MTTTS17

Dimensionality Reduction and Visualization

Spring 2020, 5cr

Jaakko Peltonen

Lecture 9: Metric Learning

Motivation – metric learning

- Metric learning means learning a better metric (better distance function) between the original high-dimensional data than the original metric that one starts with
- “Better” can mean many things, for example better separation between classes of data, better correspondence to some known properties, etc.
- Metric learning is not dimensionality reduction by itself. However, it has clear connections to dimensionality reduction, and can be used as part of dimensionality reduction.

Motivation – metric learning

- Many statistical methods rely on distances as much or more than they do on feature values:
  - nearest neighbor regression/classification uses distances to find the nearest neighbors
  - many clustering approaches such as k-means use distances as part of the algorithm to optimize the clustering
  - in information retrieval, “best” results are often the ones most similar to the query according to some distance
- Learning a good distance function between features can thus be as important as learning which features to use.
- Handcrafting good metrics is hard, automatic learning from data is needed
- Computer vision, information retrieval, and bioinformatics (e.g. learning dissimilarities between DNA sequences) are prominent application areas of metric learning

Types of metrics

- Metrics can be simple or complicated functions of data features.
- The Euclidean metric is a simple squared sum of coordinate differences.

$$d^2(\mathbf{x},\mathbf{x}') = ||\mathbf{x}-\mathbf{x}'||^2 = (\mathbf{x}-\mathbf{x}')^\top(\mathbf{x}-\mathbf{x}') = \sum_{k=1}^{d}(x_k - x'_k)^2$$

- Norm-independent distance is related to cosine similarity:

$$d^2(\mathbf{x},\mathbf{x}') = \left\|\frac{\mathbf{x}}{||\mathbf{x}||} - \frac{\mathbf{x}'}{||\mathbf{x}'||}\right\|^2 = 2 - 2\frac{\mathbf{x}^\top\mathbf{x}'}{||\mathbf{x}||\cdot||\mathbf{x}'||} = 2 - 2\cos(\mathbf{x},\mathbf{x}')$$

Types of metrics

- A Mahalanobis metric is described by a positive semidefinite metric matrix **A**:

$$d^2_{\mathbf{A}}(\mathbf{x},\mathbf{x}') = (\mathbf{x}-\mathbf{x}')^\top \mathbf{A}(\mathbf{x}-\mathbf{x}') = \sum_{k=1}^{d}\sum_{l=1}^{d}(x_k - x'_k)A_{kl}(x_l - x'_l)$$

- If **A** is diagonal the metric is just feature weighting:

$$d^2_{\mathbf{A}}(\mathbf{x},\mathbf{x}') = \sum_{k=1}^{d}(x_k - x'_k)^2 A_{kk} = \sum_{k=1}^{d}(\sqrt{A_{kk}}(x_k - x'_k))^2$$

- The traditional Mahalanobis metric uses **A**=**C**$^{-1}$ where **C** is the covariance matrix of the data. This metric appears inside the exponential term of a multidimensional Gaussian density function:

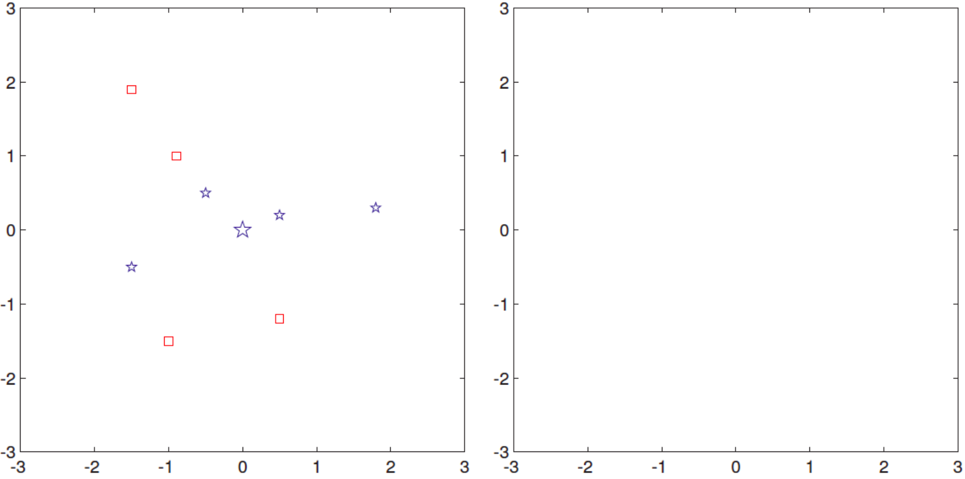
$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}\sqrt{|\mathbf{C}|}}\exp\left(-(\mathbf{x}-\boldsymbol{\mu})^\top \mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu})/2\right)$$

- We call the metric with any **A** a Mahalanobis metric.

Pictures from slides by Kerstin Bunte

Types of metrics

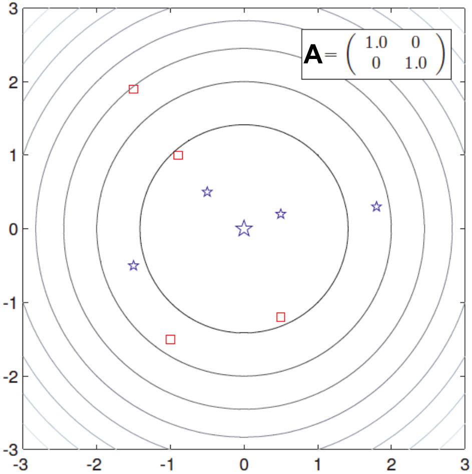
- Adjusting a Mahalanobis metric can make it easier to, for example, distinguish between classes of data: assume d(**x**,**w**) = (**x**-**w**)^T **A** (**x**-**w**)
- A non-diagonal Mahalanobis metric can take into account not just feature importances, but importance of feature combinations

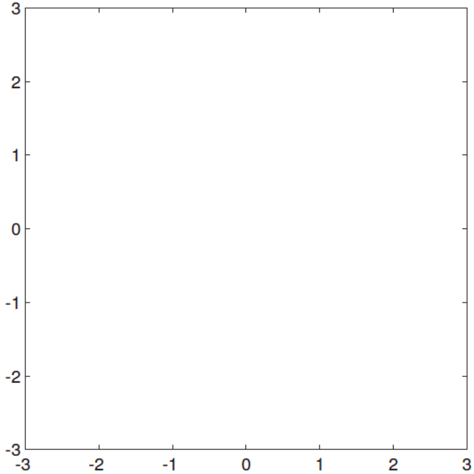


Pictures from slides by Kerstin Bunte

Types of metrics

- Adjusting a Mahalanobis metric can make it easier to, for example, distinguish between classes of data: assume d(**x**,**w**) = (**x**-**w**)ᵀ **A** (**x**-**w**)
- A non-diagonal Mahalanobis metric can take into account not just feature importances, but importance of feature combinations

