Juuti, Mika; Corona, Francesco; Karhunen, Juha

Stochastic Discriminant Analysis for Linear Supervised Dimension Reduction

Published in:
Neurocomputing

DOI:
10.1016/j.neucom.2018.02.064

Published: 01/01/2018

Document Version
Publisher's PDF, also known as Version of record

Neurocomputing 291 (2018) 136–150

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Stochastic discriminant analysis for linear supervised dimension reduction

Mika Juuti[a,*], Francesco Corona[a,b], Juha Karhunen[a]

[a] Department of Computer Science, Aalto University, P.O. Box 00076 Aalto, Espoo, Finland
[b] Department of Computing, Federal University of Ceará, Fortaleza 60455-760, Brazil

ARTICLE INFO

Article history:
Received 30 June 2016
Revised 9 August 2017
Accepted 8 February 2018
Available online 2 March 2018

Communicated by Feiping Nie

Keywords:
Dimension reduction
Classification
Linear projection
Kullback–Leibler divergence
Information visualization
Distance based probabilities

ABSTRACT

In this paper, we consider a linear supervised dimension reduction method for classification settings: stochastic discriminant analysis (SDA). This method matches similarities between points in the projection space with those in a response space. The similarities are represented by transforming distances between points to joint probabilities using a transformation which resembles Student's t-distribution. The matching is done by minimizing the Kullback–Leibler divergence between the two probability distributions. We compare the performance of our SDA method against several state-of-the-art methods for supervised linear dimension reduction. In our experiments, we found that the performance of the SDA method is often better and typically at least equal to the compared methods. We have made experiments with various types of data sets having low, medium, or high dimensions and quite different numbers of samples, and with both sparse and dense data sets. If there are several classes in the studied data set, the low-dimensional projections computed using our SDA method provide often higher classification accuracies than the compared methods.

1. Introduction

Dimension reduction is an old research topic but in the current era of big data it is at least as relevant as earlier. There are several reasons for studying and using it. First, the data vectors may have quite high dimensions, which prevents applying poorly scalable and computationally demanding methods to them even with the current high computer processing power. The computational load of such methods can be proportional for example to the third power of the dimension, and grows rapidly intolerably high with increasing dimensionality. Second, dimension reduction reduces the amount of storage needed. Third, it can remove irrelevant information and noise from the data, and may lead for these reasons in practice to improved results. Fourth, the data is often projected to two-dimensional or sometimes to three-dimensional images for understanding its properties better. This information visualization aspect is important, because it is very difficult for humans to imagine what the data looks like in high dimensions.

We call the components x_i of the data vectors $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T$ as variables in this paper. Thus the data vectors are N-dimensional column vectors. We do not consider variable selection methods in which the dimensionality is reduced by trying to select the most relevant components of the data vectors for further processing. Instead, we consider feature extraction where the data vectors $\mathbf{x}$ are transformed to feature vectors $\mathbf{z} = [z_1, z_2, \ldots, z_M]^T$ whose components are some mixtures of the components of the original data vectors. Thus the feature vectors are M-dimensional column vectors, and their dimension M is generally clearly or much smaller than the dimension N of the original data vectors $\mathbf{x}$. We call the components of these feature vectors features. It is at least preferable that when applying feature extraction, the variables should be similar type quantities such as pixel intensities in digital images. If the variables are different quantities, for example the age, sex, yearly income etc. of a person, the scaling of these quantities affects greatly the results, and one can question the meaningfulness of computing a mixture of completely different types of variables.

The goal of dimension reduction can be simply information visualization, or achieving good results after dimension reduction in clustering, regression, or classification tasks. The dimension reduction methods can be divided into unsupervised and supervised ones. In unsupervised methods such as principal component analysis (PCA) [1–3], the only available information are the data vectors themselves. On the other hand, dimension reduction for classifi-

* Corresponding author.
E-mail addresses: mika.juuti@aalto.fi (M. Juuti), francesco.corona@ufc.br (F. Corona), juha.karhunen@aalto.fi (J. Karhunen).

https://doi.org/10.1016/j.neucom.2018.02.064

cation is a supervised task in which one has always some training data set at disposal. For each data vector belonging to the training set its correct class label is known. An example of supervised dimension reduction is linear discriminant analysis (LDA) [2,4]. Another important grouping of feature extraction and dimension reduction methods is that they can be either linear or nonlinear, depending on whether the vector-valued mapping $\mathbf{f}(\cdot)$

$$\mathbf{z} = \mathbf{f}(\mathbf{x}) \tag{1}$$

from the data vectors $\mathbf{x}$ to the compressed feature vectors $\mathbf{z}$ is linear or nonlinear.

The stochastic discriminant analysis (SDA) method which we introduce and discuss in this paper is a linear supervised feature extraction method. It is intended for classification after a mapping into a relatively small-dimensional feature space, and for visualization of the data containing several classes in two dimensions.

The remainder of this paper is organized as follows. In the next section we review many related dimension reduction methods. Section 3 deals with the SDA method and minimization of its cost function. Section 4 presents experimental results of the proposed SDA method compared with traditional and state-of-the-art approaches for dimension reduction with several data sets having quite different properties. The last section contains conclusions and remarks of this study.

2. Related work

Because our stochastic discriminant analysis method is a linear supervised method for dimension reduction, we discuss here mainly such methods. The most widely used dimension reduction method is still principal component analysis (PCA) [1–3]. It is an old linear unsupervised feature extraction and dimension reduction method which maps the N-dimensional original data vectors $\mathbf{x}$ into feature vectors $\mathbf{z}$ which have a lower dimension M:

$$\mathbf{z} = \mathbf{W}^T\mathbf{x} \tag{2}$$

The row vectors of the $M \times N$ mapping matrix $\mathbf{W}^T$ consist of the eigenvectors of the data covariance matrix $\mathbf{C}_{xx} = \mathrm{E}[\mathbf{x}\mathbf{x}^T]$ corresponding to the largest eigenvalues, assuming that the data vectors have zero mean. If this is not the case, the K data vectors $\mathbf{x}_j$, $j = 1, 2, \ldots, K$ can always be preprocessed to have zero mean by first estimating their mean vector $\mathbf{m} = \frac{1}{K}\sum_{j=1}^{K}\mathbf{x}_j$, and then subtracting $\mathbf{m}$ from the data vectors $\mathbf{x}_j$. Thus PCA is easy to compute, and it is computationally not too demanding provided that the data vectors $\mathbf{x}$ are not truly high-dimensional. PCA minimizes the mean-square representation error for all linear $M \times N$ mappings $\mathbf{W}^T$, and the components of the feature vector $\mathbf{z}$ have maximal variances and are uncorrelated in directions that are mutually orthogonal [1–3]. We use PCA as a preprocessing step in our SDA method, and as a reference method in our comparison experiments. However, PCA does not often perform well in dimension reduction in classification problems, because it does not utilize the class information available in the training set in any way.

The oldest supervised linear dimension reduction method is linear discriminant analysis (LDA) [2,4] developed already in 1930's. The criterion function in LDA for the case of two classes is

$$J(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}} \tag{3}$$

where $\mathbf{w}$ is the N-dimensional projection vector for mapping the data into one dimension by computing the inner product $\mathbf{w}^T\mathbf{x}$. $\mathbf{S}_B$ is the $N \times N$ between-class covariance matrix and $\mathbf{S}_W$ is the $N \times N$ within-class covariance matrix. The criterion (3) is maximized in order to maximize the distance between the two classes and minimize the distance within the same class at the same time. The solution can be computed from a linear equation (see [2,4] for details), but it requires the inversion of the matrix $\mathbf{S}_W$. This can become computationally prohibitive for very high-dimensional data, such as digital images. LDA can have also problems with singular within-class covariance matrices $\mathbf{S}_W$, and therefore it is often coupled with dimension reduction using PCA in image recognition tasks [5].

LDA has two basic limitations (in addition to the linearity of the mapping): the probability distributions of the two classes are assumed to be Gaussian, and these Gaussian distributions are assumed to have the same covariance matrix $\mathbf{S}_W$ [4]. LDA can be extended to several classes as follows. It is assumed that each of the C classes has its own mean vector $\mathbf{m}_i$ and the same covariance matrix $\mathbf{S}_W$. Define the sample covariance matrix of the class means as

$$\mathbf{S}_C = \frac{1}{C}\sum_{i=1}^{C}(\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T \tag{4}$$

where $\mathbf{m}$ is the mean of the class means $\mathbf{m}_i$. Then the class separation in the direction $\mathbf{w}$ is given by [4]

$$J(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{S}_C\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}} \tag{5}$$

The optimal direction $\mathbf{w}$ which maximizes the separation (5) is given by the eigenvector corresponding to the largest eigenvalue of the matrix $\mathbf{S}_W^{-1}\mathbf{S}_C$.

Linear discriminant analysis (LDA) has inspired many researchers and there exist several modifications on it. We mention here the following linear supervised dimension reduction methods based on LDA. In the paper [6], the problem appearing in face recognition that the within-class covariance matrix $\mathbf{S}_W$ becomes always singular is solved by first mapping the face images to a lower dimensional space. In marginal Fisher analysis [7], new criteria that characterize intra-class compactness and inter-class separability are developed for handling cases in which the probability distributions of the classes are not Gaussian. A direction $\mathbf{w}$ which minimizes the ratio of these criteria is then sought. Essentially the same idea has been introduced in the paper [8]. Local Fisher discriminant analysis (LFDA) [9] introduces locality into the LDA method, and is particularly useful for samples consisting of intraclass separate clusters. Maxmin distance analysis (MMDA) [10] considers maximization of the minimum pairwise interclass samples.

In fact, a linear discriminant analysis type solution can be found by maximizing either a trace ratio or a ratio trace criterion which are closely related. These two criteria are compared and studied both theoretically and experimentally in [11]. The ratio trace criterion is conventionally used because it has a closed form but inexact solution, while the trace ratio criterion requires an iterative maximization method. Both these criteria yield qualitatively similar results, but the trace ratio provides somewhat better classification results, as shown by large number of experiments with various data sets in [11]. We use the closed form solution of the ratio trace criterion in the experiments of this paper and refer to it as LDA.

Partial least squares (PLS) regression is a supervised linear dimension reduction technique that tries to find from the input matrix subspaces that explain the largest amount of variance in the response matrix. When used in supervised manner with labeled data, it is referred to as PLS-DA [12]. Kernel dimension reduction (KDR) [13] is a sufficient dimension reduction method [14] for classification and regression data. A sufficient dimension reduction contains all the regression information that the original space contained about the response variable. KDR tries to find the central subspace [14] for the input data, which is the intersection of all the dimension reduction subspaces. KDR does not impose any par-

ticular assumptions on the form of the covariance matrix of the input data. However, it has high computational load and memory requirements. A gradient version of the KDR method called gKDR has been developed in [15] for faster computation.

Supervised PCA (SPCA) introduced by Barshan et al. in [16] is a regression technique that finds the principal components having the maximum dependence on the given response variable. SPCA tries to find variables that are orthogonal in a kernel space of the response variable. Using the Hilbert–Schmidt independence criterion, SPCA can be computed from an eigendecomposition. The authors have developed also a dual-space and kernel variant of the SPCA method called KSPCA in [16], extending the usage of the method.

Before proceeding, we mention briefly a few nonlinear dimension reduction methods. They are often called manifold learning methods because they assume that the data lies at least roughly in some smaller dimensional manifold which is then estimated for reducing the dimensionality. See Section 5.11.3 in [17] for a more detailed description of this idea. Belkin and Niyogi developed a nonlinear manifold learning technique called Laplacian eigenmaps for projecting high-dimensional data into a low-dimensional space in such a way that local points in the high-dimensional space are kept close in the projection [18]. Slightly later on, they developed a linear variant of Laplacian eigenmaps called locality preserving projections that projected the data points using a linear transformation of the data points [19]. This technique has the benefit that the projection is not defined only for the training data points but in the whole ambient space.

Other well-known manifold learning methods are kernel PCA [2,20] in which PCA is applied after a nonlinear mapping into a higher-dimensional kernel space, and local linear embedding (LLE) [21]. For more references and information on manifold learning methods, see [17,22].

