Jaakko Peltonen, Arto Klami, and Samuel Kaski. Improved Learning of Riemannian Metrics for Exploratory Analysis. *Neural Networks*, accepted for publication.

Improved Learning of Riemannian Metrics for Exploratory Analysis

Jaakko Peltonen [a], Arto Klami [a], Samuel Kaski [a,b,*]

[a] *Neural Networks Research Centre*

Helsinki University of Technology

P.O. Box 5400, FIN-02015 HUT, Finland

[b] *Department of Computer Science*

P.O. Box 26, FIN-00014 University of Helsinki, Finland

This work was supported by the Academy of Finland, grant 52123.

* Corresponding author. Tel.: +358-9-451 8203; fax: +358-9-451 3277.
Email address: `Samuel.Kaski@hut.fi` (Samuel Kaski).

Improved Learning of Riemannian Metrics for Exploratory Analysis

Abstract

We have earlier introduced a principle for learning metrics, which shows how metric-based methods can be made to focus on discriminative properties of data. The main applications are in supervising unsupervised learning to model interesting variation in data, instead of modeling all variation as plain unsupervised learning does. The metrics are derived by approximations to an information-geometric formulation. In this paper we review the theory, introduce better approximations to the distances, and show how to apply them in two different kinds of unsupervised methods: prototype-based and pairwise-distance based. The two examples are self-organizing maps and multidimensional scaling (Sammon's mapping).

Key words: Information geometry, information visualization, learning metrics, multidimensional scaling, self-organizing map

1 Introduction

Unsupervised learning for clustering or information visualization suffers from the garbage in—garbage out problem. The ultimate goal is to make discoveries in data, that is, to find new things without specifying them in advance. The problem is that unsupervised learning cannot distinguish relevant variation from irrelevant variation in data. Structured noise becomes modeled as well as relevant structure. The problem is particularly hard in mining high-dimensional databases, for example in bioinformatics or text mining.

Hence, all successful unsupervised learning must have been supervised implicitly or explicitly. We have introduced a learning metrics principle (Kaski et al., 2001; Kaski and Sinkkonen, 2004) to help automate some of the implicit supervision for methods that are based on distance computations. Two subproblems need to be solved: (i) how to infer what is relevant and what not, and (ii) how to use the findings in data-analysis methods.

Supervised methods are told directly what is relevant: the task is to predict the value of a dependent variable, and only variation relevant to the prediction task is interesting. Relevance is learned from a set of data pairs (x, c), where x is the primary data and c is the desired response, that is, value of the dependent variable (auxiliary data). It has been suggested that relevance could be derived from dependencies between such paired data (Becker and Hinton, 1992; Tishby et al., 1999). In practice, mutual information between representations of the data, such as clusters (Becker, 1996; Tishby et al., 1999; Friedman et al., 2001; Sinkkonen and Kaski, 2002) or linear projections (Torkkola, 2003; Kaski and Peltonen, 2003), and the class labels c would be maximized.

The question we asked was whether the relevance could be incorporated in the metric of the data space. A distance measure that would learn to gauge only relevant differences between data points would be generally applicable to a wide variety of data analysis methods that are based on distance computations. The difference from standard supervised learning is that only the metric is supervised. The ultimate task need not be prediction; in fact, the main applications for learning metrics are in exploring new things in the primary data given the supervision. This task could be called supervised unsupervised learning.

The learning metrics principle, reviewed in Section 3, was formulated in terms of idealized information-geometric concepts. Practical applications require approximations, and in this paper we introduce considerably improved approximations and methods for computing the distances.

Learning of metrics has been studied by others as well. The so-called Fisher kernels (Jaakkola and Haussler, 1999; Tsuda et al., 2002) form distance measures between data points. The goal of that work is opposite to ours, namely to derive metrics from unsupervised generative models and use them to construct distances for supervised learning. Also, although the method appears similar to ours in that both use information geometry, the similarity is actually only superficial. Recently, supervised global (non-Riemannian) metrics have been optimized based on auxiliary data (Xing et al., 2003), in this case about similarity of pairs of samples. Parametrized metrics have also been optimized in classification tasks (Hammer and Villmann, 2002).

So far we have applied the learning metrics principle only to self-organizing maps (Kaski et al., 2001), although the same principle has motivated discriminative clustering (Sinkkonen and Kaski, 2002) and projection algorithms (Kaski and Peltonen, 2003). Here we apply the metrics additionally to Sammon's mapping, a multidimensional scaling method that aims at preserving pairwise distances between data samples. The same computational methods are applicable to any prototype-based and mutual distance-based methods.

2 Preliminaries: from Euclidean to Riemannian metrics

We start by introducing the kind of metric the principle constructs.

The simplest metrics take for granted the original coordinates of the data space and their scaling. The normal Euclidean metric $d_{\mathbf{I}}$ between two data points, $\mathbf{x}$ and $\mathbf{y} \in \mathbb{R}^n$, can be expressed by

$$d_{\mathbf{I}}^2(\mathbf{x}, \mathbf{y}) \equiv \|\mathbf{x} - \mathbf{y}\|^2 = (\mathbf{x} - \mathbf{y})^T \mathbf{I} (\mathbf{x} - \mathbf{y}) \; , \tag{1}$$

that is, by a quadratic form with the identity matrix $\mathbf{I}$.

The next more general metric is a global metric $d_{\mathbf{A}}$ that re-scales the coordinates or their combinations. Such a metric can be expressed by a positive semi-definite matrix $\mathbf{A}$, $\mathbf{x}^T \mathbf{A} \mathbf{x} \geq 0$ for all $\mathbf{x}$, which replaces the identity matrix in Eq. (1). A positive semi-definite matrix can always be expressed by $\mathbf{A} = \mathbf{S}^T \mathbf{S}$ for some $\mathbf{S}$, and hence the global distance is

$$\begin{aligned} d_{\mathbf{A}}^2(\mathbf{x}, \mathbf{y}) \equiv (\mathbf{x} - \mathbf{y})^T \mathbf{A} (\mathbf{x} - \mathbf{y}) &= (\mathbf{x} - \mathbf{y})^T \mathbf{S}^T \mathbf{S} (\mathbf{x} - \mathbf{y}) \\ &= (\mathbf{S}\mathbf{x} - \mathbf{S}\mathbf{y})^T (\mathbf{S}\mathbf{x} - \mathbf{S}\mathbf{y}) = d_{\mathbf{I}}^2(\mathbf{x}', \mathbf{y}') \; , \end{aligned} \tag{2}$$

where $\mathbf{x}' = \mathbf{S}\mathbf{x}$. Hence, the global metric is equivalent to linear feature extraction with a matrix $\mathbf{S}$, followed by the standard Euclidean metric.

In the most general metric the matrix $\mathbf{A}$ depends on the location, and the distance is

$$d^2_{\mathbf{A}(\mathbf{x})}(\mathbf{x},\mathbf{y}) = (\mathbf{x}-\mathbf{y})^T \mathbf{A}(\mathbf{x})(\mathbf{x}-\mathbf{y}) \; .$$

Since a metric has to be symmetric, that is, $d(\mathbf{x},\mathbf{y}) = d(\mathbf{y},\mathbf{x})$, this direct definition is used only for local distances between very close-by $\mathbf{x}$ and $\mathbf{y} = \mathbf{x} + d\mathbf{x}$. The local distances are extended by defining global distances as minimal path integrals.

The learning metrics principle constructs these most general kinds of metrics called Riemannian metrics.

3 The learning metrics principle

3.1 Definition

We want to form a Riemannian metric that measures only relevant differences between points of the data space. The key assumption is that the data comes in pairs $(\mathbf{x}, c)$ where c indicates what is relevant. In the same way as in supervised learning, only those changes in $\mathbf{x}$ that cause changes in c are assumed to be interesting. Here $\mathbf{x} \in \mathbb{R}^n$ and c is discrete-valued. [1]

Such a metric should measure changes in the distribution of c, caused by changes in $\mathbf{x}$. When the distances between distributions are measured by the Kullback-Leibler divergence D_{KL}, it can be shown (Kullback, 1959) that for

[1] Continuous c are possible although the computation would be more difficult.

a differential $d\mathbf{x}$,

$$d_L^2(\mathbf{x}, \mathbf{x} + d\mathbf{x}) \equiv D_{KL}(p(c|\mathbf{x})||p(c|\mathbf{x} + d\mathbf{x})) = d\mathbf{x}^T \mathbf{J}(\mathbf{x}) d\mathbf{x} \ , \tag{3}$$

where $\mathbf{J}(\mathbf{x})$ is the Fisher information matrix

$$\mathbf{J}(\mathbf{x}) = E_{p(c|\mathbf{x})} \left\{ \left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x}) \right) \left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x}) \right)^T \right\} . \tag{4}$$

This is the learning metrics principle: to use such Riemannian distances d_L as a metric of the data space.

