern so it does not provide a fixed map for the entire dataset. Additionally, its transformation matrix is square so it does not learn a mapping to a lower-dimensional space.

Concluding remarks In this chapter we have introduced the SVCA algorithm. Building on earlier work on SVM kernel learning, SVCA optimizes the objective of an SVM in the transformed space. Like NCA and LiRaM LVQ, the performance of SVCA will depend on the initial **T**. SVCA and the SDR algorithms discussed in the second chapter will be put to the test in the fifth chapter of this thesis. The next chapter will focus on earlier work on the identification of significant patterns in PET-scans of people with various parkinsonisms.

Chapter 4

Finding Parkinsonian Biomarkers in PET-Scans

This chapter starts with a brief discussion of the workings of positron emission tomography (PET). We then treat the different parkinsonisms prior to reviewing conventional techniques in brain imaging analysis and earlier machine learning work on Parkinson's disease.

4.1 Positron Emission Tomography

Tomography is the practice of imaging by virtual sectioning through the use of a penetrating wave. In the case of PET, these waves are gamma ray photons emitted from inside the scanned object [8]. Positrons are emitted from a radionuclide according to:

$$\boxed{{}^{1}_{1}p \longrightarrow {}^{1}_{0}n + {}^{0}_{1}\beta + \nu}$$

After emission, positive charge ν is carried away with a positron ${}^{0}_{1}\beta$. The positron then loses kinetic energy by interaction with the surrounding matter. Once it is essentially at rest it collides with an electron. When a positron and an electron collide both particles annihilate and two photons are produced. The directions at which the photons are given off have an angle between them close to 180° as shown in figure 4.1. The photons are emitted in opposite directions to conserve momentum, which is close to zero before the annihilation. In coincidence tomography, the object being scanned is at the center of a circle of gamma sensitive detectors. When two detectors both detect a photon within a certain time window these two quanta are assumed to have been the products of the same annihilation. Together, the locations of the excited sensors define a line of response. The annihilation can be assumed to have taken place somewhere along this line, but since exact photon arrival times are not compared it is unclear where *exactly*

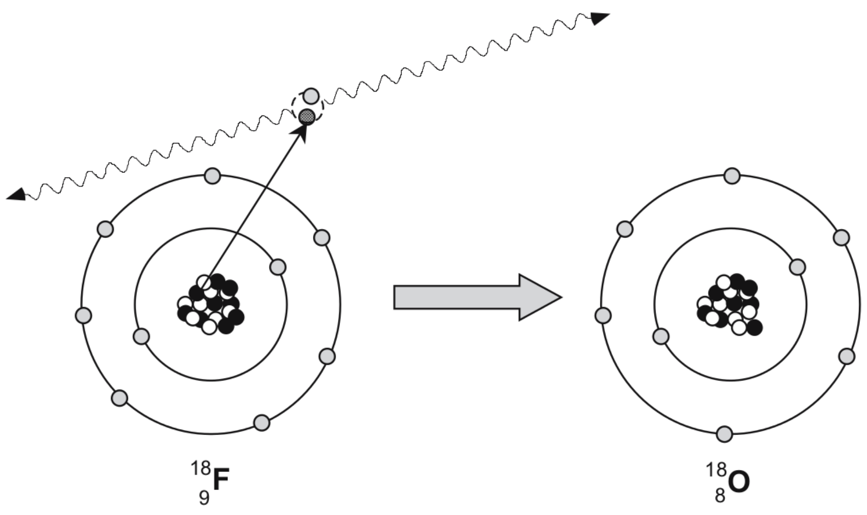


FIGURE 4.1: Illustration of annihilation radiation. Figure taken from [8]

along the line the positron collided with an electron. However, by applying tomographic reconstruction algorithms [15] to the entire set of lines of response collected during the scan, estimates of the locations of the nuclei releasing the positrons can be retrieved. For the locations of the *annihilations* to provide a decent clue to the locations of the emitting *nuclei* (since it are those location that we are interested in) it is important that the distance travelled by the positron before colliding is small.

The positron-emitting radionuclide in PET is called a *tracer*. Two popular tracers are fluorodopa (FDOPA) and fluorodeoxyglucose (FDG). In both tracers the fluorine radioisotope ^{18}F is the source of positrons. One of the benefits of ^{18}F is that its positrons have a relatively low energy, resulting in a small distance between nucleus and annihiliation locations. The difference between FDOPA and FDG is the type of activity they map, since each tracer will tend to concentrate in a different type of location. FDOPA quantifies the deficiency of dopamine synthesis and storage in nerve terminals. FDG allows for the measurement of cerebral metabolic rate of glucose.

4.2 Parkinson's Disease

In 1817 James Parkinson published a monogograph titled "An Essay on the Shaking Palsy" in which he described six individuals with a neurological condition characterized by a resting tremor and a progressive motor disability. Today we know this condition as *Parkinson's disease* [40]. The prevalence of the disease in industrialised countries is estimated to be 0.3% of the general population and about 1% of the population older than age 60 years. The mean age of onset is early to mid 60's. The *parkinsonism* syndrome is characterized by rigidity, tremor and bradykinesia. A 3-5 Hz resting tremor is the first sign in 70% of PD patients. Rigidity is the raised resistance noted during passive joint movement. Bradykinesia refers to a slowness in the execution of movement, and

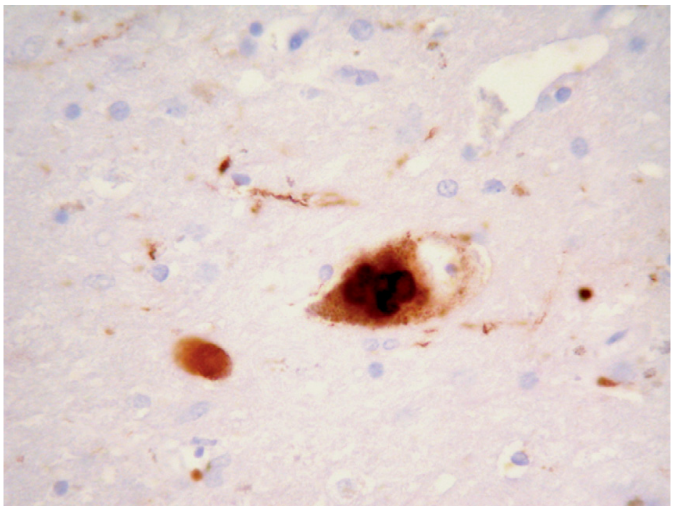


FIGURE 4.2: Light microscopy shows a surviving neuron full of Lewy bodies in the substantia nigra of a patient with Parkinson's disease. Image from [31].

initially manifests itself by difficulties with motor tasks. Idiopathic Parkinson's disease (IPD) is the main cause of parkinsonism. At a neurological level, IPD is characterized by a reduced number of dopamine neurons in the substantia nigra. In addition, Lewy bodies and Lewy neurites are present in the remaining neurons. Figure 4.2 displays a microphotograph of a PD patient which shows the presence of Lewy bodies in a remaining neuron.

While parkinsonism is mainly associated with Parkinson's disease, there are a few "Parkinson-plus" diseases that include parkinsonism with other clinical signs. The most prevalent of these atypical parkinsionian syndromes are multiple systems atrophy (MSA) and progressive supranuclear palsy (PSP). About 80% of patients misdiagnosed as having idiopathic Parkinson's disease actually have MSA or PSP [47]. Accurate early diagnosis of parkinsonian disorders is important for at least three reasons. First, prognoses of the various parkinsonism vary greatly: IPD does not significantly shorten the lifespan while MSA and PSP patients are expected to live only a few years after diagnosis. Second, the variations require different treatments. Finally, for the results of clinical trials of disease-modifying drugs on patients in an early phase of the disease to be reliable, patients' diagnoses have to be reliable. In recent years, analysis of PET scans obtained by using the FDG tracer has been particularly effective in identifying metabolic patterns that discriminate between the various parkinsonisms.

4.3 Statistical Parametric Mapping

The main paradigm in brain image analysis is *statistical parametric mapping* [5, 23, 24]. Statistical parametric mapping (SPM) is a voxel-based approach which identifies regionally specific responses to experimental factors. The procedure consists of the following steps:

1. Spatial processing
2. Estimating the voxel-by-voxel parameters of a statistical model
3. Making inferences about those parameters with appropriate statistics

In PET studies, the most important step of spatial processing is *spatial normalization.* This normalization ensures that scans of different people can be compared to each other in a sensible way. Subjects' anatomies will differ. Some voxel might record activity in one subject's substantia nigra and activity in another subject's subthalamic nucleus. By estimating a set of warping parameters each individual scan can be mapped to a standard anatomical space [24]. This step is not unique to SPM, the other methods described in this chapter perform it as well.

Once voxel values are mapped to a common anatomical template, SPM builds *statistical parametric maps* (SPMs) by estimating voxel-by-voxel parameters of a statistical model. In one of the most simple cases, this might amount to calculating the t-statistic at each voxel:

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s_{X_1X_2}\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}} \tag{4.1}$$

where $\bar{X}_1, \bar{X}_2$ will be the mean value of that voxel for two groups (for instance PD patients and a control group), n_1, n_2 the sizes of the two groups and $s_{X_1X_2}$ an estimator for the common standard deviation of the two samples.