All the linear dimension reduction methods discussed thus far except for PCA are supervised techniques. The following methods are unsupervised, and hence they do not use any training data with known class labels or outputs in computing their dimension reduction mappings. Neighborhood embedding techniques recreate a high-dimensional neighborhood structure in a low-dimensional space. These techniques cast the problem of finding a low-dimensional embedding as a problem of matching two probability distributions: one modeling a complex high-dimensional structure, and one modeling a low-dimensional manifold of the data. The methods preserve point-to-point neighborhood relations. The low-dimensional embedding is created by defining probability mass functions based on point-to-point distances in both high-dimensional and low-dimensional space. An information measure between these two joint probability distributions is then iteratively decreased. The most common information measure is the Kullback–Leibler divergence [2,23,24] which measures the difference between two probability distributions. We shall discuss it in more detail later on.

The neighbor retrieval visualizer method (NeRV) [25] matches a convex combination of divergences between the probabilities defining the high-dimensional structure and low-dimensional reconstruction. The proportion is hand-tuned, giving the user some control in penalizing precision and recall errors, see [25] for more details.

The stochastic neighbor embedding (SNE) method introduced in [26] and its various extensions have during the last years become popular in feature extraction, inspiring several modified and improved methods. Essentially the same method as SNE was introduced under the name informative discriminative analysis in [27]. The basic principle in the SNE method is to convert pairwise Euclidean distances into probabilities of selecting neighbors to model pairwise similarities. However, the basic SNE method suffers from optimization and crowding problems discussed below.

In [28], Van der Maaten and Hinton introduced the so-called t-SNE method where t refers to the Student's t probability distribution. The high-dimensional structure is modeled using Gaussian radial basis function kernels, where the authors use a binary search for determining appropriate kernel widths. Low-dimensional reconstructions are modeled with first-order t-distributed kernels. Both kernel values are normalized to sum to one and are called probabilities by the authors. The motivation for the asymmetric matchup is that it solves the crowding problem: the space available to model distant data points is too small, compared to the space available to model near data points. Yang et al. analyzed in [29] systematically the characteristics of the heavy-tailed distribution and the solutions to the crowding problem. Wu et al. explored in [30] how to measure similarity on a manifold more accurately, and introduced a feature extraction method based on SNE and t-SNE which they call manifold stochastic neighbor projection (MSNP). Even though the MSNP method has several advantages in feature extraction, it is still an unsupervised method that does not use the class information available in classification problems.

For overcoming this deficiency of the MNSP method, Zheng et al. developed a supervised method called discriminative stochastic neighbor embedding analysis (DSNE) in [31]. It resolves the problems mentioned above, but it has a high computational cost and is therefore not applicable to large-scale classification problems where the data vectors are high-dimensional. The same authors developed in [32] a faster version based on the DSNE method, which they call fast discriminative stochastic neighbor embedding analysis (FDSNE). In [32], they also introduce a nonlinear version of the FDSNE method by applying the kernel trick.

One of the authors of this paper participated in developing a method called supervised distance preserving projections (SDPP) in [33]. The SDPP method minimizes the difference between pairwise distances among projected input covariates and distances among responses locally. The SDPP method is mainly useful in regression problems. It did not work well in all the classification problems discussed in the experimental part of this paper, and therefore it is not included in our comparison experiments. In SDPP certain distances in the cost function can change the visualization to a great extent. Our new SDA method in the next section tries to avoid these problems encountered when applying the SDPP method to multiclass data in high-dimensional settings by matching probabilities instead of distances.

With point-to-point mappings it is often not easy to place out-of-sample data points. Parametric methods provide a mapping of the data points. Amongst others, parametric t-SNE method learns a mapping by using a deep neural network [34]. Out-of-sample data points can then be embedded by running them through the network. However, this is a nonlinear dimension reduction method that is pretty complicated and difficult to train even though it yields excellent results for the well-known MNIST data set [35] of handwritten digits.

3. Stochastic discriminant analysis (SDA)

3.1. The SDA method

We first define the data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K]^T \in \mathbb{R}^{K\times N}$ as a $K \times N$ matrix which has the K N-dimensional column data vectors $\mathbf{x}_j$ as its row vectors. Formally, we are reducing the number N of variables in the data matrix $\mathbf{X}$ by finding a linear subspace of it:

$$\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_K]^T = \mathbf{XW} \tag{6}$$

where $\mathbf{Z}$ is a $\mathbb{R}^{K\times M}$ matrix, $\mathbf{W} \in \mathbb{R}^{N\times M}$, and $M \leq C \ll N$ where C is the number of classes. From Eq. (6), we get for its i:th row $\mathbf{z}_i^T =$

$\mathbf{x}_i^T\mathbf{W}$, or $\mathbf{z}_i = \mathbf{W}^T\mathbf{x}_i$ which is equivalent to the PCA mapping (2). However, in the SDA method and other mapping methods than PCA the mapping matrix $\mathbf{W}$ is defined in a different way.

We are using class information from the response matrix

$$\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K]^T \in \mathbb{I}^{K\times C} \tag{7}$$

to find this projection. For each data vector $\mathbf{x}_i$, the corresponding response vector $\mathbf{y}_i$ specifies its class label. More specifically, if the data vector $\mathbf{x}_i$ belongs to the class c_j, the j:th element of its response vector $\mathbf{y}_i$ is equal to one, while the other elements of $\mathbf{y}_i$ are zero. The matrix $\mathbb{I}$ on the right hand side of Eq. (7) resembles unit matrix in that on each of its rows one element equals to one while the other elements are zero.

In the SDA method, we search for a linear subspace of the data matrix where the elements belonging to the same class are mapped close to each other, and those belonging to different classes further away. Following van der Maaten and Hinton [28], we cast the problem of finding low-dimensional embeddings as a problem of matching two probability distributions: one modeling a complex high-dimensional point-to-point structure, and another modeling a low-dimensional manifold of the data. We denote these distributions by P and Q, respectively, and their elements by p_{ij} and q_{ij}. We call the values p_{ij} target probabilities, and values q_{ij} model probabilities. Only the values q_{ij} are optimized in the algorithm presented in Section 3.2, while the values p_{ij} remain constant.

We search for a linear subspace of the data matrix by matching model probabilities q_{ij} with target probabilities p_{ij}. Denote by

$$d_{ij} = \|\mathbf{z}_i - \mathbf{z}_j\|_2 \tag{8}$$

the Euclidean distance between two points i and j in the transformed $\mathbf{Z}$-space, where $\mathbf{z}_i = \mathbf{W}^T\mathbf{x}_i$ is the low-dimensional embedding coordinate. The model probabilities characterizing the distances d_{ij} are defined by

$$q_{ij}(\mathbf{W}) = \frac{\pi^{-1}\cdot(1+d_{ij}^2)^{-1}}{\sum_{k=1}^{K}\sum_{l=1}^{K}\pi^{-1}\cdot(1+d_{ij}^2)^{-1}} \tag{9}$$

The numerator $\pi^{-1}\cdot(1+d_{ij}^2)^{-1}$ comes from the probability density function of Student's t-distribution [36] having one degree of freedom. The common factor π^{-1} can be left out of the expression:

$$q_{ij}(\mathbf{W}) = \frac{(1+d_{ij}^2)^{-1}}{\sum_{k=1}^{K}\sum_{l=1}^{K}(1+d_{ij}^2)^{-1}}. \tag{10}$$

The numbers q_{ij} are called probabilities, because they are non-negative and their sum equals one:

$$\sum_{i=1}^{K}\sum_{j=1}^{K} q_{ij}(\mathbf{W}) = 1 \tag{11}$$

The probabilities $q_{ij}(\mathbf{W})$ are inspired by the Student's t-distribution and have longer tails than the standard Gaussian distribution, but they are exactly not t-distributed, despite being called so in literature [28].

Denoting the unnormalized probability in the numerator of (10) by

$$\tilde{q}_{ij} = (1+d_{ij}^2)^{-1} \tag{12}$$

we can write the Eq. (10) simply

$$q_{ij} = \tilde{q}_{ij}/\sigma_q, \qquad \sigma_q = \sum_{i=1}^{K}\sum_{j=1}^{K}\tilde{q}_{ij} \tag{13}$$

Fig. 1 shows the profile of the unnormalized probabilities $\tilde{q}_{ij}$. The maximum value is one when the distance d_{ij} between the two points $\mathbf{z}_i$ and $\mathbf{z}_j$ is zero, and approaches zero when the distance $d_{ij}\to\infty$.

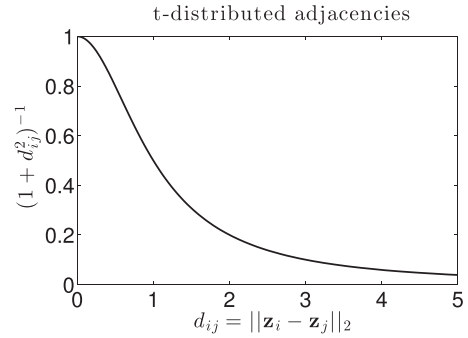

Fig. 1. Unnormalized model probabilities $\tilde{q}_{ij}$ as a function of the distance d_{ij} between two points. Their distribution has longer tail than the respective Gaussian distribution.

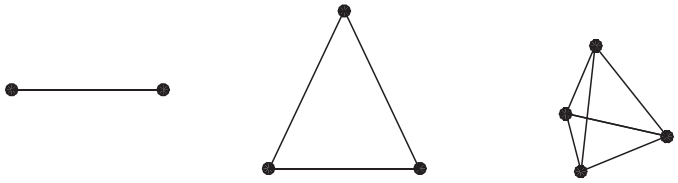

Fig. 2. From left: target manifolds of two, three, and four classes.

Unlike [28], we do not use in the SDA method high-dimensional distances in defining target probabilities p_{ij}. We want to enforce the condition that the data points belonging to the same class are projected close to each other in the $\mathbf{Z}$-space, and that the points belonging to different classes are mapped further away. In an ideal embedding, the unnormalized probability $\tilde{q}_{ij} = 1$ when the points i and j belong to the same class, corresponding to zero distance between them. Similarly, ideally $\tilde{q}_{ij} = 0$ when the points i and j belong to different classes, corresponding to an infinite distance between them. These conditions hold also for the normalized probabilities σ_q in Eq. (13).

In our SDA method, we rely only on the class information in determining the ideal embeddings. The normalized target probabilities are defined similarly as in Eq. (13)

$$p_{ij} = \tilde{p}_{ij}/\sigma_p, \qquad \sigma_p = \sum_{i=1}^{K}\sum_{j=1}^{K}\tilde{p}_{ij} \tag{14}$$

where σ_p is the normalization term, and

$$\tilde{p}_{ij} = \begin{cases} 1, & \text{if } \mathbf{y}_i = \mathbf{y}_j \\ \epsilon, & \text{otherwise} \end{cases}, \tag{15}$$

where $\epsilon > 0$ is any small number close to zero. The target probabilities in Eq. (14) define the ideal distances. Optimally both $\tilde{p}_{ij} = \tilde{q}_{ij}$ and $p_{ij} = q_{ij}$ for all $i, j \in [1, \ldots, K]$. In such a situation, all the points belonging to the same class are mapped to one dot (point), and all points belonging to different classes are at an equal distance from each other.

With a given ϵ, we can calculate the ideal point-to-point distances in $\mathbf{Z}$-space to be

$$d_{ij}^* = \begin{cases} 0, & \text{if } \mathbf{y}_i = \mathbf{y}_j \\ \sqrt{\epsilon^{-1}-1}, & \text{otherwise.} \end{cases}, \tag{16}$$

We can see that ϵ scales how close the superimposed points are to each other. Eq. (16) defines a geometric structure that has C nodes, where each node is separated by an equal distance of $\sqrt{\epsilon^{-1}-1}$. This geometric structure is called a regular simplex. Fig. 2 shows target structures for two class, three class, and four class problems. Note that the structure of the simplex is independent of the input data $\mathbf{X}$, depending only on the number classes in $\mathbf{Y}$.

