Using Discriminative Dimensionality Reduction to Visualize Classifiers

Alexander Schulz, Andrej Gisbrecht, and Barbara Hammer
CITEC centre of excellence, Bielefeld University, Germany

Preprint of the publication [34], as provided by the authors. The final publication is available at link.springer.com.

Abstract

Albeit automated classifiers offer a standard tool in many application areas, there exists hardly a generic possibility to directly inspect their behavior, which goes beyond the mere classification of (sets of) data points. In this contribution, we propose a general framework how to visualize a given classifier and its behavior as concerns a given data set in two dimensions. More specifically, we use modern nonlinear dimensionality reduction (DR) techniques to project a given set of data points and their relation to the classification decision boundaries. Furthermore, since data are usually intrinsically more than two-dimensional and hence cannot be projected to two dimensions without information loss, we propose to use discriminative DR methods which shape the projection according to given class labeling as is the case for a classification setting. With a given data set, this framework can be used to visualize any trained classifier which provides a probability or certainty of the classification together with the predicted class label.We demonstrate the suitability of the framework in the context of different dimensionality reduction techniques, in the context of different attention foci as concerns the visualization, and as concerns different classifiers which should be visualized.

1 Introduction

An increasing complexity of data as concerns its size, dimensionality, or heterogeneity poses strong challenges on automated data analysis. Often, it is no longer possible to specify a dedicated learning task in advance, rather complex settings cause the need of an interactive data analysis: humans interactively process and interpret large, heterogeneous, and high-dimensional data sets, specifying the learning goals and appropriate data analysis tools based on the obtained findings [45, 19, 36]. In this realm, interpretability of the models and data visualization play a major role since they offer an intuitive interface to the data and its analysis tools for the human practitioner [42, 24, 31]. Hence a

trained classifier is no longer judged by its classification accuracy only, rather, the question moves into the focus based on which rationale the classifier makes it decision, what are problematic regions of the classification task where refinement would be valuable, and which data correspond to outliers or noise. Possible remedies to the challenge of model interpretability are offered by relevance learning, feature selection techniques, or sparse model descriptions, for example [27, 42, 31, 32, 18]. Further, visualization plays a major role, since it addresses one of the most powerful senses of humans relying on their astonishing cognitive abilities for visual structure detection, such as grouping or outlier detection.

Visualization of data constitutes a well-investigated research topic with a plethora of different visualization techniques having been proposed in the machine learning context. Besides classical methods such as linear projections offered by principal component analysis or linear discriminant analysis and nonlinear extensions such as the self-organizing map (SOM) or generative topographic mapping (GTM), a variety of (often non-parametric) dimensionality reduction (DR) techniques has been proposed in the last decade, such as t-distributed stochastic neighbor embedding (t-SNE), neighborhood retrieval visualizer (NeRV), or maximum variance unfolding (MVU), see e.g. the articles [39, 40, 24, 43, 16, 11, 24] for overviews on DR techniques. Often, however, these methods are used to visualize a given data set in two dimensions only, not yet answering the question how to visualize the relation of these data in connection to a given classifier. The possibility to also visualize decision boundaries as provided by a given classifier would allow us to extract information beyond the mere classification accuracy of the classifier addressing questions such as: are there potential mis-labelings of data which are observable as outliers, are there noisy data regions where the classification is inherently difficult, are there regions where the flexibility of the classifier is not yet sufficient, what is the modality of single classes, etc.

At present, visualization in the context of classifiers is rather limited: visualization is often restricted to the training procedure, e.g. providing interfaces to set certain parameters or to inspect the area under the curve (AUC) results [17]. Other methods analyse the class topology in a projection space [8] and in the original data space [1]. There exists relatively little work to visualize the underlying classification function itself, including interactive tour methods [5], nomograms [20], linear projection techniques on top of the distance to the decision boundary [29], or graphs emphasizing those reagions where class affiliation changes [26]. Very few nonlinear techniques exist, one notable approach being proposed for the visualization of support vector machine (SVM) using self-organizing maps (SOM), resulting in a technique dubbed support vector machine visualization (SVMV) [44]. Recently, a first step towards a general framework how to visualize a general classifier based on nonlinear dimensionality reduction techniques has been proposed in [33]; the proposed principle allows us to project any given classifier and underlying data points to two dimensions using any dimensionality reduction method which best suits the given data. In this contribution, which constitutes an extension of the approach as presented

in [33], we present and improve this technique and extensively test it as regards different classifiers and dimensionality reduction techniques.

Given a trained classification model, typical user tasks which can be addressed with our framework include the following questions:

1. Is there multimodality in the data, i.e. are there certain classes which fall into multiple modes and how does the classifier handle them?
2. How does the classification model deal with potential outliers in the data?
3. Is there overlap in the data and how do the class boundaries look in those regions?
4. How complex are the class boundaries of the trained model? Do they potentially overfit the data?
5. If the model contains interpretable components such as data prototypical instances: What is their location in relation to the data and how do they contribute to the class boundary?

With our experiments, we show exemplarily how these questions can be addressed within our framework.

Generally speaking, the general framework for classifier visualization as proposed in this article relies on an identification of a given data manifold and a two dimensional projection. A bijective mapping between the original data manifold and a low dimensional projection enables us to directly map the decision boundaries of a given classifier in the data manifold as the set of points with output zero as concerns their distance to the decision boundary. This naive approach, however, has a few drawbacks: (i) Many powerful DR methods do not provide an explicit mapping, rather they provide a nonparametric projection of the given data points only. Relying on ideas as proposed in [13], we propose a general technique to extend non-parametric projections to explicit parametric forms, if necessary. (ii) It is infeasible to sample the usually high dimensional feature space; still we have to somehow detect the decision boundaries of a given arbitrary classifier to provide its visualization in two dimensions. We solve this problem by a trick: we sample in the low dimensional projection space rather than the feature space itself, and use the inverse projection of these sampled data to determine the decision boundary in the original data manifold. (iii) Unless the data manifold is intrinsically two-dimensional, however, there cannot exist a bijection of the data manifold and a low dimensional projection, hence no valid back-projection. More generally, the question what to visualize in a reasonable way is not clear due to the usually high data dimensionality. Hence the task of classifier visualization is essentially ill-posed. We will rely on discriminative DR to circumvent this problem.

More precisely, we will point out the necessity to integrate auxiliary information to the DR technique to make the DR problem well-posed. For classifier visualization, we do not want to visualize all aspects of the data, rather we are interested in the positioning of the data as concerns the class boundaries. As

pointed out in [43, 13], there exists a very elegant generic technique to enhance many DR techniques with auxiliary information as provided by class labels: instead of the original data and its underlying distance measure, we rely on a distance measure which is induced by the Fisher information metric for the given class labeling. This way, those aspects of the data are emphasized which are relevant for the given classifier rather than directions parallel to the classifier's decision boundaries. Since decision boundaries can often be described as the zero set of a function with a smooth parameterization, they correspond to a one-co-dimensional topological manifold. Hence the data set measured in the Fisher metric, which essentially restricts to directions orthogonal to the boundary, is locally approximately one dimensional (in the vicinity of class boundaries), and the task to visualize such data in two dimensions is well defined.

We will elaborate on this issue in the following and demonstrate the beneficial effect of taking auxiliary information as provided by class labels into account. Actually, there exist two different reasonable labelings if we address the task of classifier visualization: the ground truth which is a labeling provided by the data, and the labeling which is provided by the trained classifier. In particular for classifiers with low accuracy, a scenario where classifier inspection might be particularly interesting, these labelings do not coincide. In such settings, we would like to 'see' what causes the problems of the classifier. We will discuss that both possible labelings provide different classifier visualizations, focusing on different aspects of the setting and different insights into the classifier behavior. We will demonstrate this aspect in the following in examples. In summary, a powerful classifier visualization framework results which we will test for different classifier types including a support vector machine, a learning vector quantization scheme, and a decision tree classifier, and different DR techniques, including t-SNE, SOM, GTM, and MVU.

All in all, we propose

- a general framework for the visualization of classifiers enabling to visualize any classifier that provides a certainty measure of his decision. We demonstrate this for a SVM, a LVQ and a decision tree classifier.
- We highlight the necessity to use discriminative DR which locally emphasizes the relevant dimensions and such guides the projection to focus on those aspects of the data that are relevant for classification.
- We present examples for things that can be detected when visualizing high-dimensional classifiers.

The organization of this article is as follows: First, we shortly review popular dimensionality reduction methods and their extension towards explicit mapping prescriptions on the one hand and the incorporation of auxiliary information by means of the Fisher metric on the other hand. Afterwards, we present a general framework how to visualize a given classifier. We demonstrate the framework in a few illustrative examples, before testing its behavior as concerns different DR techniques, different choices of auxiliary label information, and different classification schemes to be visualized.

2 Dimensionality Reduction

Dimensionality reduction techniques are concerned with the following problem: given data points $\mathbf{x} \in X = \mathbb{R}^d$ in a high dimensional feature space, how to map these points to low dimensional counterparts $\pi(\mathbf{x}) = \mathbf{y} \in Y = \mathbb{R}^2$ in the two-dimensional plane such that as much structure as possible is preserved. As described in the recent overview [11] for example, one can distinguish parametric and non-parametric DR techniques.

Parametric techniques specify a functional form $\pi_{pm} : X \to Y, \mathbf{x} \mapsto \mathbf{y} = \pi_{pm}(\mathbf{x})$ (we employ the subscript $_{pm}$ to emphasize that the mapping is parametric) with free parameters which determine the form of the mapping. Given a set of examples $\mathbf{x}_1, \ldots, \mathbf{x}_n$, training takes place by optimizing these parameters such that the examples are mapped as accurately as possible. Popular methods include:

- *Principle component analysis (PCA)* defines $\pi_{pm}(\mathbf{x}) = \mathbf{W}^t\mathbf{x}$ as a linear mapping with $\mathbf{W} \in \mathbb{R}^{d\times 2}$. The linear parameters $\mathbf{W}$ are determined such that the squared reconstruction error of the given data is minimized, resulting in an eigenvalue problem with an explicit solution. Due to the particularly simple form, an approximate inverse is offered by the mapping $\pi_{pm}^{-1} : \mathbf{y} \mapsto \pi_{pm}^{-1}(\mathbf{y}) = \mathbf{W}\mathbf{y}$.

- One popular non-linear alternative is offered by the *self-organizing map (SOM)*, which maps the data to a two-dimensional regular grid (consisting of notes $\mathbf{c}_k$) by means of a winner-takes-all function, where each position j in the grid is associated with one position $\mathbf{w}_j \in \mathbb{R}^d$ in the feature space. Training takes place by Hebbian learning, thereby also respecting the neighborhood of the lattice. This way, a locally constant projection of the data to two dimensions, the lattice position of the winner, is defined: $\pi_{pm}(\mathbf{x}) = \mathbf{c}_k$ with $k = \arg\min_j d(\mathbf{x}, \mathbf{w}_j)$. By means of local interpolation, this mapping can easily be turned into a smooth function. By construction, an inverse mapping is offered by mapping a position j in the lattice to the position $\mathbf{w}_j$ of the associated place in the feature space: $\pi_{pm}^{-1}(\mathbf{y}) = \mathbf{w}_k$ with $k = \arg\min_j d(\mathbf{y}, \mathbf{c}_j)$. Again, this simple function is locally constant, but can easily be turned into a smooth mapping by means of local interpolation.