In practice the densities $p(c|\mathbf{x})$ need to be estimated from a finite data set $\{(\mathbf{x}, c)\}$, and the minimal path integrals that extend the local metric to longer distances need computational approximations. In this paper we present new methods for these tasks.

3.2 Properties of the metric

Where can it be applied? The principle assumes that the auxiliary data is well-chosen, in the sense that important changes in the primary data correspond to changes in the auxiliary data. In other words, the supervision of unsupervised learning needs to be chosen as carefully as the predicted variable in usual regression or classification tasks.

Why not normal supervised learning? If the task is pure classification or regression, that is, the only answer that is needed is the value of the dependent variable c, then normal supervised learning is the right choice.

In this paper the metric is supervised but the data analysis method used in the new metric need not be. If the goal is to make discoveries with unsupervised methods, given the supervised metric, then the learning metric is the right choice. [2]

Why preserve topology? An alternative to the Riemannian metric would be to simply use the Kullback-Leibler divergence in Eq. (3) globally, for any two points. Alternatively, any other distributional distance measure could be used. A lot of computation would be saved if the approximation of path integrals could be skipped. However, such global distance would not preserve the topology of the data space, which may be important in data analysis.

The relative usefulness of this global metric and the Riemannian metric depends on the application. The difference will be demonstrated in Section 7.1.

Why not supervised feature extraction? A straightforward and often-used alternative is to preprocess the data by a linear or nonlinear transformation and then use the standard Euclidean distance.

For local pairs of data points this is expressible by the Riemannian metric, which can be shown easily. If the transformation of $\mathbf{x}$ is denoted by $\mathbf{f}(\mathbf{x})$, the distance between differentially close-by points $\mathbf{x}$ and $\mathbf{y} = \mathbf{x} + d\mathbf{x}$ would be

$$d_{\mathbf{f}}^2(\mathbf{x}, \mathbf{y}) \equiv \|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})\|^2 = (\mathbf{x} - \mathbf{y})^T \mathbf{D_f}(\mathbf{x})^T \mathbf{D_f}(\mathbf{x})(\mathbf{x} - \mathbf{y}) \ .$$

Here $\mathbf{D_f}(\mathbf{x})$ is the Jacobian matrix evaluated at $\mathbf{x}$, and the distance matrix $\mathbf{A}(\mathbf{x}) \equiv \mathbf{D_f}(\mathbf{x})^T \mathbf{D_f}(\mathbf{x})$ defines the local metric.

[2] The distinction is not clear-cut, however, since some supervised methods can for instance make inferences about the relevance of input variables to classification.

For global distances the correspondence does not hold, and it ultimately depends on the application which approach is better. The most obvious difference is that transformed data, and hence data-analysis results, may be harder to interpret, whereas the metric is computed of the original data variables which may have domain-specific meaning.

4 Computation of the metrics

Two approaches to computation. There are two main approaches to constructing practical methods for learning metrics. The first is to develop explicit approximations to the distance d_L, Eq. (3), and use them within an algorithm.

The second approach is only mentioned briefly here. Discriminative methods for clustering (Sinkkonen and Kaski, 2002; Kaski et al., 2003) and projection (Kaski and Peltonen, 2003) have been constructed by selecting an objective function whose optimization implicitly forces the solution to correspond to learning metric distances. The connection to learning metrics is asymptotic.

Both approaches are useful. Explicit computation is generally applicable, while implicit optimization depends on a specific objective function tailored to each method. On the other hand, if such an objective function can be devised, the method can be optimized in a single stage without separate learning of the metric. In this paper we focus on explicit distance computation.

Practical approximations. In this work we take the "engineering approach" of developing general-purpose approximations: we estimate the class density $p(c|\mathbf{x})$, plug it into Eq. (3), and approximate global distances computationally. The ultimate test of such approximations is using them in practice; empirical comparisons are presented in Sections 5.3 and 6.2.

4.1 Estimation of Conditional Density

The Fisher information matrix in Eq. (4) is a function of the conditional probabilities $p(c|\mathbf{x})$ which must be estimated from a finite data set $\{\mathbf{x}_i, c_i\}_{i=1}^N$. We have used three kinds of density estimates. Their generic form is

$$\hat{p}(c|\mathbf{x}) = \frac{\sum_k \psi_{kc} \pi_k e^{-||\mathbf{x}-\boldsymbol{\theta}_k||^2/2\sigma^2}}{\sum_k \pi_k e^{-||\mathbf{x}-\boldsymbol{\theta}_k||^2/2\sigma^2}} \ , \tag{5}$$

that is, a mixture of a set of components indexed by k. Here ψ_{kc} models the conditional density of c within component k, π_k the probability of the component, and the exponentials the domain of the component. The parameters fulfill $\psi_{kc}, \pi_k \geq 0$, $\sum_c \psi_{kc} = 1$ and $\sum_k \pi_k = 1$.

The first two estimators were used in an earlier work (Kaski et al., 2001) because they could be easily derived from standard estimators. Both are optimized to model the joint density $p(\mathbf{x}, c)$, and the conditional density in Eq. (5) is derived from the result by the Bayes rule. The first option is the Mixture Discriminant Analysis (MDA2) (Hastie et al., 1995), where each mixture component generates independently both the $\mathbf{x}$ and c. MDA2 is optimized by an expectation maximization algorithm to maximize the likelihood $\prod_{i=1}^N \hat{p}(\mathbf{x}_i, c_i)$, and the number of components is selected by validation.

The second option is the standard Parzen nonparametric estimator. The terms in Eq. (5) come directly from the learning data set, $\{\mathbf{x}_k, c_k\}_{k=1}^N$, by setting $\boldsymbol{\theta}_k = \mathbf{x}_k$, $\psi_{kc} = \delta_{c_k,c}$,and $\pi_k = 1/N$. Here $\delta_{c_k,c}$ is the Kronecker delta, equal to zero unless $c_k = c$, when it equals 1. These Parzen window estimates are accurate, asymptotically consistent (see Devroye and Wagner, 1980) but computationally very expensive.

The third, new option is to estimate the conditional density directly, by maximizing the conditional likelihood $\prod_{i=1}^{N} \hat{p}(c_i|\mathbf{x}_i)$ of the model in Eq. (5). We set $\pi_k = 1/N$ for simplicity, optimize the parameters by a conjugate gradient algorithm, and again select the number of components by validation. The model will be called below the conditional mixture model (CMM).

For all these estimators of the form given by Eq. (5), the Fisher information matrix becomes

$$\mathbf{J}(\mathbf{x}) = \frac{1}{\sigma^4} E_{\hat{p}(c|\mathbf{x})}\{\mathbf{b}(\mathbf{x}, c)\mathbf{b}(\mathbf{x}, c)^T\} \tag{6}$$

where (see Kaski et al., 2001)

$$\begin{cases} \mathbf{b}(\mathbf{x}, c) & = E_{\xi(k|\mathbf{x},c;\boldsymbol{\theta}_k)}\{\boldsymbol{\theta}_k\} - E_{\xi(k|\mathbf{x};\boldsymbol{\theta}_k)}\{\boldsymbol{\theta}_k\} \\ \xi(k|\mathbf{x}, c; \boldsymbol{\theta}_k) & = \frac{\psi_{kc}\pi_k e^{-||\mathbf{x}-\boldsymbol{\theta}_k||^2/2\sigma^2}}{\sum_j \psi_{jc}\pi_j e^{-||\mathbf{x}-\boldsymbol{\theta}_j||^2/2\sigma^2}} \\ \xi(k|\mathbf{x}; \boldsymbol{\theta}_k) & = \frac{\pi_k e^{-||\mathbf{x}-\boldsymbol{\theta}_k||^2/2\sigma^2}}{\sum_j \pi_j e^{-||\mathbf{x}-\boldsymbol{\theta}_j||^2/2\sigma^2}} \end{cases} . \tag{7}$$

The operators E in the topmost equation above denote weighted sums where the weights are given by $\xi(k|\mathbf{x}, c; \boldsymbol{\theta}_k)$ and $\xi(k|\mathbf{x}; \boldsymbol{\theta}_k)$ respectively. The weights sum to 1 but in general they need not be probabilities.

4.2 Approximations to path integrals

Even though the exact form of the density or its estimate is known, in most cases it is too complex for analytically computing the minimal path integral between a pair of points $\mathbf{x}_1$ and $\mathbf{x}_2$. Hence we must approximate it.

Local distance. The least complex approximation is to use the simple local distance definition Eq. (3) as such, for any pair of points, even if they are far apart. This corresponds to assuming that $\mathbf{J}(\mathbf{x})$ is constant, and hence the shortest path is a line.

The distance between two points, $\mathbf{x}_1$ and $\mathbf{x}_2$, is then

$$d_1^2(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1 - \mathbf{x}_2)^T \mathbf{J}(\mathbf{x}_1)(\mathbf{x}_1 - \mathbf{x}_2) \ . \tag{8}$$

This approximation is called the *local approximation* below, or the *1-point approximation* since the metric is computed at one point.