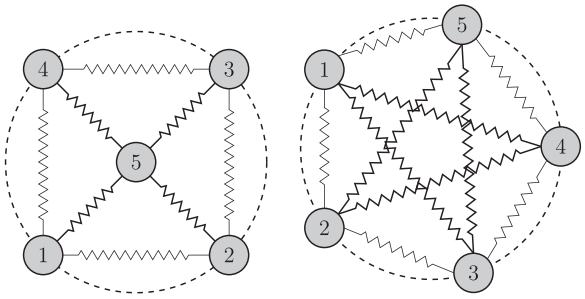

Fig. 3. Two ideal embeddings of 5 classes into two dimensions, using SDA (low ϵ, left) and HSE (high ϵ, right). With low ϵ, too small between-class distances incur a large penalty and the embedding results in the utilization of the whole volume of the hypersphere. With high ϵ, too large between-class distances incur a large penalty and the ideal embedding results in utilization of the surface of the target hypersphere.

Given sufficiently many target dimensions, and by setting $\epsilon \to 0$, the optimality criterions of SDA and LDA could yield similar embeddings, because both methods try to construct linear projections whose within-class variances are zero and between-class variances are infinite. Both methods define an embedding structure that has an intrinsic dimensionality of $C-1$ dimensions. However, the SDA and LDA methods deal with the shortcomings in the target dimensionality differently, as is explained later on in Fig. 3 and Section 3.3.

The Kullback–Leibler (KL) divergence [2,24] measures the difference between two probability distributions. Here we consider for clarity first two discrete probability densities A and B which both have J possible discrete values $a_1, a_2, \ldots, a_J$ and $b_1, b_2, \ldots, b_J$, respectively. Their KL divergence is

$$D_{KL}(A||B) = \sum_{j=1}^{J} a_j \log(a_j/b_j) \tag{17}$$

The KL divergence is zero only when the two probability distributions are the same, that is $A = B$. However, it is theoretically not a true distance because the KL divergence does not fulfill the triangle inequality, and it is not symmetric: $D_{KL}(A||B) \neq D_{KL}(B||A)$. For the two probability densities A and B, one can define two Kullback–Leibler divergences $D_{KL}(A||B)$ and $D_{KL}(B||A)$, which have different properties as discussed in Section 21.2.2 in [24]. The version (17) which we are using is called M-projection and zero-avoiding, because it becomes infinite if one of the probabilities b_j is zero.

Using the KL divergence (17) for the probabilities p_{ij} and q_{ij} defined, respectively, in Eqs. (14) and (10), we can write the cost function of our SDA method which is minimized:

$$J(\mathbf{W}) = \sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij} \log \frac{p_{ij}}{q_{ij}(\mathbf{W})} + \lambda \sum_{i=1}^{N}\sum_{j=1}^{M} w_{ij}^2 \tag{18}$$

The second term is the usual weight decay (Tikhonov) type regularizer [2,24] which penalizes for large values of the elements w_{ij} of the $N \times M$ weight matrix $\mathbf{W}$. The parameter λ determines how much one takes into account regularization compared with the first part of the cost function, the Kullback–Leibler divergence, which is the actual cost.

We are searching for the thin linear projection matrix $\mathbf{W}$ that minimizes the Kullback–Leibler divergence in which the target probability distribution P is approximated with the model probability distribution Q. The inefficiency of encoding ideal distances in the response space using realized distances in the embedding space is measured. The probability distribution (10) causes asymmetric distance penalties: the cost function is more sensitive to deviations in within-class distances than to deviations in between-class distances. Deviations from the ideal within-class distances incur a relatively large cost, but deviations from ideal between-class distances incur a much smaller cost.

If we use regularization and the value of the regularization parameter λ in (18) is searched by cross-validation, we refer to our method as Regularized SDA, abbreviated as RSDA. Normally λ is set to zero. Weight decay (Tikhonov) regularization is often applied to ill-posed problems. In SDA, we have local solutions that depend on the initialization. The initial solution in SDA is obtained with PCA, giving orthogonal vectors with maximum variance. High-dimensional problems with many non-singular dimensions have a high degree of freedom, and they can in principle have infinite parameters choices (by convex combinations) that produce the optimal solution. The KL criterion is insensitive to the weights used to find the projection. Applying Tikhonov regularization changes the optimization criteria so that optimal solutions are ranked in order of the least Frobenius norm. Additionally, the optimization process can also be made smoother by constraining the size of the elements of weight matrix $\mathbf{W}$.

3.2. Minimization of the cost function

We consider now the minimization of the cost function (18). First, we compute its gradient with respect to the weight matrix $\mathbf{W}$. Then we use this gradient in various gradient type minimization methods which are discussed later on in this subsection.

The essential steps in obtaining the gradient are as follows. We use the shorthand notation $q_{ij} = q_{ij}(\mathbf{W})$. We also write the squared distance in the embedding space as

$$\begin{aligned} D_{ij} &= D_{ij}(\mathbf{W}) = d_{ij}^2 = \|\mathbf{z}_i - \mathbf{z}_j\|^2 \\ &= \boldsymbol{\tau}_{ij}^T \mathbf{W}\mathbf{W}^T \boldsymbol{\tau}_{ij} = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{W}\mathbf{W}^T (\mathbf{x}_i - \mathbf{x}_j) \end{aligned} \tag{19}$$

where $\boldsymbol{\tau}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ is the difference between the i:th and j:th original data vectors $\mathbf{x}_i$ and $\mathbf{x}_j$, respectively.The matrices $\mathbf{P}$, $\mathbf{Q}$, $\bar{\mathbf{Q}}$ and $\mathbf{D}$ are $K \times K$ real-valued matrices, whose elements are respectively p_{ij}, q_{ij}, $\bar{q_{ij}}$, and D_{ij}.

The gradient of the first part of the cost function (18) which is the Kullback–Leibler divergence $KL(P||Q(\mathbf{W}))$, is

$$\begin{aligned} \frac{dKL(P||Q(\mathbf{W}))}{d\mathbf{W}} &= \sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij} \frac{1}{q_{ij}} (-1) \frac{dq_{ij}}{d\mathbf{W}} \\ &= \sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij}(-1) \left[\sum_{k=1}^{K}\sum_{l=1}^{K} q_{kl}\bar{q}_{kl} \frac{dD_{kl}}{d\mathbf{W}} - \bar{q}_{ij} \frac{dD_{ij}}{d\mathbf{W}} \right] \\ &= \sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij}\bar{q}_{ij} \frac{dD_{ij}}{d\mathbf{W}} - \sum_{k=1}^{K}\sum_{l=1}^{K} q_{kl}\bar{q}_{kl} \frac{dD_{kl}}{d\mathbf{W}} \\ &= \sum_{i=1}^{K}\sum_{j=1}^{K} (p_{ij} - q_{ij})\bar{q}_{ij} \frac{dD_{ij}}{d\mathbf{W}} \\ &= \sum_{i=1}^{K}\sum_{j=1}^{K} (p_{ij} - q_{ij})\bar{q}_{ij} \boldsymbol{\tau}_{ij}^T \boldsymbol{\tau}_{ij} \mathbf{W}, \end{aligned} \tag{20}$$

since

$$\sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij} k = \left(\sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij} \right) k = k \tag{21}$$

where k is an arbitrary constant. Here $(1 + D_{ij})^{-1} = \bar{q}_{ij}$ denotes the unnormalized probability in (12).

Adding the gradient $2\lambda\mathbf{W}$ of the second regularization term to the cost function (18), we get for its total gradient

$$\nabla_{\mathbf{W}} J = \frac{dJ}{d\mathbf{W}} = \sum_{i=1}^{K}\sum_{j=1}^{K}(p_{ij} - q_{ij})\bar{q}_{ij}\boldsymbol{\tau}_{ij}^{T}\boldsymbol{\tau}_{ij}\mathbf{W} + 2\lambda\mathbf{W} \tag{22}$$

This expression can be written in matrix form:

$$\nabla_{\mathbf{W}} J = 2\mathbf{X}^{T}\mathbf{L}\mathbf{X}\mathbf{W} + 2\lambda\mathbf{W}, \tag{23}$$

where the matrix

$$\mathbf{L} = \mathbf{G}^{+} - \boldsymbol{\Lambda} \in \mathbb{R}^{K\times K} \tag{24}$$

is calculated as

$$\begin{aligned} \mathbf{G} &= (\mathbf{P} - \mathbf{Q}) \odot \bar{\mathbf{Q}} \\ \mathbf{G}^{+} &= \mathbf{G} + \mathbf{G}^{T} \\ \boldsymbol{\Lambda} &= \textstyle\sum_j \mathbf{G}^{+}_{ij} \end{aligned} \tag{25}$$

Here $\odot$ denotes the Hadamard product, and $\mathbf{G}^{+}$ is a symmetrized version of the matrix of $\mathbf{G} \in \mathbb{R}^{K\times K}$. The matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{K\times K}$ is a diagonal matrix containing the row sum of $\mathbf{G}^{+}$. The matrix $\mathbf{L}$ is the difference between two Laplacian matrices:

$$\mathbf{L} = \mathbf{L}_{\mathbf{P}} - \mathbf{L}_{\mathbf{Q}}, \tag{26}$$

where $\mathbf{L}_{\mathbf{P}}$ and $\mathbf{L}_{\mathbf{Q}}$ are calculated from the adjacency matrices $\mathbf{G}_{\mathbf{P}} = \mathbf{P} \odot \bar{\mathbf{Q}}$ and $\mathbf{G}_{\mathbf{Q}} = \mathbf{Q} \odot \bar{\mathbf{Q}}$. A Laplacian matrix is a symmetric diagonally dominant matrix and therefore positive semi-definite, but the matrix $\mathbf{L}$ need not be positive semi-definite.

There are many ways of optimizing the cost function (18) based on the gradient information. Algorithm 1 presents the pseudo-code

Algorithm 1 Gradient-based minimization for stochastic discriminant analysis (SDA).

Input: Input matrix $\mathbf{X} \in \mathbb{R}^{K\times N}$, responsematrix $\mathbf{Y} \in \mathbb{I}^{K\times C}$,target dimensionality M, regularization term λ, and optimal tolerance δ.
Output: Projection matrix $\mathbf{W}$.
1. Calculate the target probabilities $\mathbf{P} \in \mathbb{R}^{K\times K}$from Eq. (15).
2. Initialize $\mathbf{W}$ using PCA, by putting as its rows the M principal eigenvectors of the covariance matrix of the data $\mathbf{X}$.
3. Calculate the model probabilities $\mathbf{Q}$ from Eq. (10).
4. Evaluate the cost function $J(\mathbf{W})$ from Eq. (18).
5. Assign $t = 0$, $\delta_J = \infty$.
while $\delta_J > \delta$ **do**
6. Compute the gradient (23).
7. Vectorize the projection matrix: $\mathbf{w}_t = \mathrm{vec}(\mathbf{W})$.
8. Vectorize the gradient: $\mathbf{g}_t = \mathrm{vec}(\nabla_{\mathbf{W}} J)$.
9. Determine the descent direction $\boldsymbol{d}_t$.
10. Determine step size η_t.
11. Update the solution vector: $\mathbf{w}_t = \mathbf{w}_{t-1} + \eta_t \boldsymbol{d}_t$.
12. Convert the vector $\mathbf{w}_t$ back into the matrix $\mathbf{W}_t$.
13. New iteration $t \leftarrow t + 1$.
14. Update the model probabilities $\mathbf{Q}$ using Eq. (10).
15. Calculate the new cost $J(\mathbf{W})$ from Eq. (18).
16. Update the change $\delta_J = J_t - J_{t-1}$ of the cost function.
end while
17. Orthogonalize $\mathbf{W}_t$ using thin singular value decomposition(SVD): $\hat{\mathbf{U}}\hat{\mathbf{S}}\mathbf{V} = \mathbf{W}_t$.
18. Return $\mathbf{W} = \hat{\mathbf{U}}\hat{\mathbf{S}}$.

for obtaining a projection matrix $\mathbf{W}$ using stochastic discriminant analysis. First, the target probabilities $\mathbf{P}$ and model probabilities $\mathbf{Q}$ collected in these matrices are calculated. Then the cost function is evaluated, and its gradient (23) is computed. The projection matrix $\mathbf{W}$ and its gradient must be vectorized in the optimization Algorithm 1. Note that the target probabilities p_{ij} are determined based on the labeling of the elements in the beginnning of the algorithm, but the model probabilities q_{ij} depend on the low-dimensional coordinates, and they must be recalculated at each iteration. The initial projection matrix is obtained using PCA.