- We will also consider the *generative topographic mapping (GTM)* as a probabilistic counterpart of SOM. Essentially, GTM relies on data being generated by a constraint mixture of Gaussians. The centers of the Gaussians are generated by a smooth mapping from regular lattice positions in a two dimensional latent space which can be used for data visualization. GTM training can be derived from a maximization of the data log likelihood function. Due to its probabilistic modeling which allows to compute probabilities of lattice points having generated a given data, a smooth mapping of data to its low dimensional projection $\pi_{pm}(\mathbf{x}) = \sum_k \mathbf{c}_k p(\mathbf{c}_k|\mathbf{x})$

and vice versa $\pi_{pm}^{-1}(\mathbf{y}) = \sum_j \mathbf{w}_j \phi(\mathbf{y})$ (where the basis function ϕ are often Gaussian kernels with predefined centers) is directly provided by GTM.

In contrast to these parametric techniques, non-parametric mappings rely on a mapping of a given set of data $\mathbf{x}_i$ to their low dimensional counterparts $\mathbf{y}_i$ only, but no explicit functional form $\pi_{pm} : X \to Y$ is priorly specified. Training takes place by tuning the projections $\mathbf{y}_i$ such that a certain criterion is optimized: usually, the structure in the data space as defined by $\mathbf{x}_i$ and the structure of the projections $\mathbf{y}_i$ are measured and compared using some suitable cost function. An overview about a generic formalization of different popular non-parametric DR techniques as cost function optimization can be found in [3]. We will exemplarily investigate the following three popular techniques:

- *Isomap* is based on the objective to preserve distances in the data space and the projection space as measured in a least squares error. Thereby, the distances in the original data space are taken along the data manifold as so-called geodesic distances. Since the exact manifold is not available, a simple numeric approximation scheme is taken: local neighborhoods of a given data point to its closest k neighbors are approximated by the euclidean distance; on a global scale, shortest paths in this neighborhood graph are considered.

- *Maximum variance unfolding (MVU)* relies on a similar idea, by first constructing a local neighborhood graph connecting every point to its k closest exemplars. Then, projection takes place by unfolding the data as much as possible in two dimensions (i.e. maximizing its covariance) thereby respecting the neighborhood structure of the constructed graph.

- *T-distributed stochastic neighbor embedding (t-SNE)* defines local neighborhoods in a probabilistic sense by using Gaussians based on pairwise distances in the feature space and student-t distributions induced by euclidean distances in the projection space. Training takes place by a minimization of the error in between these distributions as measured by the Kullback Leibler divergence. Unlike MVU, the resulting cost function can have local optima resulting in different possible visualizations.

2.1 Dimensionality reduction mapping

While parametric mappings provide an explicit functional form, non-parametric mappings such as t-SNE, MVU, or Isomap have in common that no direct out-of-sample extension is available. In [13] a general way how to extend these prescriptions to a parametric form has been proposed by means of an interpolation by Gaussian kernels. We specify a functional form π_{pm} of the mapping as follows:

$$\mathbf{x} \mapsto \pi_{pm}(\mathbf{x}) = \frac{\sum_j \boldsymbol{\alpha}_j k_j(\mathbf{x}, \mathbf{x}_j)}{\sum_l k_l(\mathbf{x}, \mathbf{x}_l)} \tag{1}$$

where $\boldsymbol{\alpha}_j \in Y$ are parameters corresponding to points in the projection space and the data $\mathbf{x}_j$ are taken as a fixed sample, usually j runs over a small subset X' sampled from the data $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. k is the Gaussian kernel parameterized by the bandwidth $\sigma_j^{\mathbf{x}}$:

$$k_j(\mathbf{x}, \mathbf{x}_j) = \exp(-0.5\|\mathbf{x} - \mathbf{x}_j\|^2/(\sigma_j^{\mathbf{x}})^2) \tag{2}$$

The idea is to determine the parameters of this mapping such that the data $\mathbf{x}_i$ and their projections $\mathbf{y}_i$ obtained with the considered projection technique are matched as far as possible. Note that the mapping has a generalized linear form such that training can be done in a particularly simple way provided a set of samples $\mathbf{x}_i$ and $\mathbf{y}_i$ is available. The parameters $\boldsymbol{\alpha}_j$ can be analytically determined as the least squares solution of the mapping: Assume $\mathbf{A}$ is the matrix of parameters $\boldsymbol{\alpha}_j$, $\mathbf{K}$ is the normalized Gram matrix with entries

$$(\mathbf{K})_{i,j} = k_j(\mathbf{x}_i, \mathbf{x}_j)/\sum_l k_l(\mathbf{x}_i, \mathbf{x}_l) \tag{3}$$

and $\mathbf{Y}$ denotes the matrix of projections $\mathbf{y}_i$, Then, a minimum of the least squares error

$$\sum_i \|\mathbf{y}_i - \pi_p(\mathbf{x}_i)\|^2 \tag{4}$$

with respect to the parameters $\boldsymbol{\alpha}_j$ has the form

$$\mathbf{A} = \mathbf{Y} \cdot \mathbf{K}^{-1} \tag{5}$$

where $\mathbf{K}^{-1}$ refers to the pseudo-inverse of $\mathbf{K}$.

The bandwidth $\sigma_i^{\mathbf{x}}$ of the mapping constitutes a critical parameter since it determines the smoothness and flexibility of the resulting kernel mapping. We use a principled approach to determine this parameter as follows: $\sigma_i^{\mathbf{x}}$ is chosen as a multiple of the distance of $\mathbf{x}_i$ from its closest neighbor in X', where the scaling factor is typically taken as a small positive value. We determine this factor automatically as the smallest value in such a way that all entries of $\mathbf{K}$ are within the range of representable numbers (respectively a predefined interval). This technique allows us to extend any given non-parametric mapping to an explicit parametric form.

2.2 Discriminative dimensionality reduction based on the Fisher metric

We are interested in mapping a given set of data points $\mathbf{x}_i$ and the classification boundary of an underlying classifier f to low dimensions. This problem is in general ill-posed since data dimensionality is typically larger than two, hence a faithful projection of data to two dimensions is not possible. More severely, the decision boundary as defined by a given classifier is also high dimensional and cannot be visualized together with the data points in a two-dimensional projection. Hence the actual visualization of the data points depends on the

chosen projection paradigm, and it is not clear how to map the class boundaries to this visualization in a meaningful way.

Therefore, we propose to consider a variation of DR techniques which takes auxiliary information into account, this way specifying which parts of the data should be visualized. This severely reduces the relevant dimensionality of the data and class boundaries and hence enables a meaningful visualization. Dealing with classifiers, we will rely on discriminative DR techniques which add given label information to the setting. The task is to visualize only those aspects of the data which are relevant for the given class labeling – hence only the information which is also relevant for the given classifier is taken into account.

A variety of different discriminative DR techniques has been proposed, such as Fisher's linear discriminant analysis (LDA), partial least squares regression (PLS), informed projections [7], global linear transformations of the metric [14, 4], or kernelization of such approaches [25, 2]. A general idea which we will use in our approach is to locally modify the metric [28, 12] by defining a Riemannian manifold which takes into account auxiliary information of the data and which measures the effect of data dimensions in the feature space on this auxiliary information. This modified metric can then be plugged into any DR technique which relies on distances only (such as the ones specified above). This can be done by replacing the commonly used Euclidean distance by the Fisher distances, as has been done for the SOM and t-SNE in [28, 13].

Basic definition

We assume that class information c assigned to $\mathbf{x}$ is available where c is one of a finite number of different classes. This information can locally be incorporated into the distance computation by setting the quadratic form of the tangential space at $\mathbf{x}$ as $\mathbf{a}^\top \mathbf{J}(\mathbf{x})\mathbf{b}$ where $\mathbf{a}$ and $\mathbf{b}$ are elements of the tangential space of the data manifold at $\mathbf{x}$ and $\mathbf{J}(\mathbf{x})$ is the local Fisher information matrix

$$\mathbf{J}(\mathbf{x}) = E_{p(c|\mathbf{x})}\left\{\left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x})\right)\left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x})\right)^\top\right\}. \tag{6}$$

Thereby, $p(c|\mathbf{x})$ denotes the probability of the class information c conditioned on $\mathbf{x}$. This local positive semidefinite bilinear form has the effect that only those dimensions are relevant for the distance computation on the data manifold which affect the given label c. Dimensions which are locally irrelevant do not contribute. These local measures can be extended to the entire manifold by taking minimum path integrals along the manifold: the distance of two points $\mathbf{x}$ and $\mathbf{x}'$ on the manifold is measured as

$$d_M(\mathbf{x}, \mathbf{x}') = \inf_P \int_0^1 \sqrt{P'(t)^\top J(P(t))P'(t)}dt \tag{7}$$

where the infimum is over all differentiable paths $P : [0, 1] \to X$ with $P(0) = \mathbf{x}$ and $P(1) = \mathbf{x}'$, the integral describing the standard path lengths of P measured using the Fisher information matrix. This integral is well defined since the derivative $P'(t)$ is element of the tangential space.

Approximation of the integral

This integral, however, is usually computationally intractable, hence approximations are used; a description of approximations with their corresponding advantages and disadvantages can be found in [28]. A popular approach is to limit paths to the straight line from $\mathbf{x}$ to $\mathbf{x}'$ only, and to approximate the integral by T piecewise constant terms induced by equidistant points on the line from $\mathbf{x}$ to $\mathbf{x}'$. Assume $\mathbf{x}_t = \mathbf{x} + (t-1)/T \cdot (\mathbf{x}' - \mathbf{x})$. Additionally approximating the integral as the sum, the exact distance on the manifold d_M can be approximated by

$$d_T(\mathbf{x}, \mathbf{x}') = \sum_{t=1}^{T} \sqrt{(\mathbf{x}_{t+1} - \mathbf{x}_t)^\top J(\mathbf{x}_t)(\mathbf{x}_{t+1} - \mathbf{x}_t)}. \tag{8}$$

The Fisher information matrix can be directly included into all DR techniques as specified above by substituting the distances by these Fisher distances.

Approximation of the probabilities

A central part of this modified distance computation consists in the estimation of the probability $p(c|\mathbf{x})$ of information c given a data point $\mathbf{x}$. In our setting, there are two essentially different possibilities how to choose this information:

(a) We can use the given class labels $c_i := l_i$ for data point $\mathbf{x}_i$, respectively, as provided in the training set. This choice emphasizes the given 'ground truth' of the data. In consequence, the visualization of the classifier will show in which regions the obtained classification is simple or complex as compared to this ground truth.