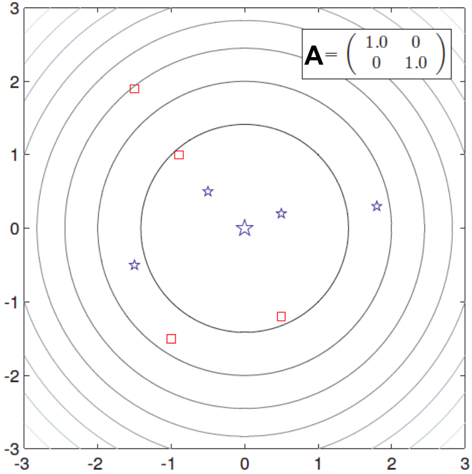


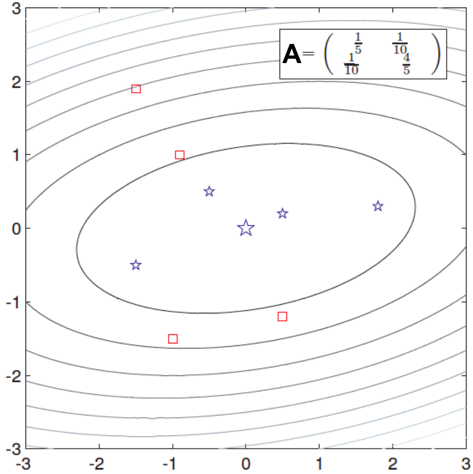


Pictures from slides by Kerstin Bunte

Types of metrics

- Adjusting a Mahalanobis metric can make it easier to, for example, distinguish between classes of data: assume d(**x**,**w**) = (**x**-**w**)ᵀ **A** (**x**-**w**)
- A non-diagonal Mahalanobis metric can take into account not just feature importances, but importance of feature combinations





Types of metrics

- Nonlinear metrics can be described in several ways:

  Globally through an explicit transformation: For example any nonlinear transformation **y** = **f**(**x**), followed by an Euclidean or Mahalanobis metric between the transformed features.

  - Thus learning any transformation **f** for the data (followed by a traditional metric) can be seen as learning a metric for the data:

$$d_{\mathbf{f}}^2(\mathbf{x}_1, \mathbf{x}_2) = ||\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2)||^2 = (\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2))^\top (\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2))$$

  - The output of the transformation can be higher-dimensional or lower-dimensional than the original features
  - In particular, learning any dimensionality reduction (feature selection/feature extraction) can be seen as learning a metric where the left-out features have no effect on distance.

Types of metrics

- Nonlinear metrics can be described in several ways:

  Globally through an implicit transformation: Sometimes the transformation does not need to be known, as long as the metric between the transformed features is known.

  - **Kernel methods** like kernel PCA use **kernel functions** to compute inner products in a transformed space.
  - Valid kernel functions (so-called “Mercer kernels”) always correspond to inner products in some transformed space, even if the transformation is unknown/hard to compute.
  - Distances can be computed using kernels only: assume **f** is the unknown nonlinear function and $k_{\mathbf{f}}(\mathbf{x}_1, \mathbf{x}_2)$ is the known kernel function. Then the distance can be computed

$$\begin{aligned} d_{\mathbf{f}}^2(\mathbf{x}_1, \mathbf{x}_2) &= ||\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2)||^2 = (\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2))^\top (\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2)) \\ &= \mathbf{f}(\mathbf{x}_1)^\top \mathbf{f}(\mathbf{x}_1) - 2\mathbf{f}(\mathbf{x}_1)^\top \mathbf{f}(\mathbf{x}_2) + \mathbf{f}(\mathbf{x}_2)^\top \mathbf{f}(\mathbf{x}_2) \\ &= k_{\mathbf{f}}(\mathbf{x}_1, \mathbf{x}_1) - 2k_{\mathbf{f}}(\mathbf{x}_1, \mathbf{x}_2) + k_{\mathbf{f}}(\mathbf{x}_2, \mathbf{x}_2) \end{aligned}$$

Types of metrics

- Nonlinear metrics can be described in several ways:

  Alternatively, a metric can be described locally:

  - In each very small (infinitesimally small) neighborhood N($\mathbf{x}$) of the feature space around point $\mathbf{x}$, distances inside the neighborhood are described by a Mahalanobis metric with a metric matrix $\mathbf{A}(\mathbf{x})$.
  - Distances between two far-apart points $\mathbf{x}_1$, $\mathbf{x}_2$ are given by integrals of local distances: for any path from $\mathbf{x}_1$ to $\mathbf{x}_2$, the distance along the path is the integral over local distances along the path.

    The shortest path (minimal integral) defines the distance d($\mathbf{x}_1$, $\mathbf{x}_2$).

    Shortest path may be difficult to compute analytically, but can be approximated.

Properties of a metric

- In general, any function $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ between pairs of data points can be used as a distance function, as long as it satisfies the properties of a metric:
  - non-negativity (separation axiom): $d(x, y) \geq 0$
  - coincidence axiom: $d(x, y) = 0 \text{ if and only if } x = y$
  - symmetry: $d(x, y) = d(y, x)$
  - triangle inequality: $d(x, z) \leq d(x, y) + d(y, z)$
- The previous examples (Euclidean, cosine, Mahalanobis, transformation + Euclidean/Mahalanobis, kernel-based, local Mahalanobis) all satisfy non-negativity, symmetry, and triangle inequality.
  - However, they can cause several x,y to have zero distance from each other (e.g. Mahalanobis with some zeros on the diagonal)
  - The satisfy the coincidence axiom if we consider all points at distance zero from each other to be “the same point”

How to Learn a Metric

- Metric learning can be done either in a supervised way or an unsupervised way.
- Unsupervised metric learning would mean optimizing a metric (e.g. optimizing the Mahalanobis matrix, or optimizing a data transformation) to make the data “look interesting” in the new metric: for example, to make the data appear well clustered.
    - Roughly speaking: try several metrics, compute for each metric an “interestingness score” such as a clustering criterion, choose the metric that gives the best score.
- We focus mostly on supervised metric learning, where there is some **annotation** available for some of the data points.
- Intuitively, the annotation tells which pairs of points should have small distance and which ones should have large distance.

