Distance Metric Learning

Technical University of Munich
Department of Informatics
Computer Vision Group

November 11, 2016

Outline

Outline

Motivation

- How do we measure similarity?
- What is a metric?
- Why to learn a metric?
- How to learn a metric?

How do we measure similarity?

Most algorithms that intend to extract knowledge from data, have to, at some stage, compute *distances* between data points. Thus, their performance, often critically, depends on their *definition of similarity* between objects.

What is a metric?

A **metric** or **distance function** is a function that defines a distance between each pair of elements of a set.

Formally, it is a mapping $\mathcal{D} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^+$ over a vector space $\mathcal{X}$, where the following conditions are satisfied $\forall x_i, x_j, x_k \in \mathcal{X}$:

1.	$\mathcal{D}(x_i, x_j) \geq 0$	**Non-negativity**
2.	$\mathcal{D}(x_i, x_j) = \mathcal{D}(x_j, x_i)$	**Symmetry**
3.	$\mathcal{D}(x_i, x_j) \leq \mathcal{D}(x_i, x_k) + \mathcal{D}(x_k, x_j)$	**Triangle inequality**
4.	$\mathcal{D}(x_i, x_j) = 0 \Leftrightarrow x_i = x_j$	**Identity of indiscernibles**

If condition 4 is not met, we are referring to a **pseudo-metric**. Usually we do not distinguish between metrics and pseudo-metrics.

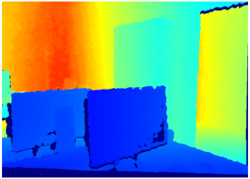
Why learn a metric?

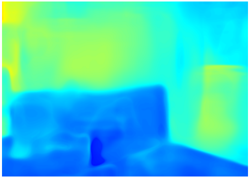
"The greatest thing by far is to be a master of metaphor; it is the one thing that cannot be learned from others; and it is also a sign of genius, since a good metaphor implies an intuitive perception of the similarity of the dissimilar."

Aristotle

Why learn a metric?

- Sometimes, the problem implicitly defines a suitable similarity measure, e.g. Euclidean distance for depth estimation:





- In many interesting problems however, the similarity measure is not easy to find. It is preferable then to learn the similarity from data, together with other parameters of the model.

A family of metrics

A family of metrics over $\mathcal{X}$ is defined by computing Euclidean distances after applying a linear transformation $\mathbf{L}$ such that $x \to \mathbf{L}x$. These metrics compute squared distances as

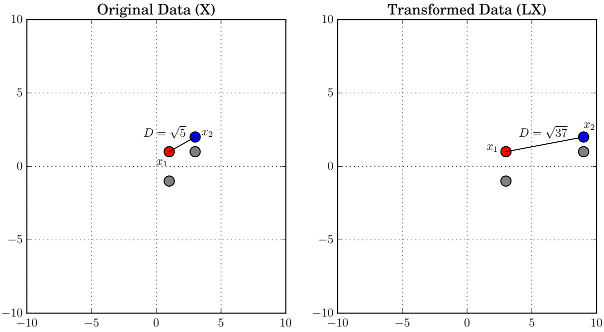
$$\mathcal{D}_L(x_i, x_j) = ||\mathbf{L}x_i - \mathbf{L}x_j||_2^2 \tag{1}$$

Equation (1) defines a valid metric if $\mathbf{L}$ is full rank and a valid pseudo-metric otherwise.

Intuitively, we want to *stretch* the dimensions that contain more information and *contract* the ones that explain less of the data.

A family of metrics - An example

Consider two data points $x_1 = (1, 1)$ and $x_2 = (3, 2)$ that are known to be dissimilar. The transformation $L = \begin{pmatrix} 3 & 0 \\ 0 & 1 \end{pmatrix}$ maps the points to $x_1' = (3, 1)$ and $x_2' = (9, 2)$ as it *weights* distances along the first axis 3 times more than the second. The squared distance of the points changed from $(3-1)^2 + (2-1)^2 = 5$ to $(9-3)^2 + (2-1)^2 = 37$.