(b) We can use the labeling as provided by the trained classifier $c := f(\mathbf{x})$. This choice emphasizes aspects of the data which are regarded by the classifier as interesting. Hence, those aspects of the data are visualized which influence the trained classification; as an example one can detect regions of the data where points are regarded as virtually identical by the classifier.

Apart from the different semantic meaning, this choice has consequences on the possibilities how to compute the probability $p(c|\mathbf{x})$. The Fisher matrix is based on the local change of the probability distribution $p(c|\mathbf{x})$, the latter of which is usually unknown and needs to be approximated. A common way to do this is to use the Parzen window non-parametric estimator as proposed in [28]. Essentially, computation takes place by estimating class probabilities as

$$\hat{p}(c|\mathbf{x}) = \frac{\sum_i \delta_{c=c_i} \exp(-0.5\|\mathbf{x} - \mathbf{x}_i\|^2/(\sigma^p)^2)}{\sum_j \exp(-0.5\|\mathbf{x} - \mathbf{x}_j\|^2/(\sigma^p)^2)}. \tag{9}$$

The Fisher information matrix based on the Parzen window estimator becomes

$$\mathbf{J}(\mathbf{x}) = \frac{1}{(\sigma^p)^4} E_{\hat{p}(c|\mathbf{x})} \left\{ \mathbf{b}(\mathbf{x}, c)\mathbf{b}(\mathbf{x}, c)^T \right\} \tag{10}$$

where

$$\mathbf{b}(\mathbf{x}, c) = E_{\xi(i|\mathbf{x},c)}\{\mathbf{x}_i\} - E_{\xi(i|\mathbf{x})}\{\mathbf{x}_i\} \tag{11}$$

$$\xi(i|\mathbf{x}, c) = \frac{\delta_{c,c_i} \exp(-0.5\|\mathbf{x} - \mathbf{x}_i\|^2/(\sigma^p)^2)}{\sum_j \delta_{c,c_j} \exp(-0.5\|\mathbf{x} - \mathbf{x}_j\|^2/(\sigma^p)^2)} \tag{12}$$

$$\xi(i|\mathbf{x}) = \frac{\exp(-0.5\|\mathbf{x} - \mathbf{x}_i\|^2/(\sigma^p)^2)}{\sum_j \exp(-0.5\|\mathbf{x} - \mathbf{x}_j\|^2/(\sigma^p)^2)} \tag{13}$$

E denotes the empirical expectation, i.e. weighted sums with weights depicted in the subscripts. If large data sets or out-of-sample extensions are dealt with, a subset of the data only is usually sufficient for the estimation of $\mathbf{J}(\mathbf{x})$.

This yields a correct estimation of the probability density but is computationally expensive, i.e. $\mathcal{O}(N^2)$ for N data points. For finite data sets, the result depends on the chosen bandwidth[1] σ^p of the estimator. The resulting estimator is differentiable and the derivatives are reported in [28], for example.

As an alternative, provided the class labels $f(\mathbf{x})$ given by a function are of interest, it is often possible to rely on the explicit functional form f as provided by the classifier if the latter yields probabilities for the class labels.

2.3 Inverse dimensionality reduction mapping

Having extended a non-parametric projection to an explicit mapping prescription, the question occurs whether an inverse mapping can also be determined. An explicit inverse would equip us with a bijective identification of the data manifold and its projection, eventually enabling us to map decision boundaries of classifiers together with the data itself.

Note that, in general, a direct inversion of a projection mapping $\pi : X \to Y$ is not possible since the projection π is usually many to one. Many parametric techniques nevertheless provide explicit inverse mappings which find a suitable inverse of the projections to the data manifold, such as discussed for PCA, SOM, and GTM above. For non-parametric mappings a picewise linear mapping is developed in [9], where the parameters have to be recomputed for each point. We propose a similar trick as before which is based on a similar interpolation idea as in [9] but being fixed and continuous.

We assume that points $\mathbf{x}_i \in X$ and projections $\pi(\mathbf{x}_i) = \mathbf{y}_i \in \mathbb{R}^2$ are available. For an inverse projection, we assume the following functional form

$$\pi^{-1} : Y \to X, \mathbf{y} \mapsto \frac{\sum_j \boldsymbol{\beta}_j k_j(\mathbf{y}, \mathbf{y}_j)}{\sum_l k_l(\mathbf{y}, \mathbf{y}_l)} \tag{14}$$

where $\boldsymbol{\beta}_j \in X$ are parameters of the mapping and $k_j(\mathbf{y}, \mathbf{y}_j) = \exp(-0.5\|\mathbf{y} - \mathbf{y}_j\|^2/(\sigma_j^{\mathbf{y}})^2)$ constitutes a Gaussian kernel with bandwidth determined by $\sigma_j^{\mathbf{y}}$. The bandwidth is determined in the same way as for the forward projection. Summation is over a random subset Y' of the given data projections $\mathbf{y}_i = \pi(\mathbf{x}_i)$,

[1] We use the estimator $\hat{h}_{rot}$ provided in the literature to specify this parameter, see e.g. [37].

or over codebooks resulting from a previously run vector quantization on the $\mathbf{y}_i$.

Depending on the choice of these data and the bandwidth, the problem of determining the parameters $\boldsymbol{\beta}_j$ can constitute an underdetermined system. One particular problem is given by the fact that the inverse mapping π^{-1} of π is not well defined: since the intrinsic data dimensionality is usually larger than two, the inverse $\mathbf{x}$ of a given projection $\mathbf{y}$ is ambiguous. Data dimensions which are not relevant for the projection π can be arbitrary. Thus a challenge is the task to find a suitable inverse projection of π which tolerates such invariances.

We solve this problem by optimization of the following costs with respect to the parameters $\boldsymbol{\beta}_j$

$$E = \sum_i \left(d_1 \left(\mathbf{x}_i, \pi^{-1}(\mathbf{y}_i) \right)^2 \right) = \sum_i \left(\mathbf{x}_i - \pi^{-1}(\mathbf{y}_i) \right)^\top \mathbf{J}(\mathbf{x}_i) \left(\mathbf{x}_i - \pi^{-1}(\mathbf{y}_i) \right) \quad (15)$$

where the matrix $\mathbf{J}$ refers to the Fisher information matrix. In contrast to a standard Euclidean error function, this function has the advantage that those dimensions in X which are locally relevant for the classification are emphasized. Invariances of the projection π due to the given class labeling are tolerated in the inverse projection. We utilize the distance d_T with $T = 1$ in order to save computational time. This local approximation works usually well since in the course of optimization the points $\mathbf{x}_i$ and $\pi^{-1}(\mathbf{y}_i)$ will get close to each other. Minimization of these costs with respect to the parameters $\boldsymbol{\beta}_j$ takes place by gradient descent.

3 General Framework

In the last section we have reviewed the following important aspects of dimensionality reduction:

- Parametric extension of non parametric dimensionality reduction techniques: We can optimize a projection of the data and map additional data points efficiently after the training.
- Supervised projections based on the Fisher metric: We can obtain a supervised projection of the data which focuses particularly on the label information.
- Parametric inverse dimensionality reduction: Having obtained a projection of the data, we can investigate interesting regions in the projection space by mapping data points back to the original data space.

In this section, we are in the position to put these pieces together towards a general framework for classifier visualization. We assume the following scenario: a data set including points $\mathbf{x}_i \in X$ is given. Every data point is labeled with $l_i \in L$ belonging to a finite set of different labels L. In addition, a classifier $f : X \to L$ has been trained on the given training set, such as a support vector

machine, a classification tree or a learning vector quantization network. The standard way to evaluate the performance of the classifier f is by inspecting the classification error of the function on the given training set or a hold out test set. This gives us an indication whether the classifier is nearly perfect, corresponding to 100% accuracy, or whether errors occur. However, the classification error does not give us a hint about the geometric distribution of the errors (are they equally distributed in the space, or do they accumulate on specific misclassified regions), whether errors are unavoidable (due to overlapping regions of the data or outliers), whether the class boundaries are complex (e.g. due to multiple modes in the single classes), etc. A visualization of the given data set and the classifier would offer the possibility to visually inspect the classification result and to answer such questions. We propose a general framework how to visualize a classifier and a given data set such as the training set of the classifier.

We extend DR methods as introduced above to also visualize the class boundaries of the classifier f. For this purpose, we assume that the label $f(\mathbf{x})$ is accompanied by a nonnegative real value $r(\mathbf{x}) \in \mathbb{R}$ which scales with the distance from the closest class boundary. As an example this could be the activation of a linear classifier such as SVM, or it could be the class probability if a probabilistic classifier such as robust soft learning vector quantization or a Bayesian classifier is considered. Note that most classifiers offer a natural way to equip the mere class output with a smooth value which correlates to the distance to the decision boundary. Since we do not assume a specific scaling of this output, any such value will do.

3.1 Naive approach

Assuming a nonlinear dimensionality reduction method is given, a naive approach to classifier visualization could be like follows:

- Sample the full data space X by points $\mathbf{z}_i$.
- Project these points nonlinearly to two dimensional points $\pi(\mathbf{z}_i)$ using some nonlinear dimensionality reduction technique.
- Display the data points $\pi(\mathbf{x}_i)$ and the contours induced by the sampled function $(\pi(\mathbf{z}_i), r(\mathbf{z}_i))$, the latter approximating the boundaries of the classifier.

This simple method, however, fails unless X is low dimensional because of two reasons:

- Sampling X sufficiently requires an exponential number of points, hence it is infeasible for high dimensional X.
- It is impossible to map a full high dimensional data set faithfully to low dimensions, hence topological distortions are unavoidable when projecting the class boundaries.

The problem lies in the fact that this procedure tries to visualize the class boundaries in the full data space X. It would be sufficient to visualize only those parts of the boundaries which are relevant for the given training data $\mathbf{x}_i$ and the underlying classification behavior as measured using the Fisher metric.

3.2 Our proposed approach

Therefore, we propose to sample in the projection plane instead of the original data manifold, and we propose to use a discriminative DR technique to make the problem of data projection well-posed (since the Fisher metric makes the data space locally low-dimensional the projection can find a compromise, at least locally). Together with the techniques presented in the last chapter, this leads to the following feasible procedure for classifier visualization (see Fig. 1):

- Project the data $\mathbf{x}_i$ using a nonlinear discriminative DR technique (for instance utilizing Fisher distances calculated with (8)) guided by the labels l_i leading to points $\pi(\mathbf{x}_i) \in Y$.
- Sample the projection space Y in a regular grid leading to points $\{\mathbf{z}'_i\}_{i=1}^n$. Determine points $\mathbf{z}_i$ in the data space X which are projected to these points $\pi(\mathbf{z}_i) \approx \mathbf{z}'_i$ by training an inverse mapping π^{-1} (minimize (15)) for these point, relying on the Fisher metric to make it well posed.
- Visualize the training points $\mathbf{x}_i$ together with the contours of the given classifier which are induced by the sampled pairs $(\mathbf{z}'_i, r(\mathbf{z}_i))$ and $(\mathbf{z}'_i, f(\mathbf{z}_i))$, where the function value r is provided by the classifier f.