How to Learn a Metric

- Different kinds of annotations:
  Class labels: some subset of training data points has a known class label, out of a set of N_C different classes. For example data points might be pictures of people, for some pictures the identity of the person is known.

  Must-link / cannot-link constraints: for some pairs of training data points, it is known that they should be similar or dissimilar.

  $$\mathcal{S} = \{(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ should be similar}\}$$

  $$\mathcal{D} = \{(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ should be dissimilar}\}$$

  For example video footage might contain several pictures of the same person, and the pictures can be considered “similar” even if the person's identity is unknown.
  Or: in social data sets data points might be people, and some people are known to be “similar” (friends/colleagues, etc.)

- Some methods instead use **constraint triplets** as annotation:

$$\mathcal{R} = \{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) : \mathbf{x}_i \text{ should be more similar to } \mathbf{x}_j \text{ than } \mathbf{x}_k \}$$

Pictures from A. Bellet, A. Habrard, and M. Sebban, ArXiv:136.6709, 2013.

How to Learn a Metric

- Class label annotation can be turned to pairwise constraints ("same-class points are similar, different-class points are dissimilar"), but class labels can provide more information. Constraint-based methods can be called "weakly supervised"
- Semi-supervised methods use also the unlabeled data
- Some approaches learn just a distance matrix (or kernel matrix) for a finite data set, but the ones we discuss learn an actual distance function that generalizes to new data points.
    - For example, metrics may be learned from a data subset that has annotations, and then applied to all data

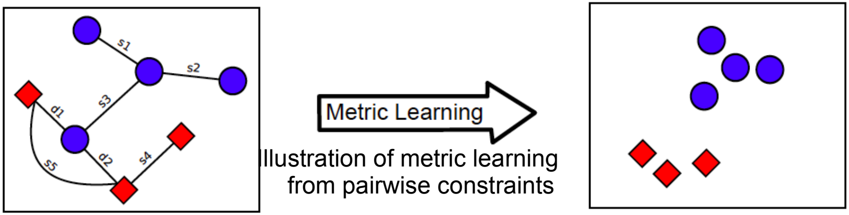
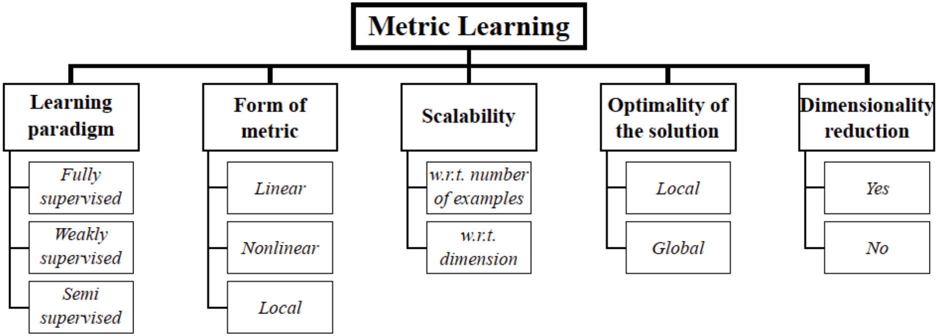


Illustration of metric learning from pairwise constraints

Pictures from A. Bellet, A. Habrard, and M. Sebban, ArXiv:136.6709, 2013.

How to Learn a Metric

- Differences between methods are: what kind of annotations are used, what form of metric is learned, and what cost function is used to evaluate how well the metric fits the annotations
- Other differences include optimization details (optimality, scalability), and whether the metric learning method can inherently perform dimensionality reduction



Using the New Metrics for Dimensionality Reduction or Visualization

- If we have a dimensionality reduction/visualization method that works based on distances (or based on a distance function), we can often simply give the optimized distances as input.

- For example the “Sammon's Mapping in the Learning Metric” method (Sammon-L; Jaakko Peltonen, Arto Klami, and Samuel Kaski, “Improved Learning of Riemannian Metrics for Exploratory Data Analysis”) infers the supervised Learning Metric, computes pairwise distances in it, and gives them as input to a traditional Sammon's mapping algorithm.

- One could similarly give supervised distances like the Learning Metric as input to Multidimensional scaling, or Curvilinear Component Analysis.

Dimensionality Reduction by Learning a Metric?

- We already saw that learning a dimensionality reduction corresponds to learning a metric where left-out features don't affect distances.
- It is possible to do the reverse: by learning a metric, we can tell which features are important to keep, and which can be left out.
- For example: if, in a diagonal Mahalanobis metric matrix **A**, some features have very small weights (close to zero), they don't affect distances much and could be left out.

Dimensionality Reduction by Learning a Metric?

- In general: if the Mahalanobis metric matrix A has an eigendecomposition **A** = **VDV**$^\top$, where **D** is diagonal, then

$$d^2_{\mathbf{A}}(\mathbf{x},\mathbf{x}') = (\mathbf{x}-\mathbf{x}')^\top \mathbf{A}(\mathbf{x}-\mathbf{x}') = (\mathbf{x}-\mathbf{x}')^\top \mathbf{V}\mathbf{D}\mathbf{V}^\top(\mathbf{x}-\mathbf{x}')$$
$$= \left(\mathbf{D}^{1/2}\mathbf{V}^\top(\mathbf{x}-\mathbf{x}')\right)^\top \left(\mathbf{D}^{1/2}\mathbf{V}^\top(\mathbf{x}-\mathbf{x}')\right) = ||\mathbf{D}^{1/2}\mathbf{V}^\top(\mathbf{x}-\mathbf{x}')||^2$$

  where D$^{1/2}$ is D with square roots taken from diagonal entries.

- Thus learning Mahalanobis metric corresponds to learning a linear data transformation!
- If some eigenvalues (diagonals of D) are zero, then the metric effectively performs dimensionality reduction
- Some methods learn a metric with penalties that encourage features to be left out.

Method 1: Informative Discriminant Analysis / Neighborhod Component Analysis

Method first proposed in Samuel Kaski and Jaakko Peltonen, ”Informative discriminant analysis”, in proceedings of ICML 2003. Soon after proposed in Jacob Goldberger, Sam Roweis, Geoffrey Hinton, Ruslan Salakhutdinov, “Neighbourhood components analysis”, proceedings of NIPS 2004.