Linear piecewise distance. The local approximation works close to the point $\mathbf{x}_1$ but neglects the changes in the auxiliary data farther away. The approximation is obviously not accurate over large distances. Note that the metric is locally at most $N_C - 1$-dimensional, where N_C is the number of classes (possible values of the auxiliary variable).[3] The distribution of auxiliary data changes locally only along a $N_C - 1$-dimensional subspace, and distances in all orthogonal directions are zero. Locally this effective dimensionality reduction is the desired solution since it gets rid of the uninteresting variation, but globally the solution is too simplified.

[3] This is not specific to our estimates. It is a general property of the way conditional class distributions change locally: $\mathbf{J}(\mathbf{x})$ is a sum of N_C outer products $\mathbf{v}\mathbf{v}^T$ where $\mathbf{v}$ are gradients of the conditional class log-probabilities (times scalars). Class probabilities sum to one, so the gradients are linearly dependent and hence $\mathbf{J}(\mathbf{x})$ has at most $N_C - 1$ nonzero eigenvalues.

A computationally manageable extension is to still assume that the shortest path is a line but compute the distances along several points on the line. Besides being more accurate, this makes the distance measure symmetric as the number of computation points increases. This will be called the *T-point approximation*

$$d_T(\mathbf{x}_1, \mathbf{x}_2) = \sum_{t=1}^{T} d_1 \left(\mathbf{x}_1 + \frac{t-1}{T}\mathbf{d}_{12}, \mathbf{x}_1 + \frac{t}{T}\mathbf{d}_{12} \right) , \tag{9}$$

where $\mathbf{d}_{12} = \mathbf{x}_2 - \mathbf{x}_1$.

Graph search distance. The T-point approximation assumes the minimal path is a straight line. The last improvement is to remove this assumption. Given a set of points X, the pairwise distances along linear paths (computed with the T-point method) are considered as edges of a graph, and the minimal path between two vertices is sought. This yields

$$d_G(\mathbf{x}_1, \mathbf{x}_2) = \min_{K, \{\mathbf{x}'_1, \ldots, \mathbf{x}'_K\} \in X} d_T(\mathbf{x}_1, \mathbf{x}'_1) + \sum_{k=1}^{K-1} d_T(\mathbf{x}'_k, \mathbf{x}'_{k+1}) + d_T(\mathbf{x}'_K, \mathbf{x}_2) \ . \tag{10}$$

This is denoted the graph approximation. It allows both linear and piecewise linear minimal paths; therefore the T-point distance is an upper bound to the graph distance. Standard graph search algorithms such as Floyd's algorithm can be used to find the minimum. An analogous graph computation scheme has earlier been suggested for distances computed from unsupervised generative models (Rattray, 2000).

When should one use which approximation? In applications like the SOM, the distances are not used as such but are compared to find the smallest distance (although the SOM adaptation does depend on the actual distance). In this case it is important to compute the close-by distances accurately to determine which one is smallest, whereas the larger distances can be allowed to be erroneous. In contrast, in applications that directly use the distances, such as MDS methods, it is important to preserve all the distances.

Another important factor in choosing the approximation is the available computation time. The 1-point approximation is relatively fast, the T-point approximation is linear in T, and the graph approximation takes the longest time ($O(N^3)$ where N is the number of samples). The graph approximation is feasible for methods where the set of points (and minimal paths) does not change and hence the distances need to be computed only once. Methods based on a pairwise distance matrix are such. In SOM the prototype vectors change and distances from them need to be computed all the time. Faster approximations are then required. In Section 7.3 we compare approximations of varying complexity.

Table 1 summarizes how to compute the distance approximations. In the next sections, we apply the approximations to unsupervised methods: in Section 5 the 1-point and T-point approximations will be applied to the SOM, and in Section 6 the T-point and graph approximations to Sammon's mapping.

5 Application I: SOM in learning metrics

The Self-Organizing Map (SOM; Kohonen, 2001) is one of the best-known neural network algorithms. In this Section we briefly review an earlier SOM that learns metrics (Kaski et al., 2001), present an improved version, and empirically compare the algorithms denoted by SOM-L against classical SOM types.

Table 1

How to compute the learning metrics.

(1) Learn the CMM estimator in Eq. (5) by maximizing the conditional likelihood $L = \sum_{(\mathbf{x},c)} \log \hat{p}(c|\mathbf{x})$.

(2) Choose a or b if distances have to be computed often, and c if only once.

  (a) For the 1-point approximation, compute the Fisher information matrix from Eq. (6) and the distance from Eq. (8).

  (b) For the T-point approximation, choose T according to computational resources and compute Eq. (9).

  (c) For the graph approximation, compute a pairwise distance matrix by Eq. (9) and find the minimum in Eq. (10) by running Floyd's algorithm on the matrix.

5.1 Computing the SOM-L

Basics. The SOM is an ordered lattice of units i with attached model vectors $\mathbf{m}_i$. The standard sequential variant of the SOM training algorithm iterates two steps: winner selection and adaptation. At each iteration t the algorithm selects a best-matching (winner) node $w(t)$ whose model vector has the smallest distance from the sample $\mathbf{x}(t)$. For the SOM-L the distances are naturally computed in the learning metric. In an earlier work (Kaski et al., 2001) we used the 1-point distance approximation d_1, Eq. (8). The winner selection step is then

$$w(t) = \arg\min_i d_1^2(\mathbf{x}(t), \mathbf{m}_i(t)) \ . \tag{11}$$

In the second step the model vectors are adapted towards the sample, in the direction of steepest descent of the distance function. For Riemannian metrics the steepest descent direction is given by the so-called natural gradient (Amari, 1998). It turns out (Kaski et al., 2001) that this yields the standard SOM adaptation rule, i.e.,

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \alpha(t) h_{w(t),i}(\mathbf{x}(t) - \mathbf{m}_i(t)) \ , \qquad (12)$$

where $\alpha(t)$ is the learning rate and $h_{w(t),i}$ is the neighborhood function around the winner. Kohonen (2001) gives instructions on choosing neighborhood functions, map topologies, and learning schedules.

Improvements. For more accurate distance computation, the 1-point distance is replaced by the T-point distance Eq. (9) in the winner search. This implies more computation since the local approximation is computed at several points along a line, separately for each model vector. Larger values of T yield more accurate results but take longer. To avoid excessive computation, a small set of W winner candidates is first chosen by a faster approximation, the 1-point or Euclidean distance; in this paper we will use the former. In Section 7.3 the effect of the choice of T and W on the results is studied empirically.

For the T-point approximation the shortest path is still assumed to be linear, so the direction of adaptation does not change. Its magnitude may, however. In preliminary tests taking the change into account did not improve results, so the standard rule will be used for simplicity.

5.2 How does the metric affect the SOM-L objective?

As mentioned in Section 4 there are two possible ways to apply learning metrics—explicit computation and implicit optimization of a tailored objective function. In this paper we use the first approach. It would be interesting to know whether the resulting algorithm can be interpreted in terms of the second approach. That is, what is the objective function of unsupervised learning methods in the new metric? Here we consider this question for the SOM-L, with some simplifying assumptions.

The basic SOM does not perform gradient descent on an energy function, which makes the question difficult to answer. However, Heskes has proposed a variant of the SOM with an energy function (Heskes, 1999). In the Euclidean metric the energy function is

$$E = E_{p(\mathbf{x})}\{\min_i \sum_j h_{ij} d^2_{\mathbf{I}}(\mathbf{x}, \mathbf{m}_j)\} \tag{13}$$

where $d^2_{\mathbf{I}}$ is the squared Euclidean distance. In the learning metric, it is replaced by d^2_L, the squared learning metric distance.

Assume that d_L can be computed with the simple local form $d^2_L(\mathbf{x}, \mathbf{m}_j) = D_{KL}(p(c|\mathbf{x}), p(c|\mathbf{m}_j))$. It can be shown (Appendix A) that optimizing the cost function Eq. (13) is equivalent to maximizing

$$E_{p(c,j)}\{\log p(c|\mathbf{m}_j)\} \tag{14}$$

where j indexes the SOM units, $p(c, j) = \int_{\mathbf{x}} p(j|\mathbf{x}, c)p(\mathbf{x}, c)d\mathbf{x}$ is the probability that a sample has class c and unit j is picked from its winner neighborhood, and $p(j|\mathbf{x}, c) \propto h_{w(\mathbf{x}),j}$. The maximization is done over $\mathbf{m}_j$ and the winner assignment function $w(\mathbf{x})$.

