Accepted Manuscript

Fast semi-supervised discriminant analysis for binary classification of large data sets

Joris Tavernier, Jaak Simm, Karl Meerbergen, Joerg Kurt Wegner, Hugo Ceulemans, Yves Moreau

PII: S0031-3203(19)30079-2
DOI: <https://doi.org/10.1016/j.patcog.2019.02.015>
Reference: PR 6821

To appear in: *Pattern Recognition*

Received date: 10 July 2017
Revised date: 16 January 2019
Accepted date: 19 February 2019

Please cite this article as: Joris Tavernier, Jaak Simm, Karl Meerbergen, Joerg Kurt Wegner, Hugo Ceulemans, Yves Moreau, Fast semi-supervised discriminant analysis for binary classification of large data sets, *Pattern Recognition* (2019), doi: <https://doi.org/10.1016/j.patcog.2019.02.015>

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

Highlights

- Computationally more efficient algorithm for semi-supervised discriminant analysis.
- Efficient handling of regularization parameter.
- Industry-scale data sets from chemogenomics analysed in a few seconds.

ACCEPTED MANUSCRIPT

Fast semi-supervised discriminant analysis for binary classification of large data sets

Joris Tavernier^{a,b,*}, Jaak Simm^b, Karl Meerbergen^a, Joerg Kurt Wegner^c, Hugo Ceulemans^c, Yves Moreau^b

^aDepartment of Computer Science, KU Leuven, Celestijnenlaan 200A, B-3001 Heverlee - Belgium

^bESAT-STADIUS, KU Leuven, Kasteelpark Arenberg 10, bus 2446, B-3001 Heverlee - Belgium

^cJanssen Pharmaceutica, Turnhoutseweg 30, B-2340 Beerse - Belgium

Abstract

High-dimensional data requires scalable algorithms. We propose and analyze four scalable and related algorithms for semi-supervised discriminant analysis (SDA). These methods are based on Krylov subspace methods and therefore exploit data sparsity and the shift-invariance of Krylov subspaces. In addition, centralization was derived for the semi-supervised setting. The proposed methods are evaluated on an industry-scale data set from a pharmaceutical company to predict compound activity on target proteins. The results show that our methods only require a few seconds, significantly improving computation time on the state of the art.

Keywords: Semi-supervised learning, Semi-supervised discriminant analysis, large-scale

2010 MSC: 65F15, 65F50, 68T10

1. Introduction

In fields such as data mining, information retrieval or pattern recognition, dimensionality reduction of high-dimensional data is often used for classification. This assumes that data can be accurately represented in a lower dimensional manifold. Linear discriminant analysis (LDA) [1, 2, 3] for example is a supervised method using dimensionality reduction to classify data while principal component analysis (PCA) [4, 1] is a widely used unsupervised feature extraction method.

Supervised methods use known labels to classify data, but sometimes only a few sample labels are available and assigning new labels can be extremely expensive. In this setting, classification methods can benefit from models that incorporate both labeled and unlabeled data in a semi-supervised setting [5]. The features of the unlabeled data points are available and semi-supervised learning is built on the assumption that taking advantage of the inherent structure of unlabeled and labeled data aids in a classification task.

*Corresponding author

Email addresses: Joris.Tavernier@cs.kuleuven.be (Joris Tavernier), jaak.simm@esat.kuleuven.be (Jaak Simm), Karl.Meerbergen@cs.kuleuven.be (Karl Meerbergen), jwegner@its.jnj.com (Joerg Kurt Wegner), hceulema@its.jnj.com (Hugo Ceulemans), moreau@esat.kuleuven.be (Yves Moreau)

The difficulty and cost of labeling samples stimulated surging interest in semi-supervised learning. This interest has led to several different methods of incorporating information of the unlabeled samples to aid the supervised classification. These samples are available before training or come in incrementally. Here, we will only consider transductive learning; for incremental approaches see for example [6, 7, 8].

Generally, semi-supervised dimensionality reduction can be established by feature selection or feature extraction. Feature selection based dimensionality reduction selects the most relevant features from high dimensional data, see [9] for a survey on semi-supervised feature selection methods. Feature extraction methods on the other hand, project the data to a lower dimensional subspace. More specifically, several semi-supervised feature extraction dimensionality reduction techniques have been developed: Semi-supervised Orthogonal Discriminant Analysis via label propagation (SODA) [10], SEmi-supervised Local Fisher discriminant analysis (SELF) [11], Trace Ratio based Flexible Semi-supervised Discriminant Analysis, [12, 13], Flexible Manifold Embedding (FME)[14], Semi-supervised Linear Discriminant Analysis [15], Semi-supervised double sparse graphs based discriminant analysis [16], Semi-supervised Local Fisher Discriminant Analysis using pseudo labels [17] and Semi-supervised local multi-manifold Isomap by linear embedding [18]. These semi-supervised methods attempt to incorporate geometric information of the unlabeled data or estimate the labels to incorporate information of the unlabeled data in a known supervised method.

One example where these semi-supervised dimensionality reduction methods could prove beneficial due to the high labeling costs is target prediction in chemogenomics. Chemical compounds are modeled by combinations of molecules. Depending on the specific target protein, the compound can be active or inactive. Due to existing drugs and research, only a few activity pairs are known and labeling these activity pairs in vitro is expensive. The structure of the chemical compounds is available and a possible feature set for these compounds are the binary substructure fingerprints, which are the substructures occurring in a compound [19]. Using the information of the unlabeled data with the few labels will likely improve the classification results and it is known that predicting which molecule affects a protein aids in the development of new drugs [20, 21]. The resulting data matrix describing these compounds is a high-dimensional sparse binary matrix.

This kind of high-dimensional data requires scalable algorithms and high performance computing. Semi-supervised algorithms based on QR-decomposition or singular value decomposition have cubic computational complexity and are unfeasible. This eliminates several state-of-the-art semi-supervised learning methods mentioned before due to computational workload. LDA has been studied for large-scale data and scalable methods are available in the literature [22, 23]. More specifically, Spectral regression discriminant analysis (SRDA) [22] solves LDA efficiently and is a special case of spectral regression (SR) [24] consisting of two phases: spectral analysis (SA) of an embedded graph and regression. Using manifold regularization [25], LDA was extended to the semi-supervised setting leading to semi-supervised discriminant analysis (SDA) [26].

Here, we present data centralization for SDA and show how to efficiently implement the centralization. Then, we use the shift-invariance property of Krylov subspace methods to find solutions for different regularization parameters in a computationally feasible manner. We suggest four scalable algorithms for SDA and compare predictive performance and execution time. These algorithms are applied on real-life large-scale data from the database of Janssen Pharmaceutica. Our main contributions are

- an efficient implementation of SDA with centralization using the shift invariance property of Krylov subspace methods for multiple regularization parameters;
- applying spectral regression to SDA and proposing classification of unlabeled data using spectral analysis in the semi-supervised setting;
- applying semi-supervised learning in the form of SDA to target prediction in chemogenomics and comparing different algorithms for solving SDA.

Section 2 provides the necessary background and preliminaries used in our methods. First, Section 2.1 presents a brief review of LDA and its properties while Section 2.3 builds on the properties of LDA to introduce SDA. In Sections 2.5 and 2.6 the necessary tools are explained to apply SDA on large-scale data sets and then our scalable algorithms are defined in Section 3. Finally SDA is compared to related dimensionality reduction techniques and applied on two data sets from chemogenomics in Section 4.

2. Preliminaries

In this section, we review LDA and present an existing scalable algorithm for LDA. Next, we detail a semi-supervised extension for LDA, named SDA. We then detail centralization for LDA and define centralization for SDA. Finally, we present the scalable algorithms to solve the SDA problem with centralization. Table 1 describes used notation.

2.1. Linear discriminant analysis

LDA [1, 2, 3, 22, 26] is a supervised dimensionality reducing method for finding the directions that maximally separate the different classes while minimizing the spread within one class. Suppose we have C classes and N samples $\mathbf{x}_i$ of dimension D . The data matrix $X \in \mathbb{R}^{N \times D}$ is defined as

$$X = [\mathbf{x}_1, \dots, \mathbf{x}_i, \dots, \mathbf{x}_N]^T \quad (1)$$

where each row corresponds to a sample. We define S_B as the between-class-scatter matrix

$$S_B = \sum_{c=1}^C N_c (\boldsymbol{\mu}_c - \boldsymbol{\mu})^T (\boldsymbol{\mu}_c - \boldsymbol{\mu}) \quad (2)$$

Symbol	Description
D	the number of features
N	the total number of data samples
l	the number of labeled data samples
C	the number of classes
N_c	the number of data samples in class c
$\mathbf{x}_i$	i -th data sample
X	the data matrix $X \in \mathbb{R}^{N \times D}$
S_B	the between-class scatter matrix
S_W	the within-class scatter matrix
S_T	the total scatter matrix
L	the laplacian matrix
$\mathbf{z}$	eigenvector of the spectral analysis eigenvalue problem, $\mathbf{z} \in \mathbb{R}^{N \times 1}$
$\mathbf{w}$	eigenvector of the spectral analysis eigenvalue problem projected onto the range of X , $\mathbf{w} \in \mathbb{R}^{D \times 1}$
SR	spectral regression
SA	spectral analysis
CSR	centralized spectral regression
SDA	semi-supervised discriminant analysis

Table 1: Notation used in the paper.