Idea: assume training data have labels. Learn a Mahalanobis metric matrix **A**. Maximize log-likelihood of predicting labels of a point from its nearby neighbors in the metric. (This method is thus a maximum-likelihood method to estimate the metric.)

Suppose the density of each class can be written as a mixture of multivariate Gaussian distributions with class-dependent weights:

$$p(c, \mathbf{x}) \propto \sum_{m=1}^{M} \psi_{mc} N(\mathbf{x};\ \boldsymbol{\mu}_{\boldsymbol{m}}, \mathbf{A}^{-1})$$

Method 1: Informative Discriminant Analysis / Neighborhod Component Analysis

If $p(c, \mathbf{x}) \propto \sum_{m=1}^{M} \psi_{mc} N(\mathbf{x};\ \boldsymbol{\mu_m}, \mathbf{A}^{-1})$ conditional class-prob. is

$$p(c|\mathbf{x}) = \frac{\sum_{m=1}^{M} \psi_{mc} N(\mathbf{x};\ \boldsymbol{\mu_m}, \mathbf{A}^{-1})}{\sum_{m=1}^{M} (\sum_c \psi_{mc}) N(\mathbf{x};\ \boldsymbol{\mu_m}, \mathbf{A}^{-1})}$$

Method 1: Informative Discriminant Analysis / Neighborhod Component Analysis

If $p(c, \mathbf{x}) \propto \sum_{m=1}^{M} \psi_{mc} N(\mathbf{x};\ \boldsymbol{\mu_m}, \mathbf{A}^{-1})$ conditional class-prob. is

$$p(c|\mathbf{x}) = \frac{\sum_{m=1}^{M} \psi_{mc} N(\mathbf{x};\ \boldsymbol{\mu_m}, \mathbf{A}^{-1})}{\sum_{m=1}^{M} (\sum_c \psi_{mc}) N(\mathbf{x};\ \boldsymbol{\mu_m}, \mathbf{A}^{-1})}$$

$$= \frac{\sum_{m=1}^{M} \psi_{mc} \frac{1}{(2\pi)^{d/2}\sqrt{|\mathbf{A}^{-1}|}} \exp(-(\mathbf{x} - \boldsymbol{\mu_m})^\top \mathbf{A}(\mathbf{x} - \boldsymbol{\mu_m})/2)}{\sum_{m=1}^{M} (\sum_c \psi_{mc}) \frac{1}{(2\pi)^{d/2}\sqrt{|\mathbf{A}^{-1}|}} \exp(-(\mathbf{x} - \boldsymbol{\mu_m})^\top \mathbf{A}(\mathbf{x} - \boldsymbol{\mu_m})/2)}$$

Method 1: Informative Discriminant Analysis / Neighborhod Component Analysis

If $p(c, \mathbf{x}) \propto \sum_{m=1}^{M} \psi_{mc} N(\mathbf{x};\ \boldsymbol{\mu_m}, \mathbf{A}^{-1})$ conditional class-prob. is

$$p(c|\mathbf{x}) = \frac{\sum_{m=1}^{M} \psi_{mc} N(\mathbf{x};\ \boldsymbol{\mu_m}, \mathbf{A}^{-1})}{\sum_{m=1}^{M} (\sum_c \psi_{mc}) N(\mathbf{x};\ \boldsymbol{\mu_m}, \mathbf{A}^{-1})}$$

$$= \frac{\sum_{m=1}^{M} \psi_{mc} \frac{1}{(2\pi)^{d/2}\sqrt{|\mathbf{A}^{-1}|}} \exp(-(\mathbf{x}-\boldsymbol{\mu_m})^\top \mathbf{A}(\mathbf{x}-\boldsymbol{\mu_m})/2)}{\sum_{m=1}^{M} (\sum_c \psi_{mc}) \frac{1}{(2\pi)^{d/2}\sqrt{|\mathbf{A}^{-1}|}} \exp(-(\mathbf{x}-\boldsymbol{\mu_m})^\top \mathbf{A}(\mathbf{x}-\boldsymbol{\mu_m})/2)}$$

$$= \frac{\sum_{m=1}^{M} \psi_{mc} \exp(-(\mathbf{x}-\boldsymbol{\mu_m})^\top \mathbf{A}(\mathbf{x}-\boldsymbol{\mu_m})/2)}{\sum_{m=1}^{M} (\sum_c \psi_{mc}) \exp(-(\mathbf{x}-\boldsymbol{\mu_m})^\top \mathbf{A}(\mathbf{x}-\boldsymbol{\mu_m})/2)}$$

Method 1: Informative Discriminant Analysis / Neighborhod Component Analysis

If $p(c, \mathbf{x}) \propto \sum_{m=1}^{M} \psi_{mc} N(\mathbf{x};\ \boldsymbol{\mu}_{\boldsymbol{m}}, \mathbf{A}^{-1})$ conditional class-prob. is

$$p(c|\mathbf{x}) = \frac{\sum_{m=1}^{M} \psi_{mc} N(\mathbf{x};\ \boldsymbol{\mu}_{\boldsymbol{m}}, \mathbf{A}^{-1})}{\sum_{m=1}^{M} (\sum_c \psi_{mc}) N(\mathbf{x};\ \boldsymbol{\mu}_{\boldsymbol{m}}, \mathbf{A}^{-1})}$$

$$= \frac{\sum_{m=1}^{M} \psi_{mc} \frac{1}{(2\pi)^{d/2}\sqrt{|\mathbf{A}^{-1}|}} \exp(-(\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{m}})^\top \mathbf{A} (\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{m}})/2)}{\sum_{m=1}^{M} (\sum_c \psi_{mc}) \frac{1}{(2\pi)^{d/2}\sqrt{|\mathbf{A}^{-1}|}} \exp(-(\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{m}})^\top \mathbf{A} (\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{m}})/2)}$$

$$= \frac{\sum_{m=1}^{M} \psi_{mc} \exp(-(\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{m}})^\top \mathbf{A} (\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{m}})/2)}{\sum_{m=1}^{M} (\sum_c \psi_{mc}) \exp(-(\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{m}})^\top \mathbf{A} (\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{m}})/2)}$$