Unlike the naive approach, sampling takes place in $\mathbb{R}^2$ only and, thus, it is feasible. Further, only those parts of the space X are considered which correspond to the observed data manifold $\mathbf{x}_i$, i.e. the class boundaries are displayed only as concerns these training data. Note that two different labelings can be used in this context: the labeling as provided by the function f, or the labeling as given by labels from the training set, i.e. the ground truth. Depending on which labeling is used to determine π and its inverse π^{-1}, we obtain a visualization of the classifier which respects invariances of the underlying ground truth labeling, or which respects invariances of the observed classifier, allowing different insights into its behavior as we will demonstrate in the following. Per default, we will refer to the labels of the classifier unless stated otherwise.

3.3 Evaluation measure

In order to evaluate the quality of a visualization of a classifier, we employ the following scheme:

- Use the trained classification model to assign to each point $\mathbf{x}_i$ a label $l_i = f(\mathbf{x}_i)$ and a certainty value $r_i = r(\mathbf{x}_i)$.

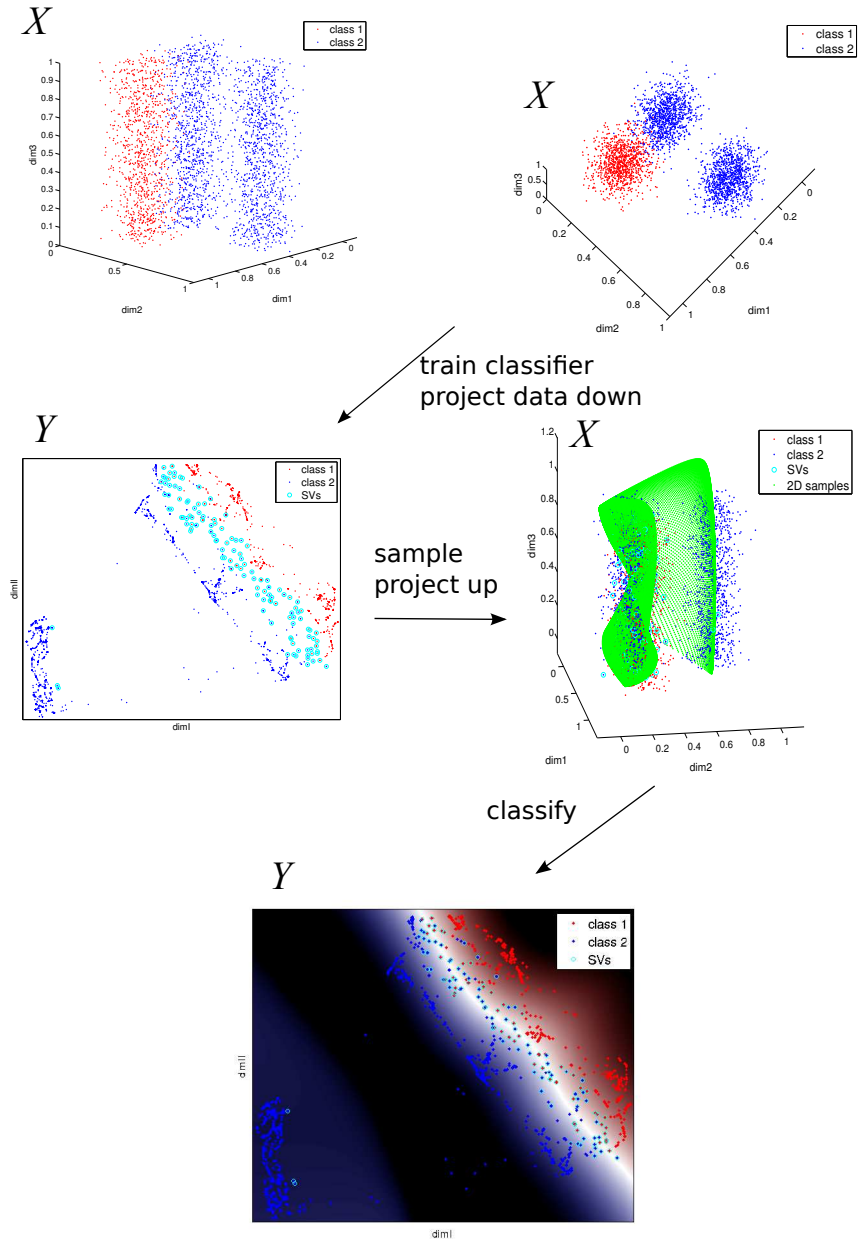

Figure 1: Principled procedure how to visualize a given data set and a trained classifier. The example displays a SVM trained in 3D.

- Utilize the visualized classifier (as described by the classified samples $\mathbf{z}'_i$) in order to assign to each low-dimensional point $\mathbf{y}_i$ a class label l'_i and a certainty value r'_i. More precisely, assume that the function $\text{findNN}(\mathbf{y}_i, \{\mathbf{z}'_j\}_{j=1}^n)$ finds the nearest neighbor of the point $\mathbf{y}_i$ among the points $\{\mathbf{z}'_j\}_{j=1}^n$. Then $l'_i = f\left(\pi^{-1}\left(\text{findNN}(\mathbf{y}_i, \{\mathbf{z}'_j\}_{j=1}^n)\right)\right)$ and $r'_i = r\left(\pi^{-1}\left(\text{findNN}(\mathbf{y}_i, \{\mathbf{z}'_j\}_{j=1}^n)\right)\right)$.
- Calculate and return the percentage of the accordance of l_i and l'_i. Further, estimate the accordance of r_i and r'_i with the Pearson correlation.

This procedure provides two quality estimates. The first measure is based on the labels and describes how many points lie on the same side of a class boundary as compared to their original positions. Such it yields a measure to how far the visualization of the classifier can be trusted. The second measure evaluates to what extend the estimated contours of the classifier are correct.

4 Experiments

In this section we demonstrate our approach for various data sets and scenarios. In the first experiments, we exemplarily visualize SVMs while later we also apply our approach to probabilistic LVQ models and classification trees.

In 4.1, we utilize two data sets addressing the user cases 1 and 2 and we investigate the influence of DR techniques on the visualization of classifiers. First, we apply PCA (being the most simple and straight forward method) and show it's limitations. Further, we compare the SOM (suggested in the literature) to non-parametric projections. In section 4.2, we perform a sanity check by visualizing classifiers based on different labellings of the same data. Here, we also address the previously specified user case 3. In section 4.3 we empirically analyze the effect of including supervised information and in section 4.4 we consider two types of supervision: given by the original labeling and by the labels assigned to by the classifier. Additionally, we investigate properties of the prototype based classifier, thus addressing user case 5. In the last section 4.5 we visualize two other classifiers: a classification tree and a Robust Soft LVQ model.

Now follows a short description of the classifiers we utilize in our experiments.

- The Support Vector Machine (SVM) [41] trains an optimal linear classifier in a feature space. The descision function has the form $f(\mathbf{x}) = \text{sign}\left(\mathbf{w}^\top \phi(\mathbf{x}) + b\right)$, where $\mathbf{w}$ and b are optimized by the method. For the SVM, we can directly compute the distance from the descision boundary by $r(\mathbf{x}) = (\mathbf{w}^\top \phi(\mathbf{x}) + b)/\sqrt{\mathbf{w}^\top \mathbf{w}}$.

  Originally, the SVM solves only two-class problems. If more classes are available we employ a "one versus one" classification scheme (i.e. training a two-class SVM for each pair of two classes) with a subsequent majority vote for classification. For this approach, the class boundaries of the resulting SVM mostly coincide with the boundaries of the two-class SVMs, which is not the case for the "one versus all" scheme (see [23] for more

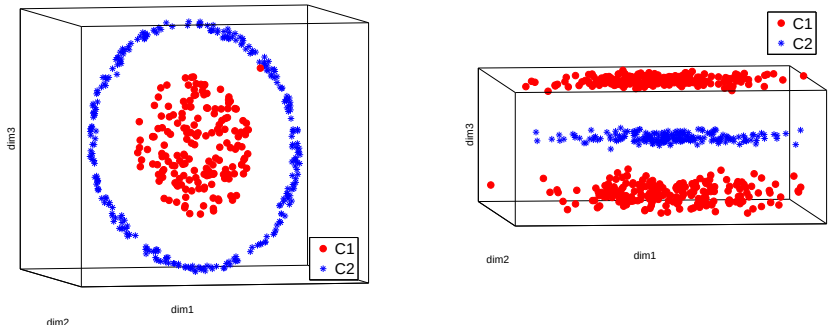

Figure 2: Toy data set 1 (left). Note the potential outlier point of class 1 in the upper right part of the data set. The right image shows toy data set 2.

details). Hence, in the case of more then two classes we specify the overall value $r(\mathbf{x})$ to be the minimum distance of $\mathbf{x}$ to the class boundary of each two-class SVM containing the class of $\mathbf{x}$. This "one versus one" scheme is also employed by the LIBSVM toolbox [6] which we utilize in the following.

- The Robust Soft LVQ (RSLVQ) classification scheme [35] learns a prototype based probabilistic model for the data such that the likelihood of correct classification is optimized. A Gaussian mixture is employed as the probabilistic model, which directly provides probability estimates for $r(\mathbf{x})$.
- Classification Trees divide the input space into several regions, thereby using axis aligned decision boundaries. They typically work in a greedy way, subdividing regions if they contain too many points form different classes. For this splitting step of cells we use the Gini index. See [22] for an review of Classification Trees. A probabilistic output for the certainty of the classification can be provided using the distribution of data points inside such cells.

4.1 Toy data examples with different DR mappings

We utilize two three-dimensional artificial data sets in order to provide an example for our approach and to demonstrate the user tasks 1,2 and 4 as defined in the introduction. Both data sets consist of two classes and are shown in Fig. 2. For both data sets we train SVMs.

Data set 1 (left) is intrinsically two-dimensional and consists of a plate surrounded by a circle. Each object represents a class. Note that one point of class 1 lies apart from the other samples of that class and close to samples of class 2. We train two SVM models for this data set: a complex one with small RBF kernels and one with larger RBF kernels. Using PCA for dimensionality reduction we obtain the two visualizations of the data together with the underlying

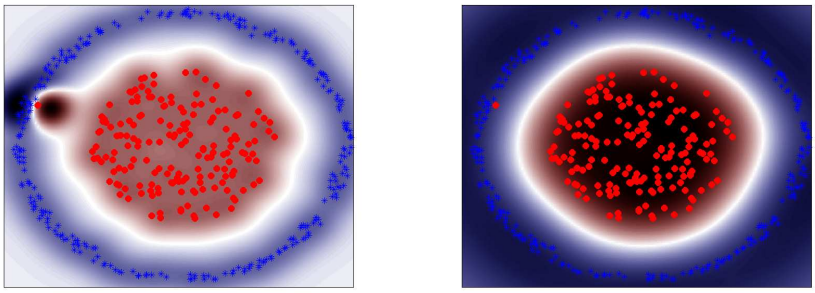

Figure 3: Visualization of two different SVMs trained on data set 1 with PCA.