with N_c the number of data points in class c , $\boldsymbol{\mu}$ the mean sample and $\boldsymbol{\mu}_c$ the mean sample of class c . The within-class-scatter matrix S_W is defined by

$$S_W = \sum_{c=1}^C \sum_{n=1}^{N_c} (\mathbf{x}_n - \boldsymbol{\mu}_c)^T (\mathbf{x}_n - \boldsymbol{\mu}_c). \quad (3)$$

The LDA objective function is then

$$\max_{\mathbf{w} \in \mathbb{R}^N} \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_W \mathbf{w}}. \quad (4)$$

Using the total scatter matrix S_T

$$S_T = \sum_{n=1}^N (\mathbf{x}_n - \boldsymbol{\mu})^T (\mathbf{x}_n - \boldsymbol{\mu}) \quad (5)$$

$$= S_B + S_W \quad (6)$$

the objective function can then equivalently be expressed as

$$\max_{\mathbf{w} \in \mathbb{R}^N} \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_T \mathbf{w}}. \quad (7)$$

Since S_B and S_T are positive semi-definite, the optimization problem can be reformulated as finding the largest eigenvalues and associated eigenvectors of

$$S_B \mathbf{w}_i = \lambda_i (S_T + \beta I) \mathbf{w}_i \quad (8)$$

which we call the LDA eigenvalue problem. The regularization parameter β is introduced to deal with the singularity of the total scatter matrix and as a countermeasure against overfitting. Suppose the data matrix X is centralized and sorted according to the class of the data points.

$$X = [X_1; X_2; \dots; X_C] \quad (9)$$

where X_i are the data samples of class i . The between-class covariance matrix can then be simplified to

$$S_B = X^T W X \quad (10)$$

with W an $N \times N$ block diagonal matrix with matrices $(W_1, \dots, W_C)$ on its main diagonal; each entry of $W_c \in \mathbb{R}^{N_c \times N_c}$ is equal to $1/N_c$ where N_c is the number of samples in class c . The total scatter matrix can be written as $S_T = X^T X$ and the LDA eigenvalue problem is then

$$X^T W X \mathbf{w}_i = \lambda_i (X^T X + \beta I) \mathbf{w}_i. \quad (11)$$

2.2. Spectral regression

Spectral regression discriminant analysis (SRDA) [22] is a state-of-the-art algorithm of LDA and is a special case of the graph embedding analysis technique called Spectral Regression (SR) [24].

The LDA eigenvalue problem (11) with $\beta = 0$ can be interpreted as the projection of the eigenvalue problem

$$W \mathbf{z}_i = \lambda_i \mathbf{z}_i \quad (12)$$

on the range of X [22]. This eigenvalue problem is quite simple due to the block diagonal structure of W where values in one block are identical. There are two eigenvalues: eigenvalue 1 with multiplicity C and eigenvalue 0 with multiplicity $N - C$ [22]. The space spanned by the eigenvectors associated with eigenvalue 1 is the space spanned by the vectors

$$\mathbf{z}_c = (\underbrace{0, \dots, 0}_{\sum_{m=1}^{c-1} N_m}, \underbrace{1, \dots, 1}_{N_c}, \underbrace{0, \dots, 0}_{\sum_{m=c+1}^C N_m})^T \quad (13)$$

with $c = 1, \dots, C$ due to the special block diagonal structure of W .

In order to gain some insight in eigenvalue problem (12), we will consider the binary case. If we only have two classes, then the matrix W has rank 2. Taking a linear combination of $\mathbf{z}_1$ and $\mathbf{z}_2$ in (13), two eigenvectors with eigenvalue 1 of eigenvalue problem (12) are

$$\mathbf{z}_{nd} = (\underbrace{1, \dots, 1}_{N_1}, \underbrace{1, \dots, 1}_{N_2})^T, \quad \mathbf{z}_d = (\underbrace{1, \dots, 1}_{N_1}, \underbrace{-1, \dots, -1}_{N_2})^T \quad (14)$$

where $\mathbf{z}_d$ has value 1 for the samples of class 1 and value -1 for the samples of class 2. It is obvious that eigenvector $\mathbf{z}_{nd}$ with all values equal to 1 has no discriminative information. The eigenvector $\mathbf{w}_d$ of (11), with $X\mathbf{w}_d = \mathbf{z}_d$, is the direction that maximizes the LDA objective function when $\mathbf{z}_d \in \text{Range}(X)$. The eigenvalue associated with $\mathbf{w}_d$ is smaller but close to 1 since (11) is the projection of (12) onto the range of X .

SRDA uses the knowledge that the space spanned by the eigenvectors $\mathbf{z}_c$ (12) includes the non-discriminative eigenvector $\mathbf{z}_{nd}$. SRDA orthogonalizes the vectors $\mathbf{z}_c$ with respect to $\mathbf{z}_{nd}$, resulting in $C - 1$ discriminative eigenvectors $\mathbf{z}_1, \dots, \mathbf{z}_{C-1}$ of eigenvalue problem (12). SRDA then solves the linear systems $X\mathbf{w}_c = \mathbf{z}_c$ for $c = 1, \dots, C - 1$ in a regularized least-squares (RLS) manner, resulting in the $C - 1$ eigenvectors of the LDA eigenvalue problem which are the directions that maximize the LDA objective function.

SRDA fits in the spectral regression (SR) framework [24]. SR provides a graph embedding interpretation to different dimensionality reduction techniques such as Semantic Subspace Projection [27] or Locality Preserving Projections [28]. The data set is represented as a graph with the data samples as nodes. The symmetric matrix W contains the weights of the edges between two nodes of the graph. Note that for LDA the resulting graph consists of C fully interconnected subgraphs, one for each class.

Spectral analysis (SA) of this graph computes the eigenvectors associated with the largest eigenvalues of

$$W\mathbf{z} = \lambda D\mathbf{z} \quad (15)$$

with D a diagonal matrix and $D_{ii} = \sum_{n=1}^N W_{in}$. As shown by Cai et al [24], this corresponds to penalizing two connected nodes i, j with a large difference $|z_i - z_j|$. For new data samples a linear function is chosen such that $z_i = \mathbf{x}_i\mathbf{w}$, resulting in $\mathbf{z} = X\mathbf{w}$. Eigenvalue problem (15) then results in

$$X^T W X \mathbf{w} = \lambda X^T D X \mathbf{w} \quad (16)$$

which is similar to (11) without regularization.

SR first computes the eigenvectors of (15) using the Lanczos method [29] which is called the spectral analysis phase. Note that for SRDA these eigenvectors were computed analytically and no eigenvalue solve is required. SR secondly solves the linear system $\mathbf{z} = X\mathbf{w}$ in a regularized least-squares manner [30, Chapter 39]

$$\mathbf{w} = (X^T X + \beta I)^{-1} X^T \mathbf{z} \quad (17)$$

to find the desired directions $\mathbf{w}$ in (11) or (16) during the regression phase.

2.3. Semi-supervised discriminant analysis

The supervised method LDA has been extended to semi-supervised learning. In the semi-supervised setting, the class of each data point is not always known and we will refer to the data points where the class is known as labeled data and unlabeled data otherwise. For both the unlabeled and labeled data, their

structure in the feature space, $\mathbf{x}_i$, is known. Semi-supervised learning combines both the class information of the labeled data and the structure in the feature space of labeled and unlabeled data. SDA [26] uses manifold regularization to extend LDA to the semi-supervised setting. Information of the geometric proximity of the data points in the feature space is contained in the regularizer.

Using a binary similarity matrix $S \in \mathbb{R}^{N \times N}$, where $s_{ij} = 1 = s_{ji}$ if two samples are similar and 0 otherwise, a natural regularizer $R(\mathbf{w})$ for LDA can be defined as

$$R(\mathbf{w}) = \sum_{i=1}^n \sum_{j=1}^n s_{ij} (\mathbf{w}^T \mathbf{x}_i - \mathbf{w}^T \mathbf{x}_j)^2 \quad (18)$$

$$= 2 \sum_{i=1}^n \left((\mathbf{w}^T \mathbf{x}_i)^2 d_i - \sum_{j=1}^n s_{ij} \mathbf{w}^T \mathbf{x}_i \mathbf{w}^T \mathbf{x}_j \right) \quad (19)$$

$$= 2\mathbf{w}^T X^T (D - S) X \mathbf{w} \quad (20)$$

where d_i is the sum of the i th row of S and $D \in \mathbb{R}^{n \times n}$ is the diagonal matrix with d_i on its main diagonal. The matrix $L = D - S$ is the Laplacian matrix of S . Suppose the number of labeled data is ℓ , the SDA eigenvalue problem is then

$$\underbrace{X^T \begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix} X}_{\text{Between-labeled-class-scatter}} \mathbf{w}_i = \underbrace{\lambda_i [X^T (1 - \alpha) \begin{pmatrix} I_\ell & 0 \\ 0 & 0 \end{pmatrix} X + X^T \alpha L X]}_{\text{Total-labeled-scatter}} + \underbrace{\beta I}_{\text{Manifold regularization}} \mathbf{w}_i. \quad (21)$$

The first ℓ rows of X are the labeled data points and $W \in \mathbb{R}^{\ell \times \ell}$ is the same matrix as for LDA. The regularization parameter α is limited to the interval $[0, 1]$ and resembles how much weight is given to the unsupervised information. Note that the information of the unlabeled data is only used in the Laplacian matrix L . SDA can be interpreted as a form of spectral regression framework where the SA formulation (15) now is

$$\begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix} \mathbf{z} = \lambda \left[(1 - \alpha) \begin{pmatrix} I_\ell & 0 \\ 0 & 0 \end{pmatrix} + \alpha L \right] \mathbf{z} \quad (22)$$

and is followed by the usual regression phase.