A quality measure. The above result motivates a measure for the accuracy of the SOM results. We had earlier used a heuristic measure defined as the class purity of auxiliary (test) data on the SOM lattice, given by

$$\sum_{(\mathbf{x},c)} \log \hat{p}(c|h_{w(\mathbf{x})}) \tag{15}$$

where $\hat{p}(c|h_j) \equiv (\sum_{(\mathbf{x}',c'):c'=c} h_{j,w(\mathbf{x}')})/(\sum_{(\mathbf{x}',c')} h_{j,w(\mathbf{x}')})$ is the proportion of class c in the neighborhood $h_j = \{h_{ji}\}_i$ of unit j, which is here Gaussian.

We will now motivate this quality measure through the objective function Eq. (14). We replace the generally unknown distribution $p(c|\mathbf{m}_j)$ at the model vectors by $\hat{p}(c|h_j)$, the distribution of samples of class c within the lattice neighborhood. This change alone would yield $I(C, J)$, the mutual information between the classes and the (soft) winner assignment. In addition, the average $E_{p(c,j)}$ is taken in the limit of zero neighborhood, to heuristically compensate for the difference between winner selection in the standard SOM algorithm and Heskes' version. This finally yields Eq. (15). If $\hat{p}(c|h_j)$ were estimated from learning data instead of test data, the measure would further become a conditional log-likelihood for predicting classes of test samples from the SOM neighborhood; this would be a reasonable alternative measure.

In conclusion, although the measure was heuristically derived, the connections to log-likelihood, mutual information and the SOM-L objective make it reasonable.

5.3 Empirical comparisons

In this section we present empirical comparisons showing that the SOM-Ls are more informative about the auxiliary data than Euclidean SOMs, and that SOM-L uses the auxiliary data better than a simple supervised SOM variant. These comparisons are sanity checks aimed to show that the learning metric in fact does what it promises. We additionally compare the different approximations of learning metrics.

Test setup. Two approximations to path integrals are included: the 1-point and T-point distances ($T = W = 10$). For most experiments the density is estimated by the conditional mixture model (CMM). The main comparison method is the classical supervised SOM, here SOM-S, where the class of the sample is concatenated into the input vectors in a (weighted) 1-out-of-N_C class encoded form. The standard Euclidean SOM, denoted here by SOM-E, is used as a baseline; it does not use the auxiliary data at all.

The comparisons can be divided into two sets. First the two SOM-L versions are compared to the other methods. The choice of density estimator is next justified by comparing SOM-Ls trained with the CMM and MDA2 estimators. The Parzen estimator was computationally too complex, at least for these experiments with the T-point distances. The choice of T and W is studied in Section 7.3.

The methods are compared on four standard machine learning data sets (Table 2). A standard cross-validation procedure and paired t-tests are used to verify the difference between the methods.

SOM-L vs. classical SOMs. The SOM-L with the T-point distance approximation is significantly better than the traditional methods, SOM-E and SOM-S, on all four data sets (Table 3). The faster 1-point distance approximation provides good results on some of the data sets, but on some others it is outperformed by SOM-S, and even by SOM-E on one data set.

Table 2

The data sets used for empirical comparisons

Data set	Dimensions	Classes	Samples
Landsat Satellite Data[a]	36	6	6435
Letter Recognition Data[a]	16	26	20000
Phoneme Data[b]	20	13	3656
TIMIT Data from (TIMIT, 1998)	12	41	14994

[a] from the UCI Machine Learning Repository (Blake and Merz, 1998)

[b] from LVQ_PAK (Kohonen et al., 1992)

Note that SOM-S outperforms SOM-E on all data sets. This gives additional empirical justification for the performance measure Eq. (15), since the measure can detect the intuitively clear difference between the supervised methods and SOM-E which does not use auxiliary data.

Comparing SOM-L versions. The different methods of approximating the learning metrics are compared in Table 4. On all data sets the combination of the conditional mixture model and the T-point distance approximation leads to significantly more accurate maps than any other combination. With the rough 1-point distance approximation there is no clear difference between the two density estimators.

In conclusion, the conditional mixture model (CMM) is the better density estimation method for learning metrics if the distance approximation is accurate enough. Even with the 1-point distance approximation MDA2 is clearly worse

Table 3

Comparison of SOM that learns metrics (SOM-L) with supervised SOM (SOM-S) and Euclidean SOM (SOM-E). Two distance approximations (1-point and T-point) are used in SOM-L. The table shows p-values of paired t-tests between the row and column methods. If an entry is present the method in that row is on the average better than the method in that column. Entries with $p < 0.01$ have been underlined for convenience.

Landsat

	1-point	SOM-S	SOM-E
T-point	4×10^{-4}	10^{-5}	3×10^{-8}
1-point	-	0.04	8×10^{-5}
SOM-S	-	-	10^{-4}

Phoneme

	SOM-S	SOM-E	1-point
T-point	0.003	2×10^{-6}	9×10^{-7}
SOM-S	-	0.008	0.001
SOM-E	-	-	0.20

Letter

	1-point	SOM-S	SOM-E
T-point	6×10^{-8}	10^{-9}	$< 10^{-10}$
1-point	-	2×10^{-8}	$< 10^{-10}$
SOM-S	-	-	$< 10^{-10}$

TIMIT

	SOM-S	1-point	SOM-E
T-point	2×10^{-5}	3×10^{-6}	3×10^{-9}
SOM-S	-	0.02	2×10^{-5}
1-point	-	-	0.004

on some data sets. Therefore CMM can be recommended, and it will be used in the further tests in Section 7.

5.4 Visualizing the SOM-L

All standard SOM visualizations are applicable to SOM-L as well. In addition, the relative importance of the components (coordinates) of primary data can be visualized on the SOM display. The relative importance of the ith coordinate at $\mathbf{x}$ is (Kaski et al., 2001)

$$r_i(\mathbf{x}) \equiv \left(\frac{d_1^2(\mathbf{x}, \mathbf{x} + \mathbf{e}_i)}{\sum_j d_1^2(\mathbf{x}, \mathbf{x} + \mathbf{e}_j)} \right)^{1/2} = \left(\frac{\mathbf{e}_i^T \mathbf{J}(\mathbf{x}) \mathbf{e}_i}{\sum_j \mathbf{e}_j^T \mathbf{J}(\mathbf{x}) \mathbf{e}_j} \right)^{1/2} \tag{16}$$

where d_1 is the 1-point distance approximation, Eq. (8), and $\mathbf{e}_i$ is the unit vector with the ith element being equal to one and others zero. The relevance

Table 4

Comparison of density estimators and distance approximation methods for SOM-L. The entries are p-values of paired t-tests, and values below 0.01 are underlined for convenience. If an entry is present the method on that row is on average better. The entries typed in boldface are exceptions; in those cases the method on that column is better. CMM: conditional mixture model, MDA2: mixture discriminant analysis 2. The distance approximations used are 1-point and T-point.

Landsat

	CMM, 1-point	MDA2, 1-point	MDA2, T-point
CMM, T-point	$\underline{4 \times 10^{-4}}$	$\underline{6 \times 10^{-7}}$	$\underline{8 \times 10^{-7}}$
CMM, 1-point	-	$\underline{8 \times 10^{-4}}$	0.04

Letter

	CMM, 1-point	MDA2, 1-point	MDA2, T-point
CMM, T-point	$\underline{6 \times 10^{-8}}$	$\underline{10^{-10}}$	$\underline{10^{-10}}$
CMM, 1-point	-	$\underline{< 10^{-10}}$	$\underline{< 10^{-10}}$

Phoneme

	CMM, 1-point	MDA2, 1-point	MDA2, T-point
CMM, T-point	$\underline{9 \times 10^{-7}}$	$\underline{2 \times 10^{-8}}$	$\underline{0.003}$
CMM, 1-point	-	**0.02**	$\underline{\mathbf{10^{-4}}}$

TIMIT

	CMM, 1-point	MDA2, 1-point	MDA2, T-point
CMM, T-point	$\underline{3 \times 10^{-6}}$	$\underline{2 \times 10^{-5}}$	$\underline{0.001}$
CMM, 1-point	-	**0.03**	$\underline{\mathbf{5 \times 10^{-5}}}$

computed at model vector $\mathbf{m}_j$ will be visualized by a gray shade on top of SOM unit j.

The Letter Recognition data is visualized in Fig. 1. One component plane and the corresponding importance plane are shown, and simple conclusions are drawn to explain how the visualizations could be used. The conclusions obtained from this visualization are very simple because of the data set, but they are intuitively clear.

Fig. 1 also demonstrates the difference between SOM-L and SOM-E. The map units with the same label (chosen by majority voting) are close-by on SOM-L, but on SOM-E some of the labels (e.g. "I" and "B") are dispersed into a number of locations on the map.