The vectorized projection matrix $\mathbf{w}$ and its vectorized gradient $\mathbf{g}$ can be plugged into any gradient-based optimization method. The basic method is the usual steepest descent method, but different versions of the conjugate gradient method [37–39] and the limited-memory BFGS algorithm [38,40] are more efficient in solving problems with a large number of variables, and converge faster.

The evaluation of the gradient is the most time consuming part of the optimization. The applied optimization method determines the descent direction, and a line search method can be used for determining the optimal step size η_t which minimizes the cost function $J(\mathbf{W})$ as much as possible in the direction of the gradient. The optimization and line search methods may require additional function evaluations. At the end, the search directions are orthogonalized using singular value decomposition (SVD). The use of thin SVD saves computational time.

3.3. Low-dimensional projections

Recall that the target embedding depends on the response space $\mathbf{Y}$ only, not on the input space $\mathbf{X}$. The optimal embedding for a response space with C classes is a simplex with C nodes, with an inherent dimensionality of $C-1$ dimensions. Thus, embedding dimensionalities M larger or equal to $C-1$ produce simplex manifolds with inherent dimensionalities of $C-1$. But in case the embedding dimensionality M is smaller than $C-1$, the solution is not intuitive to imagine. In such a case the interplay between ϵ and the shape of the t-distribution plays a role. We evaluate ϵ which is a parameter in our system.

The most important property to consider is whether it becomes prohibitively expensive to embed nodes further away than the optimal embedding. We argue that at a certain ϵ_t it becomes too expensive to embed nodes further away, so that the nodes are embedded too close to each other, which may hurt subsequent classification accuracy. However, very low values of ϵ may be slow to optimize.

Small values such as $\epsilon \to 0$ cause an optimal embedding similar to the one in Fig. 3 (left). Too large values of ϵ cause a different type of compromise similar to the one in Fig. 3 (right). The difference is clear, and we call these embeddings with different names to avoid confusion. We call the former case SDA and the latter case hypersphere embedding (HSE). The discrepancy in the shapes of the SDA and HSE embeddings results from how deviations from the ideal distances are treated: in SDA, the distances $d > d_{ideal}$ occur at a small cost, while in HSE the distances $d > d_{ideal}$ occur at an increasingly high cost.

Embeddings with SDA utilize a larger volume to separate classes and have more freedom in separating classes given a specific target dimension. However, the superiority is not obvious when we notice that embeddings with HSE are ideally embedded on a $(C-1)$-dimensional manifold in the C-dimensional space (the radius of the sphere is a constant). In certain cases, we notice that in very high-dimensional cases with extremely low-dimensional embeddings, the reward for embedding classes at the ideal distance is so strong that separate classes might be embedded on top of each other, for maximizing the number of fulfilled ideal class-to-class distances. This is contrary to our goal of separating maximally different classes. By default we choose $\epsilon = 1/C$, the inverse of the number C of classes, since the optimization criterion converges slowly with small values of ϵ. We verify the found embeddings experimentally below.

The optimization process described in Algorithm 1 decreases the KL divergence step by step towards a minimum cost structure until no significant further process can be made, measured

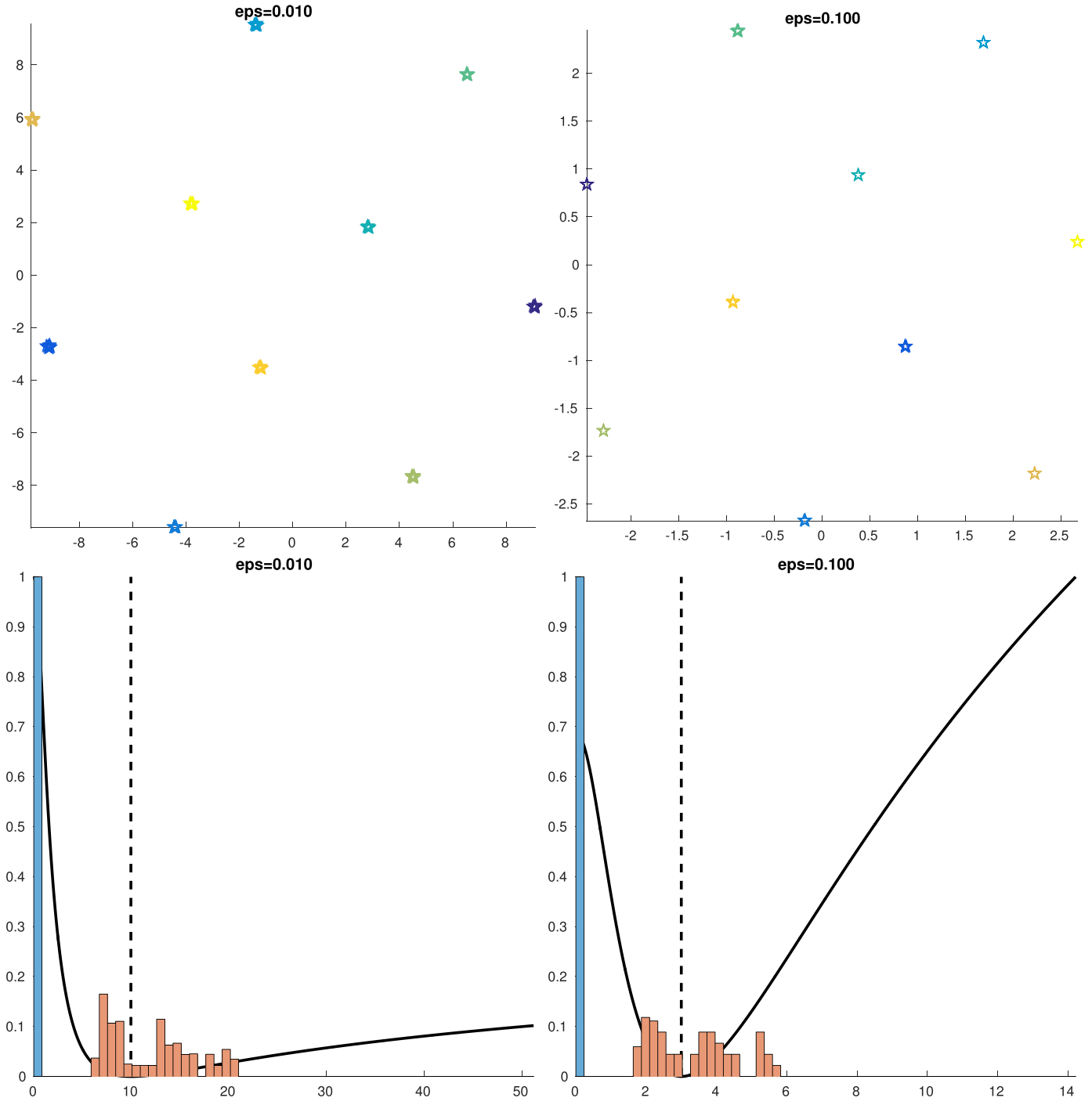

Fig. 4. Upper row: embeddings of a subset of 100 samples of the USPS dataset for the hyperparameters $\epsilon = 0.01$ (left), and 0.1 (right). Lower row: the normalized KL divergences obtained by varying all the between-class distances and the normalized histograms of the realized within-class (blue) distances and the between-class distances (orange). There are ten classes in the dataset. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

by the tolerance δ in Algorithm 1. Recall that the optimal embedding does not depend on the dataset $\mathbf{X}$. For illustrative purposes we use a smaller dataset, since the clustering behavior is clearer in a smaller dataset. Using larger datasets exhibits similar behavior, although the linearity constraint restricts the cleanness of the class clusters. The reader is advised to compare the figures below with large datasets in Fig. 11.

Fig. 4 shows the optimal two-dimensional embeddings (top row) and corresponding costs (bottom row) of projections of a 100 sample subset of the USPS dataset for the hyperparameter values $\epsilon = 0.01$ and 0.1, and Fig. 5 for $\epsilon = 0.5$ and 0.9. Notice that because the USPS dataset has $C = 10$ classes, $\epsilon = 1/C = 0.1$. The black curve shows the cost of the hypothetical KL criteria, evaluated by keeping within-class distances zero and varying all the between-class distances. The vertical dashed line shows the ideal distance, obtained with the Student's t probability distribution formula as $d_{ideal} = \sqrt{(1-\epsilon)/\epsilon}$. The cost of the KL criteria is minimized when the between-class distances precisely equal d_{ideal}, but this would require a target dimensionality of $M = C - 1 = 9$. The cost curves are normalized in each subimage.

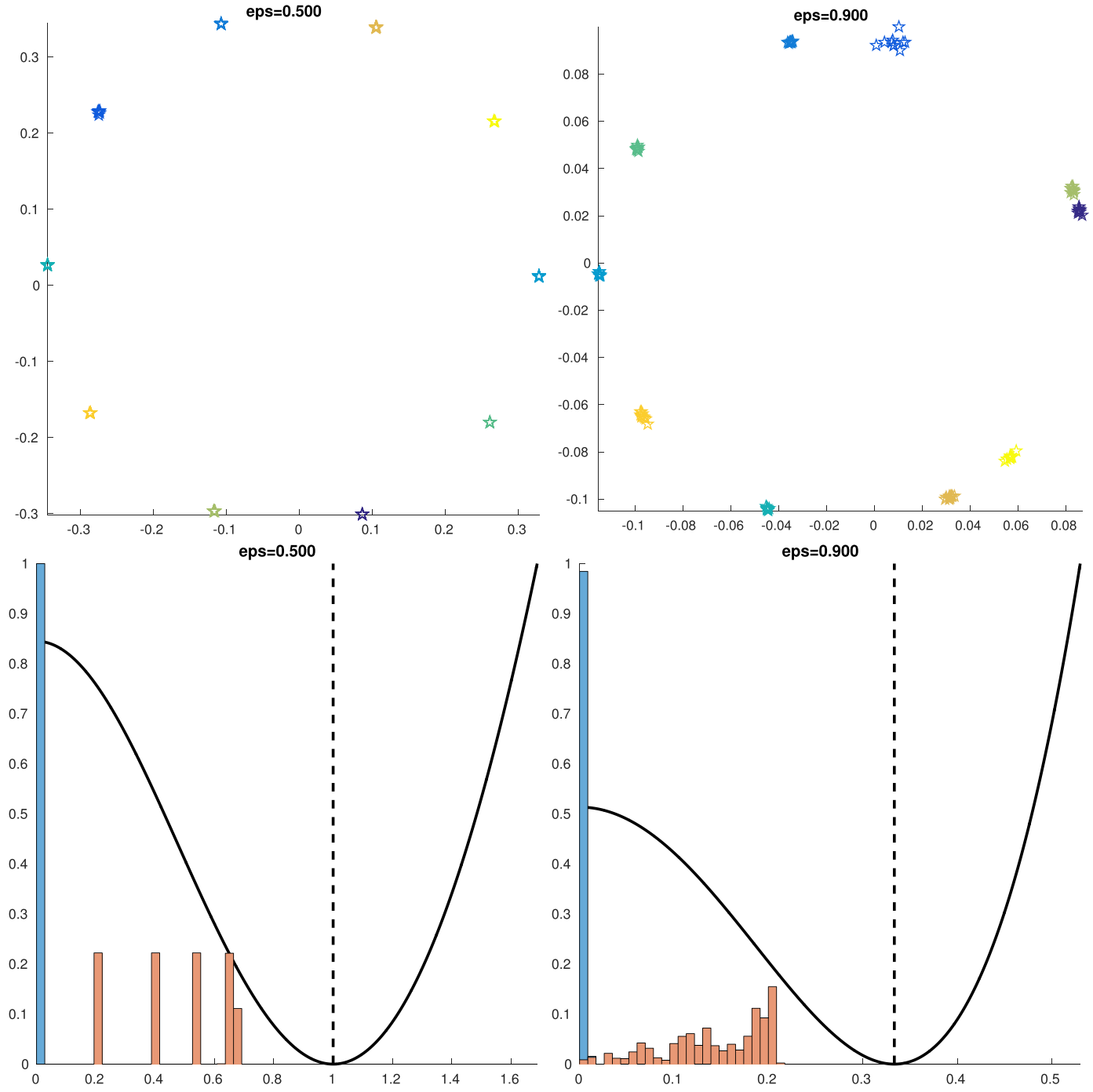

Fig. 5. Upper row: different embeddings of a subset of 100 samples of the USPS dataset for the hyperparameters $\epsilon = 0.5$ (left), and 0.9 (right). Lower row: the normalized KL divergences obtained by varying all the between-class distances and the normalized histograms of the realized within-class (blue) distances and the between-class distances (orange). There are ten classes in the dataset. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

The blue and orange histograms are the within-class and between-class realized distances, evaluated at the found optimal solution in two dimensions. The histograms are normalized in each image. The blue histogram shows that near zero within-class distances are achieved in the solution, meaning the assumption that the within-class distances are zero is valid in our analysis. The realized distances vary in each embedding. In the two subimages of Fig. 4, the distances are scattered around the ideal distance d_{ideal}, while in the two subimages of Fig. 5 the realized distances all fall short of the ideal distanc e d_{ideal}.