In the previous derivation we assumed that the similarity matrix S between samples was known. For SDA, it is assumed that similar data points in the feature space are more likely to have the same class.

Defining the similarity between these data points is an important part of the semi-supervised setting. SDA uses a binary similarity matrix but this is not required.

For a binary similarity matrix, the first approach we considered is creating a k -nearest neighbor graph [26, 27]. The value of $s_{ij} \in S$ is 1 if there exists an undirected edge in the k -nearest neighbor graph from data point i to data point j . The k -nearest neighbor graph is built using a distance measure in the feature space. The distance for each data point is calculated to all the other data points. The data point is then connected by an undirected edge to all the other points maximally separated by the distance of the k -nearest neighbor.

The second approach is to create a binary similarity matrix by thresholding. If the distance from data point i to data point j is smaller than the threshold, the data points are similar and $s_{ij} = s_{ji} = 1$. Note that using a threshold instead of the k -nearest neighbors graph may lead to samples having no neighbors at all.

Both approaches for creating the similarity matrix are computationally intensive. However, calculating the similarity matrix is only required once. If the same data set is used for multiple targets or the parameters need to be tuned, the similarity matrix does not change. Additionally, approximate similarity matrices can be computed efficiently [31].

2.4. Centralization

Centralization, as for PCA, plays an important role in LDA and SDA. We will first look into the details of the influence of centralization for LDA and extend the same principles to SDA.

2.4.1. Centralization for LDA

The non-discriminative eigenvector $\mathbf{z}_{nd}$ introduced earlier can be removed from the LDA eigenvalue problem (11) by centralization of the data [22, 23]. If $\mathbf{z}_{nd} \in \text{Range}(X)$, then the corresponding vector $\mathbf{w}_{nd}$, with $X\mathbf{w}_{nd} = \mathbf{z}_{nd}$, is an eigenvector of the LDA eigenvalue problem

$$X^T W X \mathbf{w}_{nd} = \lambda_i (X^T X \mathbf{w}_{nd} + \beta \mathbf{w}_{nd}). \quad (23)$$

Note that due to the structure of W ,

$$X^T W X \mathbf{w}_{nd} = X^T W \mathbf{z}_{nd} \quad (24)$$

$$= X^T \mathbf{z}_{nd} \quad (25)$$

$$= N \boldsymbol{\mu} \quad (26)$$

with $\boldsymbol{\mu} = X^T \mathbf{z}_{nd} / N$ the mean sample in the feature space. If the data is centralized, the non-discriminative eigenvector $\mathbf{w}_{nd}$ is moved to the common null space of S_B and S_T of the LDA eigenvalue problem (8).

2.4.2. Centralization for SDA

The centralization trick from LDA to remove the non-discriminative eigenvector can be applied for SDA as well. Consider the vector $\mathbf{z}_{nd} \in \mathbb{R}^{N \times 1}$ with all values equal to 1. From the definition of the Laplacian matrix L , we know that $L\mathbf{z}_{nd} = \mathbf{0}$. Therefore, $\mathbf{z}_{nd}$ is an eigenvector of the SA eigenvalue problem for SDA

$$\begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix} \mathbf{z}_{nd} = \lambda_{nd} \left[(1 - \alpha) \begin{pmatrix} I_\ell & 0 \\ 0 & 0 \end{pmatrix} \mathbf{z}_{nd} + \alpha L\mathbf{z}_{nd} \right] \quad (27)$$

with $\lambda_{nd} = 1/(1 - \alpha)$. The SDA eigenvalue problem can be interpreted as a projection of this eigenvalue problem onto the range of X . As in LDA, we have

$$X^T \begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix} \mathbf{z}_{nd} = \ell \boldsymbol{\mu}_\ell \quad (28)$$

with $\boldsymbol{\mu}_\ell$ the mean of the labeled data in the feature space. To remove the non-discriminative eigenvector, we centralize the data around the mean of the labeled data since the Laplacian does not influence the eigenvector $\mathbf{w}_{nd}$, found by solving the system $X\mathbf{w}_{nd} = \mathbf{z}_{nd}$. The original formulation of SDA [26] without centralization also computes the non-discriminative direction but this direction does not help in the classification task.

2.5. Subspace iteration for SDA

For the SDA eigenvalue problem (21), we have to solve a potentially large-scale singular generalized eigenvalue problem

$$A\mathbf{v} = \lambda B\mathbf{v} \quad (29)$$

where A is of low rank ($C - 1$). Exact solutions for high dimensional data are costly to calculate and are no longer feasible. Subspace iteration on the other hand is an iterative method and has geometric asymptotic convergence for eigenvalue λ_i with ratio λ_{j+1}/λ_i for $i = 1, \dots, j$ with j the dimension of the subspace [32]. Choosing $j = C - 1$ leads to instant convergence due to the low rank of A ($\lambda_{j+1} = 0$). The subspace iteration algorithm applied to eigenvalue problem (29) is given in Algorithm 1 with R randomly generated from a uniform distribution between -1 and 1 .

Algorithm 1 Subspace iteration for (29)

Input: Matrix $A \in \mathbb{R}^{D \times D}$ of rank $C - 1$ and $B \in \mathbb{R}^{D \times D}$

Output: Eigenspace $V \in \mathbb{R}^{D \times C-1}$

- 1: Initialize $R \in \mathbb{R}^{D \times C-1}$ randomly
  - 2: Solve V from $BV = AR$
-

The computational bottleneck is solving the linear system with B . When B is high-dimensional this can no longer be done exactly in a feasible manner and iterative methods are used. The convergence of subspace

iteration is no longer guaranteed in one iteration. In general, many applications such as SDA require only an approximate solution and for these applications it is not necessary to solve the problem to machine precision.

We propose to use Krylov subspace methods for solving the linear system $B\mathbf{w} = \mathbf{b}$ on line 2 of Algorithm 1. A Krylov space

$$\text{span}\{\mathbf{b}, B\mathbf{b}, \dots, B^{k-1}\mathbf{b}\}. \quad (30)$$

is built by a sequence of matrix-vector multiplications. A nice property of these methods is that if the matrix B is sparse the cost of a matrix-vector multiplication is linear with the number of nonzero entries in the matrix B . For the SDA eigenvalue problem (21), the matrix B is a product of sparse matrices, which can also be exploited efficiently in a Krylov method. Note that B is not computed explicitly since this generally leads to a dense matrix. The remainder of this subsection is technical and more detailed information can be found in the literature [29, 33].

For symmetric matrices (which is the case in SDA) the idea is to find a Krylov base $\mathcal{K}$ such that

$$\mathcal{K}^* B \mathcal{K} = T \quad (31)$$

with

$$T = \begin{bmatrix} \alpha_1 & \gamma_1 & & & \\ \gamma_1 & \ddots & & & \\ & \ddots & \ddots & & \\ & & \ddots & \gamma_{k-1} & \\ & & & \gamma_{k-1} & \alpha_k \end{bmatrix} \quad (32)$$

a tridiagonal matrix.

The approximate solution of the linear system $B\mathbf{w} = \mathbf{b}$ is found by searching the space spanned by the vectors of the Krylov space, which we denote by $\mathcal{K}$. One way to find an approximate solution is to make the residual orthogonal to the Krylov base ($B\mathbf{w}_k - \mathbf{b} \perp \text{Range}(\mathcal{K})$). This is true when $\mathbf{w}_k = \mathcal{K}\mathbf{z}_k$ where $\mathbf{z}_k$ is the solution of the linear system

$$T\mathbf{z}_k = \|\mathbf{b}\|_2 \mathbf{e}_1 \quad (33)$$

of dimension k . If the matrix is positive-definite, the solution minimizes the error under the B -norm. This method is the Lanczos algorithm to solve a linear system and is related to the conjugate gradient method (CG) for which a Cholesky factorization of T is used to define a recursive process [29].

2.6. Shift-invariance property of Krylov subspace methods

Singular eigenvalue problems as in (29) are well understood for small-scale matrices [34, 35]. However, these algorithms are unfeasible for large-scale problems, since singular subspace deflation, removing the null space from both matrices, is required and has cubic complexity. In contrast, adding a regularization parameter or shift to matrix B (29) is computationally cost-effective, equally well-defined and offers a countermeasure against overfitting.

Adding a shift and thus replacing B by $B + \beta I$ makes the eigenvalue problem well defined for which we can use a standard method, such as subspace iteration. If β is small, the eigenvalues and eigenvectors of interest do not change significantly [36]. In practice, B is not necessarily singular. In fact the singular values quickly decay, but they do not reach zero. These singular values are the result of ‘noise’ in the data or a mismatch in the imposed classification. The effect of this ‘noise’ is reduced by adding the regularization parameter β .

In general, the best solution is chosen from a range of β values in terms of a performance measure. Krylov bases are invariant to shifts or regularization parameters and exploiting this property is very efficient [37, 38, 39]. In the previous discussion, the Krylov vectors are independent of the value of β and T becomes $T + \beta I$. This means that the computationally expensive part, that is, building the Krylov base, can be reused for several β values instead of restarting the process for each β value. The parameter β only plays a role in solving the small linear system in (33), where T is replaced by $T + \beta I$. Since the value of β is not known, we propose the conjugate gradient method for several shifted linear systems (shifted CG) [39, 40] to solve

$$(B + \beta I)\mathbf{w} = A\mathbf{r} \quad (34)$$

in Algorithm 1 for the SDA eigenvalue problem (21).