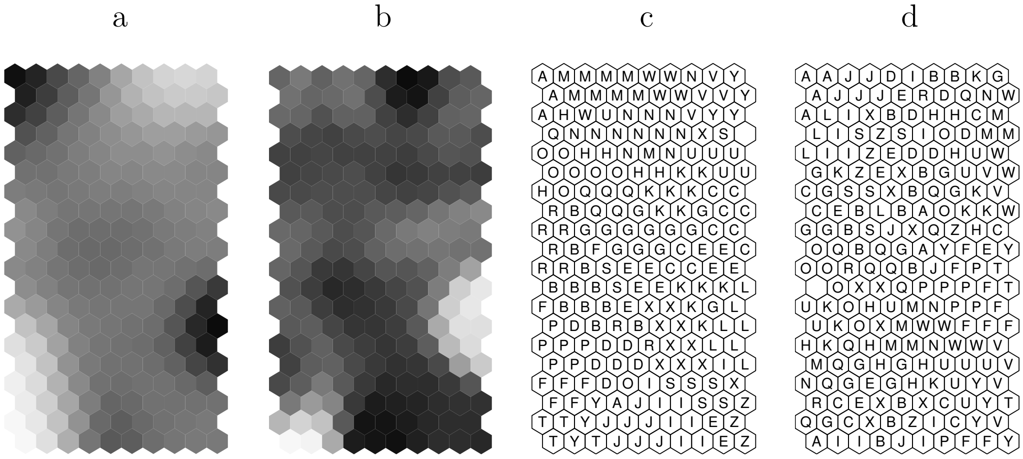


Fig. 1. Sample SOM-L visualizations for Letter Recognition data. (a) The component plane showing, with gray shades, the values of one of the components (called "height of the centroid of mass") in the map units, and (b) the importance of the component. The map has been computed with the T-point distance approximation and the units have been labeled by majority voting (c). The feature has a high importance (light shade; large contribution to the local distance) on two separate regions of the map. One region corresponds to the letter "L" and the other to the letter "T". The component plane reveals that in "L" the mass centroid is low whereas in "T" it is high. The majority voting on SOM-E (d) is given for comparison; the labels are clearly more dispersed on the map compared to the SOM-L.

6 Application II: Sammon's mapping by learning metrics

Metric multidimensional scaling (MDS) methods (see Borg and Groenen, 1997) are often used to visualize similarities of data samples; they represent the data in a low-dimensional space that tries to preserve pairwise distances. Sammon's mapping (Sammon, Jr., 1969) is a well-known MDS method which emphasizes preservation of small distances.

We show next how to compute a Sammon's mapping based on learning metric distances. As in the SOM-L, the aim is to study the primary data; the metric only emphasizes relevant differences.

6.1 Computing the Sammon-L

Sammon's mapping is based on the matrix of pairwise distances between data points; there is no need to know distances between arbitrary points. Therefore the learning metric is only needed in computing this matrix, after which standard methods to compute the Sammon's mapping can be applied. The same approach works with any algorithm operating on the pairwise distance matrix.

In principle the distance matrix could be computed with any of the distance approximations. The simple 1-point approximation is likely to be too inaccurate here but the T-point and graph approximations are suitable. Since the distances need to be computed only once in the beginning, the graph approximation is the preferred choice. Here the data points themselves form the vertices of the graph; the distances are then computed more accurately where the data is dense, which is sensible.

To our knowledge no "supervised" variant of Sammon's mapping exists. The straightforward thought of concatenating the class to data vectors as in SOM-S would not tolerate unlabeled data. Hence we only perform a basic test to ensure the Sammon-L finds class structure better than the standard Sammon's mapping that uses Euclidean distances, here denoted Sammon-E.

Test setup. The two variants of Sammon's mapping are compared with t-tests in a standard ten-fold cross-validation scheme. The data are described in Table 2. The tests are similar to the SOM-L tests; the difference is that Sammon's mapping does not generalize to new data (aside from heuristic generalizations). Hence direct validation of a computed mapping is not possible. However, the metric does generalize, so it can be used to compute a distance matrix for test data. The quality of the resulting Sammon's mapping, computed for test data but in a metric computed from learning data, can then be evaluated.

The SOM-L quality measure Eq. (15) is defined for a discrete-valued mapping and cannot be used here. Instead, the quality of the mappings is measured indirectly by the performance of a nonparametric leave-one-out K-nearest-neighbor (KNN) classifier in the output space. Each sample is classified based on K nearest samples; low error means that samples of the same class have been mapped close-by. For each method the neighborhood size K was validated from the range 1 to 100. For each distance matrix the KNN result was averaged over 20 restarts of the Sammon's mapping algorithm.

Two distance approximations, T-point ($T = 10$) and graph distance, are used For Sammon-L. The latter yields better approximations but is computationally heavier. The density is estimated with the conditional mixture model with 30 kernels; the choice was made because of its good performance in the SOM-L tests.

Results. The resulting Sammon-L mappings were significantly more informative (t-test, $p < 0.01$) than Sammon-E on all data sets (Table 5). The graph approximation further improved the results on Landsat and Letter Recognition data sets; on the other two sets the difference between the Sammon-L variants was insignificant.

Table 5

Indirect measure of the goodness for the Sammon's mappings. Average percentage of correct KNN-classifications in the output space are given over cross-validation folds for two Sammon-L variants and Sammon-E. "Graph": Sammon-L with graph search; "T-point": Sammon-L without graph search. The Sammon-L variants are both significantly better than Sammon-E on all data sets.

Data set	Graph	T-point	Sammon-E
Landsat	88.95	87.49	82.69
Letter	59.26	56.15	14.29
Phoneme	90.77	90.48	80.38
TIMIT	40.00	39.96	30.40

The Sammon-L with the graph approximation is illustrated in Fig. 2 on Letter Recognition data. The new metric has emphasized differences where the class distributions change, leading to increased class separation and more distinctive clustering of classes. The class structure is hardly visible in the Sammon-E mapping shown for comparison, whereas some classes can easily be separated in the Sammon-L.

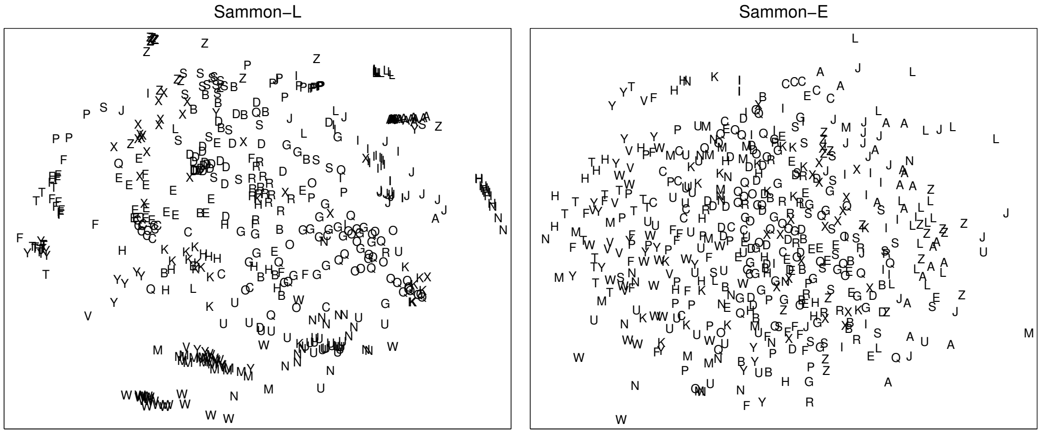


Fig. 2. The learning metric used in Sammon-L leads to clearly more separated classes in contrast to the Sammon-E. For example, the letter "W" is grouped into a tight cluster at the bottom of the Sammon-L mapping, while in Sammon-E it is dispersed to a large area on the left side of the mapping. The letters are samples of capital characters, which explains the similarity of e.g. "O" and "Q".

7 Why is this metric useful?

In Section 3 the metric was claimed to have useful properties. In this section we take a closer look at the two most important ones. We illustrate why topology preservation is useful in data exploration, and study empirically how well the metric distinguishes between relevant and irrelevant variation in data.

Finally we study the tradeoff between complexity and accuracy of the introduced approximation methods for the metric. The question is how complex approximations are needed in practice for good results.

The learning metric is first defined locally, and the local definition is extended by path integrals to the whole space. This has a rigorous interpretation: it is a Riemannian metric studied for instance in the information geometry literature.

The procedure is somewhat complicated, which may bring up the question of why not simply use the local definition globally. Distances between any two points would be measured by the Kullback-Leibler divergence in Eq. (3), or any other distributional distance measure. [4]

Both choices are sensible but the results will be very different. In particular, the globally defined metric would not be able to distinguish modes of multimodal (class) distributions, simply because points having the same class distribution would be considered identical. This tears the topology of the data space.

We argue that preserving the topology by defining the metric locally is important for instance for identifying subcategories of the classes, and for analyzing the structure and relationships of the classes in the primary data space.

A toy example. We illustrate the difference between the Riemannian distance and the Kullback-Leibler divergence on artificial data. The data are uniformly distributed over a square. One of the classes (labeled "effect") is bimodal and the other class forms the background.

