Machine Learning

Metric Learning: Algorithms and Theory

Marc Sebban & Amaury Habrard

Hubert Curien lab, UMR CNRS 5516
University of Jean Monnet Saint-Étienne (France)

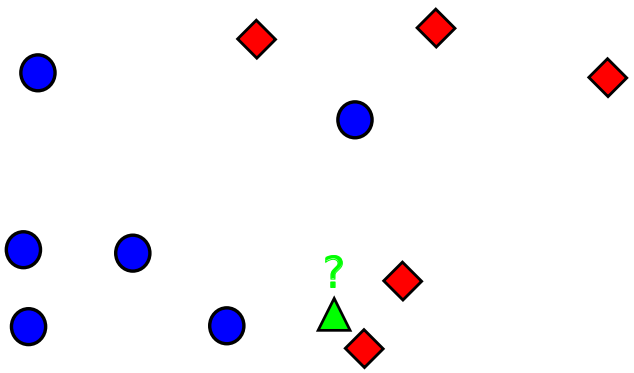
Academic year 2016-2017

Outline

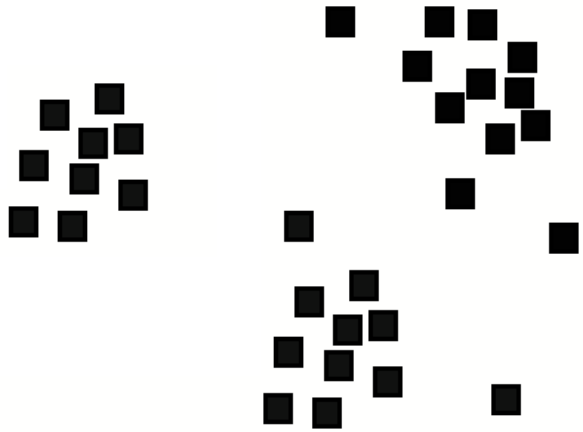
Importance of Metrics

Pairwise metric

The notion of **metric plays an important role in many domains** such as *classification*, *regression*, *clustering*, *ranking*, *information retrieval*, etc.



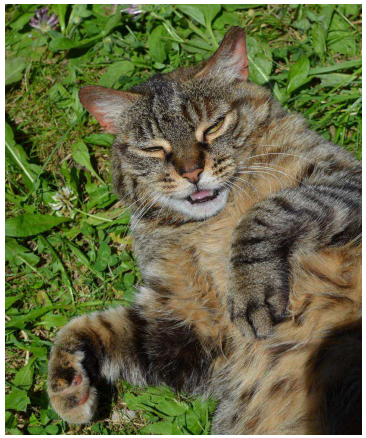
Supervised learning



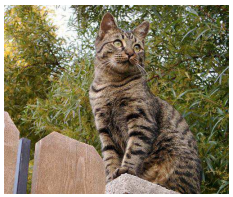
Unsupervised learning

Importance of Metrics

Query document



Most similar documents


Information retrieval: Return the documents the most similar to a query

Definition of a metric

A metric/distance function/distance d over a set X is a positive function:

$$d : X \times X \to \mathbb{R}^+$$

such that for any $x, y, z \in X$ the following conditions must be satisfied

1. $d(x, y) \geq 0$ *non-negativity*
2. $d(x, y) = 0 \Leftrightarrow x = y$ *identity of indiscernibles*
3. $d(x, y) = d(y, x)$ *symmetry*
4. $d(x, z) \leq d(x, y) + d(y, z)$ *triangle inequality*

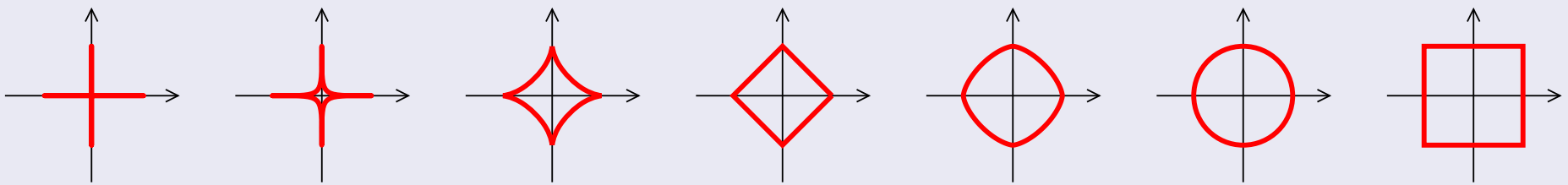
Minkowski distances: family of distances induced by ℓ_p norms

$$d_p(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_p = \left(\sum_{i=1}^{d} |x_i - x_i'|^p \right)^{1/p}$$

- For $p = 1$, the **Manhattan distance** $d_{man}(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{d} |x_i - x_i'|$.
- For $p = 2$, the "ordinary" **Euclidean distance**:

$$d_{euc}(\mathbf{x}, \mathbf{x}') = \left(\sum_{i=1}^{d} |x_i - x_i'|^2 \right)^{1/2} = \sqrt{(\mathbf{x} - \mathbf{x}')^T (\mathbf{x} - \mathbf{x}')}$$

- For $p \to \infty$, the **Chebyshev distance** $d_{che}(\mathbf{x}, \mathbf{x}') = \max_i |x_i - x_i'|$.



Some other types of "metrics"

- Distances between structured data (sequences, trees, graphs): Levensthein (edit) distance, graph isomorphism
- Distance between distributions: Wasserstein distance

Other measures

- χ^2 distance
- Divergence: Kullback-Leibler divergence, relative entropy, logDet Divergence
- Similarity measures: cosine similarity, kernel functions (SVM)

$\Rightarrow$ Importance of the choice of the metric for the task at hand

Key question

How to choose the right metric?