2.7. Implementation of the centralized multiplication

In Section 2.4 it was shown that centralization avoids calculating the non-discriminative eigenvector. Note that if we apply centralization directly on the matrix X , we will lose the sparsity of the data matrix and one of the advantages of Krylov subspace methods. In order to avoid creating a dense matrix filled with real values, the centralization is applied in the matrix-vector product. Using $\mathbf{z}_{nd} = \mathbf{1} \in \mathbb{R}^{N \times 1}$ as the vector of all ones and defining $\mathbf{z}_\ell = \mathbf{1}_\ell \in \mathbb{R}^{N \times 1}$ as the vector with value 1 for the labeled samples and 0 otherwise, the mean labeled sample in the feature space $\boldsymbol{\mu}_\ell = X^T \mathbf{1}_\ell / \ell$ can then be derived. The centralized multiplication with X can be written as

$$\left[\left(I_N - \frac{\mathbf{1}\mathbf{1}^T}{\ell} \right) X \right] \mathbf{v} = (X - \mathbf{1}\boldsymbol{\mu}_\ell^T) \mathbf{v} \quad (35)$$

$$= X\mathbf{v} - \mathbf{1}\langle \boldsymbol{\mu}_\ell^T, \mathbf{v} \rangle \quad (36)$$

and the centralized multiplication with X^T as

$$\left[\left(I_N - \frac{\mathbf{1}\mathbf{1}^T}{\ell} \right) X \right]^T \mathbf{w} = (X - \mathbf{1}\boldsymbol{\mu}_\ell^T)^T \mathbf{w} \quad (37)$$

$$= X^T \mathbf{w} - \boldsymbol{\mu}_\ell \langle \mathbf{1}^T, \mathbf{w} \rangle \quad (38)$$

were the last inner product is the sum of the elements of $\mathbf{w}$.

3. Four algorithms for SDA

Here we present four methods for solving the SDA eigenvalue problem for binary classification. The differences between these methods are subtle since they solve the same eigenvalue problem. First, we propose an efficient implementation of the spectral regression for SDA (SR-SDA). Next, we add centralization to spectral regression for SDA, resulting in centralized spectral regression for SDA (CSR-SDA). We then propose spectral analysis to classify the unlabeled data (SA-SDA) in the semi-supervised setting. Finally, we present our scalable implementation for solving the centralized SDA eigenvalue problem directly (FSDA).

For the evaluation of these algorithms, we use the area under the curve of the receiver operating characteristic (AUC-ROC). In the case of binary classification, there is only one discriminative eigenvector. This eigenvector is used to classify the data by projecting along this vector, namely $X\mathbf{w} = \mathbf{s} \in \mathbb{R}^{N \times 1}$. This projection leads to the sample rating or score $\mathbf{s}$. We use this rating directly to calculate the AUC-ROC, but other classification methods like support vector machines [41], nearest neighbor [42] or nearest centroid [43, 44] can be used to classify the data in the reduced dimension.

3.1. Spectral regression for SDA

The spectral regression framework applied to SDA (SR-SDA) first solves the spectral analysis SDA eigenvalue problem (22). The original framework finds all the eigenvectors of the spectral analysis SDA eigenvalue problem and then finds the corresponding feature vectors by solving a linear system in a regularized least squares manner. In order to find the eigenvectors, we use subspace iteration on

$$\left[(1-\alpha) \begin{pmatrix} I_\ell & 0 \\ 0 & 0 \end{pmatrix} + \alpha L \right] Z = \begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix} R \quad (39)$$

to find the eigenspace $Z \in \mathbb{R}^{N \times 2}$ with $\mathbf{R}$ randomly distributed. The block conjugate gradient method (BCG) [45] is applied to solve the linear system in (39). Subspace iteration will actually find the eigenspace Z consisting of a linear combination of both the non-discriminative eigenvector $\mathbf{z}_{nd}$ and the discriminative eigenvector $\mathbf{z}_d$. The eigenvector $\mathbf{z}_d$ can be extracted from the eigenspace Z by applying Rayleigh-Ritz subspace projection [32]. Eigenvalue problem (22) is transformed to the two-by-two eigenvalue problem

$$Z^T \begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix} Z \mathbf{q}_i = \lambda_i Z^T \left[(1-\alpha) \begin{pmatrix} I_\ell & 0 \\ 0 & 0 \end{pmatrix} + \alpha L \right] Z \mathbf{q}_i \quad (40)$$

for $i = 1, 2$ and the discriminative eigenvector is the projection of the eigenvector with second largest eigenvalue $\mathbf{z}_d = Z \mathbf{q}_2$.

Next we solve $X\mathbf{w}_d = \mathbf{z}_d$ using the regularized least-squares linear system [30, Chapter 39]

$$(X^T X + \beta I) \mathbf{w}_d = X^T \mathbf{z}_d \quad (41)$$

with shifted CG to find the discriminative eigenvector $\mathbf{w}_d$ in the feature space. The outline of the algorithm is given in Algorithm 2, where we apply the regression phase to both eigenvectors $\mathbf{z}_d$ and $\mathbf{z}_{nd}$ to match the computational workload of the original algorithm as close as possible. The LSQR method is advocated for linear systems arising from regularized least squares [46] and used in spectral regression. However, to the best of our knowledge, there is no specific LSQR variant exploiting the shift-invariance property of Krylov subspaces. As mentioned, our implementation reduces the computational cost by using shifted CG to solve the system for multiple regularization parameters.

Algorithm 2 SR-SDA

Input: Data matrix X , Laplacian matrix L , parameter α and regularization parameters β

Output: Discriminative projection vector $\mathbf{w}_d$ and sample rating $\mathbf{s}$

- 1: Initialize $R \in \mathbb{R}^{N \times 2}$ randomly generated from a uniform distribution between -1 and 1
  - 2: Solve $Z \in \mathbb{R}^{N \times 2}$ with BCG from (39)
  - 3: Solve the two-by-two eigenvalue problem (40)
  - 4: **for** $i = 1, 2$ **do**
  - 5:   Solve $\mathbf{w}_i$ with shifted CG from (41)
  - 6: **end for**
  - 7: Compute sample rating: $\mathbf{s} = X\mathbf{w}_2$
-

3.2. Centralized spectral regression for SDA

SR-SDA also computes the redundant non-discriminative eigenvector $\mathbf{z}_{nd}$ which can be avoided by modifying the SA phase in SDA (22). The SA phase is actually an SDA eigenvalue problem (21) as well with the identity matrix I_N as the data matrix X . In order to avoid the non-discriminative eigenvector $\mathbf{z}_{nd}$, we centralize the identity matrix around the mean of the labeled data ($\mathbf{1}_\ell/\ell$). Now we only need to compute one eigenvector and therefore using the power method [32] suffices,

$$\left((I_N - \frac{\mathbf{1}\mathbf{1}^T}{\ell})^T \left[(1-\alpha) \begin{pmatrix} I_\ell & 0 \\ 0 & 0 \end{pmatrix} + \alpha L \right] (I_N - \frac{\mathbf{1}\mathbf{1}^T}{\ell}) \right) \mathbf{z}_d = \left((I_N - \frac{\mathbf{1}\mathbf{1}^T}{\ell}) \begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix} (I_N - \frac{\mathbf{1}\mathbf{1}^T}{\ell}) \right) \mathbf{r} \quad (42)$$

which is basically subspace iteration with only one right-hand side with $\mathbf{r}$ randomly distributed. If we orthogonalize $\mathbf{r}$ with respect to the labeled mean

$$\mathbf{r} \leftarrow \mathbf{r} - \frac{\langle \mathbf{1}_\ell^T, \mathbf{r} \rangle}{\ell} \mathbf{1}_\ell, \quad (43)$$

the centralization can be avoided in the right-hand side of (42). Since the iterative vectors in CG are in the range of $(I_N - \frac{\mathbf{1}\mathbf{1}^T}{\ell})$, we can avoid one centralization in the left-hand side of (42) and (42) simplifies to

$$\left((I_N - \frac{\mathbf{1}\mathbf{1}^T}{\ell})^T \left[(1 - \alpha) \begin{pmatrix} I_\ell & 0 \\ 0 & 0 \end{pmatrix} + \alpha L \right] I_N \right) \mathbf{z}_d = \begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix} (\mathbf{r} - \mathbf{1} \frac{\langle \mathbf{1}_\ell^T, \mathbf{r} \rangle}{\ell}). \quad (44)$$

The difference with SR-SDA is that CSR-SDA computes one eigenvector less during the spectral analysis phase and therefore requires only one linear solve during the regression phase

$$(X^T X + \beta I) \mathbf{w}_d = X^T \mathbf{z}_d. \quad (45)$$

The outline of the algorithm for centralized spectral regression for SDA (CSR-SDA) is given in Algorithm 3.