And the log-likelihood of observed class-labels c_i of points $\mathbf{x}_i$ is

$L = \sum_{i=1}^{N} \log p(c_i|\mathbf{x}_i)$ which can be maximized with respect to **A**

Method 1: Informative Discriminant Analysis / Neighborhod Component Analysis

The locations and class-weights of the Gaussians can come from a previous density estimation. Alternatively, a simple way is to set one Gaussian at every training data point, and set its class weights according to the class of the data point:

$$M = N, \text{ for } m = 1, \ldots, N, \ \boldsymbol{\mu}_m = \mathbf{x}_m \text{ and } \psi_{mc} = \{1 \text{ if } c = c_m \text{ and zero otherwise}\}$$

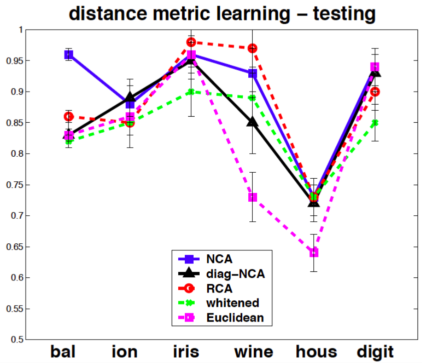
Then, to compute $p(c_i|\mathbf{x}_i)$ leave out the i:th term from the sums (leave-one-out procedure)

The log-likelihood can be maximized with respect to **A** by gradient methods.

Method 1: Informative Discriminant Analysis / Neighborhod Component Analysis

Performance of IDA/NCA can be measured for example by K-nearest neighbor classification accuracy on a test set not used for learning the metric.

IDA/NCA can give better results than LDA, because it does not make simple single-Gaussian assumptions about classes, and because it directly maximizes conditional class log-likelihood.

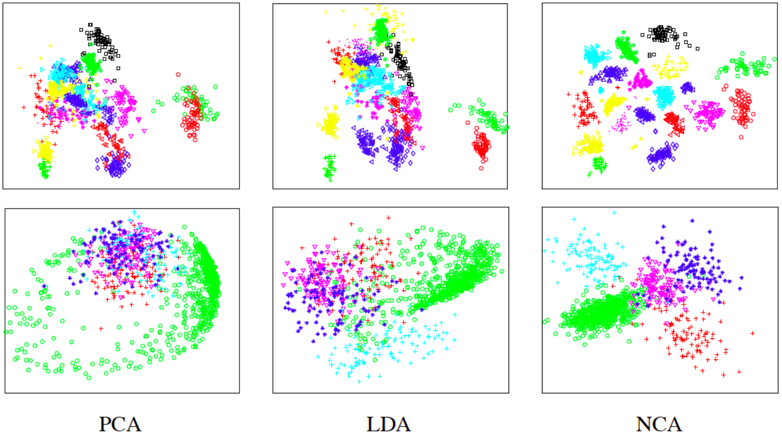


IDA/NCA compared to other metrics on six data sets. It clearly outperforms the Euclidean metric and is among the best metrics.

Pictures from J. Goldberger, S. Roweis, G. Hinton, R. Salakhutdinov, proc. NIPS 2004.

Method 1: Informative Discriminant Analysis / Neighborhod Component Analysis

IDA/NCA can also be used for dimensionality reduction by restricting the rank of the metric matrix **A**, or by directly optimizing **A** as a product of a linear projection matrix **W**, $\mathbf{A}=\mathbf{W}\mathbf{W}^T$. The probabilities and cost function are computed the same as before. The matrix W can be used to project data to lower dimensionality.



Pictures from J. Goldberger, S. Roweis, G. Hinton, R. Salakhutdinov, proc. NIPS 2004.

Method 2: Learning from Pairwise Comparisons

- A metric tells which points are similar (they have small distance in the metric) and which points are dissimilar (they have large distance). The desired metric is unknown - we want to learn it.
- If we have **examples of similar point pairs** and **examples of dissimilar point pairs,** can we learn a metric from them?
- Yes! Use probabilistic modeling: given a metric, define a **probability that two points in the metric will be labeled similar** vs. dissimilar. Then optimize the metric to maximize the likelihood of the observed pairs!
- For example, use a Mahalanobis metric (matrix **A** = **WW**^{T}) and a logistic probability:

$$p_{similar}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1}{1+\exp\left((\boldsymbol{x}_i - \boldsymbol{x}_j)^T A (\boldsymbol{x}_i - \boldsymbol{x}_j) - threshold\right)}$$

"points closer than *threshold* are probably called similar"

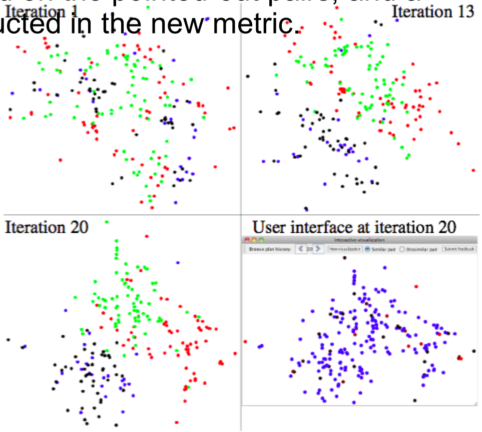
- Then maximize the log-likelihood of observed similarities with respect to elements of **W**, e.g. by gradient descent:

$$max_W \left[\sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in S_{similar}} \log p_{similar}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in S_{dissimilar}} \log\left(1 - p_{similar}(\boldsymbol{x}_i, \boldsymbol{x}_j)\right) \right]$$

Some pictures from Hoi, Liu, Lyu, and Ma, in proceedings of CVPR 2006.

Method 2: Learning from Pairwise Comparisons

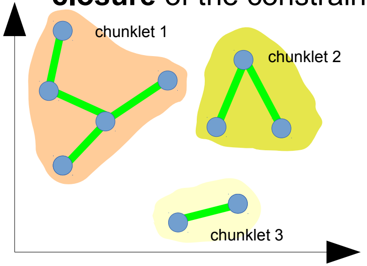
- This idea was used for **interactive visualization** (Peltonen et al.)
- Experts inspected a scatterplot of scientific documents and pointed out pairs of documents that were similar or dissimilar.
- A metric was learned for document features (=unigram content)
- The metric was learned based on the pointed-out pairs, and a new visualization was constructed in the new metric.
- The experts inspected again, and pointed out more pairs.
- The metric and visualization converged to focus on important features of data.
- The metric was learned by a so-called variational Bayes algorithm (outside the scope of the course) rather than maximizing likelihood, but the idea is essentially the same.