We can see that the cost function is nearly symmetric and locally convex in the region of the ideal distance when $\epsilon = 0.1$ and $\epsilon = 0.5$, translating into a speedy optimization. The subfigures in Fig. 4 show a lenient penalty for realized distances larger than the ideal distance. The penalty function makes it possible to embed points over the whole volume of a hypersphere. The two subimages in Fig. 5 show a harsh threshold for embedding distances too far from the optimal distance, and produce embeddings where the central area in the hypersphere remains unused. Note that when $\epsilon = 0.9$, the penalty for between-class distances becomes so large that certain classes are embedded almost on top of each other.

Table 1
Data sets used in this paper.

Data set	Samples	Variables	Classes
USPS	9298	256	10
MNIST-5000	5000	784	10
Phoneme	4509	256	5
Olivetti faces	400	4096	40
COIL-20	1440	16,384	20
COIL-100	7200	16,384	100
Iris	150	4	3
Wisconsin Breast cancer	683	9	2
Wine	178	13	3

Overall, the choice of $\epsilon = 1/C$ (0.1 in this experiment) produces well-separated embeddings which are nearly identical to the case where $\epsilon \to 0$, while still presenting a smooth, near-convex optimization surface with respect to the target distances near the optimal solution. In the experiments that follow, we set $\epsilon = 1/C$.

3.4. Outliers

Our paper focuses on the general principle behind SDA and its numerical solution, assuming that the data points are correctly labelled. However, outliers by which we mean wrong labels in the target space of a classification problem can be taken into account in the SDA formulation. More explicitly, the observed class labels are used as sure variables only in the assignment of the terms p_{ij} in the definition of the target probabilities. That is, whenever two points have the same or a different label, they are assigned some fixed distance. Since we assume no label noise, this assignment is fixed and done with probability one. To take into account label noise, this fact can be adjusted to quantify the probability of two observations actually be of the same class, given which labels are observed. A rather straightforward probabilistic model of four random variables of the discrete type (two per data point, the observed and the actual class labels) would allow to compute such probabilities. Given a measure of uncertainty, the p_{ij} could then be assigned in such a way that the distance in modulated accordingly. This outlier treatment can be considered in more detail mathematically, but it leads to a lengthy discussion, and would require new experiments. We feel that this is outside the scope of our paper.

4. Experimental results

The experimental evaluation is divided into two parts. First, our SDA method and comparison methods are applied to three different datasets in the three first subsections of the Section 4.1. In these case studies, the classification accuracies for a range of target dimensionality values are calculated, and two-dimensional projections are visualized. We also describe a regularization parameter search scheme for our SDA method in Section 4.1.1, and compare the runtime with different optimization algorithms in Section 4.1.4. In Section 4.2, a comparison of the two-dimensional projection qualities of state-of-the-art methods is carried out for several datasets. The datasets used in our experiments are summarized in Table 1.

We define the hyperparameters used in various methods here. Our proposed method SDA is initialized using standard PCA in all experiments. In SPCA, we chose the delta kernel [16] for the response space. In the kernel version of SPCA, we selected the delta kernel for response space and a Gaussian kernel for the input space, setting the width of the Gaussian to the median value of the squared interpoint distances. The gKDR method [15] was run in the partitioning mode (v) to reduce its memory requirements.

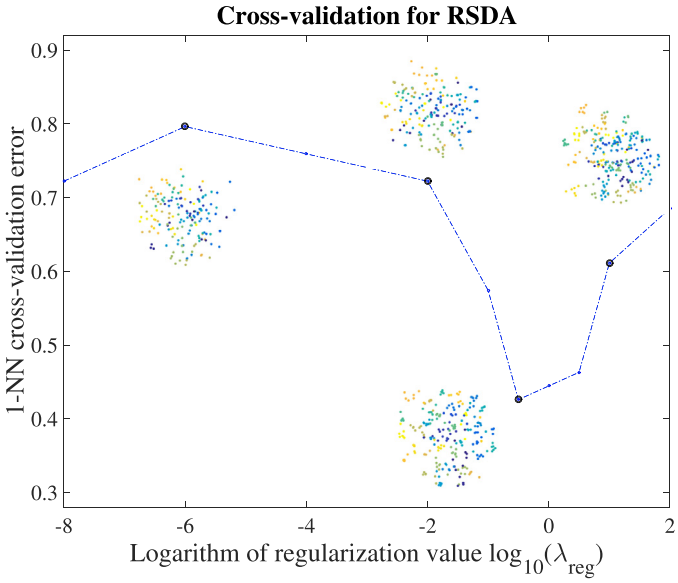

Fig. 6. Tikhonov regularization parameter search for a two-dimensional embedding. Some learned point embeddings are displayed.

The variables of each dataset were standardized by making them to have zero mean and unit variance.

4.1. Case studies with three high-dimensional datasets

Three image datasets were chosen and analyzed: Olivetti faces, USPS and COIL-20. All the data sets have multiple classes. The Olivetti face dataset [35] studied in Section 4.1.1 contains images of 40 persons, each photographed in ten pictures with both normal and unusual facial expressions. The input dimensionality is quite high, 4096. The USPS dataset [35] used in the Section 4.1.2 contains a large number of hand-written digits of ten classes in a smaller 256-dimensional space. The COIL-20 data set [41] in Section 4.1.3 consists of very high-dimensional (dimension 16,384) images of 20 rotating objects photographed at fixed angle intervals.

4.1.1. The Olivetti faces data set

Each of the 10 sample images on 40 persons in the Olivetti faces data set [35] is a 64-by-64 pixel image, leading to 4096 variables. In our tests, two thirds of the Olivetti face images were randomly selected to the training set and the remaining one third formed the test set. This random selection was repeated ten times for getting error bars.

In the Olivetti dataset, Tikhonov regularization was applied to guide the optimization process. The appropriate amount of regularization was searched by cross-validation. A random selection of 80% of the training data set was used for training and the remaining 20% were used for cross-validation. The best value was searched by trying six logarithmically spaced values of the regularization parameter λ from 10^2 to 10^{-8}. This basic search was then refined near the optimum. In total, ten regularization values were explored in the cross-validation search. Among these values, the optimal one that gives the smallest 1-NN classification error is denoted by λ^*. It was used in the tests that follow.

Fig. 6 shows one regularization term search procedure. The classification error is plotted against the logarithm $\log_{10}(\lambda)$ of the regularization parameter λ. For four values of this logarithm marked by dots in Fig. 6, the respective two-dimensional embeddings are also shown. We can observe that the search is magnified twice in the region $\lambda = 10^0$. Finally, the 1-NN classification error on the cross-validation data set was found to be the smallest when

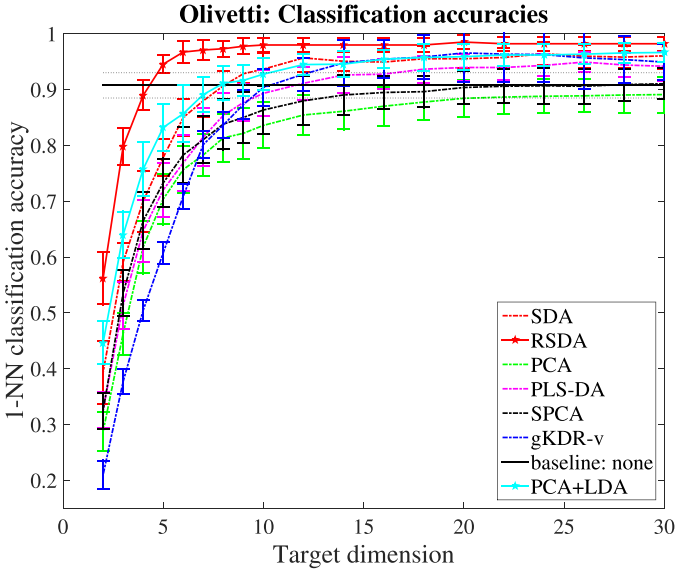

Fig. 7. Olivetti dataset. Classification accuracies after projection with different dimensionality reduction methods. The baseline is the classification accuracy in the original high-dimensional data set.

$\lambda = 10^{-0.5} \approx 0.32$. This search was continued until no further progress could be made with the tolerance 10^{-4}. The search procedure was fast, requiring approximately 3-4 seconds time per value explored. The tolerance for optimality in the main algorithm was set at 10^{-5}.

Fig. 7 shows the classification accuracy when learning the dimension reduction, using the λ search scheme described earlier. The error bars show the mean and standard deviation. The regularized algorithm has the best performance. Its mean accuracy is the highest among the compared methods and moreover its error bars are among the narrowest. The method stabilizes at 98.0% 1-NN classification accuracy at 10 dimensions, above the 90.1% accuracy obtained when using the whole input space for classification.

A two-dimensional embedding of the Olivetti faces using our regularized stochastic discriminant (RDSA) method is shown Fig. 8. The correct classifications and misclassifications have been highlighted in the figure. One can see for example that the face projected at (3, 6) is projected a bit off as it should in fact have been projected at (0.5, 4.5). We can see an overall division to dark images on the left hand side and light images appearing in the right hand side of this image. Similarly, bearded people can be found in the top part of Fig. 8. The three best performing two-dimensional projections in the Olivetti faces dataset have been compared in Fig. 9. All the figures show the 266 learning points and 134 test points for the same permutation. The same embedding is shown in both Figs. 8 and 9.

4.1.2. The USPS data set

The US Postal Service data set [35] contains 9298 handwritten images of digits. Each handwritten digit is represented by a 16×16 pixel grey-scale image, yielding 256-dimensional data vectors. In our tests, two thirds of the images were randomly selected to the training data set, and the remaining one third to the test data set. This random selection was repeated ten times for obtaining error bars.

The 1-NN classification accuracies are shown in Fig. 10. SDA provides the highest accuracies for small dimension reduction tasks. One can observe a saturation in the performance of the linear discriminant analysis (LDA), supervised PCA (SPCA), and our SDA methods. This saturation is related to the fact that the defined optimal simplex structure of the data is reached already at nine dimensions. PCA, the supervised partial least-squares method (PLS-DA), and the gKDR-v method approach or exceed the initial classification accuracy 96.3% in higher target dimensions.

Table 2
Different optimizers compared.