Another view: Mahalanobis metrics

Expanding the squared distances equation:

$$\mathcal{D}_L(x_i, x_j) = ||Lx_i - Lx_j||_2^2 = (x_i - x_j)^T L^T L(x_i - x_j) \qquad (2)$$

This allows us to express squared distances in terms of the square matrix $\mathbf{M} = \mathbf{L}^T\mathbf{L}$ which is guaranteed to be *positive semidefinite*. In terms of $\mathbf{M}$ we denote squared distances as

$$\mathcal{D}_M(x_i, x_j) = (x_i - x_j)^T \mathbf{M}(x_i - x_j) \qquad (3)$$

We refer to pseudo-metrics of this form as **Mahalanobis** metrics.

It is easy to see that by setting $\mathbf{M}$ equal to the identity matrix, we fall back to common Euclidean distances.

To learn **L** or **M**

Thus, we have two options on what to learn, which gives rise to two approaches in DML:

- Learn a linear transformation **L** of the data
  - $\mathbf{M} = \mathbf{L}^T\mathbf{L}$ is then uniquely defined
  - Optimization is unconstrained
- Learn a Mahalanobis metric **M**
  - **M** defines **L** up to rotation (does not influence distances)
  - Constraint: **M** must be positive semidefinite
  - But has certain advantages

Outline

Principal Component Analysis [Pearson, 1901]

The main goal of PCA is to find the linear transformation $\mathbf{L}$ that projects the data to a subspace that **maximizes the variance**.

The variance is expressed with the covariance matrix

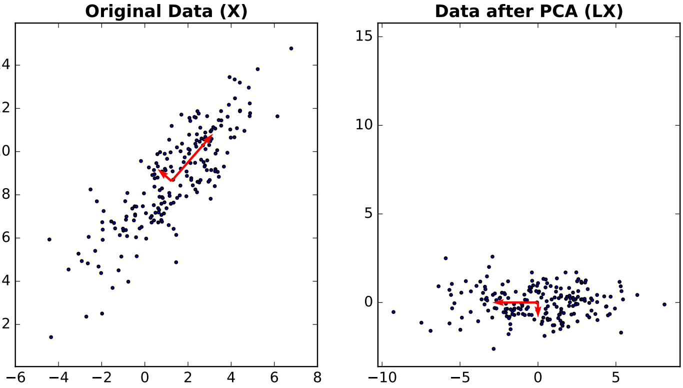
$$\mathbf{C} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T \tag{4}$$

where $\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$ is the sample mean.

It turns out that $\mathbf{C} = \frac{1}{n} X X^T$ (assuming zero-mean $X \in \mathbb{R}^{d \times n}$). The covariance of the projected inputs is then

$$\mathbf{C}' = \frac{1}{n} (\mathbf{L}X)(\mathbf{L}X)^T = \frac{1}{n} \mathbf{L} X X^T \mathbf{L}^T = \frac{1}{n} \mathbf{L} \mathbf{C} \mathbf{L}^T \tag{5}$$

Principal Component Analysis - Illustration



In red: The first two eigenvectors of the covariance matrix, scaled by the square roots of the two largest eigenvalues respectively.

Principal Component Analysis (cont'd)

We can formulate PCA as an optimization problem:

$$\max_{\mathbf{L}} \mathrm{Tr}(\mathbf{LCL}^T) \quad \text{subject to} \quad \mathbf{LL}^T = \mathbf{I} \tag{6}$$

Closed-form solution: Rows of $\mathbf{L}$ are the eigenvectors of $\mathbf{C}$.
Eigen-decomposing $\mathbf{C}$ is equivalent to computing the SVD of $\mathbf{X}$.

Remarks around PCA

- Is an unsupervised method (does **not** use data labels)
- Is widely used for dimensionality reduction: $\mathbf{L} \in \mathbb{R}^{p \times d}, p < d$
- Can be used for:
  - *De-noising*: By removing the bottom eigenvectors
  - Speeding up search of nearest neighbors.