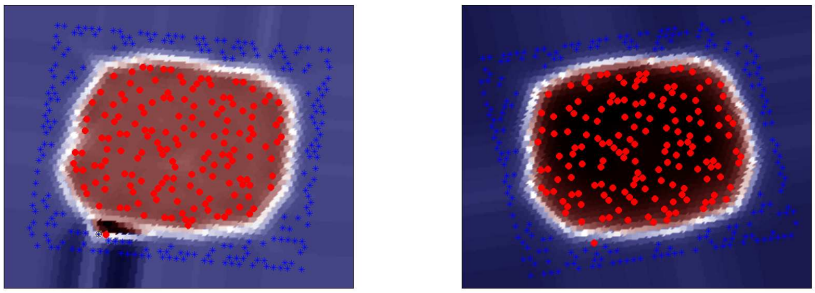

Figure 4: Visualization of two different SVMs trained on data set 1 with SOM.

classifier shown in Fig. 3.

The left image depicts the complex SVM. It can be directly observed that the class boundaries are rather complex and that the outlier is classified correctly, yielding potential generalization disadvantages. The right hand side of the figure shows the less complex SVM. Here, the rather smooth class boundaries are directly visible and a good generalization can be expected due to the large margin - at the cost of one miss-classification, however. Observing the complexity of the class boundaries might be very interest, for instance if one is addressing the bias variance dilemma of the classifier. This is an example how the user tasks 2 and 4 can be addressed with our proposed framework.

SVMV [44] uses the SOM for dimensionality reduction and yields a very similar result as can be seen for both SVMs in Fig. 4. The two SVMs can be distinguished here as well, although, the margin of the classifier is not displayed so well. This is an effect of the SOM since it is related to vector quantization and, hence, usually doesn't place nodes in regions without data (except it has to due to the neighborhood function, which is the reason for the class boundary being shown in this example at all).

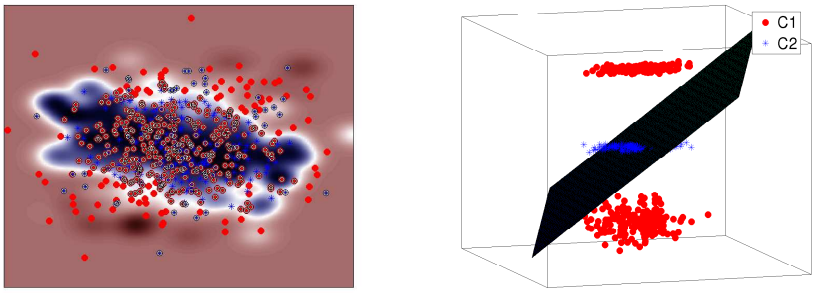

Figure 5: Visualization of data set 2 with PCA (left) and the according inverse projected samples (right).

The quality of the visualization as measured by the accordance of the class labels assigned to points by the classifier and the labeling that would be assigned to by the visualization of the classifier amounts to 100% for all visualizations shown in Figures 3 and 4. The evaluation of the contours describing the certainty of the classifier amounts to over 0.99.

Data set 2 consists of three clusters. One cluster corresponds to class 2 and it is surrounded by the other two clusters belonging to class one (see Fig. 2, right). The topmost two clusters are flat disks while noise is added to the lowest one, yielding that the lowest two clusters are closer to each other. Furthermore, for all clusters, the variance along the first two dimensions is higher then along the third one. This data set is an example for user task 1.

We use this data set to show the drawbacks of PCA and SOM visualizations. We train a SVM classifier and visualize it with PCA in Fig. 5. The accuracy of this visualization as concerns the labels amounts only to 42%, accordance of the contours only to 0.04. As can be verified in Fig. 5 (left), PCA maps the three clusters on top of each other, making a proper visualization of the classifier impossible. In these visualizations, we mark the points for which the classifier is displayed incorrectly with white circles. The right image shows the projections of the samples from the two-dimensional space into the original data space (this image is zoomed in on the Z-axis). In this case, the points are mapped to the first two principal components showing also how the dimension reduction from three to two dimensions has worked.

The same classifier is visualized by the SOM in Fig. 6. Although, the projection is much better (99% of the data points are assigned to the correct class by the map and contours agree with to the value of 0.98) it fails to show the three distinct clusters. On the contrary, it suggests that there exist two clusters of class 2. The right hand side of Fig. 6 shows again the inverse samples. Due to the fact that the inverse mapping π^{-1} for the SOM is the assignment of a point to a high-dimensional prototype, these shown samples coincide with the location of the self-organizing map. The position of this SOM grid explains how

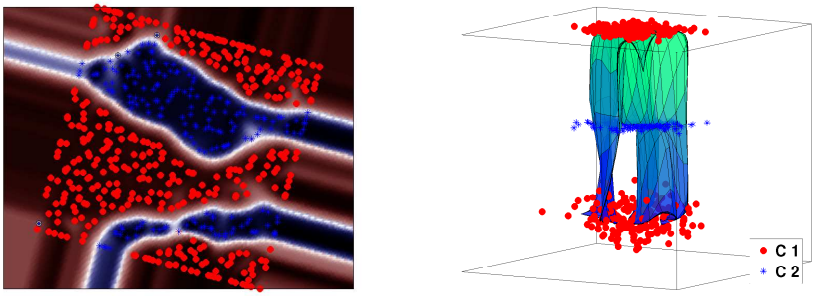

Figure 6: Visualization of data set 2 with SOM (left) and the according SOM map (right).

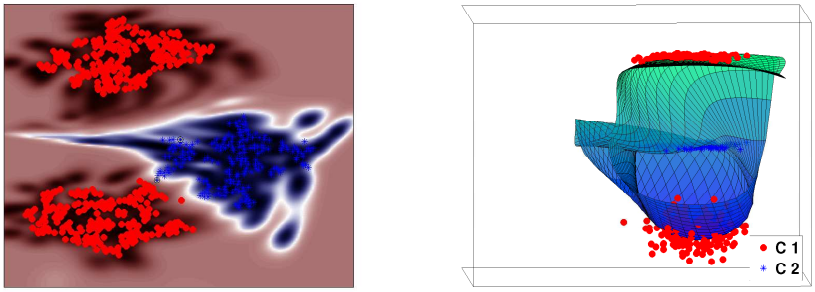

Figure 7: Visualization of data set 2 with t-SNE (left) and the according inverse projected samples (right).

the projection of the data emerged.

Using t-SNE we obtain the visualization shown in Fig. 7. Here, the three distinct clusters are visible and, further, it is shown that the class boundary between the blue cluster and one of the red clusters is more complex. The quality of the visualization of the classifier amounts to 99% and 0.98 for the labels and contours, respectively. The right hand side of Fig. 7 shows again the projected samples. The shown manifold lies smoothly in the data clouds.

Calculating the SOM on the Fisher metric (we use the relational batch SOM [15] in our experiments for this purpose) we obtain the visualization shown in Fig. 8. The projection displays much better the original data characteristics hence it shows that two regions of class one are separated by samples from class two. So for this data set, the integration of the Fisher metric yields a major improvement to the approach SVMV. However, the margin of the classifier is still not visible. The quality evaluation yields the values 99% and 0.99 for the accordance of labels and certainty.

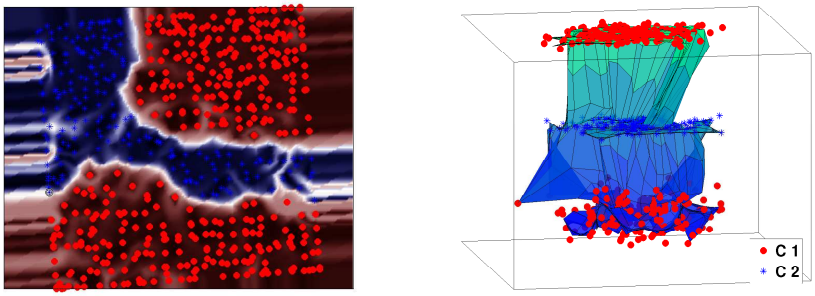

Figure 8: Visualization of data set 2 with Fisher SOM (left) and the according inverse projected samples (right).

Replacing the standard Euclidean metric by the Fisher metric seems to be advantageous for showing the class sensitive properties of the data. Whether this generalizes also to other data sets and whether it is also beneficial for the visualization of classifiers is investigated in section 4.3.

4.2 Visualizing classifiers for different class distributions of the same points

In this section we demonstrate the suitability of our approach for another artificial setting: We randomly generate data $\{\mathbf{x}_i\}_{i=1}^n$ for $n = 500$ on a three-dimensional filled cube and generate three sets of labels for two class problems. With these experiments, we address the user tasks 1 and 3. The labels are generated as follows:

1. The first class distribution consists of two clearly separated classes defined by a linear plane. We refer to the according labels by $\{l_i^1\}_{i=1}^n$.
2. For the second labelling $\{l_i^2\}_{i=1}^n$ we employ two parallel separation planes.
3. Here we utilize a random assignment with labels $\{l_i^3\}_{i=1}^n$. An overlapping class structure originates, thus addressing user task 3.

One thing that all these scenarios have in common is that, although the data set is intrinsically three-dimensional and impossible to visualize adequately in two dimensions, the class relevant structure is two dimensional, i.e. at each point in the data space only one direction is relevant for classification. Furthermore, we believe that locally this property holds for all classifiers that utilize continuous class boundaries.

Additionally, we project this data set with a random matrix to 10 dimensions. In this 10-dimensional data space, we train one SVM for each set of labels. The classification accuracies of these three classifiers are depicted in Table 1.

Table 1: Classification accuracies of the three SVMs, each trained on a different label assignment.

	l^1	l^2	l^3
training set	100%	96.5%	51.5%
test set	99.2%	95.2%	48.0%

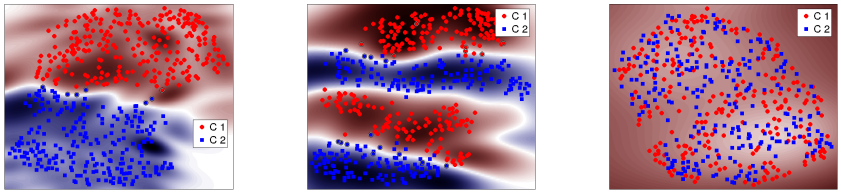

Figure 9: Visualization of SVMs trained on the 10-dimensional data set with the labels l_i^1 (left), l_i^2 (middle) and l_i^3 (right).

We utilize our approach to visualize these classifiers. Thereby, we rely on Fisher t-SNE to project the data set $\{\mathbf{x}_i\}_{i=1}^n$, while each time employing different labels and hence yielding different visualizations of the data. The three resulting visualizations of the classifiers are depicted in Figure 9.