Acronym	Method
GD:	Gradient descent [38]
BB:	GD with Barzilai and Borwein step length [38]
CG:	Conjugate gradient (Hestenes-Stiefel update) [38]
PCG:	Preconditioned CG (LBFGS preconditioning)[38]
RCG:	Conjugate gradient (Polak-Ribiere update) [39]
LBFGS:	Limited-memory BFGS [38]
SD:	Spectral direction (Modified Newton's method) [38]

The three best performing two-dimensional linear embeddings of the data points are compared in Fig. 11. We can see that the LDA and PLS-DA methods provide embeddings that resemble multidimensional simplexes projected onto a subspace with too many classes crowding near the origin and overlapping each other. Such projections are not ideal in the presence of multiple classes. On the contrary, SDA tends to fill a two-dimensional circle, leading to better class discrimination and higher classification accuracy.

4.1.3. COIL-20 Object Images

The Columbia Object Image Library contains rotated images of 20 objects, photographed at 5° intervals [41]. The images are 128-by-128 pixel grey-scale images. These images include such objects as rubber ducks, toy cars, and jars. In total, there are 1440 sample images which are 16384-dimensional when represented as vectors.

The test set and the cross-validation set were generated differently for the COIL-20 images when compared with Olivetti and USPS images for exploiting the structure in the dataset. The test data set was generated with 24-fold partitioning, and the cross-validation data set by selecting five elements from the training set. This selection of test points made it easier to analyze the scatter plots, resulting in less clutter and more expected visual structure.

Fig. 12 shows the classification accuracies for the previous techniques calculated for the target dimensions two, three, four, and five. The mean and error bars were calculated by leaving three elements out of each class at each round, and repeating the runs 24 times, thus going through the whole data. The tolerance for the SDA algorithms was set to 10^{-5}. SDA and RSDA can on average identify over 90% of the classes with two target variables only. When the dimensionality of the mapped data is five, most algorithms perform similarly. The three best performing embeddings of the COIL-20 dataset are shown in Fig. 13.

4.1.4. Computational complexity and running time comparison

The computational complexity of stochastic discriminant analysis (SDA) is largely determined by the number of times the gradient in Eq. (23) must be evaluated. The matrix evaluation has the complexity $O(LK^2 + LNK)$, where N is the dimensionality of the input space, L is dimensionality of target space, and K is the number of samples. As such, optimizers that require as few function evaluations as possible would be efficient choices.

The processing times of the algorithms in Table 2 are compared on the three tested datasets for two-dimensional SDA embeddings. Figs 14–16 show the results for the USPS, Olivetti, and COIL20 data, respectively. The fastest algorithm differs depending on the characteristics of the dataset. The spectral direction method converges faster and at a lower level than the other algorithms in the USPS dataset. Convergence is reached in about two minutes. The number of variables is still small enough so that the partial Hessian information can be utilized cost efficiently. The Olivetti and COIL-20 datasets contain a much larger number of variables. The Hessian

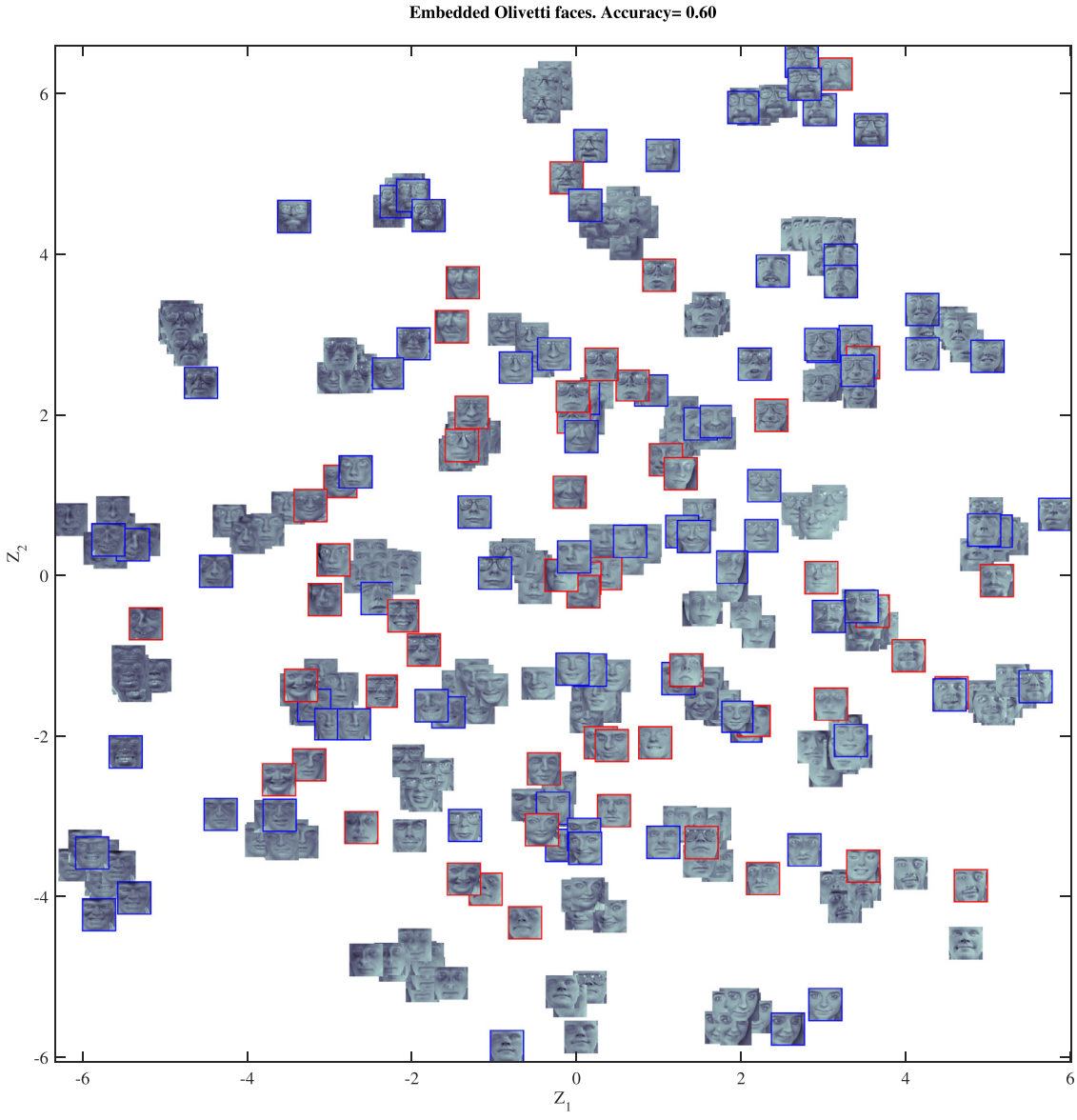

Fig. 8. A representative linear embedding of the Olivetti faces dataset using regularized SDA. Colored borders denote projected test points. Red borders denote a misclassification, while blue borders denote a correct classification.

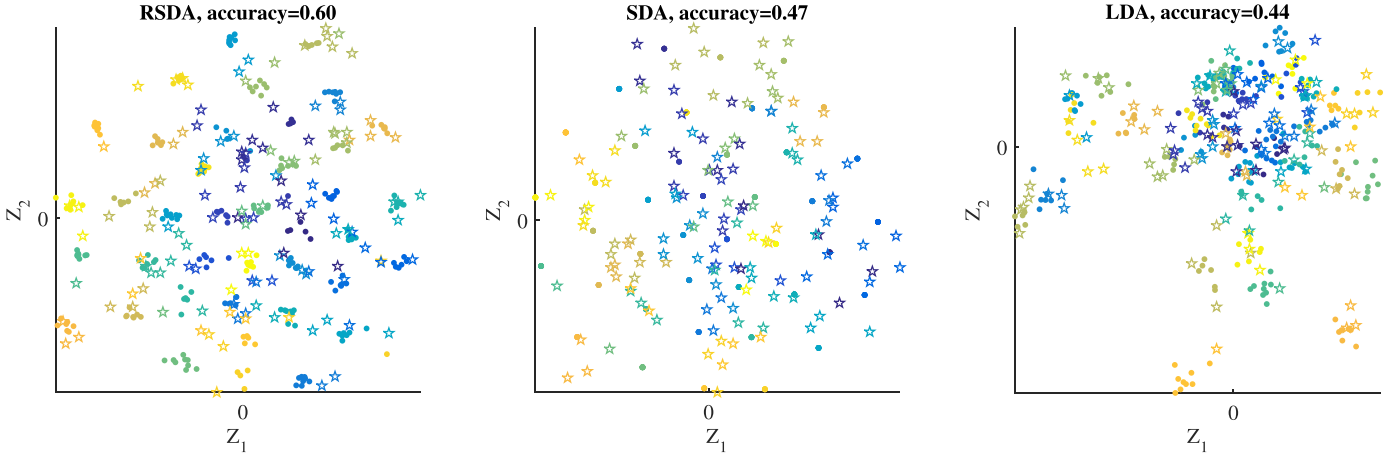

Fig. 9. Three linear embeddings of the Olivetti faces dataset. Dots denote projected learning points. Stars denote projected test points. The 1-NN classification accuracy provided by each embedding is shown above each subfigure. Samples belonging to different classes are depicted using different colors.

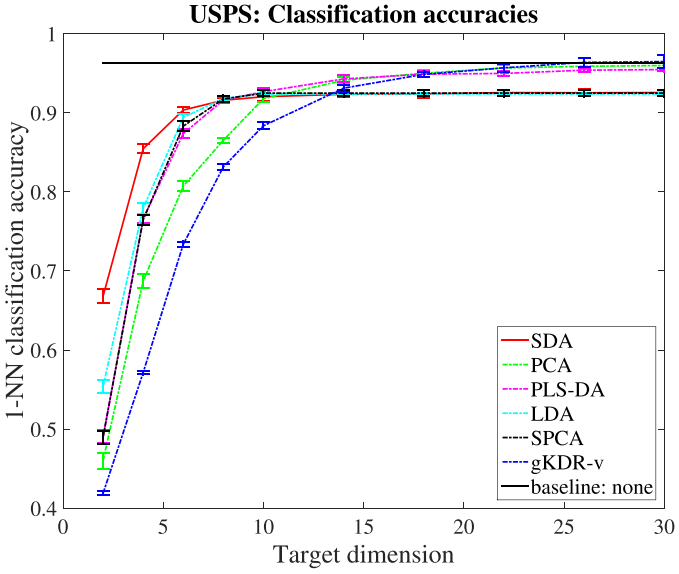

Fig. 10. Classification accuracies for different dimension reduction methods for the USPS data set. The baseline is the classification accuracy for the original high-dimensional data set.

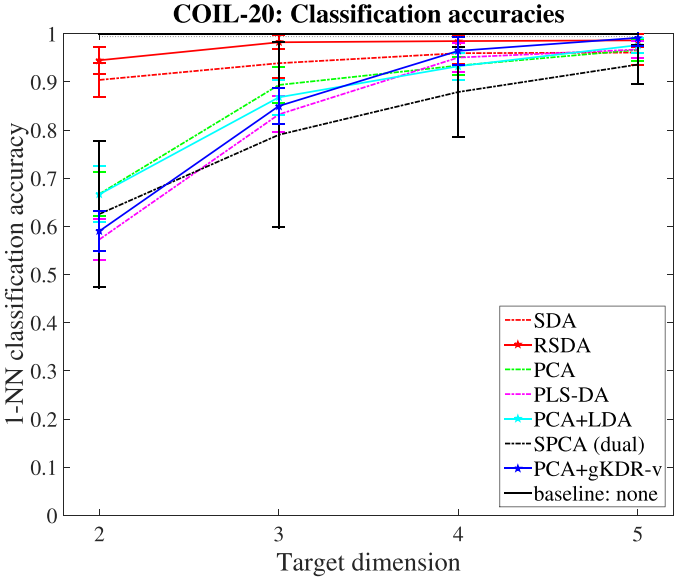

Fig. 12. COIL-20 dataset. Classification accuracies for different dimension reduction methods. The baseline is the classification accuracy with no dimension reduction.

is a *LN*-by-*LN* matrix, resulting in costly operations involving the Hessian.