The notion of good metric is problem-dependent

Each problem has its own notion of similarity, which is often badly captured by standard metrics.

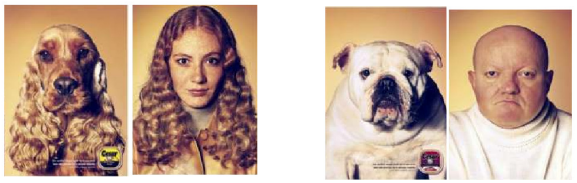
Limitations of standard metrics

Class 1: Humans

Class 2: Dogs

Predicted label using the 1-nearest neighbor algorithm?

Limitations of standard metrics



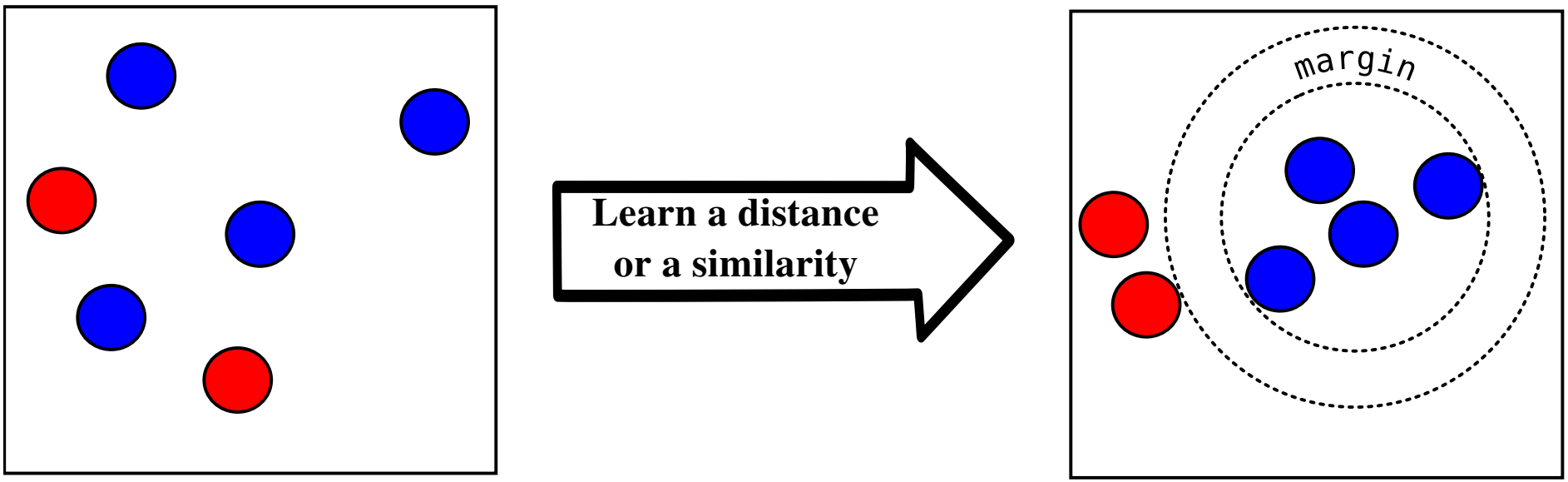
It's not what it looks like...

Metric learning

Adapt the metric to the problem of interest

Learn the metric from data

Basic idea: learn a metric that assigns small (resp. large) distance to pairs of examples that are **semantically** similar (resp. dissimilar).

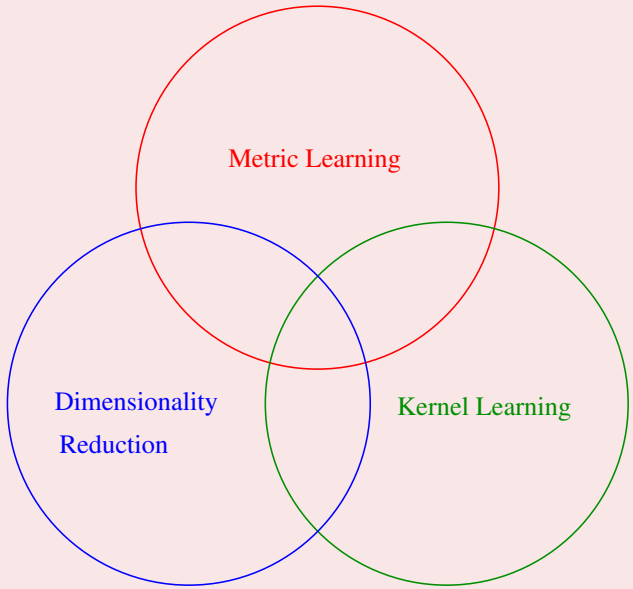


It typically **induces a change of representation space** which satisfies constraints.

Related topics

Related topics

- Kernel learning: typically learns the Gram matrix.
- Multiple Kernel Learning: learn to combine a set of predefined kernels.
- Dimensionality reduction: often unsupervised, primary goal is really to reduce data dimension. It usually assumes that data lie on an embedded low-dimensional manifold.



"Learnable" Metrics

The Mahalanobis distance

$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$, the Mahalanobis distance is defined as follows:

$$d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^T \mathbf{M} (\mathbf{x} - \mathbf{x}')},$$

where $\mathbf{M} \in \mathbb{R}^{d \times d}$ is a symmetric PSD matrix ($\mathbf{M} \succeq 0$).

The original term refers to the case where $\mathbf{x}$ and $\mathbf{x}'$ are random vectors from the same distribution with covariance matrix $\mathbf{\Sigma}$, with $\mathbf{M} = \mathbf{\Sigma}^{-1}$.