Outline

Linear Discriminant Analysis [Fisher, 1936]

Unlike PCA, LDA is supervised: it uses labels of the inputs.
Goal: Find the $\mathbf{L}$ that **maximizes the between-class variance w.r.t. the within-class variance**.

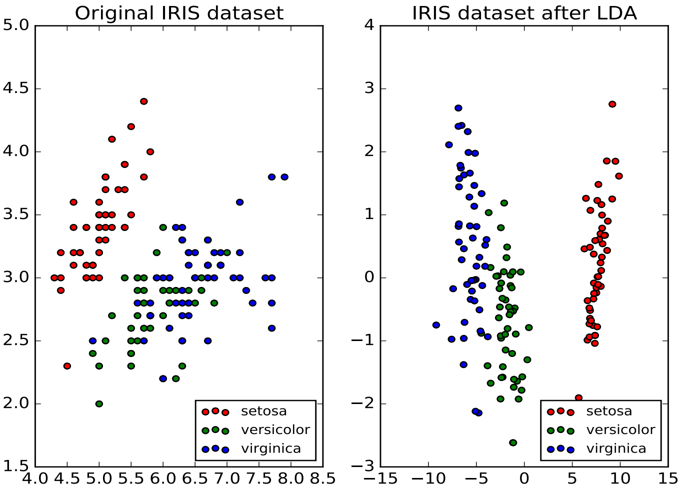
Assuming we have m classes, the covariance matrices are

$$\mathbf{C}_b = \frac{1}{m}\sum_{c=1}^{m} \mu_c \mu_c^T \tag{7}$$

$$\mathbf{C}_w = \frac{1}{n}\sum_{c=1}^{m}\sum_{i\in\Omega_c}(x_i - \mu_c)(x_i - \mu_c)^T, \tag{8}$$

where Ω_c is the set of indices of inputs that belong to class c,
μ_c is the sample mean of class c.
We assume that the data are globally centered.

Linear Discriminant Analysis - Illustration



Linear Discriminant Analysis (cont'd)

Corresponding optimization problem:

$$\max_{\mathbf{L}} \mathrm{Tr}\left(\frac{\mathbf{L}\mathbf{C}_b\mathbf{L}^T}{\mathbf{L}\mathbf{C}_w\mathbf{L}^T}\right) \quad \text{subject to} \quad \mathbf{L}\mathbf{L}^T = \mathbf{I} \tag{9}$$

Closed form solution: Rows of $\mathbf{L}$ are the eigenvectors of $\mathbf{C}_w^{-1}\ \mathbf{C}_b$.

Remarks around LDA

- Is a supervised method (makes use of label information)
- Is widely used as a preprocessing step for pattern classification
- Works well when class distributions are Gaussians

Neighborhood Component Analysis [Goldberger et al., 2004]

Idea: Learn a Mahalanobis metric explicitly to improve ***k-nn*** classification.
Goal: Estimate the $\mathbf{L}$ that minimizes the expected LOO error.

Observations

- LOO error is highly discontinuous w.r.t. the distance metric. ☹
- In particular, an infinitesimal change in the metric can alter the neighbour graph and thus change the validation performance.
- We need a smoother (or at least continuous) function

Idea 2: Instead of picking a fixed number of k nearest neighbors, select a single neighbor **stochastically** and count the expected votes.

Neighborhood Component Analysis (cont'd)

The reference samples x_j for each point x_i are drawn from a softmax pdf:

$$p_{ij} = \begin{cases} \frac{\exp(-||(\mathbf{L}x_i - \mathbf{L}x_j||^2))}{\sum_{k \neq i} \exp(-||(\mathbf{L}x_i - \mathbf{L}x_k||^2))} & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \tag{10}$$

The fraction of the time that x_i will be correctly labeled is:

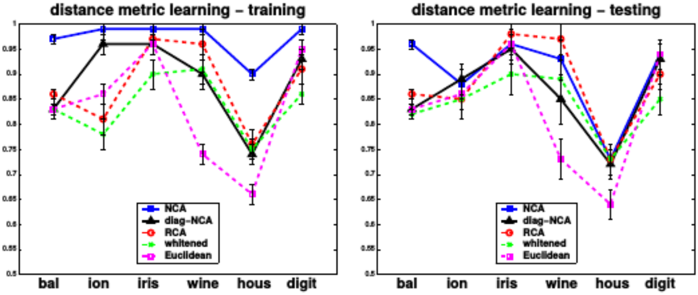
$$p_i^+ = \sum_{j \in C_i} p_{ij} \tag{11}$$

The expected error then is

$$\varepsilon_{NCA} = 1 - \frac{1}{n} \sum_{ij} p_{ij} y_{ij} \quad \text{where} \quad y_{ij} = \begin{cases} 1 & \text{if } y_i = y_j \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

Neighborhood Component Analysis (cont'd)

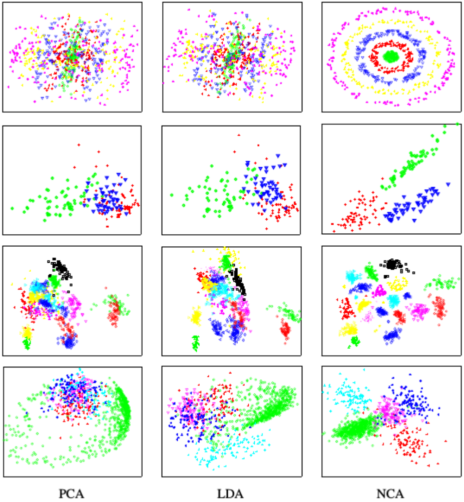
- We don't have to choose a parameter k ☺
- The stochastic nature makes ε_{NCA} differentiable w.r.t. $\mathbf{L}$ ☺
- But ε_{NCA} is not convex $\rightarrow$ no globally optimal $\mathbf{L}$ ☹



k-nn classification accuracy

Dimensionality Reduction - PCA vs LDA vs NCA

Dataset	**Dimensions**
concentric rings	3
wine	13
faces	560
digits	256



Large Margin Nearest Neighbor [Weinberger et al., 2005]

- Idea: Enforce the **maximum margin** possible between intra-class and inter-class samples (as in SVMs)
- *Target neighbors* of $\vec{x}_i$: samples desired to be closest to $\vec{x}_i$
- *Impostors*: samples that violate the margin
- Loss function
  - Pulling target neighbors together

$$\varepsilon_{\text{pull}}(\mathbf{L}) = \sum_{i,j \rightsquigarrow i} ||\mathbf{L}(\vec{x}_i - \vec{x}_j)||^2.$$

  - Pushing impostors away

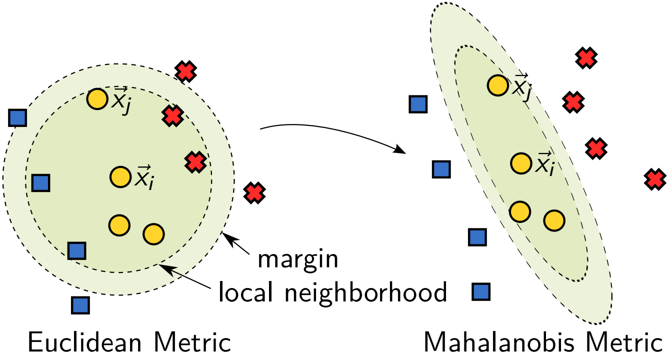
$$\varepsilon_{\text{push}}(\mathbf{L}) = \sum_{i,j \rightsquigarrow i} \sum_{l} (1 - y_{il})[1 + ||\mathbf{L}(\vec{x}_i - \vec{x}_j)||^2 - ||\mathbf{L}(\vec{x}_i - \vec{x}_l)||^2]_+$$

  - Convex combination

$$\varepsilon(\mathbf{L}) = \mu \varepsilon_{\text{pull}}(\mathbf{L}) + (1 - \mu)\varepsilon_{\text{push}}(\mathbf{L}), \quad \mu \in [0, 1]$$

Large Margin Nearest Neighbor (cont'd)

- Supervised Distance Metric Learning for classification
- Considers triplets of points at a time.