In a normal univariate t-test, we would look up what t-value would constitute a significant difference between the two groups. For instance, we could want that the probability that we incorrectly reject the null hypothesis that the two groups actually have the same means in an experiment with 30 degrees of freedom is not greater than 5%. This corresponds to a t-value larger than 1.70. Now suppose we are considering 1000 voxels in total. By the definition of the t-value, any separate voxel will have a 5% chance of having an associated t-value larger than 1.70. That means if the two samples are drawn from the same distribution, we can expect to find about 50 voxels with a significant t-value. Seeing one or more t-values above 1.70 in this familiy of tests is not good evidence against the *family-wise null hypothesis* [9] that all these values have been drawn from a null distribution. Instead, we would like to find a threshold such that in a family of 1000 t-values there is a 5% probability of there being one or more values above that threshold. We can obtain such a threshold through *Bonferroni correction.* The family-wise error rate P^{FWE} is the probability that one or more values will be greater than α:

$$P^{\text{FWE}} = 1 - (1-\alpha)^n \tag{4.2}$$

Where n is the total number of voxels measured. Since α is small, P^{FWE} can be approximated by

$$P^{\text{FWE}} \leq n\alpha \tag{4.3}$$

so the threshold value given a family-wise p-value is given by

$$\alpha = P^{\mathrm{FWE}}/n. \tag{4.4}$$

In most cases, the threshold in 4.4 will be much too conservative. In calculating the family-wise error rate, we have assumed that all voxels are *statistically independent*. In imaging studies this will hardly be the case. In PET scans particularly, voxels are related because of the way in which the scanner collects and reconstructs the image. Due to this spatial correlation, the number of independent observations is smaller than the number of voxels.

In random field theory (RFT) [9, 25] a measure similar to the family-wise error rate for smooth statistical maps has been formulated. Due to the spatial correlation in brain images, they can be considered to be smooth statistical maps and results from RFT can be used in their analysis. Rather than aiming for a certain family-wise error rate, most SPM studies try to find a threshold value yielding the desired *Euler characteristic*. In addition to the threshold this characteristic depends on the smoothness of the map, which is a measure that can be estimated. The Euler characteristic can roughly be interpreted as the probability that the two sample groups would have at least one *region* above the threshold under the null hypothesis [9].

Recent SPM based research into differential diagnosis of parkinsonian disorders using FDG-PET found that all IPD, MSA and PSP patient show significantly lower metabolic rate in the cerebral neocortex compared to the control group [28]. The MSA group showed a significantly lower metabolic rate in the putamen, pons, nucles, the thalamus, midbrain and the cingulate gyrus compared to the normal controls, the IPD and the MSA groups. The difference between each condition and the control group are depicted graphically in figure 4.3. In another study [19] similar patterns were obtained for the different parkinsonisms using SPM. These patterns were subsequently used to train a non-expert in performing a diagnosis based on an image. Both these non-expert diagnoses and expert diagnoses obtained by visual assessment were compared with 2-year follow-up clinical assessments. It was found that the SPM-assisted non-experts were better in distinguishing the various parkinsonisms than experts who drew on their knowledge and experience.

4.4 Scaled Subprofile Modelling

Scaled Subprofile Modelling (SSM) uses principal component analysis (PCA) to identify abnormal functional patterns in multivariate imaging data by analyzing the combined patient and control group mean values [3, 34, 45]. After spatial normalization and initial smoothing, the procedure consists of applying the following steps to the combined group reference (REF) image data [44]:

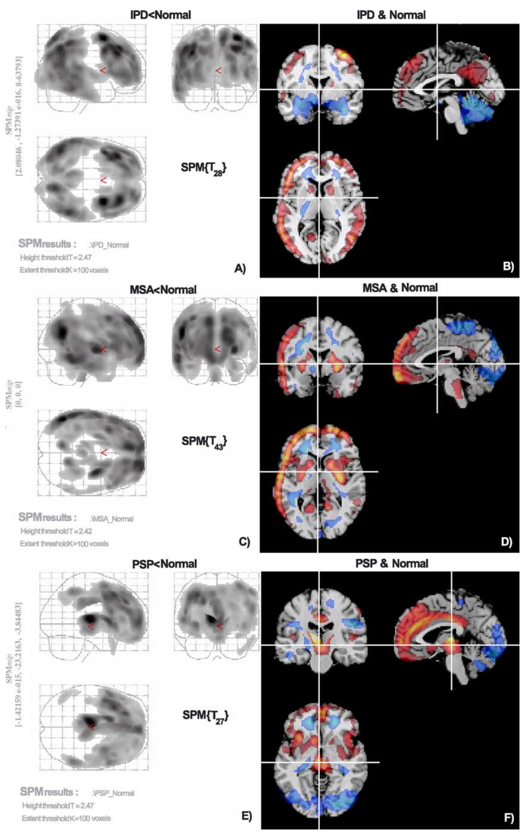


FIGURE 4.3: Characteristic parkinsonian metabolic patterns obtained using SPM in [28].

1. Mask the brain data to reduce low values and noise. This can be done by using predefined maps of gray matter. Another method is to define a mask for each subject such that only voxels with a value higher than 35% of the maximum are included, and then multiply these masks elementwise to obtain a mask for the entire group.
2. Take the log transformation of the combined group reference.
3. Center each subject's scan data by subtracting mean voxel value for that subject.
4. Center the value of each voxel by subtracting its mean over all subjects. In SSM, the resulting data matrix is called the *subject residual profile* (SRP).
5. Apply principal component analysis to the SRP data.

SSM then tries to find a linear combination of principal components that can be used to project the data on a line on which the patient group can be linearly discriminated from the control group. Such a vector of voxel-weights is called a Group Invariant Subprofile (GIS) and the dot product between a subject's scan and a GIS is called that subject's *score*. A GIS can be considered a disease-related pattern if the scores of the patients and the scores of the control group are separated at a pre-specified threshold, typically

$p \leq 0.001$. There are multiple strategies to select the principal components which make up the linear combination [44]:

- Select only the single components with the lowest p-value.
- Combine components whose p-values are less than a fixed value
- Select the combination of components with the lowest Akaike information criterion value [2, 48].
- Only retain the first few successive principal components that explain a pre-defined amount of the variance in the data and then apply an additional selection criterion.

Once the relevant principal components have been found, the corresponding coefficients in the linear combination are typically determined by logistic regression [44]. Once such a discriminatory metabolic pattern is obtained, new subjects are evaluated as deviations from the REF group score mean offset by the REF control mean score μ_{CTRL} divided by the REF score standard deviation σ_{REF}:

$$Z = \frac{y - \mu_{\text{CTRL}}}{\sigma_{\text{REF}}} \tag{4.5}$$

Finally, an appropriate cut-off value θ_Z is determined by maximizing the corrected total likelihood ratio and the sign of patterns will be such that the Z-scores of the patient group will be higher than those of the control group.

The metabolic pattern thus obtained shows which regions are typically (in)active in patients with a certain parkinsonism when compared to a control group, but that does not necessarily mean that it allows us to discriminate between the different varieties. Ideally, the Z-scores of a patient whose condition was in the j-th combined group would satisfy

$$Z_j > \theta_{Z_j}$$

and

$$Z_i \leq \theta_{Z_i} \text{ if } i \neq j$$

where we use Z_i to denote the Z-score of that patient with respect to the i-th group and θ_{Z_i} to denote the threshold of the i-th group. However, due to the fact that the characteristic metabolic patterns of the different parkinsonisms might show overlap w.r.t. (in)active regions, a scan might score positive on multiple tests. A diagnosis can be singled out by choosing the patterns with the highest Z-score or the highest Z-score to cut-off ratio Z/θ_Z. Alternatively, it has been proposed to apply discriminant analysis to Z-scores alone and use the classifier obtained this way to make new predictions [45].

In [45], a number of strategies are listed to decrease the false positive ratio when using SSM to perform a differential diagnosis. In addition to improving the reference group selection, these strategies include:

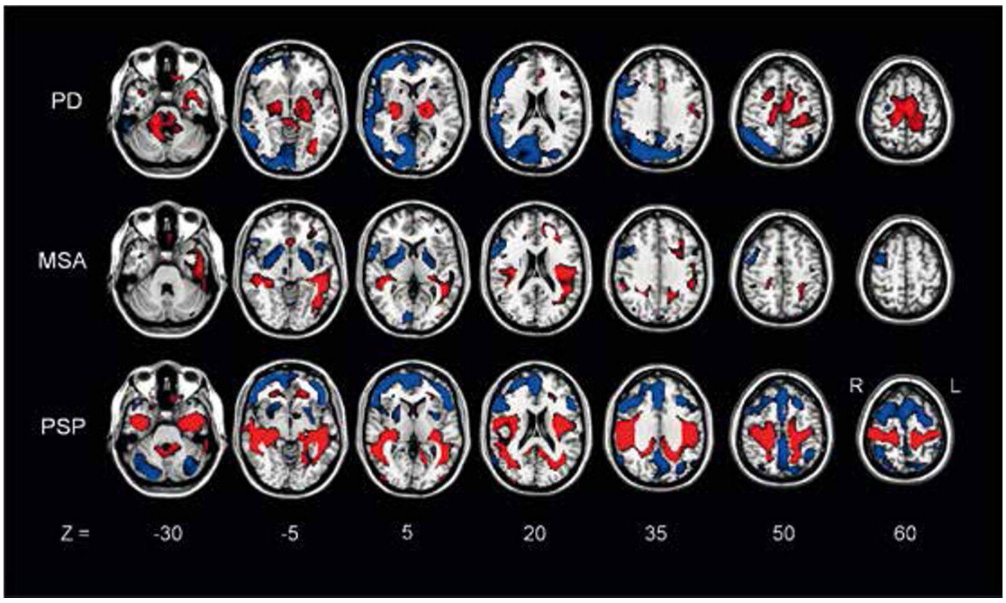


FIGURE 4.4: Characteristic metabolic patterns of PD, MSA and PSP obtained in [48]. Red (blue) regions shows statistically significantly more active (inactive) regions ($p < 0.001$) in comparison to healthy controls. Image from [48]
.