Useful properties

If $\mathbf{M} \succeq 0$, then

- $\mathbf{x}^T \mathbf{M} \mathbf{x} \geq 0 \ \forall \mathbf{x}$ (as a linear operator, can be seen as nonnegative scaling).
- $\mathbf{M} = \mathbf{L}^T \mathbf{L}$ for some matrix $\mathbf{L} \in \mathbb{R}^{k \times d}$ with k the rank of $\mathbf{M}$.

Mahalanobis distance learning

Using the decomposition $\mathbf{M} = \mathbf{L}^T\mathbf{L}$, where $\mathbf{L} \in \mathbb{R}^{k \times d}$, where k is the rank of $\mathbf{M}$, one can rewrite $d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}')$.

$$\begin{aligned} d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}') &= \sqrt{(\mathbf{x} - \mathbf{x}')^T\mathbf{L}^T\mathbf{L}(\mathbf{x} - \mathbf{x}')} \\ &= \sqrt{(\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}')^T(\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}')} \\ &= \sqrt{(\widetilde{\mathbf{x}} - \widetilde{\mathbf{x}}')^T(\widetilde{\mathbf{x}} - \widetilde{\mathbf{x}}')} \\ &= d_{euc}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{x}}') \end{aligned}$$

Mahalanobis distance learning = Learning a linear projection

If $\mathbf{M}$ is learned, a Mahalanobis distance implicitly corresponds to **computing the Euclidean distance after a learned linear projection** of the data by $\mathbf{L}$ in a k-dimensional space.

Metric learning in a nutshell

A hot research topic in Machine Learning

- Really started with a paper at NIPS 2002 (MMC [Xing et al.]).
- Ever since, several papers each year in top conferences and journals.
- Since 2010, tutorials and workshops at major machine learning (NIPS, ICML) and computer vision (ICCV, ECCV) venues.

Metric learning in a nutshell

General formulation

Given a parameterized metric, estimate its parameters $\mathbf{M}$ s.t. the metric best fulfills the semantic constraints

$\rightarrow$ Solve an optimization problem of the form

$$\mathbf{M}^* = \underset{\mathbf{M} \succeq 0}{\arg\min} \left[\ell(\mathbf{M}, \mathcal{S}, \mathcal{D}, \mathcal{R}) + \lambda R(\mathbf{M}) \right],$$

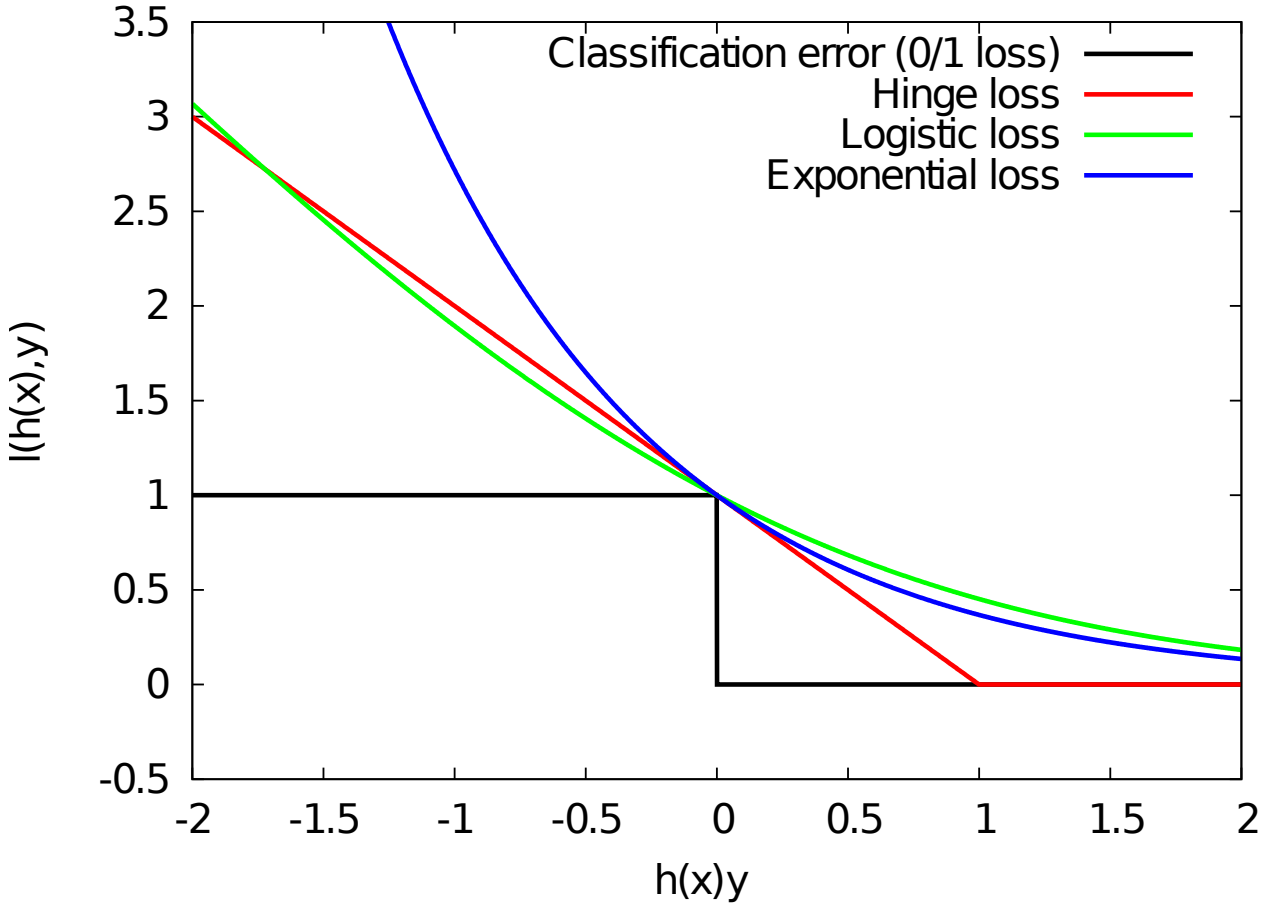
where

- $\ell(\mathbf{M}, \mathcal{S}, \mathcal{D}, \mathcal{R})$ is a loss function that penalizes violated constraints,
- $R(\mathbf{M})$ is some regularizer on $\mathbf{M}$,
- and $\lambda \geq 0$ is the regularization parameter.