This artificial data is effectively one-dimensional by both the Kullback-Leibler divergence and the Riemannian metric. Therefore, one-dimensional Sammon's

[4] Jensen-Shannon distance would probably be a better choice since it is symmetric.

mappings were computed to visualize their difference (Fig. 3).

Both mappings group the "background" into one connected area. The Kullback-Leibler divergence does not distinguish the two "effect" clusters, since they have identical class distributions. The mapping based on the Riemannian metric, however, preserves the multimodality. Euclidean Sammon's mapping is included for comparison; naturally, it cannot separate the categories of the uniformly-distributed data.

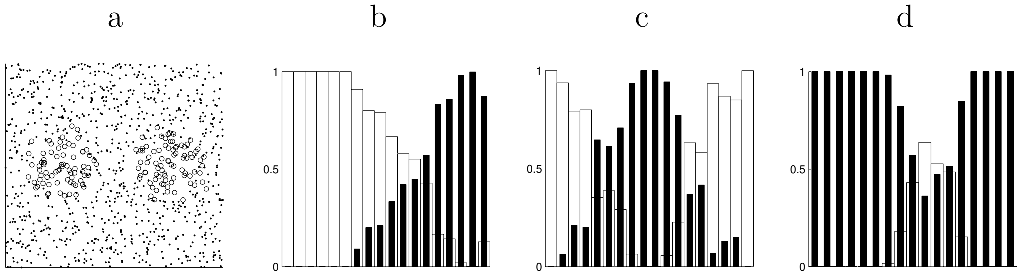


Fig. 3. Demonstration of the importance of topology preservation. The data (a) has two classes: "background" (dots) and "effect" (circles). (b-d) A one-dimensional Sammon's mapping was computed in three different metrics to project the two-dimensional data samples to the horizontal axis. The vertical axis shows probabilities of the background (black bars) and effect (white bars) classes, estimated within bins of fixed width. (b) Kullback-Leibler divergence. (c) Riemannian learning metric. (d) Euclidean metric.

7.2 Ability to focus on important variation

Although the learning metric theoretically depends only on changes that affect the auxiliary variable C, the approximations used in practical computation are not perfect. Approximation of the distances from finite data becomes the more difficult the more unimportant variation (noise) there is in the data. Here we empirically study how increasing the amount of unimportant variation affects the organization of SOM-L.

Test setup. To create unimportant variation in a dataset while keeping its other properties as constant as possible, we randomly picked one of the dimensions (variables) of primary data and randomly permuted the values of that dimension among the data samples. This removes statistical dependency between the selected variable and the remaining (primary and auxiliary) variables, but otherwise preserves the data distribution. More noise is added by repeating the procedure for more variables.

The quality measure Eq. (15) will be used to evaluate the overall quality of the SOMs. The quality will naturally drop when more dimensions are permuted, for two reasons: (i) information about auxiliary data is lost simply because fewer informative dimensions are left, and (ii) the noise may disturb distance approximation. We are interested in (ii), and hence introduce a new complementary measure that focuses on it.

If the unimportant dimensions do not affect the distances in the learning metric, winner search is determined by the remaining dimensions. Therefore the data won by each map node should have the same distribution over the unimportant dimensions, and each model vector should converge to the same value (the mean) along them. We can then measure the noise caused by the unimportant dimensions by the average (Euclidean) variance of the model vectors along them.

Data. Two data sets are used: Letter Recognition data (Table 2) and a new gene expression data (Su et al., 2002). The gene data are derived from sample pairs $(\mathbf{x}, \mathbf{y})$ of expression level vectors of human genes and corresponding (in the sense of having similar DNA sequences) mouse genes in a set of their respective tissues. Human expressions are regarded as primary data and mouse tissues are used as categories of an auxiliary variable: if the expression in a mouse tissue is high after preprocessing (over one standard deviation above the gene-wise average), the corresponding human gene was assigned to that category. Up to 16 dimensions were permuted for gene data (out of 42) and up to 8 for Letter Recognition (out of 16).

Results. As expected, the performance of both SOM-L and SOM-E decreases when uninteresting variation (number of permuted dimensions) increases. The SOM-L seems to lose performance roughly at the same rate as SOM-E (Fig. 4), but the performance is clearly better at any given point, and even with a few permuted dimensions the SOM-L outperforms the SOM-E that was computed using the intact data.

The effect of unimportant variation on the approximation errors is visualized in Fig. 5, which shows average variances for the permuted SOM-L dimensions, relative to the variance of the same dimensions in the SOM-L of original data. SOM-E results are again included for comparison; SOM-E variances change because permutation removes dependencies between permuted and non-permuted primary variables.

The SOM-L variance for permuted dimensions is close to zero, and clearly smaller than the SOM-E variance on both sets. Therefore, the permutation

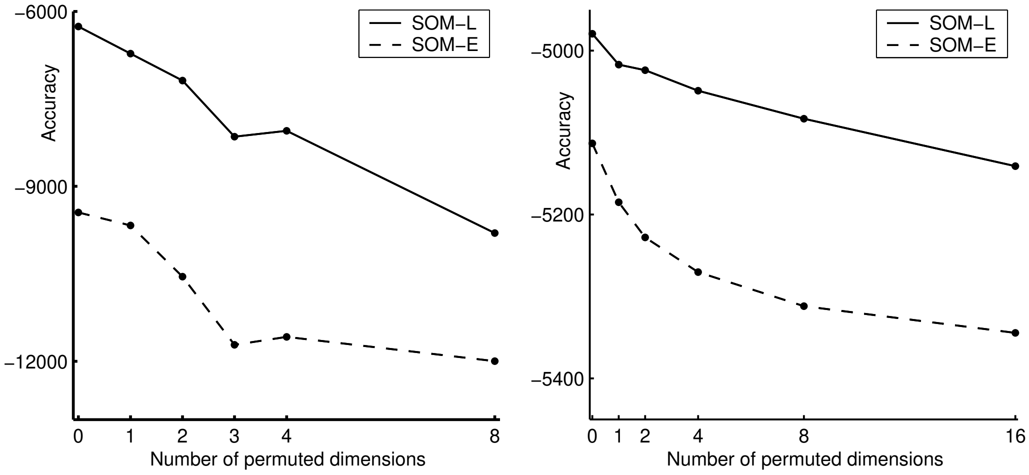


Fig. 4. Replacing interesting with uninteresting variation decreases the accuracy of both SOM-L and SOM-E at a similar rate. On the Letter Recognition data (left), even when half of the dimensions are permuted, SOM-L performs as well as the SOM-E for the original (unpermuted) data. On the gene data (right) permuting roughly 10 out of 42 dimensions reduces the SOM-L performance to the level of a SOM-E for original data.

does not seem to cause significant noise for SOM-L training.

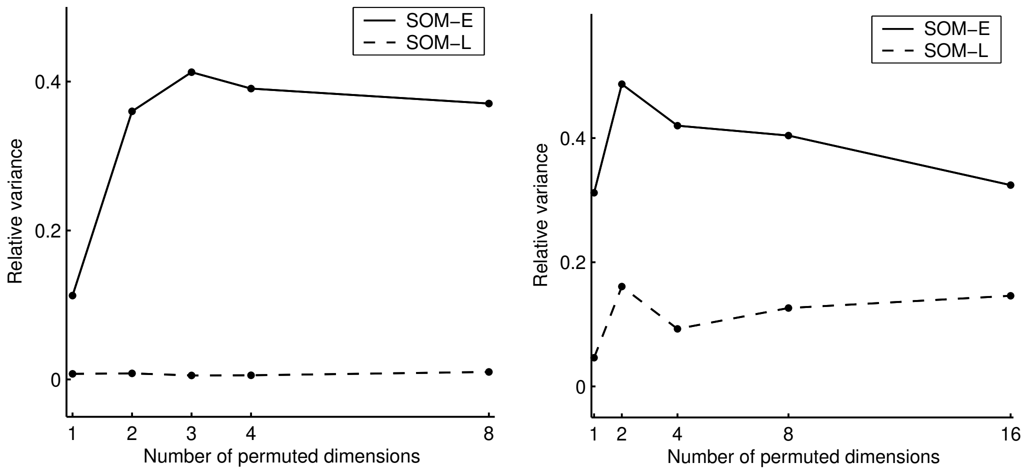


Fig. 5. Effect of irrelevant variation on SOM-L, measured by variances of model vectors along permuted dimensions relative to variances in the original data. Solid line: SOM-E, dashed: SOM-L.

The complexity-quality tradeoff. To apply explicit learning metrics in practice, we need to estimate the density and approximate the distances, as described in Section 4. Here we study the complexity-quality tradeoff, that is, how much the accuracy improves by increasing the complexity of the computation (of the estimates and approximations). This is done in order to provide a practical recommendation for "sufficient" complexity. The experiments also justify the earlier heuristic choices. The density estimators were already compared in Section 5.3, and here we focus on the distance approximation.