Unique sign effects The MSA pattern initially found in [45] mainly emphasized metabolic decreases. By repeating the whole SSM sequence on the subset of voxels that showed a metabolic decrease in the pattern, the overlap between the different patterns was reduced.

Elimination of similar voxels Similar to the unique signs procedure, the SSM sequence can be repeated on just that subset of voxels that have different signs between conflicting groups (i.e. eliminating the voxels which have the same signs).

Generate a new discriminatory network A new network discriminating two conflicting conditions can be trained by using one of the two conditions as the control group.

In recent years, SSM has been used to obtain disease-related patterns for various parkinsonisms. In [33] a PD-related pattern is obtained that shows relative increases in pallidothalamic, pontine and cerebellar metabolism and relative decreases in the premotor and posterior parietal areas. In [21] typical patterns for parkinsonisms MSA and PSP are presented. The MSA pattern is characterized by metabolic decreases in the putamen and the cerebellum. The PSP pattern shows covarying metabolic decreases in the medial prefrontal cortex, the frontal eye fields, the ventrolateral prefrontal cortex, the caudate nuclei, the medial thalamus and the upper brainstem. These patterns were validated in [48] (see fig. 4.4). In [20] it was shown that the "network score" linearly correlates with the progression of the disease. Aforementioned patterns were used in [47] as well: patients from the New York area with parkinsionian features but without a clear diagnosis were classified using the metabolic networks. When these predictions were compared with a final clinical diagnosis ~ 2.6 years later they proved to be highly accurate.

4.5 Machine Learning Methods in Brain Imaging Analysis

This thesis is by no means the first research to use machine learning methods in the analysis of brain images. This section provides a brief review of earlier work.

Support vector machines have been used in a straightforward manner to recognize early Parkinson's disease [32] and Alzheimer's disease [29]. In [35] it was found that a support vector machine outperforms Fisher's discriminant analysis in distinguishing two groups of fMRI data obtained under different experimental conditions. A multi-layer perceptron is trained to distinguish PD patients from a control group using dopaminergic images in [1]. In that research, the system was only trained on voxels in the striatum by using an explicit striatal mask. While the classification was fairly accurate, it should be noted that it was not possible to visualize the regions of interest within the striatum found by this method. The authors emphasize the point we raised in our introduction by noting that "as with all application of [artificial neural networks], it was difficult to interpret precisely what triggers in the images were detected by the network."

The *recursive feature elimination* (RFE) algorithm is presented in [14]. RFE trains a two-class, linear SVM and subsequently eliminates the voxels whose corresponding weights have a low absolute value. This process is repeated until some stop criterion is met. Finally, in [26] relevance vector machines are combined with error-correcting output codes (see section 3.1.2) to solve the multi-class classification problem of distinguishing PD, MSA, PSP and corticobasal syndrome patients based on FDG PET scans.

Concluding remarks This chapter has treated PET scans, Parkinson's disease, conditions associated with Parkinson's disease and two strands in brain imaging analysis: Statistical Parametric Mapping and the Scaled Subprofiling Model. We also gave a brief review on earlier work on using machine learning in brain imaging. The next chapter will report on the experiments we performed using the techniques described in chapter two. The final part of the next chapter will report on what the merits of supervised dimensionality reduction methods are in the classification of various parkinsonisms.

Chapter 5

Experiments and Results

This chapter treats the results obtained in experiments performed with Support Vector Components Analysis and the other SDR techniques described in chapter 2. The first part of this chapter will show some general properties of the algorithm discovered empirically using both a toy problem consisting of artificial data and a dataset from the UCI repository. The final part will concern experiments conducted on datasets of FDG-PET scans of subjects with various parkinsonisms.

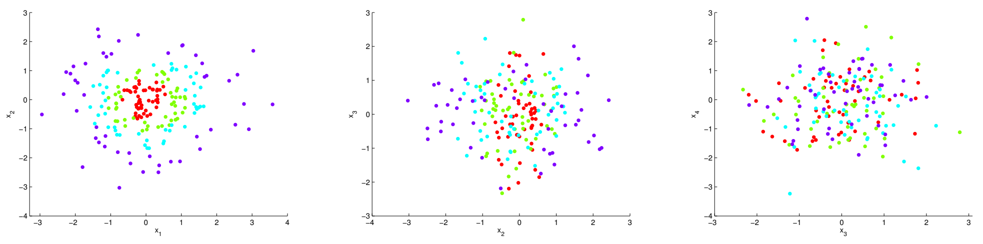
Implementation The SVCA algorithm was implemented in MATLAB. We use SVM implementations as provided by LIBSVM [12]. MATLAB implementations of LiRaM LVQ[1] and LFDA[2] were retrieved from the Web. We implemented NCA ourselves, also in MATLAB.

Metaparameter values For SVCA, we use the RPROP algorithm to optimize the transformation matrix as explained in chapter 2. In all experiments $(\eta^-, \eta^+) = (0.5, 1.2)$. Initially, all Δ_{ij}'s are set to 10^{-3}. We limit the range of learned Δ_{ij}'s by setting $(\Delta_{\min}, \Delta_{\max}) = (10^{-6}, 1)$. Other metaparameter values differed per experiment and will be explicitly stated.

5.1 Experiment on Artificial Data

The first problem we consider is more of a sanity check than a thorough experiment. Inspired by the concentric ring data in [27], we create an artificial dataset in the following way: First, we create patterns $\mathbf{x}_1 \dots \mathbf{x}_P$ in $\mathbb{R}^8$ by drawing from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\mu} = \mathbf{0}$ and $\boldsymbol{\Sigma}$ is the 8×8 identity matrix. Then we assign labels based solely on the distance from the origin in the first two dimensions, i.e. $\sqrt{x_1^2 + x_2^2}$. This results in classes that take the shape of concentric rings in the class-relevant subspace. In total, we create 200

[1] `http://matlabserver.cs.rug.nl/gmlvqweb/web/`
[2] `http://sugiyama-www.cs.titech.ac.jp/~sugi/software/LDA/`



(A) (B) (C)

FIGURE 5.1: The artificial dataset of concentric rings. Shown are scatter plots in which both features are relevant to the labels (A), only one feature is relevant (B) and none of the features are relevant (C).

patterns belonging to four different classes. In defining class boundaries, we ensure that each class has about the same number of patterns. Figure 5.1 shows the dataset thus generated.

We stress two important properties of this problem: First, the underlying structure of the data is based solely on the class labels. Without those labels, the data is simply Gaussian noise so an unsupervised technique such as PCA will never discover the underlying structure. Second, the class boundaries in the class-relevant subspace are non-linear, so classical Fisher's discriminant analysis will not perform well either.

We compare the performance of SVCA with the three other SDR technique introduced in chapter 2: NCA, LiRaM LVQ and LFDA. The LiRaM LVQ algorithm was not designed with this type of problem in mind, although in principle it should be able to solve it if it has enough prototypes for each class. We have tried to give LiRaM LVQ the fairest of chances by setting the number of prototypes per class in the following way: We observe that the ratio between the areas of the three inner rings $A_1 : A_2 : A_3$ is about $5 : 8 : 13$. It seems reasonable that each class should have about the same prototype density, so we initialized the number of prototypes of the first three classes as 5, 8 and 13 respectively. Since the area of the "outer ring" is infinite in principle, we observe that the sequence $(a_k) = (5, 8, 13)$ is modelled by $k^2 + 4$ and set the number of prototypes of the outermost class to 20.

In chapter 2 we stressed the importance of the initialization of the transformation matrix for the LiRaM LVQ, NCA and SVCA algorithms. To be able to compare their performances in a sensible way we generated 100 initial matrices with elements drawn from $\mathcal{N}(0, 1)$ that were subsequently made orthonormal. During each separate run of the experiment, a new dataset was generated and the three initialization-dependent algorithms were supplied the same initial matrix. The four algorithms were then tested on the problem just created. We emphasize that in each run, both the dataset and the initialization matrix were different. The experiment consisted of a total of 100 runs.

Figure 5.2 shows the projections obtained by the algorithm in one particular run. It is illustrative of the general results in that LiRaM LVQ and LFDA never succeed in finding

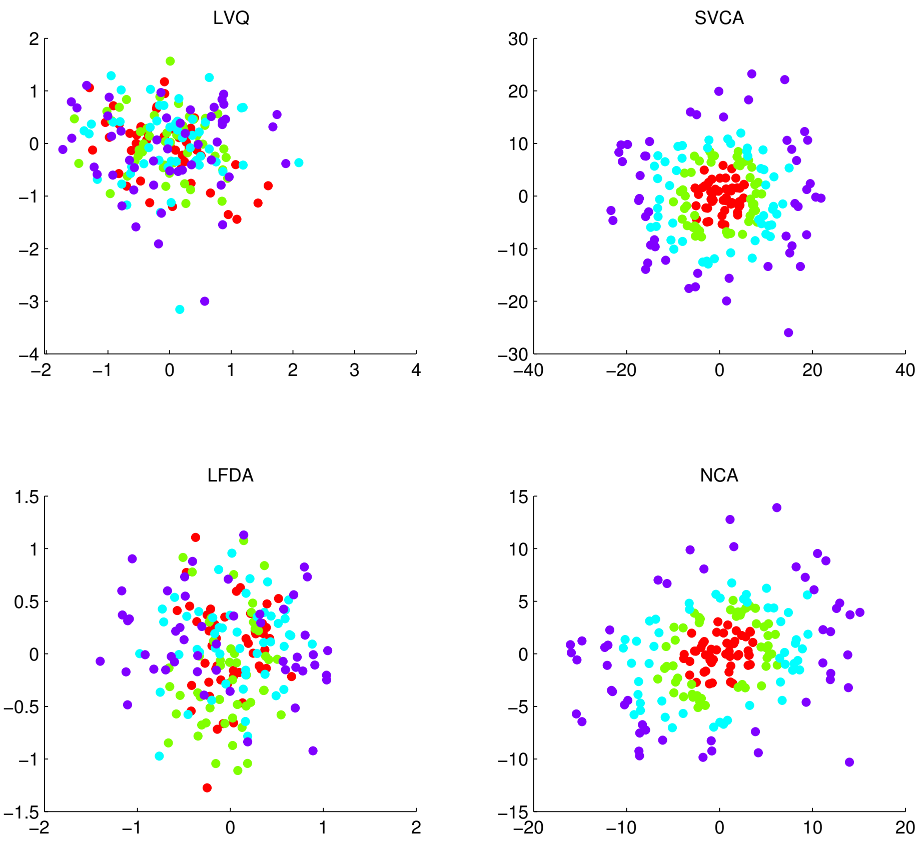


Figure 5.2: Projections obtained by LiRaM LVQ, SVCA, LFDA and NCA on the concentric ring problem in one particular run. LiRaM LVQ, SVCA and NCA used the same orthonormal random initialization matrix.