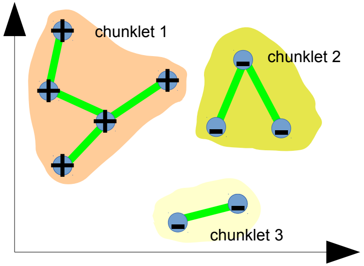
Pictures from Peltonen et al., Information Retrieval Perspective to Interactive Data Visualization, in Eurovis 2013.

Method 3: Relevant Component Analysis

- Proposed in Aharon Bar-Hillel, Tomer Hertz, Noam Shental, and Daphna Weinshall, “Learning Distance Functions using Equivalence Relations”, in proceedings of ICML 2003, and in Shental, Hertz, Weinshall, Pavel, “Adjustment Learning and Relevant Component Analysis”, in ECCV 2002.
- Idea: data often comes in “chunklets” known to arise from the same unknown class. For example several images from a video sequence of a person: all images have the same person.
- Given a set of pairwise “must-link” constraints (“x1 and x2 have the same unknown class”), identify the chunklets by **transitive closure** of the constraints:



Must-link constraints shown as green lines



Example of underlying classes of the chunklets

Method 3: Relevant Component Analysis

- Given N chunklets (data subsets) H_n, each of size H_n , compute the average within-chunklet covariance:

$$S_{ch} = \frac{1}{|\Omega|} \sum_{n=1}^{N} |H_n| Cov(H_n) = \underbrace{\frac{1}{|\Omega|}}_{\text{number of all data}} \sum_{n=1}^{N} \sum_{j=1}^{|H_n|} \left(\boldsymbol{x}_n^j - \hat{\mu}_n\right) \left(\boldsymbol{x}_n^j - \underbrace{\hat{\mu}_n}_{\text{mean of chunklet}}\right)^T$$

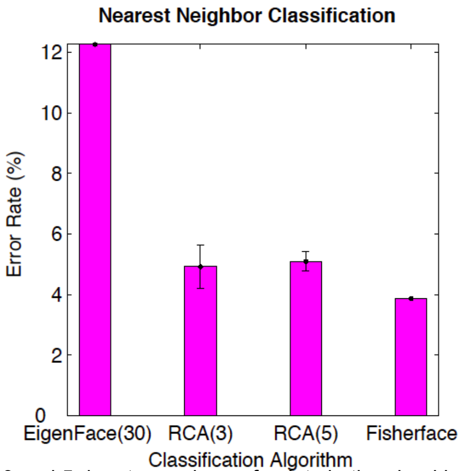
- Use the inverse of the within-chunklet covariance as a Mahalanobis distance matrix! (Alternatively: compute a corresponding data transformation, we saw earlier that Mahalanobis metrics correspond to linear data transformations)

- This means that distances grow slowly in directions where within-chunklet variance is large, and fast in directions where within-chunklet variance is small. If chunklets are shaped like the classes they come from, this means distances grow fast in directions where within-class variance is small.

Pictures from Shental, Hertz, Weinshall, Pavel, “Adjustment Learning and Relevant Component Analysis”, in ECCV 2002.

Method 3: Relevant Component Analysis

- Good property: very simple method. Bad property: only uses “must-link” constraints, does not use “cannot-link” constraints at all.



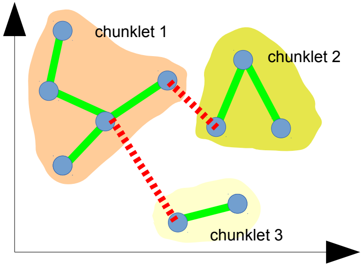
Performance of various metrics in nearest neighbor classification of face images. EigenFace essentially means PCA. FisherFace essentially means Linear Discriminant Analysis. FisherFace performs best but it is fully supervised (knows class labels), RCA only uses must-link constraints.

3 and 5 denote numbers of points in the chunklets

Pictures from Shental, Hertz, Weinshall, Pavel, “Adjustment Learning and Relevant Component Analysis”, in ECCV 2002.

Method 4: Discriminative Component Analysis

- Steven C. Hoi, Wei Liu, Michael R. Lyu, and Wei-Ying Ma, “Learning Distance Metrics with Contextual Constraints for Image Retrieval”, in proceedings of CVPR 2006.
- Again, similar names have been used in other methods...
- Idea: similar to Relevant Component Analysis, but use also “cannot-link” information.
- For each chunklet i, identify “discriminative set” D_i: other chunklets that have at least one cannot-link constraint to chunklet i.



Must-link constraints shown as green lines, cannot-link as red dashed lines.

Discriminative set for chunklet 1: chunklets 2, 3.
Discriminative set for chunklet 2: chunklet 1.
Discriminative set for chunklet 3: chunklet 1.

Some pictures from Hoi, Liu, Lyu, and Ma, in proceedings of CVPR 2006.

Method 4: Discriminative Component Analysis

- Compute matrix similar to the previous within-chunklet covariance, and also a matrix of between-discriminative-chunklets covariance:

mean of chunklet j

$$\hat{C}_b = \frac{1}{n_b} \sum_{j=1}^{n} \sum_{i \in D_j} (\mathbf{m}_j - \mathbf{m}_i)(\mathbf{m}_j - \mathbf{m}_i)^\top$$

discriminative set of chunklet j

$$\hat{C}_w = \frac{1}{n} \sum_{j=1}^{n} \frac{1}{n_j} \sum_{i=1}^{n_j} (\mathbf{x}_{ji} - \mathbf{m}_j)(\mathbf{x}_{ji} - \mathbf{m}_j)^\top$$

number of points in chunklet j

$$n_b = \sum_{j=1}^{n} |D_j|$$

total number of chunklets in all discriminative sets

- Then optimize a transformation matrix **A** with a criterion similar to linear discriminant analysis (corresponding Mahalanobis matrix is **M=A$^\top$A**:

$$J(A) = \arg\max_A \frac{|A^\top \hat{C}_b A|}{|A^\top \hat{C}_w A|}$$

(The solution can be found by an eigenvalue approach)