Previous choices. For the tests in Sections 5.3 and 6.2, the distance approximations were fixed. For the density estimate, the number of mixture components was chosen by preliminary SOM-L experiments. The variance of Gaussian components was validated separately in each cross-validation fold by dividing the training data into learning and validation subsets. The best variance for SOM-L training seems to be somewhat larger than the maximum likelihood value (best density estimator), especially for SOM-Ls trained with the 1-point approximation. Hence, good candidate variances can be picked near the maximum likelihood value.

The tests. We studied how the complexity parameter T used for the T-point and graph approximations affects SOM-L and Sammon-L performance.

For SOM-L we in practice must also consider the speedup parameter W. For the fastest value $W = 1$ the distance approximation equals the simpler speedup approximation (here 1-point distance); the other extreme of W equal to the number of SOM nodes implies no speedup.

We tested the effect of increasing the computational complexity on two data sets (Landsat and Phoneme), by computing SOM-Ls and Sammon-Ls for a range of parameter values, and measuring their goodness on a separate test set. SOM-E, SOM-S, and Sammon-E are used as baselines.

For SOM-L the parameter σ of the CMM density estimator is validated for each value of T and W. For SOM-S the class weight is validated. For Sammon-L, σ and the neighborhood of the KNN-classifier (performance measure) are validated for each value of T. For Sammon-E the KNN neighborhood is validated.

The results. Fig. 6 shows the SOM-L performances and Fig. 7 the Sammon-L performances on the data sets. The SOM-L performance varies somewhat but the SOM-L is better than the other methods for all values $T \geq 5$, $W \geq 10$. For both SOM-L and Sammon-L the performance increases with the complexity, but compared to the increase in computation time the gain is small already with quite low values. The SOM-L computation time is proportional to TW, whereas the Sammon-L distance computation is linear in T. The values $T = W = 10$ used in Section 5.3 seem sufficient with both methods and on both data sets; performance does not increase markedly for larger values.

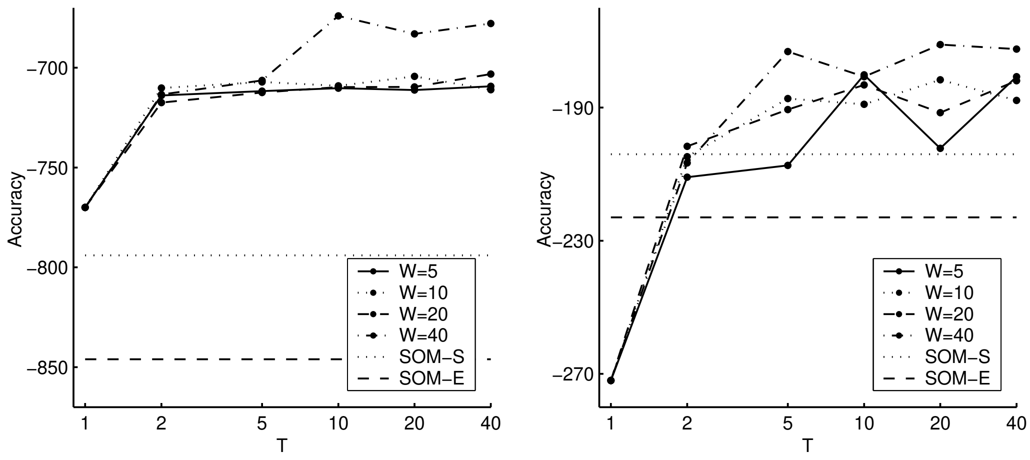


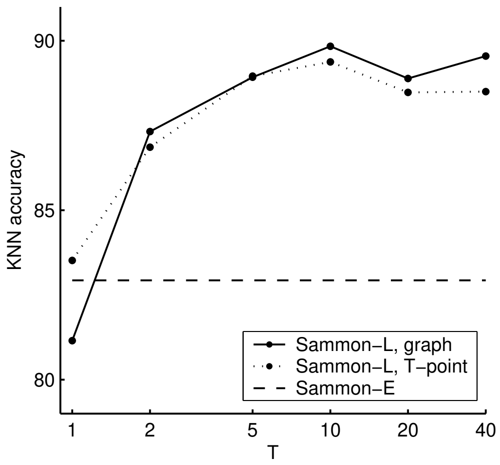
Fig. 6. SOM accuracy for Landsat (left) and Phoneme (right) data as a function of the computational complexity. The effect of the speedup parameter W is not altogether clear, but in general the larger W work better. On the Landsat data the SOM-L results are always better than the comparison methods (SOM-S and SOM-E), but on the Phoneme data the least accurate approximations are not accurate enough.

8 Conclusions and discussion

The learning metrics principle addresses the garbage in—garbage out problem of unsupervised learning, by using auxiliary data to separate the important variation from unimportant variation. Choosing auxiliary data specifies what is important; this is often easier than hand-tuning the features.

The learning metrics share one aspect with supervised learning: both use auxiliary data. The difference is that here only the metric is supervised; the data analysis task itself can be unsupervised. Note also that analyzing partially labeled data is easy; after the metric has been estimated it applies to all data, labeled or not.

In a sense the concept of learning metrics is complementary to so-called



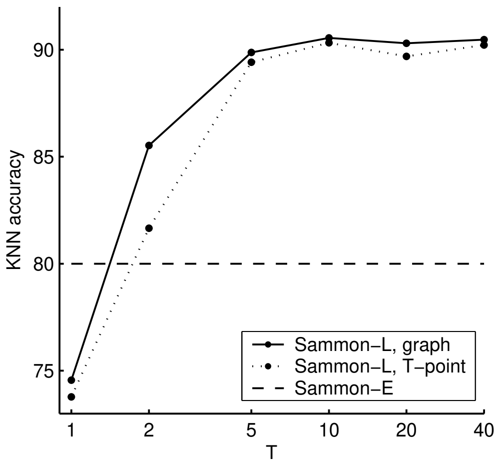


Fig. 7. The KNN-classification accuracy of Sammon-L mapping increases rather quickly with the distance parameter T on both the Landsat (left) and Phoneme (right) data sets. The difference to Sammon-E performance is clear already with $T = 2$ and using values larger than $T = 5$ seems unnecessary. The local approximation ($T = 1$) is not sufficient, as was to be expected.

semisupervised learning, where unlabeled samples are used to help in a supervised task. Here, auxiliary labels are used to help in unsupervised data analysis.

The choice of auxiliary data is important. It specifies everything that is relevant, and may suppress unknown but possibly interesting properties of the data. In a knowledge discovery task one could "regularize" the learning metrics by the Euclidean distance as suggested in (Kaski et al., 2001), allowing effects that are strong enough to turn up even when considered irrelevant with respect to the auxiliary data.

The learning metrics have a flexible Riemannian form, and are well suited for data analysis since they do not tear the topology of the data space. Data analysis can even provide new insights into the importance of the features. Such metrics are also more general than those found by extracting a smaller set of features. The downside is that computing the metrics analytically is usually not possible, and approximations must make a tradeoff between quality and computational complexity.

In this paper we presented generally applicable, practical approximations to computing the metrics and showed that they improved the performance of two well-known unsupervised methods, the self-organizing map and Sammon's mapping. We also demonstrated and studied properties of the metrics.

Based on the results the following guidelines for approximating the metrics computationally can be given. For self-organizing maps, use the T-point approximation with speedup. For Sammon's mappings, use either the T-point or graph approximation, depending on computational resources. For both methods, approximate densities by the conditional mixture model (CMM).

Although the metric was applied here to only two data analysis methods, it is more general. In particular the method of computing pairwise distances between all data pairs for Sammon's mapping (Section 6) is readily applicable to a variety of methods such as other MDS methods or hierarchical clustering algorithms.

The main unsolved theoretical question in the current approach is how to combine the density estimation and distance approximation steps of computing the metrics, in other words, what is the optimal density estimator for a particular distance approximation. In this paper we presented practical tools that clearly improve data analysis methods but still leave room for further work.

References

Amari, S., 1998. Natural gradient works efficiently in learning. Neural Computation 10, 251–276.

Becker, S., 1996. Mutual information maximization: models of cortical self-organization. Network: Computation in Neural Systems 7, 7–31.

Becker, S., Hinton, G. E., 1992. Self-organizing neural network that discovers surfaces in random-dot stereograms. Nature 355, 161–163.

Blake, C. L., Merz, C. J., 1998. UCI repository of machine learning databases. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Borg, I., Groenen, P., 1997. Modern Multidimensional Scaling. Springer, New York.

Devroye, L. P., Wagner, T. J., March 1980. Distribution-free consistency results in nonparametric discrimination and regression function estimation. The Annals of Statistics 8 (2), 231–239.