State of the art methods essentially differ by the choice of the **loss function**, the **constraints** and the **regularizer** on $\mathbf{M}$.

Margin-based loss functions for binary classification

$$\mathbf{M}^* = \underset{\mathbf{M} \succeq 0}{\arg\min} \left[\ell(\mathbf{M}, \mathcal{S}, \mathcal{D}, \mathcal{R}) + \lambda R(\mathbf{M}) \right]$$



Constraints in Metric learning

$$\mathbf{M}^* = \arg\min_{\mathbf{M} \succeq 0} \left[\ell(\mathbf{M}, \mathcal{S}, \mathcal{D}, \mathcal{R}) + \lambda R(\mathbf{M}) \right]$$

Constraints

Three main categories of semantic constraints from data pairs/triplets

- $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are similar}\}$
- $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are dissimilar}\}$
- $\mathcal{R} = \{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) : \mathbf{x}_i \text{ is more similar to } \mathbf{x}_j \text{ than to } \mathbf{x}_k\}$

Importance of the regularization term $R(\mathbf{M})$

$$\mathbf{M}^* = \underset{\mathbf{M} \succeq 0}{\arg\min} \left[\ell(\mathbf{M}, \mathcal{S}, \mathcal{D}, \mathcal{R}) + \lambda R(\mathbf{M}) \right]$$

- Simple and classic choice: $\|\mathbf{M}\|_{\mathcal{F}}^2 = \sum_{i,j}^{d} \mathbf{M}_{ij}^2$ (Frobenius norm)
  (Use in [Schultz and Joachims, 2003] and many others)
- Feature selection with Mixed $L_{2,1}$ norm: $\|\mathbf{M}\|_{2,1} = \sum_{i}^{d} \|\mathbf{M}_i\|$, convex but non smooth (*e.g.* [Ying et al., 2009])
  Needs efficient proximal gradient algorithm (*e.g.* [Bach et al., 2012])
- Favoring low rank matrices for dimensionality reduction with trace (or nuclear) norm $\|\mathbf{M}\|_* = \sum_{i=1}^{d} \sigma_i(\mathbf{M})$ (*e.g.* [McFee and Lanckriet, 2010]), convex but non smooth efficient Frank-Wolfe algorithms [Jaggi, 2013]
- Use of the LogDet divergence (used in ITML [Davis et al., 2007]) - see later)

MMC (Xing et al. 2002)

MMC (Xing et al. 2002)

Pioneering work in Mahalanobis distance learning.

$$\max_{\mathbf{M} \in \mathbb{S}_+^d} \quad \sum_{(x_i, x_j) \in \mathcal{D}} d_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_j})$$

$$\text{s.t.} \quad \sum_{(x_i, x_j) \in \mathcal{S}} d^2_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_j}) \leq 1.$$

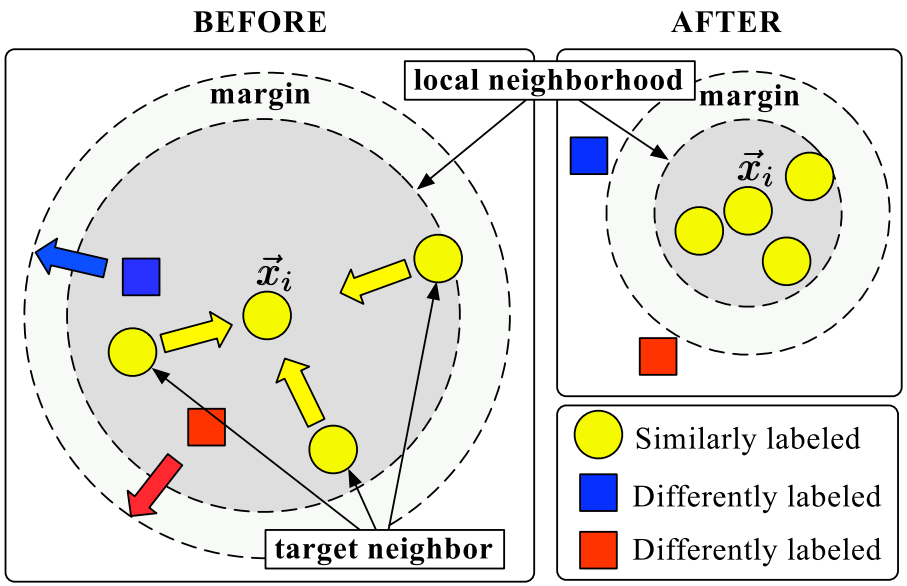
The algorithm is a basic SDP approach based on eigenvalue decomposition. This makes it intractable for medium and high-dimensional problems.