In the COIL-20 data set, the partial Hessian is re-evaluated only at every 20 iterations for making the computations faster. We can see that the LBFGS algorithm and different forms of the nonlinear conjugate gradient method are faster choices when performing dimensionality reduction for very high-dimensional data sets. The spectral gradient method works better for the USPS data set having a smaller input dimension but larger amount of data.

4.2. Comparison over multiple data sets

In this subsection we compare the proposed method with state-of-the-art linear embeddings especially in two-dimensional information visualization tasks. The algorithms were run over three standard UCI datasets [42], three large datasets having more than 4000 data points, and three very high-dimensional datasets which were more than 4000-dimensional. In general, the algorithms were run for different selections of training and test points 10 times to obtain the confidence intervals. The COIL-20 and COIL-100 datasets were evaluated in the principle of leave-three-out, as discussed in Section 4.1.3. As a preprocessing step, the original color images in COIL-100 were transformed to gray-scale images and all datasets were normalized [41]. In the tables that follow, a distinction is made between different dimension reduction types: *none*, *supervised*, and *unsupervised*. PCA is in our comparison the only unsupervised method. These different types are separated by horizontal lines.

4.2.1. UCI datasets

In the Iris dataset, three species of Iris flowers are identified by quantitative measurements of the flowers. In the Wine dataset, wine species are identified based on chemical test results. In the Wisconsin Breast Cancer dataset, tumors are classified as benign or malign ones based on physical measurements. The datasets are all standard small datasets with small dimensional data vectors. The results of low-dimensional projections are shown in Table 3. For these three UCI datasets, all the compared methods perform rather similarly. The tests were repeated 20 times for obtaining the standard deviations of the errors.

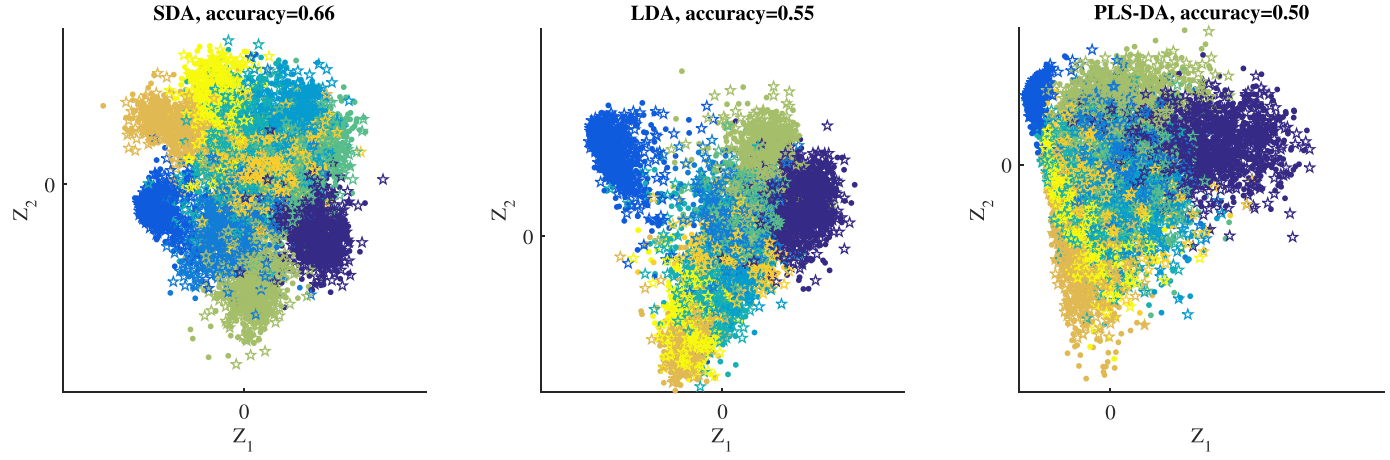

Fig. 11. Three linear embeddings of the USPS dataset. Dots denote projected learning points and stars denote projected test points. The 1-NN classification accuracy resulting from this embedding is shown above each subfigure. Samples belonging to different classes are depicted using different colors.

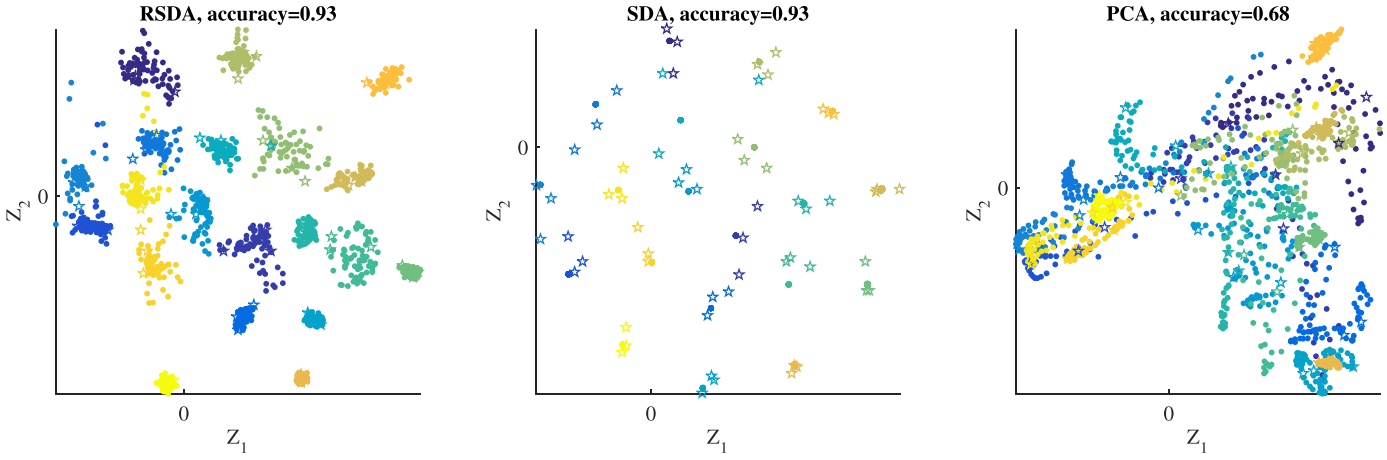

Fig. 13. Three linear embeddings of the COIL-20 dataset. Dots denote projected learning points and stars denote projected test points. The 1-NN classification accuracy resulting from this embedding is added above each subimage. Colors denote different classes.

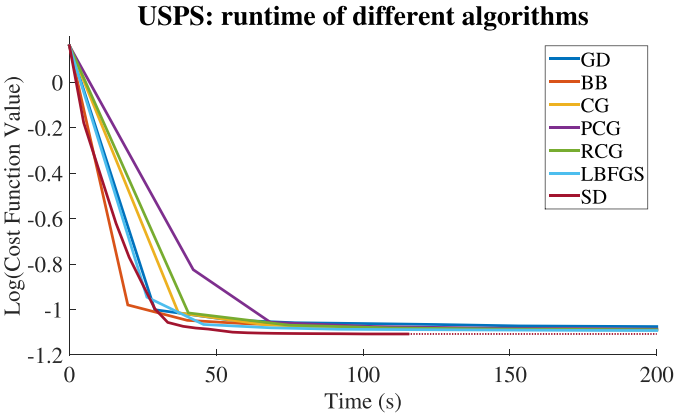

Fig. 14. Running times of different optimization algorithms for the USPS data.

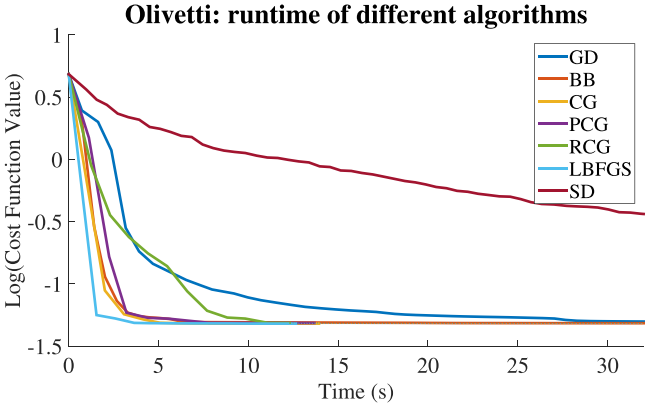

Fig. 15. Running times of different optimization algorithms for the USPS data.

4.2.2. Large high-dimensional datasets

Three large datasets were compared. Two datasets were for handwritten digit recognition tasks (MNIST, USPS), and one was a dataset for phoneme recognition. The phoneme dataset contains three vowel pronunciations (aa, ao, iy), and two consonants (dcl, sh), where the vowels are difficult to separate [43,44]. In SDA, the optimality tolerances for the large datasets were set to 10^{-5}, and each test was repeated ten times. The results are shown in Table 4. The SDA method performs favorably in all these tests.

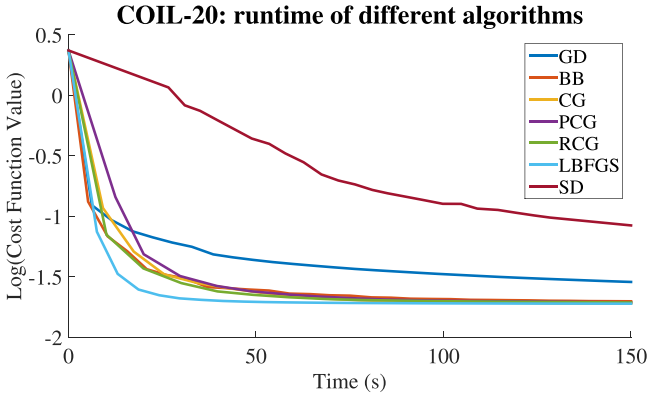

Fig. 16. Running times of different optimization algorithms for the COIL-20 data.

Table 3
The generalization accuracy (mean ± standard deviation) of the nearest neighbor classification method on test set. The datasets were reduced to two-dimensional aside from *None*, in which no dimension reduction was done.

Method	Iris	Wine	W. Breast Cancer
None	0.941 ± 0.026	0.949 ± 0.026	0.957 ± 0.014
SDA	0.948 ± 0.030	**0.983 ± 0.017**	0.957 ± 0.008
RSDA	0.957 ± 0.023	0.982 ± 0.016	0.955 ± 0.011
LDA	**0.962 ± 0.025**	0.981 ± 0.016	**0.961 ± 0.009**
PLS-DA	0.879 ± 0.040	0.974 ± 0.021	0.957 ± 0.008
gKDR	0.960 ± 0.021	0.959 ± 0.030	0.956 ± 0.013
SPCA	0.892 ± 0.026	0.974 ± 0.018	**0.961 ± 0.011**
KSPCA	0.893 ± 0.047	0.971 ± 0.019	0.893 ± 0.087
PCA	0.860 ± 0.034	0.938 ± 0.024	**0.961 ± 0.011**

Table 4
The 1-NN generalization accuracy (mean ± std) on test set for three large high-dimensional datasets. The datasets were reduced to two-dimensional except for *None*.

Method	Phoneme	MNIST5k	USPS
None	0.889 ± 0.010	0.936 ± 0.002	0.962 ± 0.002
SDA	0.875 ± 0.009	**0.557 ± 0.006**	0.668 ± 0.009
RSDA	**0.877 ± 0.009**	0.550 ± 0.005	**0.669 ± 0.007**
LDA	0.664 ± 0.010	0.461 ± 0.011[3]	0.554 ± 0.008
PLS-DA	0.779 ± 0.014	0.301 ± 0.006	0.490 ± 0.008
gKDR-v	0.809 ± 0.015	0.323 ± 0.024	0.453 ± 0.009
SPCA	0.780 ± 0.008	0.401 ± 0.008	0.490 ± 0.008
KSPCA	0.781 ± 0.009	0.401 ± 0.009	0.354 ± 0.010
PCA	0.765 ± 0.007	0.383 ± 0.006	0.460 ± 0.010

Table 5
The 1-NN generalization accuracy (mean ± std) on test sets for three very high-dimensional datasets. The datasets were reduced to two-dimensional except for *None*.