the underlying structure. For SVCA and NCA, we have simply counted the number of times each algorithm finds the right projection, resulting in the contingency table shown in table 5.1. Under the null hypothesis that NCA and SVCA have the same error rate, we expect $e_{01} = e_{10}$ and these to be equal to $(e_{10} + e_{01})/2$. Using McNemar's test we obtain a test statistic $\chi_1^2 = 6.5$, corresponding to a p-value of 0.011. We therefore reject our null hypothesis and conclude that SVCA significantly outperforms NCA in this experiment.

		SVCA	
		incorrect	correct
NCA	incorrect	$e_{00} = 30$	$e_{01} = 20$
	correct	$e_{10} = 6$	$e_{11} = 44$

Table 5.1: Contingency table of SVCA and NCA error rates in the concentric ring experiment.

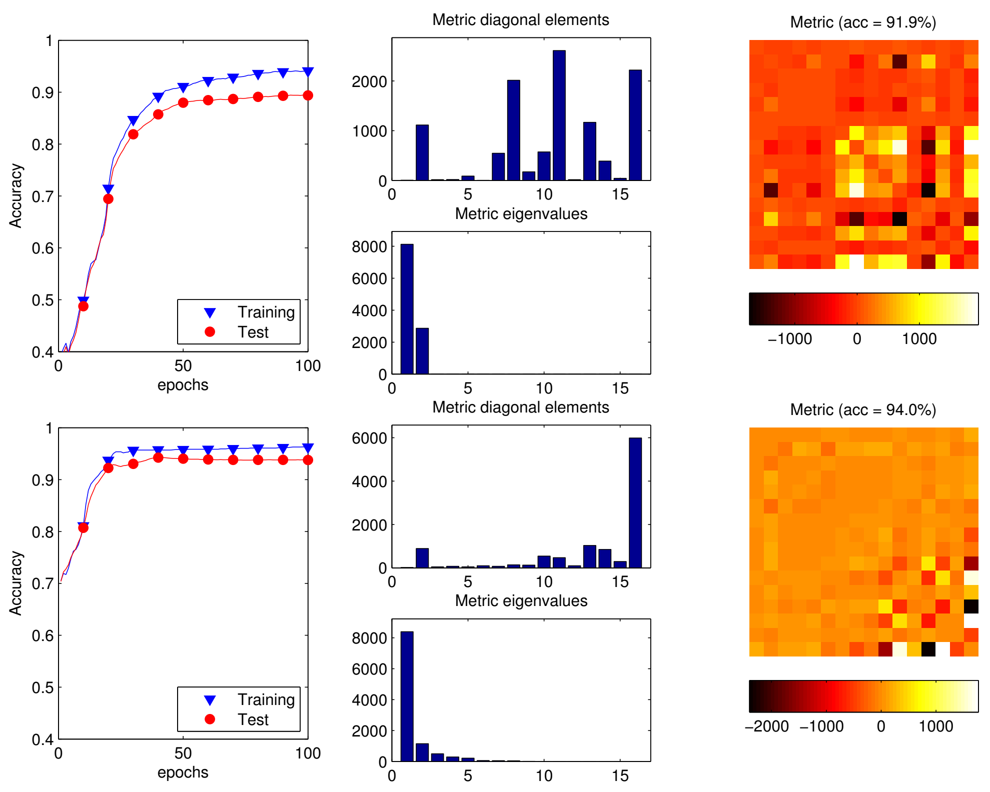


FIGURE 5.3: Left panels: learning curves of the SVCA algorithm for $K = 2$ (top) and $K = 16$ (bottom) averaged over ten runs on the UCI image segmentation dataset. Right panels: eigenvalues, diagonal values and a heatmap of the learned metric for $K = 2$ (16) on top (bottom) in the best run. Diagonal values in heat maps are set to zero. In all experiments, $(C, \gamma) = (1, 10^{-3})$.

5.2 Experiments on UCI Image Segmentation Dataset

In [10], Bunte et al. illustrate properties of the LiRaM LVQ algorithm by presenting the results obtained in experiments on the UCI Image Segmentation dataset [6]. In this section we will do the same for the SVCA algorithm.

The UCI Image Segmentation dataset consists of 19-dimensional feature vectors that have been constructed from regions of 3×3 pixels, randomly drawn from a set of 7 manually segmented outdoor images. The class of each pattern indicates the object or material shown in the corresponding 3×3 image patch, with the set of classes consisting of “brickface”, “sky”, “foliage”, “cement”, “window”, “path” and “grass”. The dataset has been pre-split into 210 training patterns and 2100 test patterns, with each class occurring an equal number of times in both subsets. We follow [10] in not using features 3, 4 and 5 because they have zero variance. We preprocess the data using a Z-transform based on training set mean and variance.

Influence of K We first compare the dependence of the performance of SVCA on the dimensionality of the range K. Figure 5.3 shows the learning curves of the SVCA

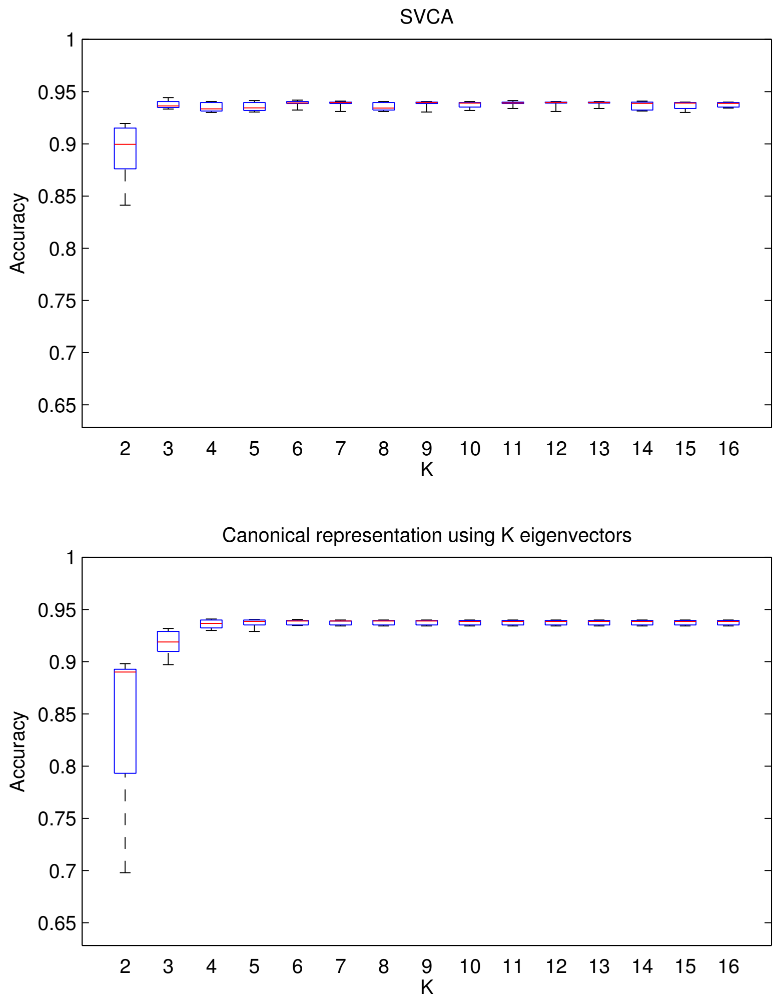


FIGURE 5.4: Accuracy dependence on the number of rows K of $\mathbf{T}$ on the test split of the UCI image segmentation dataset. Boxplots on top (bottom) correspond to the performance obtained by the SVCA algorithm (decomposition method) averaged over ten runs of 100 epochs each. For each value of K, the same ten orthonormal initialization matrices were used. $(C, \gamma) = (1, 10^{-3})$. Details in text.

algorithm for $K = 2$ and $K = 16$. The curves in figure 5.3 were obtained by averaging the test and training accuracies over ten different runs. Like in the concentric ring experiment we pre-generated orthonormal random initialization matrices and used the same ten initialization matrices for the two values of K, where in case of $K = 2$ only the first two rows were used. In these particular experiments, we optimize the objectives of radial kernel SVMs with $(C, \gamma) = (1, 10^{-3})$.