LMNN (Weinberger et al. 2005) +2200 citations

Main Idea

Define constraints tailored to k-NN in a local way: the k nearest neighbors should be of same class ("target neighbors"), while examples of different classes should be kept away ("impostors"):

$$\begin{aligned} \mathcal{S} &= \{(\mathbf{x}_i, \mathbf{x}_j) : y_i = y_j \text{ and } \mathbf{x}_j \text{ belongs to the } k\text{-neighborhood of } \mathbf{x}_i\}, \\ \mathcal{R} &= \{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) : (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S}, y_i \neq y_k\}. \end{aligned}$$



Hard Formulation

$$\min_{\mathbf{M} \succeq 0} \quad \sum_{(\mathbf{x_i}, \mathbf{x_j}) \in \mathcal{S}} d^2_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_j})$$

$$\text{s.t.} \quad d^2_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_k}) - d^2_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_j}) \geq 1 \quad \forall (\mathbf{x_i}, \mathbf{x_j}, \mathbf{x_k}) \in \mathcal{R}.$$

Soft Formulation

$$\min_{\mathbf{M} \succeq 0} \quad (1-\mu) \sum_{(\mathbf{x_i}, \mathbf{x_j}) \in \mathcal{S}} d^2_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_j}) + \mu \sum_{i,j,k} \xi_{ijk}$$

$$\text{s.t.} \quad d^2_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_k}) - d^2_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_j}) \geq 1 - \xi_{ijk} \qquad \forall (\mathbf{x_i}, \mathbf{x_j}, \mathbf{x_k}) \in \mathcal{R},$$

where μ controls the "pull/push" trade-off.

Remarks

- **Advantages**: Convex, with a solver based on working set and subgradient descent. Can deal with millions of constraints and very popular in practice.
- **Drawback**: Subject to overfitting in high dimension.

ITML (Davis et al. 2007)

Information-Theoretical Metric Learning (ITML) introduces LogDet divergence regularization. This Bregman divergence on PSD matrices is defined as:

$$D_{ld}(\mathbf{M}, \mathbf{M_0}) = trace(\mathbf{M}\mathbf{M_0}^{-1}) - \log\det(\mathbf{M}\mathbf{M_0}^{-1}) - d.$$

where d is the dimension of the input space and $\mathbf{M_0}$ is some PSD matrix we want to remain close to. ITML is formulated as follows:

$$\begin{aligned} \min_{\mathbf{M} \succeq 0} \quad & D_{ld}(\mathbf{M}, \mathbf{M_0}) \\ \text{s.t.} \quad & d^2_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_j}) \leq u \quad \forall (\mathbf{x_i}, \mathbf{x_j}) \in \mathcal{S} \\ & d^2_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_j}) \geq v \quad \forall (\mathbf{x_i}, \mathbf{x_j}) \in \mathcal{D}, \end{aligned}$$

The LogDet divergence is finite iff $\mathbf{M}$ is PSD (cheap way of preserving a PSD matrix).

Non-Linear Extensions

Limitation

LMNN and ITML generate linear transformations that are unable to capture nonlinear structures from data.

3 main categories of approaches to fix the problem:

- **Kernelized metrics**: learn the metric in a new feature space induced by a kernel - requires often a technical work
  Other solution: use KPCA as a pre-process [Chatpatanasiri et al., 2010].
- **Learn non linear mappings** based on regression trees [Kedem et al., 2012], Deep neural networks [Chopra et al., 2005, Hu et al., 2014] (non convex)
- **Local metrics**: learn many metrics for different part of the space [Shi et al., 2014]. (space splitting, blow-up of parameters)

Nonlinear metric learning

Kernelization of linear methods

- Some algorithms have been shown to be kernelizable, but in general this is not trivial: a new formulation of the problem has to be derived, where interface to the data is **limited to inner products**, and sometimes a different implementation is necessary.
- When the number of training examples n is large, **learning n^2 parameters may be intractable**.

A solution: KPCA trick (Chatpatanasiri et al., 2010)

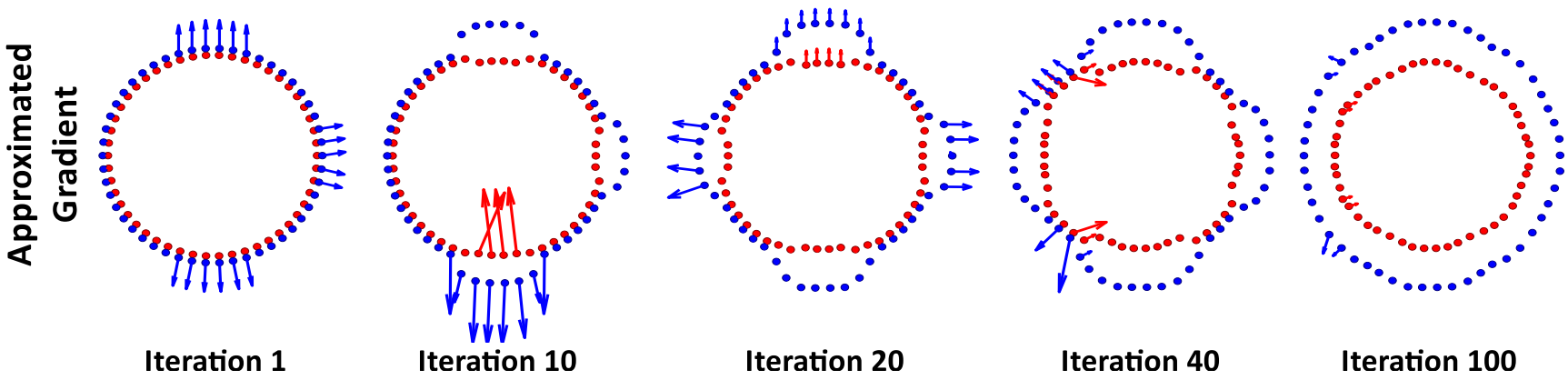
- Use KPCA (PCA in kernel space) to get a nonlinear but low-dimensional projection of the data.
- Then use unchanged algorithm!