Method	Olivetti faces	COIL-20	COIL-100
None	0.908 ± 0.023	0.999 ± 0.005	0.988 ± 0.006
SDA	0.393 ± 0.056	0.904 ± 0.035	0.277 ± 0.024
RSDA	$\mathbf{0.562 \pm 0.047}$	$\mathbf{0.944 \pm 0.026}$	$\mathbf{0.605 \pm 0.026}$
LDA	0.446 ± 0.039	0.656 ± 0.079	0.300 ± 0.054
PLS-DA	0.310 ± 0.042	0.573 ± 0.042	0.481 ± 0.049
gKDR-v	0.210 ± 0.046	0.565 ± 0.057	0.142 ± 0.038
SPCA	0.325 ± 0.033	0.623 ± 0.152	0.437 ± 0.061
KSPCA	0.322 ± 0.037	0.567 ± 0.191	0.397 ± 0.055
PCA	0.289 ± 0.029	0.667 ± 0.046	$0.288 + 0.036$

4.2.3. Very high-dimensional datasets

A face recognition dataset (Olivetti faces) and two object recognition datasets (COIL-20 and COIL-100) were compared. The regularized version RSDA of the SDA method was also computed. The 1-NN out-of-sample classification accuracies are shown in Table 5. Our regularized algorithm RSDA has the highest accuracy among the tested algorithms on all datasets. The tests were repeated 10 times to obtain the error ranges. The tolerance for optimality was set at 10^{-5} in Olivetti and COIL-20 and at 10^{-4} in COIL-100 data sets. The tolerances for the regularization search were set one magnitude higher (10^{-3}) than in the final algorithm (10^{-4}). Optimization with RSDA, including the regularization parameter λ search procedure, was on an average faster than using no regularization ($\lambda = 0$) in the COIL-100 data set, with the median time of 88 min versus 215 min.

5. Conclusions

We have introduced in this paper a linear supervised dimension reduction method for classification settings, which we call Stochastic Discriminant Analysis (SDA). The SDA method matches similarities between points in the projection space with those in a response space. The similarities are represented by transforming distances between points to joint probabilities using a transformation resembling Student's t-distribution. The matching is done by minimizing the Kullback–Leibler divergence between the two probability distributions.

The proposed stochastic discriminant analysis (SDA) method is useful especially when two-dimensional projections of datasets having several or many classes are needed. In such situations, ordinary discriminant analysis algorithms perform poorly. The generalization ability of the SDA method increases until the optimal structure is found in $C-1$ dimensions, where C is the number of classes. It should be noted that due to the definition of the optimization criterion, neither SDA nor closed-form solutions of linear discriminant analysis (LDA) can obtain improved results once the target dimensionality surpasses $C-1$ dimensions, since both the methods try to reconstruct an intrinsically $C-1$-dimensional simplex. For combatting overlearning in very high-dimensional datasets, Tikhonov regularization was used. It improves the generalization ability of the SDA method, and increases classification accuracies for very high-dimensional datasets.

In the extensive experimental part of this paper, we compare the performance of our SDA method against several state-of-the-art methods in supervised linear dimension reduction. The SDA method performs in most cases better than the compared linear projection methods when low two or three-dimensional projections are used. We have made experiments with various types of data sets having low, medium, or high dimensions and quite different numbers of samples. Our experiments with both sparse and dense data sets confirm the good performance of the SDA method and its regularized version.

References

[1] I. Jolliffe, Principal Component Analysis, 2nd Ed., Springer Series in Statistics, 2004.
[2] C. Bishop, Pattern Recognition and Machine Learning, Springer Series on Information Science and Statistics, 2006.
[3] Article "Principal component analysis" in Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/wiki/principal_component_analysis.
[4] Article "Linear disriminant analysis" in Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/wiki/linear_discriminant_analysis.
[5] J. Yang, J.-Y. Yang, Why can LDA be performed in PCA transformed space? Pattern Recognit. 36 (2) (2003) 563–566.
[6] P. Belhumeur, J. Hespanha, D. Kriegman, Eigenfaces versus fischerfaces: recognition using class specific linear projection, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 711–720.
[7] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Graph embedding: a general framework for dimensionality reduction, in: Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05), 2005, pp. 830–837.
[8] H.-T. Chen, H.-W. Chang, T.-L. Liu, Local discriminant embedding and its variants, in: Proceedings of the 2005 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'05), 2005, pp. 846–853.
[9] M. Sugiyama, Dimensionality reduction of multimodal labeled data by local Fisher discriminant analysis, J. Mach. Learn. Res. 8 (2007) 1027–1061.
[10] W. Bian, D. Tao, Max–min distance analysis by using sequential SDP relaxation for dimension reduction, IEEE Trans. Pattern Anal. Mach. Intell. 33 (5) (2011) 1037–1050.
[11] H. Wang, S. Yan, D. Xu, X. Tang, T. Huang, Trace ratio vs. ratio trace for dimensionality reduction, in: Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07), 2005, pp. 830–837.
[12] M. Prez-Enciso, M. Tenenhaus, Prediction of clinical outcome with microarray data: a partial least squares discriminant analysis PLS-DA approach, Human Genet. 112 (5–6) (2003) 581–592.
[13] K. Fukumizu, F. Bach, M. Jordan, Kernel dimension reduction in regression, Ann. Stat. 37 (2009) 1871–1905.
[14] K. Adragni, D. Cook, Sufficient dimension reduction and prediction in regression, Philos. Trans. R. Soc. Lond. A Math. Phys. Eng. Sci. 367 (1906) (2009) 4385–4405.
[15] K. Fukumizu, C. Leng, Gradient-based kernel dimension reduction for regression, J. Am. Stat. Assoc. 109 (505) (2014) 359–370.
[16] E. Barshan, A. Ghodsi, Z. Azimifar, M. Zolghadri Jahromi, Supervised principal component analysis: visualization, classification and regression on subspaces and submanifolds, Pattern Recognit. 44 (7) (2011) 1357–1371.
[17] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, The MIT press, 2016.
[18] P. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, Neural Comput. 15 (2003) 1373–1396.
[19] X. He, P. Niyogi, Locally preserving projections, in: Proceedings of the Advances in Neural Information Processing Systems 16, NIPS 2003, 2004, pp. 153–160.
[20] B. Schölkopf, A. Smola, K. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput. 10 (17) (1998) 1299–1319.
[21] S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326.
[22] Article "Nonlinear dimensionality reduction" in Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/wiki/nonlinear_dimensionality_reduction.
[23] D. Barber, Bayesian Reasoning and Machine Learning, Cambridge University Press, 2012.
[24] K. Murphy, Machine Learning: a Probabilistic Perspective, The MIT Press, 2012.
[25] J. Venna, J. Peltonen, K. Nybo, H. Aidos, S. Kaski, Information retrieval perspective to nonlinear dimensionality reduction for data visualization, J. Mach. Learn. Res. 11 (2010) 451–490.
[26] G. Hinton, S. Roweis, Stochastic neighbor embedding, in: S. Becker, S. Thrun, K. Obermayer (Eds.), Proceedings of the Advances in Neural Information Processing Systems 15 (NIPS 2002), Vancouver, Canada, December 2002, pp. 833–840.
[27] S. Kaski, J. Peltonen, Informative discriminant analysis, in: T. Fawcett, N. Mishna (Eds.), Proceedings of the 20th International Conference on Machine Learning (ICML 2003), AAAI Press, Menlo Park, California, USA, 2003, pp. 329–336.
[28] L. Van der Maaten, G. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (2008) 2579–2605.
[29] Z. Yang, I. King, L. Xu, E. Oja, Heavy-tailed symmetric stochastic neighbor embedding, in: Proceedings of the Advances in Neural Information Processing Systems (NIPS), Vol. 22, 2009, pp. 2169–2177.
[30] S. Wu, M. Sun, J. Yang, Stochastic neighbor projection on manifold for feature extraction, Neurocomputing 74 (17) (2011) 2780–2789.
[31] J. Zheng, H. Qiu, Y. Jiang, W. Wang, Discriminative stochastic neighbor embedding analysis method, Comput. Aided Des. Comput. Graph. 24 (11) (2013) 1477–1484.
[32] J. Zheng, H. Qiu, X. Xu, W. Wang, Q. Huang, Fast discriminative stochastic neighbor embedding analysis, Comput. Math. Methods Med. 2013 (2013). Article ID 106867
[33] Z. Zhu, T. Similä, F. Corona, Supervised distance preserving projections, Neural Process. Lett. 38 (2013) 445–463.

[34] R. Salakhutdinov, G. Hinton, Learning a nonlinear embedding by preserving class neighbourhood structure, in: Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS 2007), San Juan, Puerto Rico, March 2007, pp. 409–416.
[35] S. Roweis, Data for Matlab hackers: handwritten digits, https://cs.nyu.edu/~roweis/data.html.
[36] Article "Student's t-distribution" in Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/wiki/student's_t-distribution.
[37] D. Bertsekas, Nonlinear Programming, Athena Scientific, 1999.
[38] M. Schmidt, Minfunc (A Matlab function for unconstrainedoptimization of differentiable real-valued multivariate functions using line-search methods), 2005, https://www.cs.ubc.ca/~schmidtm/Software/minFunc.html.
[39] C. Rasmussen, Matlab function: nonlinear conjugate gradient minimizer, 2001, http://learning.eng.cam.ac.uk/carl/code/minimize/.
[40] D. Luenberger, Y. Ye, Linear and Nonlinear Programming, Springer-Verlag, 2008.
[41] S. Nene, S. Nayar, H. Murase, Columbia object image library (COIL-20 and COIL-100), Technical Report CUCS-005-96, Dept. of Computer Science, Columbia University, USA, 1996.
[42] K. Bache, M. Lichman, UCI machine learning repository, 2013, https://archive.ics.uci.edu/ml/index.php.
[43] T. Hastie, R. Tibshirani, J. Friedman, Data sets on the home page of the book "The Elements of Statistical Learning", https://web.stanford.edu/~hastie/ElemStatLearn/data.html.
[44] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, 2nd Ed., Springer-Verlag, 2009.

Mika Juuti completed in 2015 his M.Sc. (Tech.) degree with distinction at Aalto University Department of Computer Science, Espoo, Finland. He investigated information visualization topics under the supervision of Dr. Francesco Corona in the Environmental and Industrial Machine Learning group led by Prof. Juha Karhunen. He is currently a Ph.D. candidate at Aalto University Dept. of Computer Science, working there on machine learning and security problems in the Secure Systems group led by Prof. N. Asokan.

Francesco Corona is a professor of Machine Learning of the Department of Computing at the Federal University of Ceará (Brazil) and a docent of Information and Computer Science at the Department of Computer Science of the Aalto University (Finland). Francesco joined the Aalto University/Helsinki University of Technology as a post-doctoral researcher in January 2007, in 2009 he was appointed the docentship in Industrial applications of Machine Learning and in 2014 the docentship in Information and Computer Science. In 2016 he was appointed the professorship in Machine Learning and Logic at the Department of Computer Science of the Federal University of Ceará Francesco's interest concentrate on statistical machine learning, probabilistic modelling, and information visualisation: Emphasis is on full-scale applications to supervision, control, and optimisation of industrial and environmental systems. He is member of IFAC, the International Federation of Automatic Control (since 2007). In IFAC, he is member of the Technical Commitee 8.4 on Biosystems and Bioprocesses(since 2011) where he serves as vice-chair (for industry, since 2017) and he is member of the IFAC Industry Committee (since 2018).

Juha Karhunen received the D.Sc. (Tech.) degree from Helsinki University of Technology in 1984. In 1999, he became there an associate professor, with specialization area neural networks and signal processing. After several structural changes, he is now full professor in the Aalto University Department of Computer Science in Espoo, Finland. His current research interests include machine learning and neural networks, especially extreme learning machines, deep learning, and their different applications. As a supervising professor, he has been formally responsible for several different research groups. Prof. Karhunen has published more than 100 conference and journal papers, and given invited talks in international conferences. He is a co-author of the textbook and monograph A. Hyvärinen, J. Karhunen, and E. Oja, "Independent Component Analysis", Wiley 2000, which became a standard reference on independent component analysis and blind source separation. He has been a member in numerous conference program committees and in the editorial board of the journals Neurocomputing and Neural Processing Letters.