For both values of K SVCA shows typical learning curves: both the training and test accuracies increase during training, with the train accuracy being higher than the test accuracy. Overfitting does not seem to occur. Figure 5.3 also shows the diagonal elements, eigenvalues and heatmap of the distance metric $\mathbf{\Lambda} = \mathbf{T}'\mathbf{T}$ obtained in the best run in terms of test accuracy for both values of K. Note that when using a radial kernel function a metric $\mathbf{\Lambda}$ appears in the following way:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma ||\mathbf{T}\mathbf{x}_i - \mathbf{T}\mathbf{x}_j||_2^2\right) = \exp\left(-\gamma(\mathbf{x}_i - \mathbf{x}_j)'\mathbf{\Lambda}(\mathbf{x}_i - \mathbf{x}_j)\right) \tag{5.1}$$

Both metrics indicate the sixteenth feature (the hue of the image patch, as in HSV) to be salient. The eigenvalues for $K = 16$ sorted in descending order rapidly decrease, with only the first 5 being visible in the bar plot. This seems to suggest that even in learning a full-rank distance metric SVCA finds a low-rank represenation of the training data. Given this observation, we could also try to perform supervised dimensionality reduction by reducing the rank of the full-rank metric $\mathbf{\Lambda}$. Note that we can decompose $\mathbf{\Lambda}$ in the following way:

$$\mathbf{\Lambda} = \mathbf{V}\mathbf{D}\mathbf{V}' \tag{5.2}$$

where $\mathbf{V}$ is a matrix with the eigenvectors of $\mathbf{\Lambda}$ as its columns and $\mathbf{D}$ is a diagonal matrix with corresponding eigenvalues on its diagonal. We can obtain a transformation matrix $\mathbf{T}$ from (5.2) by setting $\mathbf{T}$ to $\mathbf{V}'\mathbf{D}^{1/2}$. Then, if we assume the values of $\mathbf{D}$ are sorted in descending order, we can obtain $\mathbf{T}_K$ mapping to a K-dimensional space by only using the first K eigenvectors and eigenvalues.

Figure 5.4 shows the dependencies of the test accuracy on K for both SVCA and the decomposition method described above. As expected, the performance barely increases for $K > 5$. More interestingly, we find that for low K the performance of SVCA is better than that of the canonical decomposition method. For $K = 2$, the mean test accuracies differ by 4%.

Influence of $\boldsymbol{\gamma}$ When using the radial kernel, the parameter γ determines how the similarity measure of two patterns scales with the distance between those patterns. For a low γ even patterns far away from each other are considered relatively similar. In support vector components analysis, the transformation matrix determines the magnitude of the vectors in the low-dimensional space and thus influences the distances between patterns. During training, we hope the magnitudes of the patterns in the low dimensional space are optimized with regard to γ which is fixed beforehand. The *norm* of the transformation matrix, $||\mathbf{T}||$, can serve as a measure of the magnitude of the transformed vectors by providing an upper bound on transformed vector lengths:

$$||\mathbf{T}\mathbf{x}|| \leq \max_{\mathbf{x}'} ||\mathbf{T}\mathbf{x}'|| \leq c||\mathbf{T}|| \tag{5.3}$$

Since the argument of the exponentional function in equation 5.1 is the product of γ and the length of a difference vector *squared*, we expect optimized transformation matrices for different values of γ to yield a roughly constant product

$$\gamma ||\mathbf{T}_\gamma||^2 = c \tag{5.4}$$

γ	$\mathbf{10^{-5}}$	$\mathbf{10^{-4}}$	$\mathbf{10^{-3}}$	$\mathbf{10^{-2}}$	$\mathbf{10^{-1}}$
$\lVert\mathbf{T}\rVert$	1.077×10^3	360.2	134.9	44.71	13.16
$\gamma\lVert\mathbf{T}\rVert^2$	11.61	12.98	18.21	19.99	17.31

TABLE 5.2: Dependence between γ and norm of $\mathbf{T}$ in the experiment on the UCI Image Segmentation dataset. Shown are the average values obtained over ten runs of 1000 epochs each, using the same initialization matrices for different values of γ.

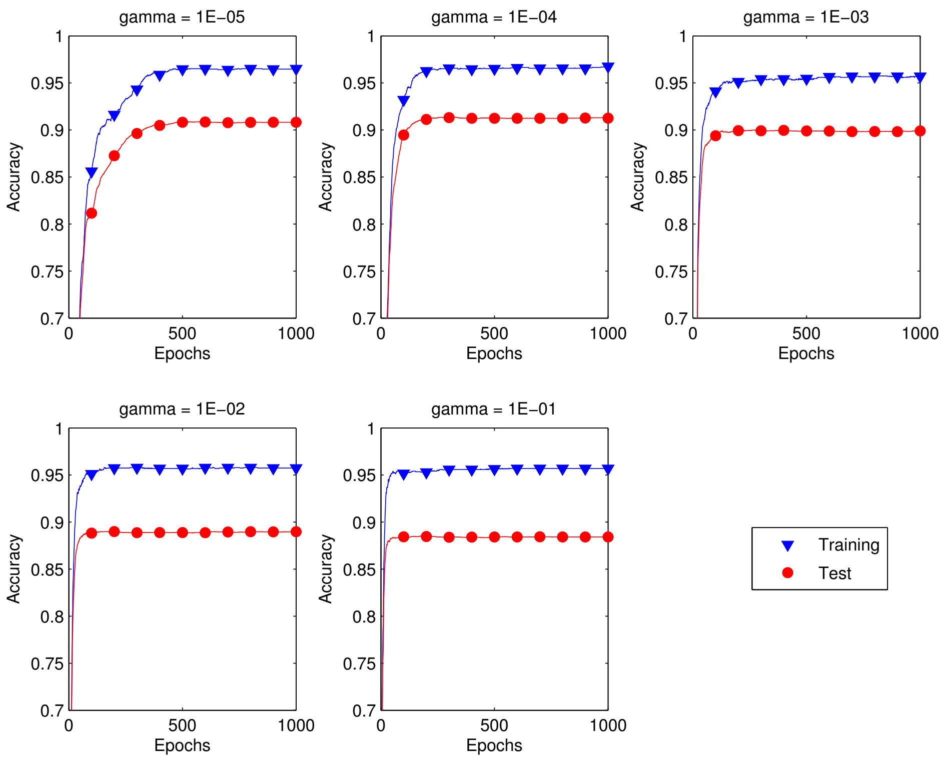


FIGURE 5.5: Learning curves for different values of γ in experiments on the UCI Image Segmentation dataset. In all experiments, $C = 1$.

where we assume that c is specific to the classification problem. To test this hypothesis, we optimize the same orthonormal initialization matrices for five different values of γ logarithmically scaled between 10^{-5} and 10^{-1}, again setting C to 1. Table 5.2 shows the results of this experiment for $K = 2$. While $\gamma\lVert\mathbf{T}_\gamma\rVert^2$ is not *identical* for all values of γ, the products are still rather similar given the strongly varying values of $\lVert\mathbf{T}\rVert$.

The fact that the magnitude of transformed vectors is adjusted to γ might lead us to think that the value of γ is unimportant for the eventual performance of the system. However, we find that this is not the case. Figure 5.5 shows the learning curves of the experiments with the logarithmically scaled values of γ. It can be clearly seen that for lower γ values, the performance converges to a higher accuracy. This actually illustrates an interesting property of SVCA with a radial kernel: given normalized data and an orthonormal initialization matrix, a relatively small γ will make the initial adjustments of $\mathbf{T}$ only be based on coarse structures in the data such as the structural difference between

	SVCA	LRMLVQ	NCA	LFDA
$K = 2$	0.9078 ± 0.0114	0.8441 ± 0.0527	0.8554 ± 0.0121	0.7165
$K = 16$	0.9418 ± 0.0004	0.8953 ± 0.0114	0.9228 ± 0.0000	0.9313

Table 5.3: Average test accuracies on the UCI Image Segmentation Dataset over 100 runs (except for LFDA). In experiments with SVCA, LiRaM LVQ and NCA the same initialization matrices wered used.

the two most dissimilar classes. Then, as the norm of $\mathbf{T}$ becomes larger, increasingly fine-grained differences will start to influence the training process until training converges. Setting γ too high will result in skipping the phase in which the algorithm only uses the most coarse class differences, and might emphasize noisy differences due to noise in the data or the random initialization. Figure 5.5 illustrates that setting γ too low has its drawbacks as well: the training procedure will take more epochs until convergence.

As a final experiment on the UCI dataset, we compare the performances of SVCA, LiRaM LVQ, NCA and LFDA. Like in the concentric ring experiment, we pregenerate 100 initial matrices to be able to compare the three initialization-dependent algorithms. Since NCA and LFDA do not provide an explicit classification, we perform k-nearest-neighbor classification in the range of the learned transformation. Comparing different values of k we found that both NCA and LFDA perform best when setting k to 1. The results of these experiments for both $K = 2$ and $K = 16$ are shown in table 5.3. Running paired t-tests, we find that the performance is SVCA is significantly better than each of the other algorithms for both $K = 2$ and $K = 16$ using a significance level of $p = 0.05$. Somewhat surprisingly, the experiments with NCA for $K = 16$ always yield the same accuracy while the transformation matrix found is not necessarily the same for each experiment.

5.3 Experiments on FDG-PET Scans

Here we apply the algorithms dicussed in chapter 2 to the same set of PET scans as used in [26]. These scans were obtained in two different locations between 1993 and 2009 and are comprised of 42 PD patients, 31 MSA patients, 26 PSP patients and 21 corticobasal syndrome (CBS) patients. In all of the experiments we preprocess the data through through the SSM routine:

1. Define a gray matter mask by only using those voxels which are above 35% of the maximum of each individual scan
2. Take the log transformation.
3. Center each subject's scan data by substracting the mean voxel value for that subject.
4. Center the value of each voxel by subtracting its mean over all subjects.