The accuracy of the three visualizations as measured by the method introduced in section 3.3 based on the labels yields an accordance of 98.6% for the set $\{l_i^1\}_{i=1}^n$ (the left visualization), 96.0% for $\{l_i^2\}_{i=1}^n$ (middle) and 100% for set $\{l_i^3\}_{i=1}^n$ (right). The quality based on the certainty yields 0.91 (left), 0.90 (middle) and 0.82 (right).

In addition to the high accordance of the labelling regions, in this case we know the underlying class structure and, hence, can judge the visualization qualitatively. The structure of the projected points and of the class regions agrees largely to the labelling of the asociated case, i.e for case 1 two coherent structures are present, for case 2 there are 4 coherent regions while for case 3 the labelling does not have any structure.

4.3 Evaluating discriminative dimensionality reduction techniques for classifier visualization

In this section, we utilize the DR techniques t-SNE, Isomap, MVU, SOM and GTM to visualize classifiers. Exemplarily, we use the SVM here (other classifiers are visualized in sections 4.4 and 4.5). We apply these methods on the Euclidean and on the Fisher metric and we use the prefix "Fisher" in front of the DR name to indicate the latter.

In order to evaluate the effect this change of the metric has, we utilize three benchmark data sets. Similarly as in [43], we use a randomly chosen subsample of 1500 samples for each data set to save computational time.

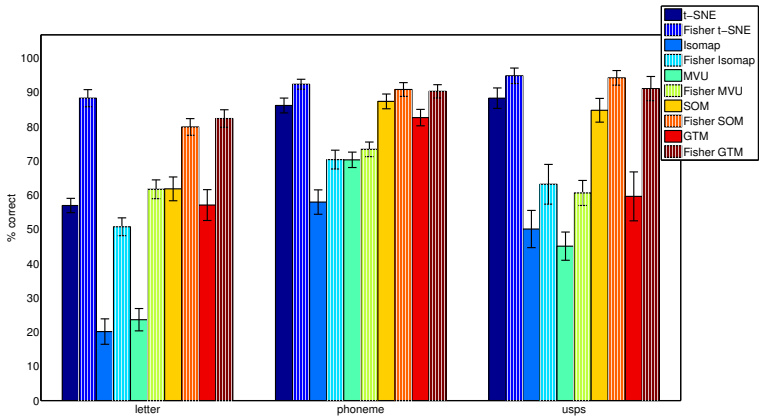

Figure 10: Empirical comparison of different DR techniques with and without supervision.

- The *letter recognition* data set (referred to as letter in the following) comprises 16 attributes of randomly distorted images of letters in 20 different fonts. The data set contains 26 classes and is available at the UCI Machine Learning Repository [10].
- The *phoneme* data set (denoted phoneme) consists of phoneme samples which are encoded with 20 attributes. 13 classes are available and the data set is taken from LVQ-PAK [21].
- The *U.S. Postal Service* data set (abbreviated via usps) contains 16×16 images of handwritten digits, and hence comprises 10 classes. It can be obtained from [30]. This data set has been preprocessed with PCA by projecting all data samples on the first 30 principal components.

As described previously, we employ SVMs with a "one versus one" classification scheme for the following data sets with more then two classes.

For each data set we apply the ten DR methods to project all points from that set. Afterwards, we utilize a ten-fold cross-validation scheme to evaluate the inverse mapping π^{-1}: The data set is randomly divided into ten parts, where nine subsets are used to train π^{-1} and the remaining subset is used for evaluation with our scheme proposed in section 3.3. This procedure is repeated ten times yielding a mean and standard deviation shown in Fig. 10 for all methods and all data sets.

For each data set and each DR projection the supervised variant achieves a better performance for the purpose of classifier visualization. This also holds for the SOM projection, yielding an improvement to the SVMV method. Further, the methods Fisher t-SNE, Fisher SOM and Fisher GTM yield the best results in our experiments.

Example visualizations of the SVM trained on the phoneme data set are shown in Fig. 11 and 12. In both, the left column displays the unsupervised visualizations and the right one the supervised ones. In the right column, the cluster structure is better visible and, hence, allows a better visualization of the class boundaries.

4.4 Utilizing supervised information based on a trained classifier

In this section we apply supervised projections based on the Fisher information metric which is based on the conditional class probability $p(c|\mathbf{x})$. We illustrate the difference between estimating $p(c|\mathbf{x})$ from the given labeling $c := l$ (i.e. from the ground truth) and estimating $p(c|\mathbf{x})$ from the labels of the classification model $c := f(\mathbf{x})$, i.e. the difference between using $p(l|\mathbf{x})$ and $p(f(\mathbf{x})|\mathbf{x})$ for the estimation of the local Fisher information matrix. Both can be done with the Parzen window estimator. If a probabilistic model is available and if it provides differentiable probabilities $p(f(\mathbf{x})|\mathbf{x})$, however, an alternative for the latter is to utilize $p(f(\mathbf{x})|\mathbf{x})$ directly to compute the local Fisher information matrices.

In this section we do the latter, and for this purpose utilize the Robust Soft LVQ (RSLVQ) classifier which has been briefly summarized in the beginning of this section. With this classifier, we can demonstrate the user task 5, i.e. how did the RSLVQ algorithm choose the prototype positions in order to solve the task.

We create an artificial data set (referred to as data set 3) which is intrinsically three-dimensional and, hence, cannot be projected to two dimensions without information loss. The data points are uniformly sampled in a filled ball. A posterior labeling is assigned to them such that a nonlinear class structure emerges. This set is shown in Fig. 13 from two perspectives. Class two (shown in blue) consists of a continuous tube which is, however, separated by a gap. Further, there is a distinct noisy region.

An unsupervised projection of this data set with t-SNE is shown in Fig. 14. As expected, the projection distorts the continuous class structure since in an unsupervised scenario no information about the labeling is available. This illustrates that unsupervised visualization techniques might not always be well suited if intrinsically high-dimensional data should be projected to low dimensions. In this example, the displayed information looks almost arbitrary.

For the training of the classifier, we use only four prototypes per class, which is small considering the complexity of the data set. The trained classifier achieves a classification accuracy of 90%. Now, a typical use case for the classifier visualization method occurs: How did the classification method solve this problem? Which simplifications of the data did the classifier use and which data points are regarded as similar by the classifier?

In order to answer these questions we visualize the classifier using Fisher t-SNE build on the original class labels l_i on the one hand and on the provided classification $f(\mathbf{x}_i)$ on the other hand. We build the visualization of the classifier on top of these two projections. The two resulting visualizations are depicted

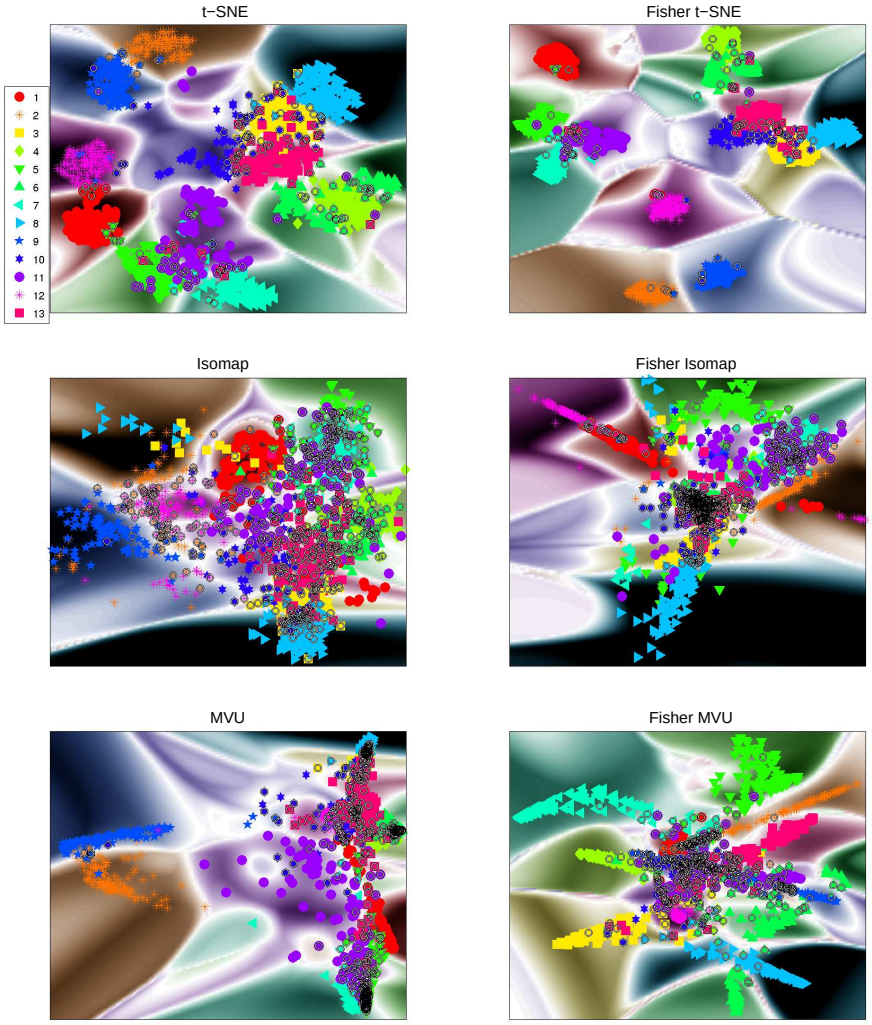

Figure 11: Visualization of the phoneme data set with the methods t-SNE, Fisher t-SNE, Isomap, Fisher Isomap, MVU and Fisher MVU.

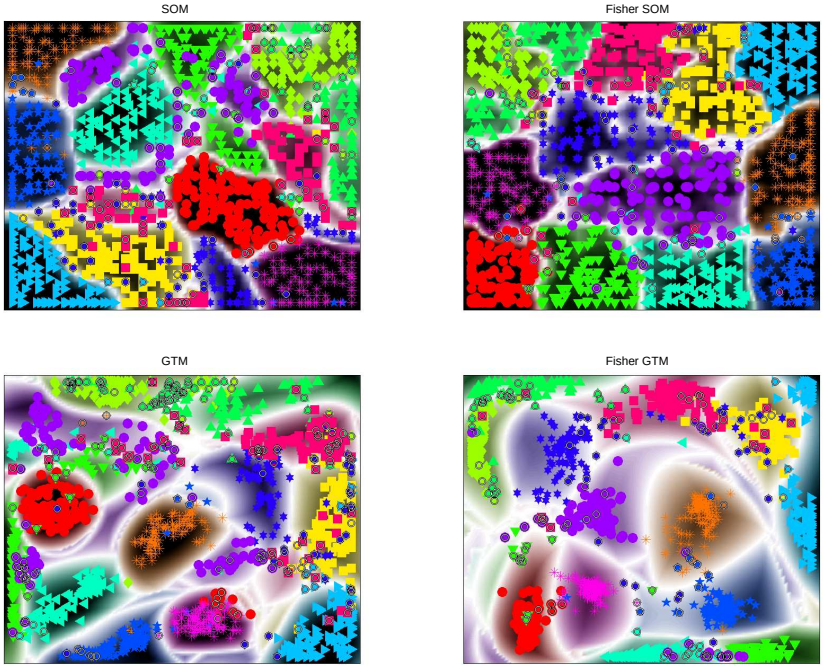

Figure 12: Visualization of the phoneme data set with the methods SOM, Fisher SOM, GTM and Fisher GTM.