Nonlinear metric learning

Learning a nonlinear metric: GB-LMNN (Kedem et al. 2012)

Main idea

- Learn a nonlinear mapping ϕ to optimize the Euclidean distance $d_\phi(\mathbf{x}, \mathbf{x}') = \|\phi(\mathbf{x}) - \phi(\mathbf{x}')\|_2$ in the transformed space.
- $\phi = \phi_0 + \alpha \sum_{t=1}^{T} h_t$, where ϕ_0 is the mapping learned by linear LMNN, and $h_1, \ldots, h_T$ are **gradient boosted regression trees**.
- Intuitively, each tree divides the space into 2^p regions, and instances falling in the **same region are translated by the same vector**.



Nonlinear metric learning

Local metric learning

Motivation

- Simple linear metrics perform well locally.
- Since everything is linear, can keep formulation convex.

M^2-LMNN (Weinberger and Saul 2008,2009)

- Partition in C clusters (in a supervised or unsupervised way).
- C Mahalanobis distances are learned.

Pitfalls

- How to split the space?
- How to avoid a blow-up in number of parameters to learn, and avoid overfitting?
- How to obtain a proper (continuous) global metric?
- ...

Issues for large scale learning: Large n

Large datasets

Number of pairwise/triplet constraints can grow in $\mathcal{O}(n^2)/\mathcal{O}(n^3)$.

- Online Learning algorithm (*e.g.* [Chechik et al., 2010], [Jain et al. 2008])
  - Receive one (small) constraints at a time
  - Optimize the loss wrt the current metric and the constraint
  - Update the metric and iterate
- Stochastic (gradient descent) and distributed optimization (See [Xie and Xing, 2014, Clémençon et al., 2015])

Mahalanobis distance learning

LEGO (Jain et al. 2008)

Formulation

At each step, receive $(\mathbf{x}_t, \mathbf{x}'_t, y_t)$ where y_t is the target distance between $\mathbf{x}_t$ and $\mathbf{x}'_t$, and update as follows:

$$\mathbf{M}^{t+1} = \underset{\mathbf{M} \succeq 0}{\arg\min}\, D_{ld}(\mathbf{M}, \mathbf{M}^t) + \lambda \ell(\mathbf{M}, \mathbf{x}_t, \mathbf{x}'_t, y_t),$$

where ℓ is a loss function (square loss, hinge loss...).

Remarks

- It turns out that the above update has a closed-form solution which maintains $\mathbf{M} \succeq 0$ automatically.
- Can derive a regret bound.

Issues for large scale learning: Large d

High dimensional data

Computational complexity in $\mathcal{O}(d^2)/\mathcal{O}(d^3)$

- Constraints on the matrix: diagonal matrix
  [Xing et al., 2002, Schultz and Joachims, 2003].
- Perform dimensionality reduction as a pre-process (PCA).
- Use matrix decomposition
  - Low-rank decomposition $\mathbf{M} = \mathbf{L}^T\mathbf{L}$ with $\mathbf{L} \in \mathbb{R}^{d \times r}$
    [Goldberger et al., 2004]
    - Generally non convex in classic metric learning settings
    - Must tune the target rank r
  - Rank-1 decomposition: $\mathbf{M} = \sum_{i=1}^{K} w_k \mathbf{b}_k \mathbf{b}_k^T$ [Shi et al., 2014]
    - Learn K parameters but hard to generate good bases in high dimensions
  - Sparsity constraints [Liu et al., 2015]
    - Decomposition as rank-1 of matrices with sparsity structure

Theoretical Guarantees in Metric Learning

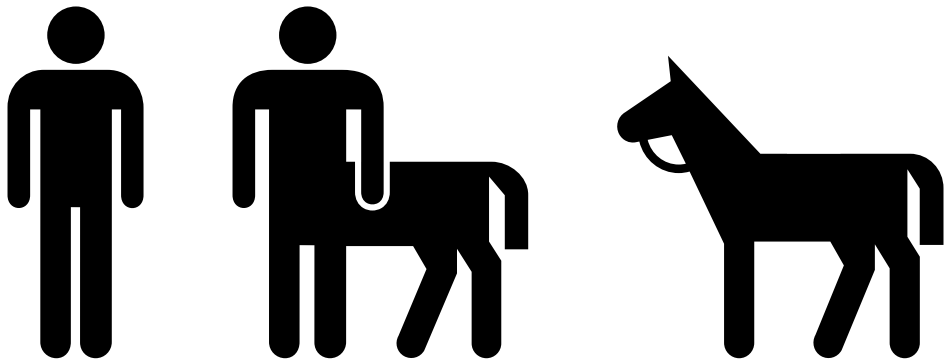
- Guarantees on the learned metric
- Guarantees on the algorithm that uses the learned metric

Framework

- From distance functions to similarity functions in binary linear classification (where $y \in \{-1; +1\}$).
- (ϵ, γ, τ)-**good** similarity functions [Balcan et al. 2006]
- Uniform stability to derive generalization bounds.

Beyond Mahalanobis distance learning

- Maintaining $\mathbf{M} \succeq 0$ is often costly, especially in high dimensions.
- Objects must have same dimension.
- Distance properties can be useful (e.g., for fast neighbor search), but restrictive. Evidence that our notion of (visual) similarity violates the triangle inequality.



Motivation to learn similarity functions!!

Similarity learning

Cosine similarity

The cosine similarity (widely used in data mining) measures the cosine of the angle between two instances, and can be computed as

$$K_{cos}(\mathbf{x}, \mathbf{x}') = \frac{\mathbf{x}^T \mathbf{x}'}{\|\mathbf{x}\|_2 \|\mathbf{x}'\|_2}.$$

Bilinear similarity

The bilinear similarity is related to the cosine but does not include normalization and is parameterized by a matrix $\mathbf{M}$:

$$K_{\mathbf{M}}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{M} \mathbf{x}',$$

where $\mathbf{M}$ is not required to be PSD nor symmetric.