RVM Classification	Diagnostic Classes				PPV/NPV
	PD	MSA	PSP	CBS	
PD	**38**	6	2	2	.79/.94
MSA	1	**14**	5	1	.67/.83
PSP	1	7	**14**	5	.52/.87
CBS	2	4	5	**13**	.54/.92
Class accuracy	.90	.45	.55	.62	

Table 5.4: Confusion matrix obtained in [26] using bootstrap aggregating with relevance support vector machines. Details in text

5. Apply principal components analysis and determine principal component scores for each pre-procesed scan.

We stress that any of above steps that require some measure of the whole data set (such as PCA or the mask definition), only use the training part of the dataset to calculate that measure when the dataset is split into a test part and a training part. After these five steps, we normalize the principal component scores such that they each have zero mean and unit variance.

5.3.1 Assessing Accuracy

If we want to compare the classification accuracy of dimensionality reduction methods the results presented in [26] provide a benchmark. However, their way of assessing the accuracy of their classifier is a bit unorthodox: They create 100 bootstrap samples by drawing – with replacement – 84 scans from the dataset, ensuring that each sample has the same number of scans for each class. The bootstrap sample is used to train a classifier, and then the class predictions for the scans *not* in the sample are determined. Finally, they determine the final classification of a particular scan by a majority vote among all the classifiers in which that scan was not part of the bootstrap sample. This differs from the usual bootstrap procedure in which you would report the average classification accuracy on test splits, not the accuracy of the 'average' classification. Nor is it the usual bagging procedure in which you would report the performance on a split of the data which in no part was part of the training. Still, we adopt this procedure to be able to compare results. Table 5.4 shows the confusion matrix obtained in [26].

We note that in [26] the classifier is trained on pre-processed voxel values rather than principal component scores. With 120 scans in total and each scan consisting of 153,594 voxels, the risk of overfitting seems very high. The bagging scheme described above is one way to smooth the hypothesis of the classifier, effectively retaining only those voxel relevances which were discovered in all of the bootstrap samples. Using principal component scores (like we do) is another way to prevent overfitting. We can set an upper threshold on the variance explained by the principal components by only using the first n scores. In the experiment, we set this threshold to 90% of the total variance.

In this experiment, we are interested in transformations to a 2 or 3-dimensional space, since in that case we can make a scatter plot of the transformed data that might lead to new insights. Tables 5.5 and 5.6 show the confusion matrices for each dimensionality reduction methods for $K = 2$ and $K = 3$, respectively. NCA and LFDA do not provide an explicit prediction for new patterns. We have chosen to assign labels according the nearest neighbor classification in the transformed space, where the number of nearest neighbors was determined through cross-validation.

Metaparameter values and other specifics In these experiments, we use one prototype per class in LiRaM LVQ. We use the 'local' definition of the affinity matrix in LFDA. The parameters of SVCA have been coarsely optimized through cross-validation and are set to $(C, \gamma) = (1, 10^{-4})$. For NCA, we use a majority vote among the nearest 9 neighbors and for LFDA among the nearest 11 neighbors. Initialization-dependent algorithms are supplied with the identity matrix appended with zeros, meaning that the initial transformed representation is the same as the first two principal components scores of each pattern. In the case of NCA we multiply this initialization matrix by 0.01 since we find that that increases the performance, for reasons similar to the advantage of a low γ in SVCA.

To give an idea of relative performances, we report the accuracies of the various algorithms in table 5.7 in addition to the accuracy of the results reported in [26] and that of an RBF SVM with coarsely optimized parameters $(C, \gamma) = (2^9, 2^{-7})$ used in the same bagging prodedure on the same bootstrap samples. The RBF SVM uses the pairwise multiclass extension described in chapter 2. While it is nice to see that all SDR algorithms have a better accuracy than that reported in [26], it is hard to draw any statistically sound conclusions since this is just the accuracy of one particular prediction and because contrary to [26], we use principal component scores rather than direct voxel values. Comparing the confusion matrices of the SDR methods we find that they have similar accuracies overall but that their accuracies per class differ, especially for $K = 2$. However, since these results are ultimately based on particular prediction we are unable to draw any conclusions from these differing class accuracies.

To be able to at least compare the SDR algorithms to each other, we also run a series of experiments in which we try to get a better idea of the test error of each algorithm. We predefine 100 splits of the data. In each split, 10% of the patterns has been randomly assigned to the test set, the rest of the patterns are used for training. We report mean test accuracies and their standard deviations in table 5.8. Running paired t tests on the different fold error rates, we find no significant difference between the various algorithms. However, we do find that the performance of the SDR algorithms rivals that of the optimized RBF SVM for $K = 3$, all while the RBF SVM does not have to push the data through a 3-dimensional bottleneck.

LFDA	Diagnostic Classes				PPV/NPV
	PD	MSA	PSP	CBS	
PD	**34**	4	1	1	.85/.90
MSA	7	**20**	8	5	.50/.86
PSP	1	5	**15**	6	.56/.88
CBS	0	2	2	**9**	.69/.89
Class acc.	.81	.65	.58	.43	

(A)

SVCA	Diagnostic Classes				PPV/NPV
	PD	MSA	PSP	CBS	
PD	**33**	4	2	1	.83/.89
MSA	9	**24**	10	8	.47/.90
PSP	0	1	**13**	2	.81/.88
CBS	0	2	1	**10**	.77/.90
Class acc.	.79	.77	.50	.48	

(B)

NCA	Diagnostic Classes				PPV/NPV
	PD	MSA	PSP	CBS	
PD	**35**	5	1	1	.83/.91
MSA	7	**21**	6	6	.53/.88
PSP	0	4	**17**	4	.68/.91
CBS	0	1	2	**10**	.77/.90
Class acc.	.83	.68	.65	.48	

(C)

LRMLVQ	Diagnostic Classes				PPV/NPV
	PD	MSA	PSP	CBS	
PD	**37**	5	3	2	.79/.93
MSA	3	**19**	4	1	.70/.87
PSP	2	6	**15**	9	.49/.88
CBS	0	1	4	**9**	.64/.89
Class acc.	.88	.61	.58	.43	

(D)

TABLE 5.5: Confusion matrices for $K = 2$ for LFDA (A), SVCA (B), NCA (C) and LiRaM LVQ (D).

LFDA	Diagnostic Classes				PPV/NPV
	PD	MSA	PSP	CBS	
PD	**36**	4	4	1	.80/.92
MSA	5	**20**	4	3	.63/.88
PSP	1	5	**15**	6	.56/.88
CBS	0	2	3	**11**	.69/.90
Class acc.	.86	.65	.58	.52	

(A)

SVCA	Diagnostic Classes				PPV/NPV
	PD	MSA	PSP	CBS	
PD	**40**	4	3	1	.83/.97
MSA	1	**19**	3	3	.73/.87
PSP	1	5	**16**	6	.57/.89
CBS	0	3	4	**11**	.61/.90
Class acc.	.95	.61	.62	.52	

(B)

NCA	Diagnostic Classes				PPV/NPV
	PD	MSA	PSP	CBS	
PD	**38**	4	4	1	.81/.95
MSA	3	**19**	2	3	.70/.87
PSP	1	6	**17**	5	.57/.90
CBS	0	2	3	**12**	.59/.91
Class acc.	.90	.61	.65	.57	

(C)

LRMLVQ	Diagnostic Classes				PPV/NPV
	PD	MSA	PSP	CBS	
PD	**39**	4	4	1	.81/.96
MSA	2	**19**	2	3	.73/.87
PSP	1	5	**16**	5	.59/.89
CBS	0	3	4	**12**	.63/.91
Class acc.	.93	.61	.62	.57	

(D)

TABLE 5.6: Confusion matrices for $K = 3$ for LFDA (A), SVCA (B), NCA (C) and LiRaM LVQ (D).

	SVCA	LRMLVQ	NCA	LFDA
Bag-2	0.6667	0.6917	0.6500	0.6667
Bag-3	0.7167	0.7167	0.6833	0.6833

RVM [26]:	0.6583
RBF SVM:	0.7250

TABLE 5.7: Accuracies of the various algorithms when used in the bagging procedure from [26] on the same 100 bootstrap samples. Accuracies of the RVM ensemble reported in [26] and an optimized RBF SVM included for reference. Note that in [26] a different preprocessing procedure is used.

	SVCA	LRMLVQ	NCA	LFDA
$K = 2$	0.5825 ± 0.1501	0.5617 ± 0.1322	0.5925 ± 0.1230	0.6133 ± 0.1205
$K = 3$	0.6842 ± 0.1180	0.6717 ± 0.1297	0.6567 ± 0.1215	0.6767 ± 0.1272

RBF SVM:
0.6842 ± 0.1250

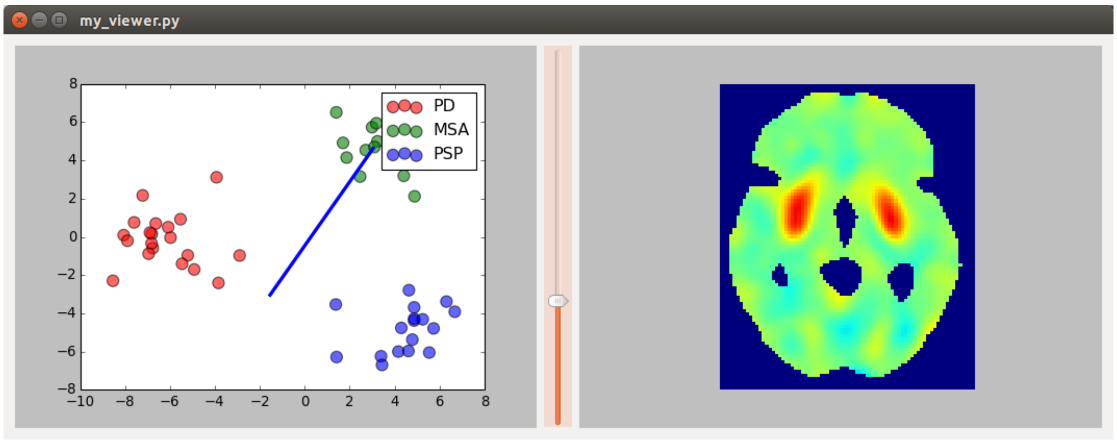
TABLE 5.8: Average test accuracies and standard deviations of the various algorithms on 100 test/train splits on the data from [26]. In each split, 10% of the patterns was randomly assigned to the test set. Performance of an RBF SVM on the same 100 splits included as a reference.