- Goal: Find a metric to maximize k-NN accuracy
- Advantage: Convex formulation ☺

Metric Learning Variants

Most metric learning algorithms improve by looking at pairs, triplets or even quadruplets of points.
Many noteworthy algorithms exist:

- Relevant Component Analysis (RCA)
- Information Theoretic Metric Learning (ITML)
- Pseudo-metric Online Batch Learning Algorithm (POLA)
- LogDet Exact Gradient Online (LEGO)
- BoostMetric (combines boosting and metric learning)
- Large Scale Online Learning of Image Similarity Through Ranking (OASIS)
- . . .

This is definitely not an exhaustive list.

Outline

Metric Learning and Kernel Methods

Kernel methods

- Express similarity with the Gram matrix K which is $n \times n$.
- The feature space Φ is usually high-dimensional (theoretically can be infinite-dimensional).
- The training takes place in the kernel space. The algorithm no longer sees the raw inputs $\mathcal{X}$.

Metric Learning

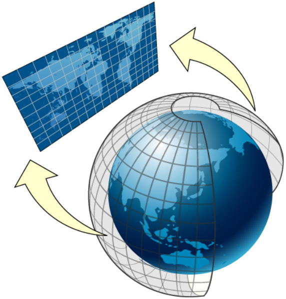
- Learns a transformation L, which is $p \times d$ or a Mahalanobis matrix M which is $d \times d$, like the covariance matrix C .
- Usually $p < d \rightarrow$ learning also results in dimensionality reduction.
- Therefore usually more efficient than kernel methods.

Metric learning can be combined with kernel methods for better results.

Multidimensional Scaling [Torgerson, 1952]

Inverse Problem: Given dissimilarities, find an embedding.
Goal of MDS is to find coordinates of the data points in some subspace of $\mathbb{R}^n$ such that the given distances are preserved.

A famous problem in cartography:
Find a 2-dimensional map of the earth, so that distances between cities are distorted as little as possible.
Notice that the original distances are not Euclidean, but measured along the earth's surface.



Multi-dimensional Scaling (cont'd)

We are given an $n \times n$ matrix D of distances d_{ij} between all pairs of points. Metric MDS minimizes the distortion of distances in terms of a residual sum of squares, called the "stress":

$$stress(x_1, x_2, \ldots, x_n) = \sqrt{\frac{\sum_{i,j}(d_{ij} - ||x_i - x_j||)^2}{\sum_{i,j} d_{ij}^2}} \tag{13}$$

so

$$\{x_1, x_2, \ldots, x_n\}^* = \underset{\{x_i\}}{\arg\min}\, stress(x_1, x_2, \ldots, x_n) \tag{14}$$

- No unique solution. For example, all rotations of a solution would produce the same distances.
- MDS is often used for data visualization.

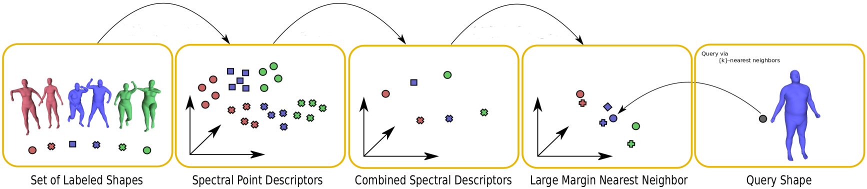
Outline

Non-rigid 3D Shape Retrieval via LMNN [Chiotellis et al., 2016]

Dataset: SHREC'14 [Pickup et al., 2014]