Generalization guarantees in linear classification

Definition (Balcan et al., 2008)

A similarity function $K \in [-1, 1]$ is (ϵ, γ, τ)-**good** w.r.t. to an indicator function $R(x)$ defining a set of "reasonable points" if:

1. A $1 - \epsilon$ probability mass of examples (x, y) satisfy:

$$\mathbb{E}_{(x',y')\sim P}\left[yy'K(x,x')|R(x')\right] \geq \gamma.$$

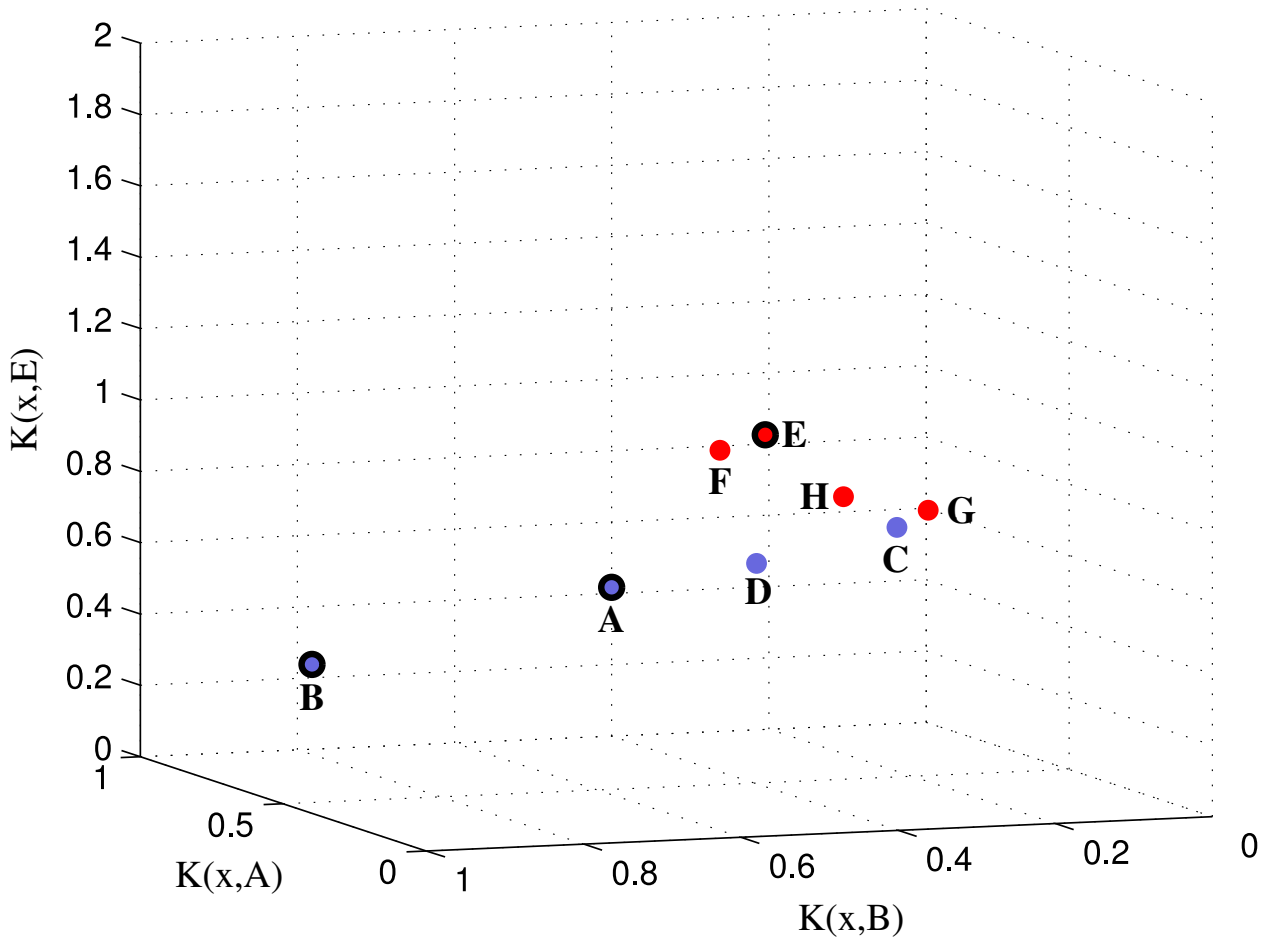
2. $\Pr_{x'}[R(x')] \geq \tau.$

$\epsilon, \gamma, \tau \in [0, 1]$

- The first condition requires that a $1 - \epsilon$ proportion of examples x are **on average** more similar to reasonable examples of the same class than to reasonable examples of the opposite class by a margin γ.
- The second condition means that at least a τ proportion of the examples are reasonable.