5.3.2 Validation and Visualization

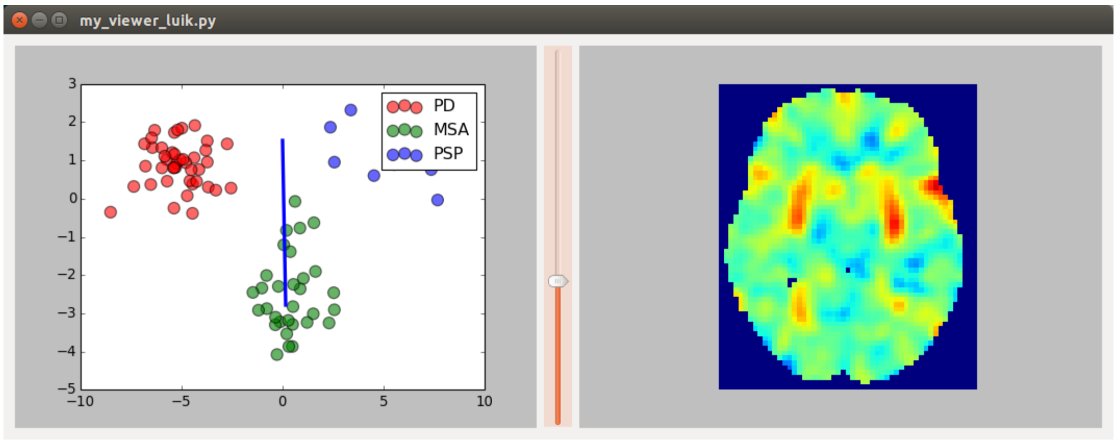
One of the goals of this thesis was to make classification to some extent intelligible. To this end, we have developed a prototype classifier viewer of which screenshots are displayed in figure 5.6. The prototype only works for transformations to a 2-dimensional space. In the left panel of the GUI, a scatter plot showing patterns' representations in the lower dimensional space is displayed. By clicking in this scatter plot, users can draw lines corresponding to discriminative patterns in the input space. These discriminative patterns are displayed in the right panel interactively. A slider in the middle allows the user to change the slice currently being displayed.

One of the applications of such a viewer might be the validation of patterns. To demonstrate this, we use LFDA to learn a 2-dimensional representation of two datasets: the PD, MSA and PSP patterns from [26] and scans with the same conditions used earlier in [48]. We use LFDA since we estimated that algorithm to have the highest test accuracy in our earlier experiments for $K = 2$. We load the transformation thus obtained into our viewer and visualize the patterns distinguishing MSA from the other two conditions. Interestingly, we find similar structures within the two classifiers as displayed in figure 5.6. In the visualization, red indicates areas relatively active in the first point clicked in the scatter plot while blue indicates areas relatively inactive in the point first clicked. For both plots, we first clicked in the MSA area.

Concluding remarks This chapter reported on all the experiments conducted on SVCA and the other SDR algorithms. We found that SVCA is at least as good as NCA in finding a low-dimensional representation of data with non-linear class boundaries. The experiments on the UCI dataset illustrated two important properties of the algorithm: Learning a low dimensional representation by limiting K beforehand is better than limiting K by a canonical decomposition of a full-rank transformation matrix, and a low gamma has the beneficial effect of first focusing on the most general class differences. In the experiments on the PET-scans, we were unable to show that SVCA has a clear benefit over other SDR algorithms. However, we were able to show that the decrease in accuracy due to the low-dimensional representation in SDR is negligible, while it does enable us the produce interactive visualizations such as those in the prototype displayed in figure 5.6. We will discuss these results more generally in the next, concluding chapter of this thesis.



(A)



(B)

FIGURE 5.6: Screenshots of the prototype transformation visualization. On top, the lower-dimensional representation obtained by LFDA of the scan data in [48] is displayed, while on the bottom the same procedure has been applied to a subset of the dataset from [26]. In both panels, the pattern on the right visualizes the structural difference between MSA and the other two groups. A user has indicated his wish to see this pattern by drawing the blue line from the MSA cluster to somewhere between the other two clusters. While not identical, the two discriminative patterns do show similarities.

Chapter 6

Conclusion

This thesis has considered the idea of 'intelligible classification': using supervised dimensionality reduction methods on imaging data to obtain class-relevant differential patterns. We have tried to answer two loosely related related research questions, and at the momentboth of them can only be answered to some extent. How the performance of SVCA compares to other SDR algorithms remains an open question. We do not know yet which type of problems the SVCA algorithm is particularly well suited for. In addition, the role of the cost parameter C has not been investigated yet. Although we formulated SVCA to be able to work with any type of kernel, in practice we have only used the RBF kernel. Linear kernels correspond to hyperplanes in the transformed space, and thus aren't very useful for multi-class problems. Intuitively, a polynomial kernel is somewhere between a linear kernel and an RBF kernel in terms of complexity. However, we found that the hypotheses suggested by SVCA when using a polynomial kernel are much more complex than when using a radial kernel with sufficiently low γ.

Our implementation of SVCA leaves room for improvement. By optimizing the alpha-values through SMO we introduced problems for our gradient-based method which we subsequently had to work around by using RPROP. Earlier work on dual objective optimization [50] used gradient ascent to optimize the alpha values. Doing so for SVCA would give us more control over the alpha values for the initial projections, which in the case of SMO are to some extent random since the classes can't be separated well initially. It also creates the possibility of a NCA/SVCA hybrid algorithm: We can initialize the alpha values in such a way that the dual objective initially is similar to the objective of NCA, and use the support vector machine algorithm to gradually identify boundary patterns during training. In addition, optimizing alpha values through gradient ascent would allow the algorithm to be made parallel and implemented on a GPU.

In the particular case of the differential diagnosis of parkinsonisms, there is a promise on which the SDR algorithms did not deliver. In principle, it should be possible that multiple clusters exist within one patient group. Such a discovery would be interesting and it is a disadvantage of LiRaM LVQ that the number of protypes per class have to be defined beforehand, precluding automating such a discovery. However, in projections with reasonable accuracies of any of the algorithms patient groups were always contained within one cluster each.

We should note that there is also a clear disadvantage to the SDR approach in classification in medical applications. Methods that find a single discriminative pattern distinguishing, for instance, a patient group from a control group can use various thresholds on the manifestation of such a pattern. This allows them to investigate the characteristics of the pattern by drawing ROC curves. In medical applications we often want to be able to control the sensitivity/specificity trade-off, and this is not possible in SDR methods considering more than two classes. We emphasize that SDR methods will be particularly useful in the exploratory analysis of a dataset, they do not provide the most flexible classification methods.

Using the results obtained in the previous chapter, we now return to the research questions posed in the introduction:

- **Question** How does support vector components analysis compare to other supervised dimensionality reduction methods?
  Answer From table 5.8, we can conclude that that SVCA performs on par with the SDR algorithms in terms of predictive accuracy for the FDG-PET scans from [26]. From the experiment on the concentric ring data we can conclude that SVCA is at least as able as NCA in finding a suitable low-dimensional representation for data with non-linear class boundaries and where the structure is solely based on class labels. LFDA and LiRaM LVQ seem unable to solve this type of problem. SVCA significantly outperforms all the three other SDR algorithms in the experiment on the UCI Image Segmentation dataset for both a low rank and a full-rank transformation matrix.

- **Question** What are the merits of supervised dimensionality reduction methods in the differential diagnosis of parkinsonisms based on FDG-PET scans?
  Answer Table 5.8 shows that all SDR methods achieve predictive accuracies similar to that of an RBF SVM when using a 3-dimensional representation of the data. The SDR methods have a clear advantage over the RBF SVM in that their transformations can be interactively visualized in an application such as depicted in figure 5.6.

We have refrained from giving a neuroanatomical interpretation of the discriminative direction found by the SDR methods since it is not within our expertise. In future work, we would like to collaborate with PD experts giving their interpretation of the patterns. From table 5.8 we know that classification results for the four-class problem are best when using a transformation to a 3-dimensional space. Extending our viewer to three dimensions is non-trivial and introduces usability issues. In addition, our viewer should be clearer about how the visualization corresponds to the line drawn in the scatter plot. Adding visualizations of reconstructions corresponding to the endpoints of the line seems to be an essential extension as well. The scatter plot can also be augmented by explicity indicating which regions in the scatter plot will be classified in what way. Finally, the visualization would also benefit from some contextual information such as a template skull overlay.