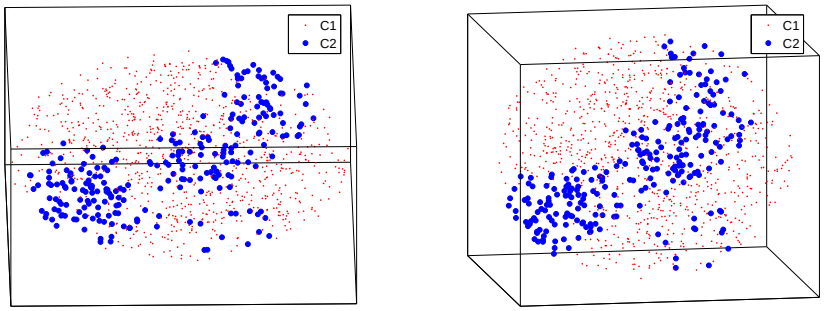

Figure 13: The three-dimensional data set 3 shown from two different perspectives.

in Fig. 15. The left visualization is based on the Parzen window estimator for the class labels $p(l|\mathbf{x})$: Basically, two clusters of points from class blue are shown and these are distinct from each other. The visualization quality of the classifier amounts to 92%. Interestingly, albeit this is not yet perfect, the visualization looks much more reasonable than direct unsupervised t-SNE on the data. The right visualization shows the same classifier, but this time based on the discriminative projection obtained by using the probabilities $p(f(\mathbf{x})|\mathbf{x})$ of the classifier itself. The data from class two form again two clusters, but this time, they are close to each other. The quality is estimated to 95%. Furthermore, the shape of the class boundaries resembles more the expected shape of the classifier, the latter usually being related to convex regions. In the visualization based on the ground truth, the original spherical shape of the data is much more pronounced.

The Parzen window estimator used in the left visualization estimates the probability density accurately and finds the gap in the blue class tube. In his part of the data space, the class distribution changes rapidly and, therefore, the distances in this region grow large, which can directly be observed in the visualization. The prototype distribution does not fit very well to the visualized classifier, since in one region of the blue class there are three prototypes of that class on top of each other and in another region there are none. But since the visualized class distribution is correct as concerns a large part of the points, we can see from this visualization that the largest part of the blue class tube is classified correctly.

In the right visualization which is based on the labeling of the classifier, the two parts of the tube lie close together. This suggests that the labeling of the classifier does not change much in this region, i.e. that the data lying in this gap of the tube are classified incorrectly. This can also be seen directly in the visualization. For few points the visualization of the classifier is inaccurate, but these lie close to the class boundary, i.e. imply only small inaccuracies. The

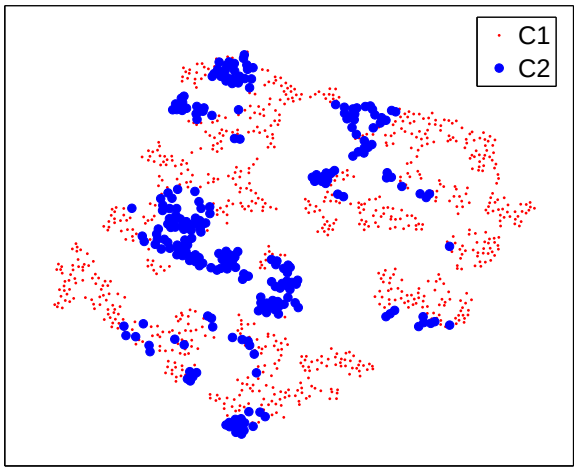

Figure 14: Projection of data set 3 with t-SNE.

most points (95%) are displayed in the correct region of the classifier. This time, the location of the prototypes is plausible in relation to the data: the prototypes of the blue class are surrounded by those of the red class. Such a constellation is plausible in the original data space.

From the latter visualization we can deduce more information as regards potential errors as compared to the previous one; we see directly the source of the remaining classification error: the classifier is not powerful enough and is not able to classify this gap in the data correctly. Furthermore, there are a few points from the blue class which lie in the cluster of points from the red class. From the perspective of this visualization we would deduce that these are either overlapping regions or too complex regions for our classifier (both aspects are probably correct: in the high-dimensional data we can see that there is indeed a region of overlapping classes).

For this toy example we can verify our interpretation by visualizing the positions of the prototypes in the original data space. Fig. 16 depicts the original data set in conjunction with the prototypes of the classifier. The same positioning of the prototypes as in the low-dimensional visualization emerges: the prototypes of the blue class are surrounded by those of the red class.

4.5 Visualize different classification models

In this section we demonstrate our approach on the real world benchmark data set USPS for the two classifiers Robust Soft LVQ and Classification Tree.

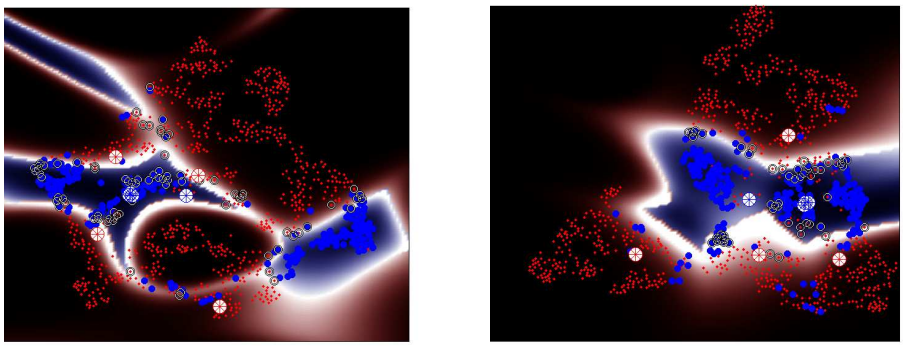

Figure 15: Two visualization of the same RSLVQ classification model: The projection methods Fisher t-SNE based on the original labeling (left) and Fisher t-SNE based on the labels from the trained classifier (right) are applied.

Visualization of a Classification Tree

We train a Classification Tree on the USPS data set used in the previous section. The resulting classifier obtains a classification accuracy of 89% on the training set and 66% on the test set.

Fig. 17 shows two Fisher SOM visualizations of this classifier: For the left we employ the Fisher information defined by the labels of the classifier and for the right one we utilize the original labels for the Fisher information (we use the Parzen window estimator in both cases). Due to this choice the left visualization rather shows the '"view of the classifier" on the data while the right one shows the true distribution. However, the first one can be better suited to interpret the trained classification model. In this case the quality of the left visualization of the classifier is 92.3% and the quality of the right one is 87.8%.

In the left visualization we can see that in the region of class 9 some instances of class 8 are mixed. In the right visualization this is not the case. Therefore, we can deduce that the separation of class 8 and 9 is particularly hard for the given classifier. Further, the classes 5 and 3 seem to overlap (left visualization). However, these two classes only have very little overlap in the right visualization. This indicates that the classifier is not complex enough in this region of the data space, as well.

Furthermore, we re-plot both visualizations with from Fig. 17 in Fig. 18 with the labeling assigned to by the classifier. The visual impression of the two images shown in Fig. 18 agrees with the result of the formal evaluation measure suggesting that the left one visualizes the classifier more accurately.

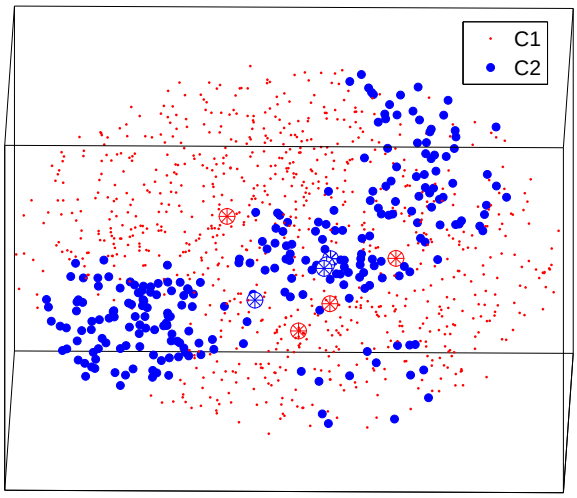

Figure 16: Data set 3 together with the prototypes of the trained RSLVQ model.

Visualization of a Robust Soft LVQ model

As a next step, we train a RSLVQ classifier with two prototypes per class on the USPS data set. The trained model obtains a classification accuracy of 97,2% on the training and of 87.3% on the test set.

Using the Fisher information as defined by the labels of the classifier and the Fisher SOM technique, the visualization shown in Fig. 19 (left) results. This visualization of the classifier has an accordance of 97.9%. The high classification accuracy can be observed in this visualization, as well. In addition, we can see directly which classes are mixed up the most time. For example, there are a few instances of class 7 classified as class 9. Having this knowledge, we could improve our classifier by increasing the complexity of the class boundary between these two classes (in this case we could employ more prototypes for these classes). On the other hand, the visualization suggests that the classes are unimodal. Furthermore, some prototypes of the same class seem to be located close to each other (e.g. those of class 0).

In order to obtain another view on the data, we project the classifier also with Fisher t-SNE (shown on the right of Fig. 19). This method tends to show clustering information (in contrast to the SOM, which doesn't show gaps between clusters). The Fisher t-SNE projection indicates further that the complexity of the model could be reduced without loosing much accuracy, since many prototypes lie on top of each other. More precise, for all except three classes (1,3 and 5) the two prototypes are positioned on top of each other. We examine this

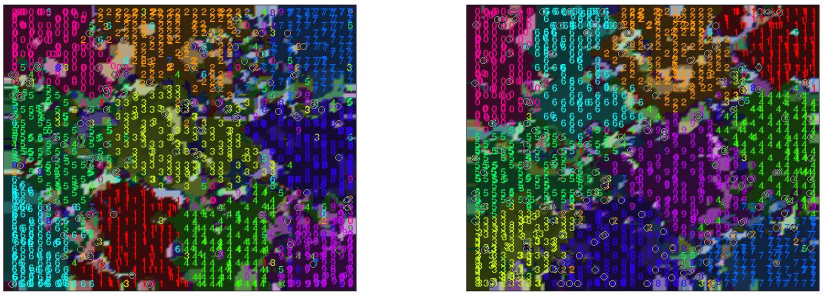

Figure 17: Two Fisher SOM visualization of the same Classification Tree classier. The left visualization is based on labeling provided by the classifier and the right on the original labels.

hypotheses by training a RSLVQ classifier with only one prototype per class. Indeed, this model has only a slight accuracy loss: the model classifies 95,1% of the training set and 86.9% of the test set correct (using the same training/test set partition as before).