Algorithm 3 CSR-SDA

Input: Data matrix X , Laplacian matrix L , parameter α and regularization parameters β

Output: Discriminative projection vector $\mathbf{w}_d$ and sample rating $\mathbf{s}$

- 1: Initialize $\mathbf{r} \in \mathbb{R}^{N \times 1}$ randomly generated from a uniform distribution between -1 and 1
  - 2: Orthogonalize: $\mathbf{r} = \mathbf{r} - \mathbf{1} \frac{\langle \mathbf{1}_\ell^T, \mathbf{r} \rangle}{\ell}$
  - 3: Solve $\mathbf{z}_d$ with CG from (44)
  - 4: Solve $\mathbf{w}_d$ with shifted CG from (45)
  - 5: Compute sample rating: $\mathbf{s} = X \mathbf{w}_d$
-

3.3. Spectral analysis for SDA

CSR-SDA computes the sample rating by projecting the data

$$\mathbf{s} = X \mathbf{w}_d \quad (46)$$

where the projection vector $\mathbf{w}_d$ is the solution of

$$X \mathbf{w}_d = \mathbf{z}_d \quad (47)$$

with $\mathbf{z}_d$ the discriminative eigenvector of SA (22). It follows from (46) and (47) that the rating of the samples approximates the discriminative eigenvector of SA ($\mathbf{s} \simeq \mathbf{z}_d$). This assumes that the samples of interest are present in X . In the semi-supervised setting, the data samples requiring classification are often available in the data matrix X as unlabeled data. For the rating of these unlabeled data, SA-SDA calculates their rating as their corresponding values of the eigenvector of SA. The outline of SA-SDA is shown in Algorithm 4 where we added regularization to the SA phase as countermeasure against overfitting. Note that the information of the unlabeled samples is available in the Laplacian L and if $\alpha = 0$ in 22, the Laplacian is neutralized and the unlabeled data receive a rating of zero. Therefore, SA-SDA only returns valid ratings if $\alpha \neq 0$.

Algorithm 4 SA-SDA

Input: Data matrix X , Laplacian matrix L , parameter α and regularization parameters β

Output: Sample rating $\mathbf{s}$

- 1: Initialize $\mathbf{r} \in \mathbb{R}^{N \times 1}$ randomly generated from a uniform distribution between -1 and 1
  - 2: Orthogonalize: $\mathbf{r} = \mathbf{r} - \mathbf{1} \frac{\langle \mathbf{1}_\ell^T, \mathbf{r} \rangle}{\ell}$
  - 3: Solve $\mathbf{z}_d$ with shifted CG from (44)
  - 4: Compute sample rating: $\mathbf{s} = \mathbf{z}$
-

3.4. Fast semi-supervised discriminant analysis

CSR-SDA solves the eigenvalue problem (21) in two phases and uses an iterative solver during each phase. One difficulty with this approach is that the residual of the SDA eigenvalue problem is not necessarily smaller than the required tolerance in each individual phase. Since we are working with sparse data and Krylov subspace methods, the complexity of the algorithms is proportional to the number of nonzeros. Hence, we propose an efficient implementation to solve the centralized SDA eigenvalue problem directly by using the power method for

$$\left((X - \mathbf{1}\mu_\ell^T)^T \left[(1 - \alpha) \begin{pmatrix} I_\ell & 0 \\ 0 & 0 \end{pmatrix} + \alpha L \right] X + \beta I_D \right) \mathbf{w}_d = (X - \mathbf{1}\mu_\ell^T)^T \begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix} X \mathbf{r}. \quad (48)$$

with $\mathbf{r}$ randomly distributed. The outline of the algorithm for fast semi-supervised discriminant analysis (FSDA) is given in Algorithm 5.

Algorithm 5 FSDA

Input: Data matrix X , Laplacian matrix L , parameter α and regularization parameters β

Output: Discriminative projection vector $\mathbf{w}_d$ and sample rating $\mathbf{s}$

- 1: Initialize $\mathbf{r} \in \mathbb{R}^{D \times 1}$ randomly generated from a uniform distribution between -1 and 1
  - 2: Calculate mean labeled sample: $\mu_\ell = \frac{X^T \mathbf{1}_\ell}{\ell}$
  - 3: Solve $\mathbf{w}_d$ with shifted CG from (48)
  - 4: Compute sample rating: $\mathbf{s} = X \mathbf{w}_d$
-

3.5. Comparison of the SDA algorithms

All four algorithms solve the same underlying problem. In a perfect world without regularization and if the solution of $\mathbf{z}_d = X \mathbf{w}_d$ exists, these algorithms would have the same performance. In reality regularization is required and iterative methods are used to solve linear systems within a certain tolerance. The difference between SR-SDA and CSR-SDA is only computational and these two algorithms result in the same classification. The differences between CSR-SDA, SA-SDA and FSDA are more subtle. FSDA

solves the SDA eigenvalue problem (21) directly and the tolerance used in CG reflects the true error of the eigenvector. Using the same tolerance in both linear systems of CSR-SDA does not guarantee this accuracy for the SDA eigenvalue problem.

All presented iterative methods build a Krylov subspace and each iteration is linear in the number of samples, features and nonzeros of the matrices. Furthermore, centralization requires 4 operations for each iteration each of which has complexity linear in the column size of the matrix. Table 2 shows the complexity of the different algorithms given in this section with k_1 the number of CG-iterations with dimension N , k_2 the number of CG-iterations with dimension D , nz_L the number of nonzeros of L , nz_X the number of nonzeros of X , N_β the number regularization parameters β and P_β the extra cost related to N_β used in shifted CG. Applying centralization for CSR-SDA results in a speed-up of approximately $C/(C-1)$ with respect to SR-SDA, reducing execution time by almost a factor 2 for binary classification.

Table 2 also shows the main difference between CG and Shifted CG. We did not take into account the influence of the β -value on the number of iterations required to reach a certain tolerance and assumed that each solve requires the same number of iterations. CG has to rebuild the Krylov subspace for each value of β , while shifted CG reuses the basis. Note that the number of nonzeros nz is generally significantly larger than the column size of the matrix, and thus shifted CG results in a speed-up approximately the number of regularization parameters used.

Algorithm	Complexity
SR-SDA	$k_1 C(nz_L + 10N) + k_2 C(2nz_X + 10D)$
CSR-SDA	$k_1 (C-1)(nz_L + 14N) + k_2 (C-1)(2nz_X + 10D)$
SA-SDA	$k_1 (C-1)(nz_L + 14N)$
FSDA	$k_2 (C-1)(2nz_X + nz_L + 14D)$
CG	$k_1 N_\beta (nz + 10N)$
Shifted CG	$k_1 (nz + N_\beta 10N + P_\beta)$

Table 2: The complexity of the 4 SDA algorithms.

4. Experimental results

In this section we compare FSDA to other related dimensionality reduction methods for small to medium sized data sets. Next we apply the four algorithms for SDA on two large-scale drug-protein activity prediction data sets.

4.1. Comparison of related methods

Semi-supervised learning has gained traction and several semi-supervised dimensionality reduction techniques were developed in recent years. We used the following data sets for comparing AUC and execution time for related dimensionality reduction methods.

1. The Ionosphere data set is from the UCI machine learning repository and consists of radar returns from the ionosphere [47, 48]. The data has 351 samples and 34 continuous features.
2. The ovarian cancer data set consists of high-resolution ovarian cancer data generated using the WCX2 protein array and is available in MATLAB [49]. The data set includes 95 controls and 121 ovarian cancers resulting in 216 samples and 4000 features.
3. Dexter is a binary classification task about filtering texts. The data consists of sparse and continuous input variables and is available on the UCI Machine Learning repository [50, 48]. This dataset is one of five datasets of the NIPS 2003 feature selection challenge and consists of 2600 samples and 20000 features.
4. The RNAseq is a part of the collection of data of the RNA-Seq (HiSeq) PANCAN data set, it is a random extraction of gene expressions of patients having different types of tumor: BRCA, KIRC, COAD, LUAD and PRAD [51, 48]. We binarized the classification task by considering the classes LUAD, KIRC and COAD as active. The data set has 801 samples and 20531 features.
5. GISETTE is a handwritten digit recognition problem from the UCI machine learning repository [50, 48]. The classification task is to identify the digits 4 and 9. GISETTE consists of 13500 samples and 5000 features.
6. Nlpdata is natural language processing data extracted from the MathWorks documentation and is available in MATLAB [49]. We considered the classes *comm*, *dsp*, *ecoder*, *hdlcoder*, *phased* and *physmod* as active. The data set has in total 31572 samples and 34023 features.

Using these data sets, FSDA is compared with the following dimensionality reduction techniques.

1. Flexible Manifold Embedding (FME) [14] aims to find the optimal prediction labels for all training samples. The objective function aims at minimizing the label fitness and manifold smoothness, resulting in a flexible framework. The described algorithm requires the inverse of $X^T X + I$ and loses data sparsity in the process.
2. Trace Ratio based flexible Semi-supervised discriminant analysis (TR-FSDA) [13] relaxes the constraint that the low-dimensional data representation lies in the subspace spanned by the data matrix by

using a flexible regularizer. The method requires an eigenvalue decomposition as main computational bottleneck.

3. Semi-supervised orthogonal discriminant analysis (SODA) [10] uses a specially designed label propagation. Orthogonal discriminant analysis is then applied using the soft labels learned by label propagation. The main computational step in the algorithm is an eigenvalue decomposition.
4. In Globality-Locality Preserving Projections (GLPP) [52] the data-manifold is a combination of two sub-manifolds, namely the sub-manifold of the dynamic factors of samples and the sub-manifold of static sample factors. The method requires an eigenvalue decomposition.
5. Semi-supervised Linear Discriminant Analysis (SLDA) [15] is an iterative procedure where the projection is computed and next the soft labels are calculated using the projected data. These two steps are repeated iteratively until convergence. Calculating the projection requires an eigenvalue decomposition.