In conclusion, the major achievement of this thesis has been to show that SDR methods can achieve accuracies similar to 'unintelligible' classification techniques in the classification of various parkinsonisms based on FDG-PET scans. Whether the discriminative directions found by the methods provide any new insights remains an open question. In addition, we have introduced a novel member in the family of dual-objective optimization methods, SVCA, that performs on-par with existing SDR methods.

Bibliography

[1] Acton, P. D., and Newberg, A. Artificial neural network classifier for the diagnosis of Parkinson's disease using [99m]TRODAT-1 and SPECT. *Physics in Medicine and Biology 51* (2007), 3057–3066.

[2] Akaike, H. A new look at the statistical model identification. *IEEE Transactions on Automatic Control 19* (1974), 716–723.

[3] Alexander, G., and Moeller, J. Application of the scaled subprofile model to functional imaging in neuropsychiatric disorders: a principal component approach to modeling brain function in disease. *Human Brain Mapping 2* (1994), 79–94.

[4] Allwein, E. L., Schapire, R. E., and Singer, Y. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research 1* (2000), 113–141.

[5] Ashburner, J., and Friston, K. Voxel-based morphometry – the methods. *NeuroImage* (2000), 805–821.

[6] Bache, K., and Lichman, M. UCI machine learning repository, 2013.

[7] Bailey, D. Data acquisition and performance characterization in PET. In *Positron Emission Tomography*. Springer-Verlag, 2005.

[8] Bailey, D., Karp, J., and Surti, S. Physics and instrumentation in PET. In *Positron Emission Tomography*. Springer-Verlag, 2005.

[9] Brett, M., Penny, W., and Kiebel, S. Introduction to random field theory. In *Human Brain Function*. Elsevier, 2003.

[10] Bunte, K., Schneider, P., Hammer, B., Schleif, F.-M., Villmann, T., and Biehl, M. Limited rank matrix learning, discriminative dimension reduction and visualization. *Neural Networks 26* (2012), 159–173.

[11] Burges, C. J. C. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery 2* (1998), 121–167.

[12] Chang, C.-C., and Lin, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology 2* (2011), 27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[13] Crammer, K., and Singer, Y. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research 2* (2001), 265–292.

[14] De Martino, F., Valente, G., Staeren, N., Ashburner, J., Goebel, R., and Formisano, E. Combining multivariate voxel selection and support vector machines for mapping and classifcation of fMRI spatial patterns. *NeuroImage 43* (2008), 44–58.

[15] Defrise, M., Kinahan, P., and Michel, C. Image reconstruction algorithms in PET. In *Positron Emission Tomography.* Springer-Verlag, 2005.

[16] Dietterich, T. G., and Bakiri, G. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research 2* (1995), 263–286.

[17] Domeniconi, C., Gunuulos, D., and Peng, J. Large margin nearest neighbor classifiers. *IEEE Transactions on Neural Networks 16* (2005), 899–909.

[18] Duda, R. O., Hart, P. E., and Stork, D. G. *Pattern Classification.* John Wiley & Sons, 2001.

[19] Eckert, T., Barnes, A., Dhawan, V., Frucht, S., Gordon, M., Feigin, A., and Eidelberg, D. FDG PET in the differential diagnosis of parkinsonian disorders. *NeuroImage 26* (2005), 912–921.

[20] Eckert, T., Tang, C., and Eidelberg, D. Assessment of the progression of Parkinson's disease: a metabolic network approach. *The Lancet Neurology 6*, 926–932.

[21] Eckert, T., Tang, C., Ma, Y., Brown, N., Lin, T., Frucht, S., Fegin, A., and Eidelberg, D. Abnormal metabolic networks in atypical parkinsonism. *Movement disorders 23* (2008), 727–733.

[22] Fisher, R. A. The use of multiple measurements in taxonomic problems. *Annals of Eugenics 7* (1936), 179–188.

[23] Friston, K. Statistical parametric mapping: Ontology and current issues. *Journal of Cerebral Blood Flow and Metabolism* (1995), 361–370.

[24] Friston, K. Experimental design and statistical parametric mapping. In *Human Brain Function.* Elsevier, 2003.

[25] Friston, K., Holmes, A., Poline, J.-B., Price, C., and Frith, C. Detecting activations in PET and fMRI: Levels of inference and power. *NeuroImage 40* (1996), 223–235.

[26] Garraux, G., Phillips, C., Schrouff, J., Kreisler, A., Lemaire, C., C, D., Delcour, C., Hustinx, R., Luxen, A., Desée, A., and Salmon, E. Multiclass classification of FDG PET scans for the distinction between Parkinson's disease and atypical parkinsonian syndromes. *NeuroImage: Clinical* (2013), 883–893.

[27] Goldberger, J., Roweis, S., Hinton, G., and Salakhutdinov, R. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems 17* (2004), MIT Press, pp. 513–520.

[28] Juh, R., Kim, J., Moon, D., Choe, B., and Suh, T. Different metabolic patterns analysis of parkinsonism on the 18F-FDG PET. *European Journal of Radiology 51* (2004), 223–233.

[29] Klöppel, S., Stonnington, C. M., Chu, C., Draganski, B., Scahill, R. I., Rohrer, J. D., Fox, N. C., Jack Jr, C. R., Ashburner, J., and Frackowiak, R. S. J. Automatic classification of MR scans in Alzheimer's disease. 681–689.

[30] Kohonen, T. *Self-Organzing Maps.* Springer-Verlag, 1997.

[31] Lees, A. J., Hard, J., and Revesz, T. Parkinson's disease. *The Lancet 373*, 2055.

[32] Long, D., Wang, J., Xuan, M., Gu, Q., Xu, X., Kong, D., and Zhang, M. Automatic classification of early Parkinson's with multi-modal MR imaging. *PLoS ONE 7* (2012).

[33] Ma, Y., Tang, C., Spetsieris, P. G., Dhawan, V., and Eidelberg, D. Abnormal metabolic network activity in Parkinson's disease: test-retest reproducibility. *Journal of Cerebral Bloof Flow & Metabolism 27* (2007), 597–605.

[34] Moeller, J., and Strother, S. A regional covariance approach to the analysis of functional patterns in positron emission tomographic data. *Journal of Cerebral Blood Flow and Metabolism 11* (1991), A121–135.

[35] Mourão Miranda, J., Bokde, A. L. W., Born, C., Hampel, H., and Stetter, M. Classifying brain states and determining the discriminating activation patterns: Support vector machine on functional MRI data. *NeuroImage 28* (2005), 980–995.

[36] Novembre, J., Johnson, T., Bryc, K., Kutalik, Z., Boyko, A. R., Auton, A., Indap, A., King, K. S., Bergmann, S., Nelson, M. R., et al. Genes mirror geography within Europe. *Nature 456*, 7218 (2008), 98–101.

[37] Pietersma, A.-D., Schomaker, L. R. B., and Wiering, M. A. Kernel learning in support vector machines using dual-objective optimization. In *Proceedings of the 23rd Belgian-Dutch Conference on Artificial Intelligence* (2011), pp. 167–174.

[38] Platt, J. Sequential minimal optimisation: a fast algorithm for training support vector machines. Tech. Rep. MSR-TR-98-14, Microsoft Research, 1998.

[39] Riedmiller, M., and Braun, H. A direct adaptive method for faster backpropagation learning: The Rprop algorithm. In *Proceedings of the IEEE International Conference on Neural Networks* (1993), pp. 586–591.

[40] Samii, A., Nutt, J., and Ransom, B. Parkinson's disease. *The Lancet 363* (2004), 1783–1793.

[41] Sato, A., and Yamada, K. Generalized learning vector quantization. In *Advances in Neural Information Processing Systems 8* (1996), MIT Press, pp. 423–429.

[42] Scheider, P., Bunte, K., Hammer, B., and Biehl, M. Adaptive relevance matrices in learning vector quantization. *Neural Computation 21* (2009), 3532–3561.

[43] Schölkopf, B., and Smola, A. *Learning with Kernels.* MIT Press, 2002.

[44] Spetsieris, P., and Eidelberg, D. Scaled subprofile modeling of resting state imaging data in Parkinson's disease: Methodological issues. *NeuroImage 54* (2011), 2899–2914.

[45] Spetsieris, P., Ma, Y., Dhawan, V., and Eidelberg, D. Differential diagnosis of parkinsonian syndromes using PCA-based imaging features. *NeuroImage 45* (2009), 1241–1252.

[46] Sugiyama, M. Dimensionality reduction of multimodal labeled data by local Fisher discriminant analysis. *Journal of Machine Learning Research 8* (2007), 1027–1061.

[47] Tang, C. C., Poston, K. L., Eckert, T., Feign, A., Frucht, S., Gudesblatt, M., Dhawan, V., Lesser, M., Vonsattel, J.-P., Fahn, S., and Eidelberg, D. Differential diagnosis of parkinsonism: a metabolic imaging study using pattern analysis. *The Lancet Neurology 9* (2010).

[48] Teune, L. K., Renken, R. J., Mudali, D., de Jong, B. M., Dierckx, R. A., Roerdink, J. B. T. M., and Leenders, K. L. Validation of parkinsonian disease-related metabolic brain patterns. *Movement Disorders 28* (2013), 547–551.

[49] Vapnik, V. N. *The Nature of Statistical Learning Theory.* Springer-Verlag, 1995.

[50] Wiering, M. A., Van der Ree, M. H., Embrechts, M. J., Stollenga, M. F., Meijster, A., Nolte, A., and Schomaker, L. R. B. The neural support vector machine. In *Proceedings of the 25th Belgian-Dutch Conference on Artificial Intelligence* (2013), pp. 247–254.

[51] Zhu, X. Semi-supervised learning literature survey. Tech. Rep. 1530, Computer Sciences, University of Wisconsin-Madison, 2005.