5 Discussion

In this paper we present a general framework to visualize nonlinear classifiers trained on potentially high-dimensional data sets. This framework makes it possible to visualize arbitrary classifiers, with the only restriction that they have to provide some measure of certainty for the classification. We demonstrate this for Support Vector Machines, Classification Trees and probabilistic LVQ classifier.

We utilize ten dimensionality reduction methods to visualize classifiers and state experimentally that among them, supervised DR techniques are particularly well suited for this task.

Further, we demonstrate that a visualization of a trained classifier can give insights into the classification process. Hence, it can also help to improve the process itself.

This framework is general, such that it allows to combine arbitrary classifiers with arbitrary projection methods. As such, it also includes methods from the literature, as for instance SVMV. Furthermore, we combine SOM with the Fisher metric, yielding an improvement of this approach.

The evaluation of these visualized classifiers is currently based on the classification and certainty accordance of the projected and original classifier. Although we also evaluate the generalization of such visualizations to new points, other properties of the classifier are not evaluated, yet. Such include the topological structure of the class boundaries and the size of the margin. These

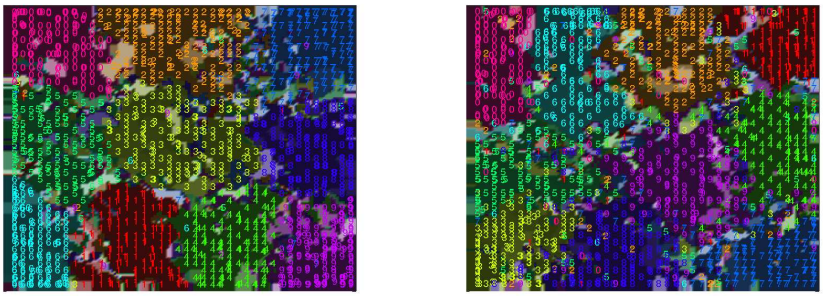

Figure 18: Fisher SOM visualization of the Classification Tree where the data points are labeled according to the classifier. The same projections as shown in Fig. 17 are utilized.

aspects will be the focus of future work.

Another source for potential improvement comprises the complexity of this approach. Many non-parametric methods such as t-SNE require squared computational time. Fortunately, approximations in log-linear time have been proposed, recently [46, 38]. Such ideas could also help to reduce the number of Fisher matrices to be estimated.

Acknowledgements

Funding from DFG under grant number HA2719/7-1 and by the CITEC center of excellence is gratefully acknowledged. We also would like to thank the reviewers for many helpful comments and ideas concerning the evaluation.

References

[1] M. Aupetit and T. Catz. High-dimensional labeled data analysis with topology representing graphs. *Neurocomputing*, 63:139–169, 2005.

[2] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12:2385–2404, 2000.

[3] K. Bunte, M. Biehl, and B. Hammer. A general framework for dimensionality reducing data visualization mapping. *Neural Computation*, 24(3):771–804, 2012.

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

[5] D. Caragea, D. Cook, H. Wickham, and V. Honavar. Visual methods for examining svm classifiers. In Simoff et al. [36], pages 136–153.

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

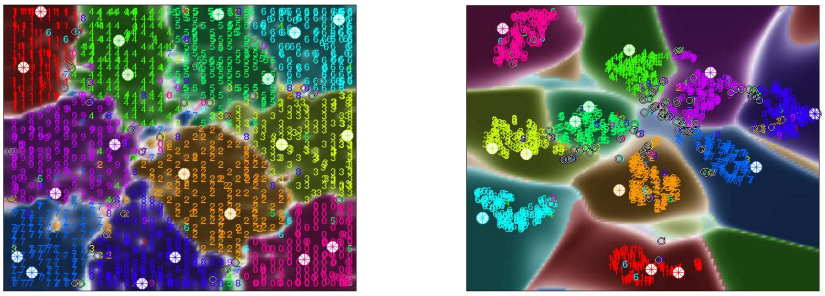

Figure 19: Visualization of the RSLVQ classifier with Fisher SOM (left) and Fisher t-SNE (right). Both projections are based on the Fisher information as defined by the labels of the classifier (but the original labeling is shown).

[7] D.Cohn. Informed projections. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 849–856. MIT Press, 2003.

[8] I. S. Dhillon, D. S. Modha, and W. S. Spangler. Class visualization of high-dimensional data with applications. *Computational Statistics & Data Analysis*, 41(1):59–90, November 2002.

[9] E. P. dos Santos Amorim, E. V. Brazil, J. D. II, P. Joia, L. G. Nonato, and M. C. Sousa. ilamp: Exploring high-dimensional spacing through backward multidimensional projection. In *IEEE VAST*, pages 53–62. IEEE Computer Society, 2012.

[10] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[11] A. Gisbrecht and B. Hammer. Data visualization by nonlinear dimensionality reduction. *WIREs Data Mining and Knowledge Discovery*, accepted.

[12] A. Gisbrecht, D. Hofmann, and B. Hammer. Discriminative dimensionality reduction mappings. In J. Hollmén, F. Klawonn, and A. Tucker, editors, *IDA*, Lecture Notes in Computer Science, pages 126–138. Springer, 2012.

[13] A. Gisbrecht, A. Schulz, and B. Hammer. Parametric nonlinear dimensionality reduction using kernel t-sne. *Neurocomputing*, 147(0):71 – 82, 2015. Advances in Self-Organizing Maps Subtitle of the special issue: Selected Papers from the Workshop on Self-Organizing Maps 2012 (WSOM 2012).

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

[15] B. Hammer and A. Hasenfuss. Topographic mapping of large dissimilarity datasets. *Neural Computation*, 22(9):2229–2284, 2010.

[16] B. Hammer, D. Hofmann, F.-M. Schleif, and X. Zhu. Learning vector quantization for (dis-)similarities. *Neurocomputing*, 2013. In Press.

[17] J. Hernandez-Orallo, P. Flach, and C. Ferri. Brier curves: a new cost-based visualisation of classifier performance. In *International Conference on Machine Learning*, June 2011.

[18] D. Hofmann, F.-M. Schleif, B. P. en, and B. Hammer. Learning interpretable kernelized prototype-based models. *Neurocomputing*, revised, 2013.

[19] T. W. House. Big data research and development initiative, 2012.

[20] A. Jakulin, M. Možina, J. Demšar, I. Bratko, and B. Zupan. Nomograms for visualizing support vector machines. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 108–117, New York, NY, USA, 2005. ACM.

[21] T. Kohonen, J. Hynninen, J. Kangas, J. Laaksonen, and K. Torkkola. LVQ_PAK: The Learning Vector Quantization program package. Report A30, Helsinki University of Technology, Laboratory of Computer and Information Science, Jan. 1996.

[22] R. Kothari and M. Dong. Decision trees for classification: A review and some new results.

[23] U. H.-G. Kreß el. Advances in kernel methods. chapter Pairwise classification and support vector machines, pages 255–268. MIT Press, Cambridge, MA, USA, 1999.

[24] J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer, 2007.

[25] B. Ma, H. Qu, and H. Wong. Kernel clustering-based discriminant analysis. *Pattern Recognition*, 40(1):324–327, 2007.

[26] O. Melnik. Decision region connectivity analysis: A method for analyzing high-dimensional classifiers. *Machine Learning*, 48(1-3):321–351, 2002.

[27] C. Otte. Safe and interpretable machine learning: A methodological review. In C. Moewes and A. Nürnberger, editors, *Computational Intelligence in Intelligent Data Analysis*, volume 445 of *Studies in Computational Intelligence*, pages 111–122. Springer Berlin Heidelberg, 2013.

[28] J. Peltonen, A. Klami, and S. Kaski. Improved learning of riemannian metrics for exploratory analysis. *Neural Networks*, 17:1087–1100, 2004.

[29] F. Poulet. Visual svm. In C.-S. Chen, J. Filipe, I. Seruca, and J. Cordeiro, editors, *ICEIS (2)*, pages 309–314, 2005.

[30] S. Roweis. Machine learning data sets, 2012. Available at http://www.cs.nyu.edu/ roweis/data.html.

[31] S. Rüping. *Learning Interpretable Models*. PhD thesis, Dortmund University, 2006.

[32] P. Schneider, M. Biehl, and B. Hammer. Adaptive relevance matrices in learning vector quantization. *Neural Computation*, 21:3532–3561, 2009.

[33] A. Schulz, A. Gisbrecht, and B. Hammer. Using nonlinear dimensionality reduction to visualize classifiers. In I. Rojas, G. J. Caparrós, and J. Cabestany, editors, *IWANN (1)*, volume 7902 of *Lecture Notes in Computer Science*, pages 59–68. Springer, 2013.

[34] A. Schulz, A. Gisbrecht, and B. Hammer. Using Discriminative Dimensionality Reduction to Visualize Classifiers. *Neural Processing Letters*, 42(1), 2015.

[35] S. Seo and K. Obermayer. Soft learning vector quantization. *Neural Computation*, 15(7):1589–1604, 2003.

[36] S. J. Simoff, M. H. Böhlen, and A. Mazeika, editors. *Visual Data Mining - Theory, Techniques and Tools for Visual Analytics*, volume 4404 of *Lecture Notes in Computer Science*. Springer, 2008.

[37] B. A. Turlach. Bandwidth Selection in Kernel Density Estimation: A Review. In *CORE and Institut de Statistique*, pages 23–493, 1993.

[38] L. van der Maaten. Barnes-hut-sne. *CoRR*, abs/1301.3342, 2013.

[39] L. van der Maaten and G. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[40] L. van der Maaten, E. Postma, and H. van den Herik. Dimensionality reduction: A comparative review. Technical report, Tilburg University Technical Report, TiCC-TR 2009-005, 2009.

[41] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[42] A. Vellido, J. Martin-Guerroro, and P. Lisboa. Making machine learning models interpretable. In *ESANN'12*, 2012.

[43] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *Journal of Machine Learning Research*, 11:451–490, 2010.

[44] X. Wang, S. Wu, X. Wang, and Q. Li. Svmv - a novel algorithm for the visualization of svm classification results. In J. Wang, Z. Yi, J. Zurada, B.-L. Lu, and H. Yin, editors, *Advances in Neural Networks - ISNN 2006*, volume 3971 of *Lecture Notes in Computer Science*, pages 968–973. Springer Berlin / Heidelberg, 2006.

[45] M. Ward, G. Grinstein, and D. A. Keim. *Interactive Data Visualization: Foundations, Techniques, and Application*. A. K. Peters, Ltd, 2010.

[46] Z. Yang, J. Peltonen, and S. Kaski. Scalable optimization of neighbor embedding for visualization. In S. Dasgupta and D. Mcallester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, volume 28, pages 127–135. JMLR Workshop and Conference Proceedings, May 2013.