The eigenvalue problems in these methods are for matrices of almost full rank and subspace iteration would converge slowly and is thus not feasible for these problems. In contrast, it is the low rank of the between-labeled-class-scatter matrix in (21) that allows efficient use of subspace iteration in FSDA.

To allow fair comparison between these methods, nested 3-fold cross-validation was used to choose between 4 sets of parameters for all methods. The computational complexity of the methods did not allow for full grid-search in a nested cross-validation setting for all parameters. This limits one advantage of applying SDA since the shift-invariance property allows the regularization parameter β to be chosen during one run, leaving only the parameter α to be chosen over multiple runs. The number of labeled data in the experiments was restricted to three for each class in the inner folds.

After the projection was calculated, a linear regression classifier was trained in the reduced dimension for the training data. This classifier was then used on the test data, available during training as unlabeled data, to calculate the AUC-ROC. The AUC-ROC and execution times are shown in Table 3. The MATLAB implementation for these results is available¹. Table 3 shows that although FSDA might not be the best method for classification for all data sets, FSDA remains competitive and clearly has the smallest execution time. FSDA is thus best suited for large-scale data sets.

4.2. SDA for drug-protein activity prediction

This work is performed with drug-protein activity prediction in mind. We have tested our methods on different targets for two different data sets. The first data set is described in section 4.2.1 and is publicly

---

¹<https://people.cs.kuleuven.be/~joris.tavernier/>

Data set	FSDA		FME		TR-FSDA		SODA		GLPP		SLDA	
	AUC	Time(s)	AUC	Time(s)	AUC	Time(s)	AUC	Time(s)	AUC	Time(s)	AUC	Time(s)
Ionosphere	87.21	0.0008	87.83	0.0049	82.16	0.052	76.02	0.0054	71.09	0.0013	81.13	0.0080
Ovarian cancer	93.42	0.0118	83.62	0.0360	82.00	40.3340	90.30	84.0769	94.19	16.6654	89.71	18.51
Dexter	94.65	0.0745	94.14	17.9739	62.92	2277.059	78.90	1405.62	54.835	3587.09	61.72	6200.33
RNAseq	95.68	0.3165	99.69	1.9414	67.28	1257.60	65.74	4198.47	79.63	4537.90	60.96	5833.04
GISETTE	80.76	16.6482	95.47	109.187	62.04	3419.79	77.47	5206.99	66.25	1309.00	59.47	1591.92
Nlpdata	69.65	14.8245	Out of memory		Out of memory		Out of memory		Out of memory		Out of memory	
Average Rank	2	1	1	2	5	3	3	5	4	4	6	6

Table 3: Comparison of FSDA and related dimensionality reduction methods for different data sets. The average AUC-ROC and outer fold execution time are given for 3-fold nested cross-validation. The classification results of FSDA are competitive with related methods and FSDA has the smallest execution time of all compared methods.

available. The C++ implementation for this data set was compiled with gcc 5.3.1 and OPENMP 4.0 with compile option -O3 and executed on a machine with an Intel(R) Core(TM) i7-6560U (2.20GHz) processor with an L3 cache memory of 4096 KB and 16 GB of DRAM where all 4 cores were used². The second industry-scale data set is described in section 4.2.3 and stronger hardware was required. These experiments were run using a computing node with dual Intel(R) Xeon(R) E5-2699 v3 (2.30GHz) processors with a L3 cache memory of 46080 KB and 264 GB of DRAM. For these experiments we used 12 cores.

First we compare FSDA with a deep generative model [53] for the sparse ChEMBL data set. Next, we present the AUC-ROC in function of the execution time to evaluate the different SDA methods for these data sets. Since these methods solve the same underlying problem, we are interested in convergence speed and difference in the algorithms predictive performances.

4.2.1. Comparison of FSDA with a deep generative model for the ChEMBL data

For drug-protein activity prediction, one of the possible indicators is the IC50 value. IC50 or the half maximal inhibitory concentration measures the substance concentration required to inhibit the activity of a protein by 50%. Using the bioactivity database ChEMBL version 19 [54], we selected protein targets that had at least 200 or more IC50 values for the compounds. A compound was considered active with an IC50 value less than 1000 nM. We then kept the protein targets that had at least 30 actives and inactives, resulting in 330 targets.

The extended-connectivity fingerprints (ECFP [19]) were computed for the compounds using rdkit [55] with 3 layers. In summary, ECFP generates the features for a compound in layers. In the first layer each individual atom receives an initial identifier based on its atomic symbol. In the following layer this value is concatenated with the bond and the identifiers of the neighboring atoms and then hashed to a new identifier for this atom. This process is repeated for a fixed number of layers. These generated substructures

²<https://people.cs.kuleuven.be/~joris.tavernier/>

Figure 1: Example of the ECFP feature generation.

represented by their hash value are the features for this compound. Duplicate identifiers or duplicate substructures are removed from the feature list. Figure 1 depicts the creation of the ECFP features for a small chemical compound. The characteristics of the resulting ECFP data matrix X are given in Table 4. The similarity matrix was designed using a 5-nearest neighbor graph and the Tanimoto similarity. To

Parameter	Value
Compounds	167 668
Features	291 714
Nonzeros	12 246 376
Density	0.025038%

Table 4: The characteristics of the ChEMBL data matrix.

evaluate the convergence and execution time for the different methods, we randomly selected 6 targets with 2000 or more labeled compounds. The number of active and inactive compounds for each target is given in Table 5.

Name	Active	Inactive	Known labels	% labeled
CHEMBL203	2225	1729	3954	2.36
CHEMBL220	1365	1275	2640	1.57
CHEMBL228	1621	491	2112	1.26
CHEMBL230	1409	1170	2579	1.54
CHEMBL240	817	3144	3961	2.36
CHEMBL260	2514	516	3030	1.81

Table 5: The characteristics of the chosen targets for the 167 668 chemical compounds of the ChEMBL data set.

We compared the performance of FSDA with a deep generative model for this large-scale data set. Deep generative models have been developed for semi-supervised learning [53]. Table 6 shows the AUC and execution time for 5-fold cross-validation of FSDA and the M2 deep generative model[53] (DGM) with added regularization. The experiments with the deep generative network were run on an NVIDIA Quadro GV100

GPU with 32GB RAM. The implementation³ uses PyTorch 1.0. As can be seen from Table 6 the accuracy of the deep generative model is competitive with FSDA in terms of AUC but the average execution time is about two orders of magnitude slower. This is mainly due to the fact that DGM generates the ECFP features requiring a dense high-dimensional output layer.

Gene	FSDA		DGM (8, 32, 16)		DGM (16, 64, 32)		DGM (32, 128, 64)		DGM (64, 256, 128)	
	AUC	Time(s)	AUC	Time(s)	AUC	Time(s)	AUC	Time(s)	AUC	Time(s)
CHEMBL203	93.00	1.77	90.98	142	91.01	175	92.01	245	91.34	392
CHEMBL220	93.05	1.80	92.80	142	93.45	178	93.88	246	93.45	392
CHEMBL228	91.17	1.81	90.63	142	90.38	176	90.59	245	90.19	391
CHEMBL230	88.33	1.82	87.08	141	87.37	179	87.01	246	87.45	393
CHEMBL240	85.29	1.81	80.98	142	81.22	178	82.75	245	83.22	391
CHEMBL260	90.10	1.80	89.77	142	90.01	176	89.72	246	89.55	392
Average	90.16	1.80	88.49	141.83	88.91	177.00	89.33	245.50	89.20	391.83

Table 6: Comparison of AUC and average execution time of one fold of FSDA with a deep generative model (DGM) of the chosen targets of the ChEMBL data set. We varied the latent dimension n_l , number of nodes in the first hidden layer n_1 and number of nodes in the second hidden layer n_2 for the DGM (n_l, n_1, n_2).

4.2.2. Comparison of the four SDA algorithms for the ChEMBL data

Using the same setup as described in Section 4.2.1, we varied the number of CG iterations which increases the complexity, but also the accuracy of CG for the different SDA methods. Note that the spectral regression methods require two linear solves and the best pair (execution time, AUC-ROC) is reported. The different number of iterations for the linear solvers are given in Table 7. 5-Fold nested cross-validation was used for the experiments, with the inner-folds used to pick the best value of the regularization parameter β . Starting from $\beta = 10^{-9}$, we let β vary by multiplying β with 10 until $\beta = 10^3$. We report the average execution time for the outer cross-validation using only one optimal β . The reported AUC-ROC is the average along with the standard deviation for all outer folds and 5 different seed initializations of the random number generator.

Figure 2 shows the AUC-ROC in function of the execution time for the different methods. Solving SDA directly in the feature space (FSDA) gives higher or equal average AUC-ROC for all 6 targets in comparison with the spectral regression methods. There is no difference in AUC-ROC between SR-SDA and CSR-SDA, but Figure 2 clearly shows that CSR-SDA requires less execution time to reach the same predictive performance. For this data, the AUC-ROC in function of execution times is higher for FSDA than for CSR-SDA. Ideally, we prefer to reach high AUC-ROC with the smallest execution time (top left of the graph). We conclude that FSDA is more suitable than CSR-SDA for the ChEMBL data set. Note

³ <https://github.com/wohlert/semi-supervised-pytorch>

Figure 2: Comparison of spectral regression methods and FSDA for the 6 chosen targets of the ChEMBL data set. The AUC-ROC is given in function of the execution time in seconds. Converging to large AUC-ROC in small execution time (top left of the graph) is preferable.