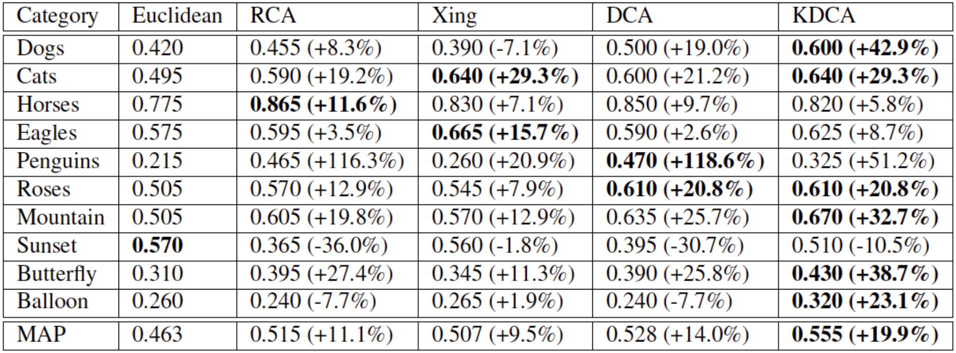
Some pictures from Hoi, Liu, Lyu, and Ma, in proceedings of CVPR 2006.

Method 4: Discriminative Component Analysis

- Performance in an information retrieval study using various metrics:

another comparison method

kernel version of discriminative component analysis, not discussed here



Category	Euclidean	RCA	Xing	DCA	KDCA
Dogs	0.420	0.455 (+8.3%)	0.390 (-7.1%)	0.500 (+19.0%)	**0.600 (+42.9%)**
Cats	0.495	0.590 (+19.2%)	**0.640 (+29.3%)**	0.600 (+21.2%)	**0.640 (+29.3%)**
Horses	0.775	**0.865 (+11.6%)**	0.830 (+7.1%)	0.850 (+9.7%)	0.820 (+5.8%)
Eagles	0.575	0.595 (+3.5%)	**0.665 (+15.7%)**	0.590 (+2.6%)	0.625 (+8.7%)
Penguins	0.215	0.465 (+116.3%)	0.260 (+20.9%)	**0.470 (+118.6%)**	0.325 (+51.2%)
Roses	0.505	0.570 (+12.9%)	0.545 (+7.9%)	**0.610 (+20.8%)**	**0.610 (+20.8%)**
Mountain	0.505	0.605 (+19.8%)	0.570 (+12.9%)	0.635 (+25.7%)	**0.670 (+32.7%)**
Sunset	**0.570**	0.365 (-36.0%)	0.560 (-1.8%)	0.395 (-30.7%)	0.510 (-10.5%)
Butterfly	0.310	0.395 (+27.4%)	0.345 (+11.3%)	0.390 (+25.8%)	**0.430 (+38.7%)**
Balloon	0.260	0.240 (-7.7%)	0.265 (+1.9%)	0.240 (-7.7%)	**0.320 (+23.1%)**
MAP	0.463	0.515 (+11.1%)	0.507 (+9.5%)	0.528 (+14.0%)	**0.555 (+19.9%)**

evaluation criterion: average precision on top-20 returned images

Method (family) 5: Multiple Kernel Learning

- Idea: suppose we have a set of known fixed kernel functions:
  $$\{k_1(\cdot,\cdot), k_2(\cdot,\cdot), \ldots, k_M(\cdot,\cdot)\}$$
  Then any weighted linear combination of them is also a valid kernel function.
- Learn the “best” linear combination of the kernel functions, based on annotations:

$$k_{\mathbf{w}}(\mathbf{x},\mathbf{x}') = w_1 k_1(\mathbf{x},\mathbf{x}') + w_2 k_2(\mathbf{x},\mathbf{x}') + \cdots + w_M k_M(\mathbf{x},\mathbf{x}')$$

- Optimize the weights **w** based on cost functions like the ones we have seen for other methods (note that distances involved in the cost functions can be computed using the kernel)

Method 6: The Learning Metric

- Learns a local metric from class labels of data.
- Suppose we know conditional probabilities of classes at different points of the feature space.
- Idea: Locally, distances should increase the most in directions where the class probabilities (class distribution) changes the most. If we have good local distances, we can derive a full metric from them.
- Difference between two class probability distributions can be measured by Kullback-Leibler divergence

$$D_{KL}(p||q) = \sum_c p(c) \log \frac{p(c)}{q(c)}$$

- It turns out Kullback-Leibler divergence between conditional class distributions at nearby points (**x**, **x**+d**x**) can be expressed as a squared Mahalanobis distance!

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

Method 6: The Learning Metric

- The Mahalanobis matrix **J**(**x**) is the **Fisher information matrix** computed from the change of the local class distributions:

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

- Under suitable assumptions **J**(**x**) can be computed analytically
- Assume the conditional probabilities have a form, corresponding to a Gaussian mixture density in each class:

weight of class c in kth Gaussian

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

mean of kth Gaussian

prior weight of kth Gaussian

Method 6: The Learning Metric

- Then the Fisher information matrix can be computed as:

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

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

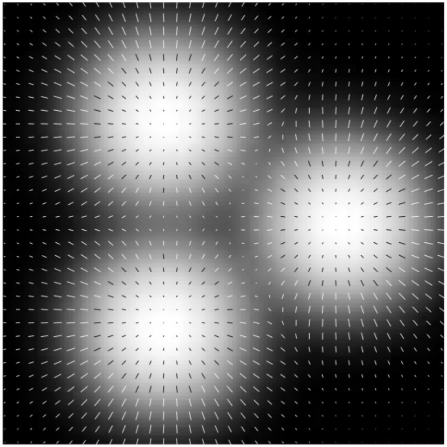
- As mentioned before, local Mahalanobis distances can be extended to global distances between two points as minimal integrals of the local distances (minimal integral over possible paths between the points)

Method 6: The Learning Metric

- As mentioned before, local Mahalanobis distances can be extended to global distances between two points as minimal integrals of the local distances (minimal integral over possible paths between the points)
- Simple approach: just compute the local Mahalanobis from **x** to **x**+d**x**, regardless how large the difference d**x** is.
- More advanced approach: compute an approximate integral over the line connecting **x** to **x**+d**x** (e.g.divide line into 10 segments, compute local Mahalanobis over each segment).
- Even more advanced approach (similar to Isomap): compute initial distance matrix as above, then compute minimal path using other data points as possible waypoints. Can be done by Dijkstra's algorithm / Floyd's algorithm.