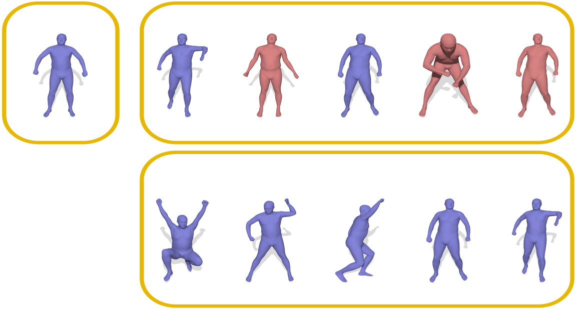
Synthetic dataset: 300 models
(**15** persons × 20 poses)

Real dataset: 400 models
(**40** persons × 10 poses)



Non-rigid 3D Shape Retrieval via LMNN (cont'd)

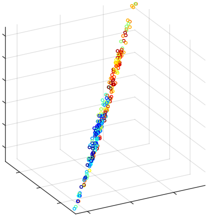
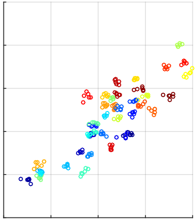
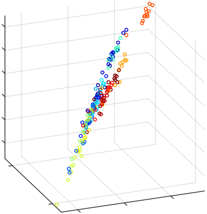
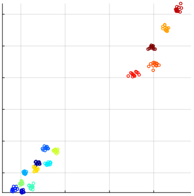
Retrieval Example



Top left: A query model. Top row: 5 best matches retrieved by the Supervised Dictionary Learning method [Litman et al., 2014]. Bottom row: 5 best matches retrieved by the proposed method (CSD+LMNN). Blue indicates that a match corresponds to the correct class. Red indicates an incorrect class.

Non-rigid 3D Shape Retrieval via LMNN (cont'd)

Embeddings Visualization

Dataset	$y_f(\mathcal{S})$ before learning	learned $L \cdot y_f(\mathcal{S})$
SHREC'14 Real		
SHREC'14 Synthetic		

Bibliography I

[Chiotellis et al., 2016] Chiotellis, I., Triebel, R., Windheuser, T., and Cremers, D. (2016).
Non-rigid 3d shape retrieval via large margin nearest neighbor embedding.
In *European Conference on Computer Vision*, pages 327–342. Springer.

[Fisher, 1936] Fisher, R. A. (1936).
The use of multiple measurements in taxonomic problems.
Annals of eugenics, 7(2):179–188.

[Goldberger et al., 2004] Goldberger, J., Hinton, G. E., Roweis, S. T., and Salakhutdinov, R. (2004).
Neighbourhood components analysis.
In *Advances in neural information processing systems*, pages 513–520.

[Litman et al., 2014] Litman, R., Bronstein, A., Bronstein, M., and Castellani, U. (2014).
Supervised learning of bag-of-features shape descriptors using sparse coding.
In *Computer Graphics Forum*, volume 33, pages 127–136. Wiley Online Library.

[Pearson, 1901] Pearson, K. (1901).
On lines and planes of closest fit to system of points in space. philiosophical magazine, 2, 559-572.

[Pickup et al., 2014] Pickup, D., Sun, X., Rosin, P. L., Martin, R. R., Cheng, Z., Lian, Z., Aono, M., Ben Hamza, A., Bronstein, A., Bronstein, M., Bu, S., Castellani, U., Cheng, S., Garro, V., Giachetti, A., Godil, A., Han, J., Johan, H., Lai, L., Li, B., Li, C., Li, H., Litman, R., Liu, X., Liu, Z., Lu, Y., Tatsuma, A., and Ye, J. (2014).
SHREC'14 track: Shape retrieval of non-rigid 3d human models.
In *Proceedings of the 7th Eurographics workshop on 3D Object Retrieval*, EG 3DOR'14. Eurographics Association.

Bibliography II

[Torgerson, 1952] Torgerson, W. S. (1952).
Multidimensional scaling: I. theory and method.
Psychometrika, 17(4):401–419.

[Weinberger et al., 2005] Weinberger, K. Q., Blitzer, J., and Saul, L. K. (2005).
Distance metric learning for large margin nearest neighbor classification.
In *Advances in neural information processing systems*, pages 1473–1480.