Method	Description	Iterations
FSDA	Iterations linear solve for SDA	[2, 3, 5, 10, 20, 40, 60, 80]
CSR-SDA	Iterations linear solve for spectral SDA	[2, 3, 5, 10, 20, 40, 60, 80]
	Iterations RLS solve with X (regression)	[2, 3, 5, 10, 20, 40, 60, 80]
SA-SDA	Iterations linear solve for spectral SDA	[3, 5, 10, 20, 40, 60, 80]
SR-SDA	Iterations linear solve for spectral SDA	[2, 3, 5, 10, 20, 40, 60, 80]
	Iterations RLS solve with X (regression)	[2, 3, 5, 10, 20, 40, 60, 80]

Table 7: The different parameters used in the experiments for each method.

that SA-SDA reaches high AUC-ROC values in even less execution time since there is no regression step. Currently, we do not know if the difference in AUC-ROC after convergence between CSR-SDA, SA-SDA and FSDA is specifically for the data or inherently present due to the way the direction $\mathbf{w}$ is calculated.

N_β	β	CG	shifted CG	Speed-up
12	[1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 1e2]	43.11 s	6.02 s	7.16
12	[1.0e-6, 1.1e-6, 1.2e-6, 1.3e-6, 1.4e-6, 1.5e-6, 1.6e-6, 1.7e-6, 1.8e-6, 1.9e-6, 2.0e-6, 2.1e-6]	50.98 s	5.68 s	8.98

Table 8: The computation time of CG versus shifted CG for the regression phase using a tolerance of 10^{-3} , the ChEMBL data matrix and given β -values.

Table 8 shows the execution time for CG and shifted CG solving the regularized least squares system in CSR-SDA (Algorithm 3, line 5) with a tolerance of 10^{-3} and several β -values. Using shifted CG leads to a speed-up of 7.16 in computation time for 12 β -values. Note that for the larger β -values the number of iterations of CG required to reach the desired tolerance is less than for the small β -values. Higher regularization parameters result in a smaller condition number and faster convergence for CG [29]. Table 8 additionally shows the speed-up if β would not vary significantly, resulting in a larger speed-up of 8.98. If the β -values do not vary drastically, the number of iterations of CG do not change significantly and the achieved speed-up is larger.

4.2.3. Janssen data

Next, we investigate the performance of the proposed methods on a industrial-scale data set containing a few million of compounds. This large-scale data set was provided by Janssen Pharmaceutica. Out of more than 1000 protein targets, we chose 6 with a large number of IC50 measurements, see Table 9 for an overview of these targets. For these targets, a compound was considered active with an IC50 value below $1.0 \mu M$. Using ECFP over 10 million compound features were generated and the SDA similarity matrix was created using the Tanimoto similarity with a threshold of 0.4.

Figure 3: Comparison of spectral regression methods and FSDA for the 6 Janssen targets. The AUC-ROC is given in function of the execution time in seconds with a fixed number of iterations equal to 70. Our methods improve the execution time by a factor 2 in comparison with SR-SDA or more for SA-SDA.

Figure 3 shows the AUC-ROC in function of the execution time for the targets. The number of iterations for the linear solves was set to 70. As can be seen from the figure, the centralization improves the execution time by a factor 2 in comparison with SR-SDA. SA-SDA even gains a factor 7-8. All methods reach the same predictive performance with a small advantage for FSDA as seen for the ChEMBL data set as well.

Name	Active	Inactive	Total labels	% labeled
389	20 000	275 000	295 000	15
685	10 000	475 000	485 000	25
739	5000	750 000	755 000	40
1448	2500	300 000	302 500	15
1736	5000	500 000	505 000	25
1833	2500	300 000	302 500	15

Table 9: The characteristics of the Janssen targets.

The relatively large assays for the Janssen data allows us to subsample the labeled data. Figure 4 gives the AUC-ROC in function of randomly sub-sampled labeled data for different values of the parameter α . Using only 0.1 to 1% labels, SDA outperforms LDA for all the targets. Incorporating information of the unlabeled data through SDA results in larger predictive performance if the number of labels is small.

Figure 4: The AUC-ROC for the Janssen data in function of number of used labels (randomly subsampled) for 6 different targets. Using only 0.1 to 1% labels, SDA outperforms LDA for all the targets.

5. Discussion

Semi-supervised learning has mainly been developed for cases where there is a small number of labels. In our application, there is a medium number of labels and an abundance of unlabeled data samples. Further research is required specifically for this kind of large-scale data sets in the semi-supervised setting. The information of the unlabeled data is incorporated in the classification task. To include this information, SDA uses geometric distance in the feature space to define similarity between samples. Defining similarity in the feature space is sensitive to noise and redundant information of the defined features.

For the chemogenomics application, the features are exact but do contain redundant information. The sample similarity could be further improved by removing non-informative features as suggested in [15]. Note that for our application, even the smallest speed-up for an algorithm is multiplied by the number of specific targets, resulting in a significant reduction of the total execution time.

6. Conclusion

We have presented efficient algorithms for semi-supervised discriminant analysis for large-scale data. We have shown how to use the centralization to avoid the calculation of the non-discriminative eigenvector and an efficient way to implement the centralization. We used the shift-invariance property of Krylov subspaces to our advantage to solve SDA efficiently for different regularization parameters β .

We have applied the spectral regression framework to SDA and showed how to centralize the data in spectral analysis. We additionally showed how to use SA as a classification method in the semi-supervised setting. We applied SDA for target prediction in chemogenomics using data with medium number of labeled data and an abundance of unlabeled data.

Using our proposed methods and implementations, it was possible to apply SDA on real-life large-scale data from the pharmaceutical industry in seconds, improving on previous state of the art by a factor 2 or more. Solving the problem directly in the feature space had experimentally for almost all targets better or equal predictive performance than the spectral regression methods. In the future, we will look further at semi-supervised methods using neural networks and more specifically exploiting the data sparsity for the deep generative model.

Conflict of interest

There is no conflict of interest.

Acknowledgement

The authors would like to thank the reviewers for their comments and suggestions. The authors would additionally like to thank Jean-Marc Neefs for annotating assays with protein and gene identifiers and

Vladimir Chupakhin for helping with the compound normalization of the Janssen data. This work was supported by Flanders Innovation & Entrepreneurship (IWT, No. IWT130406) and Research Foundation - Flanders (FWO, No. G079016N). GPU support was provided by NVIDIA.

References

References

- [1] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer-Verlag New York, 2006.
- [2] G. J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*, John Wiley & Sons, Inc., 2005.
- [3] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning*, Springer-Verlag New York, 2009.
- [4] H. Hotelling, Analysis of a complex of statistical variables into principal components., *Journal of Educational Psychology* 24 (6) (1933) 417–441.
- [5] X. Zhu, A. B. Goldberg, Introduction to semi-supervised learning, *Synthesis lectures on artificial intelligence and machine learning* 3 (1) (2009) 1–130.
- [6] F. Dornaika, Y. E. Traboulsi, A. Assoum, Inductive and flexible feature extraction for semi-supervised pattern categorization, *Pattern Recognition* 60 (2016) 275 – 285.
- [7] T. I. Dhamecha, R. Singh, M. Vatsa, On incremental semi-supervised discriminant analysis, *Pattern Recognition* 52 (2016) 135 – 147.
- [8] F. Dornaika, R. Dahbi, A. Bosaghzadeh, Y. Ruichek, Efficient dynamic graph construction for inductive semi-supervised learning, *Neural Networks* 94 (2017) 192 – 203.
- [9] R. Sheikhpour, M. A. Sarram, S. Gharaghani, M. A. Z. Chahooki, A survey on semi-supervised feature selection methods, *Pattern Recognition* 64 (2017) 141 – 158.
- [10] F. Nie, S. Xiang, Y. Jia, C. Zhang, Semi-supervised orthogonal discriminant analysis via label propagation, *Pattern Recognition* 42 (11) (2009) 2615 – 2627.
- [11] M. Sugiyama, T. Idé, S. Nakajima, J. Sese, Semi-supervised local fisher discriminant analysis for dimensionality reduction, *Machine Learning* 78 (1) (2009) 35.
- [12] H. Wang, S. Yan, D. Xu, X. Tang, T. Huang, Trace ratio vs. ratio trace for dimensionality reduction, in: *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.
- [13] Y. Huang, D. Xu, F. Nie, Semi-supervised dimension reduction using trace ratio criterion, *IEEE Transactions on neural networks and learning systems* 23 (3) (2012) 519–526.
- [14] F. Nie, D. Xu, I. W. H. Tsang, C. Zhang, Flexible manifold embedding: A framework for semi-supervised and unsupervised dimension reduction, *IEEE Transactions on Image Processing* 19 (7) (2010) 1921–1932.
- [15] S. Wang, J. Lu, X. Gu, H. Du, J. Yang, Semi-supervised linear discriminant analysis for dimension reduction and classification, *Pattern Recognition* 57 (2016) 179 – 189.
- [16] P. Chen, L. Jiao, F. Liu, J. Zhao, Z. Zhao, S. Liu, Semi-supervised double sparse graphs based discriminant analysis for dimensionality reduction, *Pattern Recognition* 61 (2017) 361 – 378.
- [17] H. Wu, S. Prasad, Semi-supervised dimensionality reduction of hyperspectral imagery using pseudo-labels, *Pattern Recognition* 74 (2018) 212 – 224.
- [18] Y. Zhang, Z. Zhang, J. Qin, L. Zhang, B. Li, F. Li, Semi-supervised local multi-manifold isomap by linear embedding for feature extraction, *Pattern Recognition* 76 (2018) 662 – 678.
- [19] D. Rogers, M. Hahn, Extended-connectivity fingerprints, *Journal of Chemical Information and Modeling* 50 (5) (2010) 742–754, pMID: 20426451.