Friedman, N., Mosenzon, O., Slonim, N., Tishby, N., 2001. Multivariate information bottleneck. In: Proc. Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI). Morgan Kaufmann Publishers, San Francisco, CA, pp. 152–161.

Hammer, B., Villmann, T., 2002. Generalized relevance learning vector quantization. Neural Networks 15, 1059–1068.

Hastie, T., Tibshirani, R., Buja, A., 1995. Flexible discriminant and mixture models. In: Kay, J., Titterington, D. (Eds.), Neural Networks and Statistics. Oxford University Press, Oxford.

Heskes, T., 1999. Energy functions for self-organizing maps. In: Oja, E., Kaski, S. (Eds.), Kohonen Maps. Elsevier, Amsterdam, pp. 303–316.

Jaakkola, T. S., Haussler, D., 1999. Exploiting generative models in discriminative classifiers. In: Kearns, M. S., Solla, S. A., Cohn, D. A. (Eds.), Advances in Neural Information Processing Systems 11. Morgan Kaufmann Publishers, San Mateo, CA, pp. 487–493.

Kaski, S., Peltonen, J., 2003. Informative discriminant analysis. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003). AAAI Press, Menlo Park, CA, pp. 329–336.

Kaski, S., Sinkkonen, J., 2004. Principle of learning metrics for data analysis. The Journal of VLSI Signal Processing-Systems for Signal, Image, and Video Technology, Special Issue on Data Mining and Biomedical Applications of Neural Networks 37, 177–188.

Kaski, S., Sinkkonen, J., Klami, A., 2003. Regularized discriminative clustering. In: Molina, C., Adali, T., Larsen, J., Van Hulle, M., Douglas, S., Rouat, J. (Eds.), Neural Networks for Signal Processing XIII. IEEE, New York, NY, pp. 289–298.

Kaski, S., Sinkkonen, J., Peltonen, J., 2001. Bankruptcy analysis with self-organizing maps in learning metrics. IEEE Transactions on Neural Networks 12, 936–947.

Kohonen, T., 2001. Self-Organizing Maps, 3rd Edition. Springer, Berlin.

Kohonen, T., Kangas, J., Laaksonen, J., Torkkola, K., 1992. LVQ_PAK: A program package for the correct application of Learning Vector Quantization algorithms. In: Proceedings of IJCNN'92, International Joint Conference on Neural Networks. Vol. I. pp. 725–730.

Kullback, S., 1959. Information Theory and Statistics. Wiley, New York.

Rattray, M., 2000. A model-based distance for clustering. In: Proceedings of IJCNN-2000, International Joint Conference on Neural Networks. IEEE Service Center, Piscataway, NJ, pp. 4013–4016.

Sammon, Jr., J. W., 1969. A nonlinear mapping for data structure analysis.

IEEE Transactions on Computers C-18, 401–409.

Sinkkonen, J., Kaski, S., 2002. Clustering based on conditional distributions in an auxiliary space. Neural Computation 14, 217–239.

Su, A. I., Cooke, M. P., Ching, K. A., Hakak, Y., Walker, J. R., Wiltshire, T., Orth, A. P., Vega, R. G., Sapinoso, L. M., Moqrich, A., Patapoutian, A., Hampton, G. M., Schultz, P. G., Hogenesch, J. B., 2002. Large-scale analysis of the human and mouse transcriptomes. PNAS 99, 4465–4470.

TIMIT, 1998. TIMIT. CD-ROM prototype version of the DARPA TIMIT acoustic-phonetic speech database.

Tishby, N., Pereira, F. C., Bialek, W., 1999. The information bottleneck method. In: 37th Annual Allerton Conference on Communication, Control, and Computing. Urbana, Illinois, pp. 368–377.

Torkkola, K., 2003. Feature extraction by non-parametric mutual information maximization. Journal of Machine Learning Research 3, 1415–1438.

Tsuda, K., Kawanabe, M., Rätsch, G., Sonnenburg, S., Müller, K.-R., 2002. A new discriminative kernel from probabilistic models. Neural Computation 14, 2397–2414.

Xing, E. P., Ng, A. Y., Jordan, M. I., Russell, S., 2003. Distance metric learning, with application to clustering with side information. In: Becker, S., Thrun, S., Obermayer, K. (Eds.), Advances in Neural Information Processing Systems 15. MIT Press, Cambridge, MA.

Appendix A: Heskes' cost function in learning metrics

We consider two cases: one where the conditional class probabilities are known and one where they are not. We derive a direct cost function for the first case and an upper bound for the second case.

Known class probabilities. The learning metrics version of the energy function in Eq. (13) can be rewritten as

$$E = E_{p(\mathbf{x})}\{\min_i \sum_j h_{ij} d_L^2(\mathbf{x}, \mathbf{m}_j)\} = \min_w E_{p(\mathbf{x})}\{\sum_j h_{w(\mathbf{x}),j} d_L^2(\mathbf{x}, \mathbf{m}_j)\} \quad (17)$$

where the minimum on the right is taken over all functions $w(\mathbf{x})$ that output an index of a SOM node; they are here denoted *winner assignment functions.*

Assuming the local distance definition (Kullback-Leibler divergence) can be used, the function can be further rewritten as

$$\begin{aligned} E &= \min_w E_{p(\mathbf{x})}\{\sum_j h_{w(\mathbf{x}),j} D_{KL}(p(c|\mathbf{x}), p(c|\mathbf{m}_j))\} \\ &= \min_w E_{p(\mathbf{x})}\{\sum_j h_{w(\mathbf{x}),j} \sum_c p(c|\mathbf{x}) \log \frac{p(c|\mathbf{x})}{p(c|\mathbf{m}_j)}\} \\ &= C_1 - \max_w E_{p(\mathbf{x})}\{\sum_{j,c} h_{w(\mathbf{x}),j} p(c|\mathbf{x}) \log p(c|\mathbf{m}_j)\} \\ &= C_1 - \max_w E_{p(\mathbf{x},c)}\{\sum_j h_{w(\mathbf{x}),j} \log p(c|\mathbf{m}_j)\}\ , \end{aligned} \quad (18)$$

where we assumed a fixed neighborhood total $\sum_j h_{w(\mathbf{x}),j} = C_0$ and denoted $C_1 = -C_0 \cdot E_{p(\mathbf{x})}\{H(C|\mathbf{x})\} = -C_0 \cdot H(C|X)$, which is constant with respect to the map parameters (model vectors $\mathbf{m}_j$). Therefore minimizing E with respect to the map parameters is equivalent to maximizing

$$E_{p(\mathbf{x},c)}\{\sum_j h_{w(\mathbf{x}),j} \log p(c|\mathbf{m}_j)\} \quad (19)$$

with respect to both the prototype vectors and the winner assignment function $w(\mathbf{x})$. Denoting $p(j|\mathbf{x}, c) = p(j|\mathbf{x}) \equiv h_{w(\mathbf{x}),j}/C_0$ this further simplifies to

$$C_0 E_{p(\mathbf{x},c,j)}\{\log p(c|\mathbf{m}_j)\} = C_0 E_{p(c,j)}\{\log p(c|\mathbf{m}_j)\} \quad (20)$$

where $p(c,j) = \int_{\mathbf{x}} p(j|\mathbf{x},c)p(\mathbf{x},c)d\mathbf{x}$ and C_0 is constant. This yields Eq. (14).

Unknown class probabilities. When the class probabilities are not known, an upper bound for Eq. (14) can be derived as follows. The above can be rewritten using an estimate $\hat{p}$, as

$$E_{p(c,j)}\{\log p(c|\mathbf{m}_j)\} = E_{p(c,j)}\{\log \hat{p}(c|\mathbf{m}_j)\} + E_{p(c,j)}\{\log \frac{p(c|\mathbf{m}_j)}{\hat{p}(c|\mathbf{m}_j)}\} \qquad (21)$$

The first term is maximized with respect to the class estimate when $\hat{p}(c|\mathbf{m}_j) = p(c|j)$, the proportion of samples having class c and assigned to j, where again $p(j|\mathbf{x},c) \equiv h_{w(\mathbf{x}),j}/C_0$. In that case the first term is simply $-H(C|J)$, the second term is $-E_{p(j)}\{D_{KL}(p(c|j), p(c|\mathbf{m}_j))\}$, and we may maximize the upper bound

$$\begin{aligned} -H(C|J) &= E_{p(c,j)} \log p(c|\mathbf{m}_j) + E_{p(j)}\{D_{KL}(p(c|j), p(c|\mathbf{m}_j))\} \\ &\geq E_{p(c,j)} \log p(c|\mathbf{m}_j)\ . \qquad (22) \end{aligned}$$

The "tightness" of the bound depends on the average Kullback-Leibler divergence; it is small when Eq. (17) is small.

Notice that $-H(C|J)$ is just a constant $H(C)$ away from $I(C,J)$, the mutual information between a (soft) winner assignment and the auxiliary variable. Maps with a high $I(C,J)$ can therefore be called informative.