Method 6: The Learning Metric

- Example of local Mahalanobis metrics



2 classes,
grayscale background:
probability of class 1.

lines: direction where
local Mahalanobis
distance increases

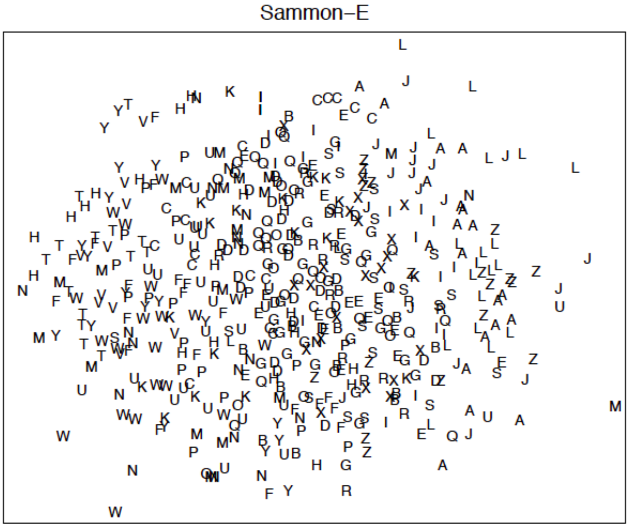
distance increases only
in directions where
class probability changes!

- The learning metric can be computed between any points (either the known training data points or any other points). Thus it can be applied to any method that works based on distances.

Sammon's Mapping in the Learning Metric

Traditional Sammon's mapping on a data set of images of different letters A-Z, using various geometrical descriptors about the letter shapes as the features, with the Euclidean metric (“Sammon-E”).

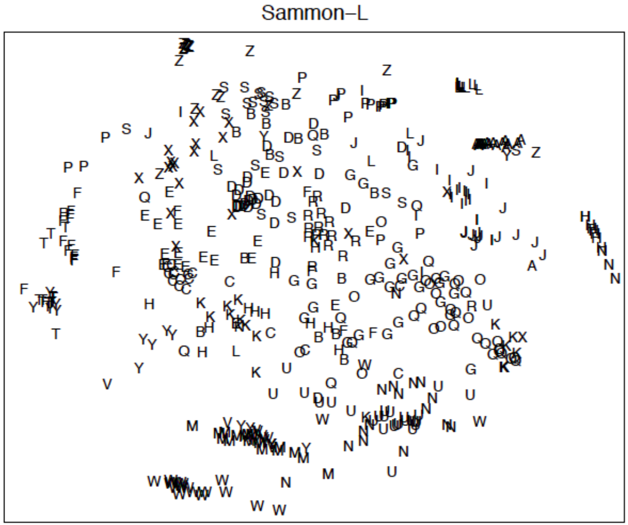
From: Jaakko Peltonen, Arto Klami, and Samuel Kaski. Improved Learning of Riemannian Metrics for Exploratory Data Analysis. Neural Networks, vol. 17, pages 1087-1100, 2004.



Sammon's Mapping in the Learning Metric

Traditional Sammon's mapping on a data set of images of different letters A-Z, using various geometrical descriptors about the letter shapes as the features, with the local supervised Learning Metric (“Sammon-L”).

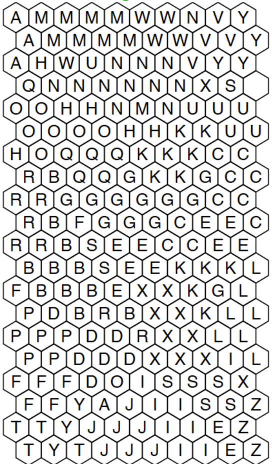
From: Jaakko Peltonen, Arto Klami, and Samuel Kaski. Improved Learning of Riemannian Metrics for Exploratory Data Analysis. Neural Networks, vol. 17, pages 1087-1100, 2004.



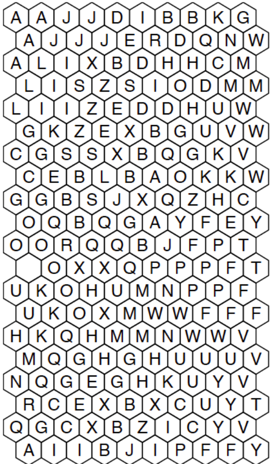
Self-Organizing Map in the Learning Metric

Idea: at each iteration of Self-Organizing Map training, find the nearest prototype for a data point using distances in the learning metric, instead of the simple Euclidean metric. The rest of the Self-Organizing Map training (the way propotypes are adapted towards data) is the same as before.

Learning metric



Euclidean metric



Self-Organizing Map trained for letter images (features = geometric descriptions of the letters), with classes A-Z shown on the map.

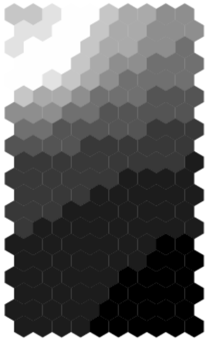
The Learning Metric leads to better class organization

From: Samuel Kaski, Janne Sinkkonen, and Jaakko Peltonen. Bankruptcy analysis with self-organizing maps in learning metrics. IEEE Transactions on Neural Networks, 12:936-947, 2001.

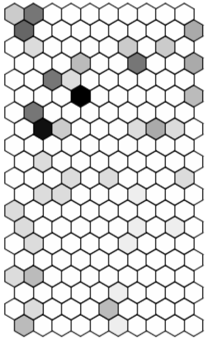
Self-Organizing Map in the Learning Metric

Idea: at each iteration of Self-Organizing Map training, find the nearest prototype for a data point using distances in the learning metric, instead of the simple Euclidean metric. The rest of the Self-Organizing Map training (the way propotypes are adapted towards data) is the same as before.

Probability of bankruptcy



Actual per-node frequency of bankrupt companies



Self-Organizing Map trained for company data (features = financial indicators) in the learning metric. Classes = whether the company went bankrupt.

Metric Learning References

Aurélien Bellet, Amaury Habrard, and Marc Sebban. **A Survey on Metric Learning for Feature Vectors and Structured Data.** ArXiv:136.6709, 2013.

Liu Yang and Rong Jin. **DistLearnKit: A Matlab Toolkit for Distance Metric Learning.**
www.cs.cmu.edu/~liuy/distlearn.htm

Tutorials on metric learning have been given at conferences ICML 2010, ECCV 2010, and workshops on metric learning have been held at conferences ICCV 2011, NIPS 2011, and ICML 2013. Some of their material may be available online.