- [20] M. J. Keiser, B. L. Roth, B. N. Armbruster, P. Ernsberger, J. J. Irwin, B. K. Shoichet, Relating protein pharmacology by ligand chemistry, *Nature biotechnology* 25 (2) (2007) 197–206.
- [21] M. J. Keiser, V. Setola, J. J. Irwin, C. Laggner, A. I. Abbas, S. J. Hufeisen, N. H. Jensen, M. B. Kuijter, R. C. Matos, T. B. Tran, et al., Predicting new molecular targets for known drugs, *Nature* 462 (7270) (2009) 175–181.
- [22] D. Cai, X. He, J. Han, Srda: An efficient algorithm for large-scale discriminant analysis, *IEEE Transactions on Knowledge and Data Engineering* 20 (1) (2008) 1–12.
- [23] L. Zhu, D.-S. Huang, A rayleigh-ritz style method for large-scale discriminant analysis, *Pattern Recognition* 47 (4) (2014) 1698 – 1708.
- [24] D. Cai, X. He, J. Han, Spectral regression: A unified subspace learning framework for content-based image retrieval, in: *Proceedings of the 15th ACM International Conference on Multimedia, MM '07*, ACM, New York, NY, USA, 2007, pp. 403–412.
- [25] M. Belkin, P. Niyogi, V. Sindhwani, Manifold regularization: A geometric framework for learning from labeled and unlabeled examples, *Journal of machine learning research* 7 (Nov) (2006) 2399–2434.
- [26] D. Cai, X. He, J. Han, Semi-supervised discriminant analysis, in: *2007 IEEE 11th International Conference on Computer Vision*, 2007, pp. 1–7.
- [27] J. Yu, Q. Tian, Learning image manifolds by semantic subspace projection, in: *Proceedings of the 14th ACM International Conference on Multimedia, MM '06*, ACM, New York, NY, USA, 2006, pp. 297–306.
- [28] X. He, P. Niyogi, Locality preserving projections, in: S. Thrun, L. K. Saul, B. Schölkopf (Eds.), *Advances in Neural Information Processing Systems 16*, MIT Press, 2004, pp. 153–160.
URL <http://papers.nips.cc/paper/2359-locality-preserving-projections.pdf>
- [29] J. Demmel, *Applied Numerical Linear Algebra*, Society for Industrial and Applied Mathematics, 1997.
- [30] L. Hogben, *Handbook of Linear Algebra*, New York: Chapman and Hall/CRC, 2007.
- [31] M. Muja, D. G. Lowe, Scalable nearest neighbor algorithms for high dimensional data, *IEEE Transactions on Pattern Analysis & Machine Intelligence* 36 (11) (2014) 2227–2240.
- [32] B. Parlett, *The Symmetric Eigenvalue Problem*, Society for Industrial and Applied Mathematics, 1998.
- [33] L. N. Trefethen, D. Bau III, *Numerical Linear Algebra*, Vol. 50, Siam, 1997.
- [34] J. Demmel, B. Kågström, The generalized schur decomposition of an arbitrary pencil $A - \lambda B$ —robust software with error bounds and applications. part i: Theory and algorithms, *ACM Trans. Math. Softw.* 19 (2) (1993) 160–174.
- [35] J. Demmel, B. Kågström, The generalized schur decomposition of an arbitrary pencil $A - \lambda B$ — robust software with error bounds and applications. part ii: Software and applications, *ACM Trans. Math. Softw.* 19 (2) (1993) 175–201.
- [36] A. Tikhonov, V. Y. Arsenin, *Methods for solving ill-posed problems*, John Wiley and Sons, Inc, 1977.
- [37] J. van den Eshof, G. L. Sleijpen, Accurate conjugate gradient methods for families of shifted systems, *Applied Numerical Mathematics* 49 (1) (2004) 17 – 37, *numerical Algorithms, Parallelism and Applications*.
- [38] A. Frommer, P. Maass, Fast cg-based methods for tikhonov-phillips regularization, *SIAM Journal on Scientific Computing* 20 (5) (1999) 1831–1850.
- [39] B. Jegerlehner, Krylov space solvers for shifted linear systems, *arXiv preprint hep-lat/9612014*.
- [40] A. Sciarra, CG algorithm for shifted matrices.
URL <http://code.compeng.uni-frankfurt.de/attachments/178/CG-M.pdf>
- [41] V. N. Vapnik, An overview of statistical learning theory, *IEEE Transactions on Neural Networks* 10 (5) (1999) 988–999.
- [42] T. Cover, P. Hart, Nearest neighbor pattern classification, *IEEE Transactions on Information Theory* 13 (1) (1967) 21–27.
- [43] R. M. McIntyre, R. K. Blashfield, A nearest-centroid technique for evaluating the minimum-variance clustering procedure, *Multivariate Behavioral Research* 15 (2) (1980) 225–238.
- [44] R. Tibshirani, T. Hastie, B. Narasimhan, G. Chu, Diagnosis of multiple cancer types by shrunken centroids of gene

- expression, *Proceedings of the National Academy of Sciences* 99 (10) (2002) 6567–6572.
- [45] D. P. O’Leary, The block conjugate gradient algorithm and related methods, *Linear Algebra and its Applications* 29 (1980) 293 – 322, special Volume Dedicated to Alson S. Householder.
- [46] C. C. Paige, M. A. Saunders, Lsqqr: An algorithm for sparse linear equations and sparse least squares, *ACM Trans. Math. Softw.* 8 (1) (1982) 43–71.
- [47] V. G. Sigillito, S. P. Wing, L. V. Hutton, K. B. Baker, Classification of radar returns from the ionosphere using neural networks, *Johns Hopkins APL Technical Digest* 10 (3) (1989) 262–266.
- [48] M. Lichman, UCI machine learning repository (2013).
URL <http://archive.ics.uci.edu/ml>
- [49] Matlab statistics and machine learning toolbox, the MathWorks, Natick, MA, USA (2018).
- [50] I. Guyon, S. Gunn, A. Ben-Hur, G. Dror, Result analysis of the nips 2003 feature selection challenge, in: *Advances in neural information processing systems*, 2005, pp. 545–552.
- [51] J. N. Weinstein, E. A. Collisson, G. B. Mills, K. R. M. Shaw, B. A. Ozenberger, K. Ellrott, I. Shmulevich, C. Sander, J. M. Stuart, C. G. A. R. Network, et al., The cancer genome atlas pan-cancer analysis project, *Nature genetics* 45 (10) (2013) 1113.
- [52] S. Huang, A. Elgammal, L. Huangfu, D. Yang, X. Zhang, Globality-locality preserving projections for biometric data dimensionality reduction, in: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2014.
- [53] D. P. Kingma, S. Mohamed, D. Jimenez Rezende, M. Welling, Semi-supervised learning with deep generative models, in: Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems* 27, Curran Associates, Inc., 2014, pp. 3581–3589.
URL <http://papers.nips.cc/paper/5352-semi-supervised-learning-with-deep-generative-models.pdf>
- [54] A. P. Bento, A. Gaulton, A. Hersey, L. J. Bellis, J. Chambers, M. Davies, F. A. Krger, Y. Light, L. Mak, S. McGlinchey, M. Nowotka, G. Papadatos, R. Santos, J. P. Overington, The chembl bioactivity database: an update, *Nucleic Acids Research* 42 (D1) (2014) D1083–D1090.
- [55] RDKit: Open-source cheminformatics, <http://www.rdkit.org>, [Online; accessed 18-Januari-2017].

Joris Tavernier received his master's degree in mathematical engineering at KU Leuven and is currently pursuing his doctoral degree. His research interests include numerical linear algebra, high performance computing and machine learning.

Jaak Simm obtained his doctoral degree at Tokyo Institute of Technology in the field of machine learning. His current research interests are bioinformatics, matrix factorization, deep learning and high performance computing for machine learning.

Karl Meerbergen is professor at the department of computer science of KU Leuven (Belgium). His field of research is numerical linear algebra, with focus on large-scale and structured eigenvalue problems, model order reduction, preconditioning, and high performance computing and their use for applications from engineering and data analysis.

Joerg Kurt Wegner works as principal scientist for Janssen Pharmaceutica. He is mainly interested in rational data-driven decision impacting business and research challenges. We are seeking to improve the combinations of multiple types of warehouse and high-dimensional biological data to facilitate decision support via machine learning.

Hugo Ceulemans is Scientific Director Discovery Data Sciences at Janssen, and heads a team that develops and applies machine learning for the prediction of the bioactivity of chemicals integrating various industry scale data sources. A second interest lies in the identification of predictive biomarkers from different sources of single-cell data.

Yves Moreau is a Professor at KU Leuven. His research focuses on developing computational methods for (1) governance of clinical genomic data, including variant discovery and privacy management, (2) data fusion of omics data for gene and variant prioritization in Mendelian and oligogenic disorders, and (3) data fusion for the prediction of drug-target interactions in drug discovery.