If K is good and R is known

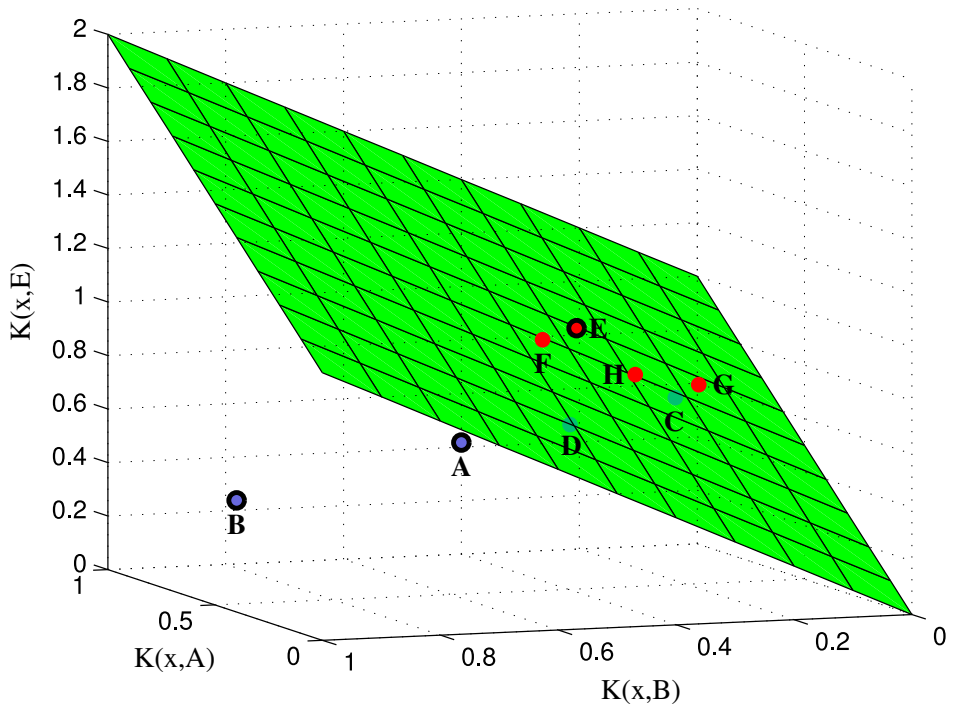
Use K to map the examples to the space ϕ of "the similarity scores with the reasonable points" (**similarity map**).



Generalization guarantees in linear classification

A trivial linear classifier

By definition of (ϵ, γ, τ)-goodness, we have a linear classifier in ϕ that achieves true risk ϵ at margin γ.



First framework which establishes a strong relationship between the quality of the similarity function and the quality of the induced linear classifier.

Generalization guarantees in Supervised Learning

Theorem (Balcan et al., 2008)

If R is unknown, given K is (ϵ, γ, τ)-good and enough points to create a similarity map, with high probability **there exists a linear separator** α *that has true risk $\epsilon + \epsilon_1$ at margin γ.*

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \left[1 - \sum_{j=1}^{m} \alpha_j y_i K(x_i, x_j) \right]_+ + \lambda \|\boldsymbol{\alpha}\|_1$$

Looks like ℓ_1-SVM but:

- K is not required to be a kernel.
- It opens the door to (ϵ, γ, τ)-good similarity learning.
- Bilinear similarity: $K_M(x, x') = x^T M x'$.

Optimization of the goodness - SLLC algorithm

[Bellet et al. 2012] Bellet A., Habrard H., Sebban M. - *Similarity Learning for Provably Accurate Sparse Linear Classification - ICML 2012.*

The performance of the linear classifier theoretically depends on how well the similarity function satisfies the definition of goodness.

$$\mathbb{E}_{(x',y')\sim P}\left[yy'K(x,x')|R(x')\right] \geq \gamma.$$

SLLC optimizes the empirical goodness $\hat{\epsilon}$ of K over the training set.

Formulation of SLLC

$$\min_{\mathbf{M}\in\mathbb{R}^{d\times d}} \quad \frac{1}{n}\sum_{i=1}^{n}\left[1 - y_i\frac{1}{\gamma|R|}\sum_{\mathbf{x}_j\in R} y_j K_{\mathbf{M}}(\mathbf{x}_i,\mathbf{x}_j)\right]_+ \quad + \quad \beta\|\mathbf{M}\|_{\mathcal{F}}^2,$$

where $K_{\mathbf{M}}(\mathbf{x},\mathbf{x}') = \mathbf{x}^T\mathbf{M}\mathbf{x}'$.

Guarantees on the learned metrics

Generalization guarantees with uniform stability

Definition (Uniform stability for metric learning)

A learning algorithm $\mathcal{A}$ has a **uniform stability** in κ/n, where $\kappa > 0$, if

$$\forall (T, \mathbf{x}), \forall i, \sup_{\mathbf{x}_1, \mathbf{x}_2} |\ell(\mathcal{A}_T, \mathbf{x}_1, \mathbf{x}_2) - \ell(\mathcal{A}_{T^{i,\mathbf{x}}}, \mathbf{x}_1, \mathbf{x}_2)| \leq \frac{\kappa}{n},$$

where $\mathcal{A}_T$ is the metric learned by $\mathcal{A}$ from T, and $T^{i,\mathbf{x}}$ is the set obtained by replacing $\mathbf{x}_i \in T$ by a new example $\mathbf{x}$.

Theorem (Uniform stability bound)

For any algorithm $\mathcal{A}$ with uniform stability κ/n, with probability $1 - \delta$ over the random sample T, we have:

$$R^{\ell}(\mathcal{A}_T) \leq R^{\ell}_T(\mathcal{A}_T) + \frac{2\kappa}{n} + (2\kappa + B)\sqrt{\frac{\ln(2/\delta)}{2n}},$$

where B is a problem-dependent constant.

Stability of SLLC

Formulation of SLLC

$$\min_{\mathbf{M}\in\mathbb{R}^{d\times d}} \quad \frac{1}{n}\sum_{i=1}^{n}\left[1 - y_i \frac{1}{\gamma|R|}\sum_{\mathbf{x}_j\in\mathcal{R}} y_j K_{\mathbf{M}}(\mathbf{x}_i,\mathbf{x}_j)\right]_+ \quad + \quad \beta\|\mathbf{M}\|_{\mathcal{F}}^2,$$

where

$$K_{\mathbf{M}}(\mathbf{x},\mathbf{x}') = \mathbf{x}^T\mathbf{M}\mathbf{x}'.$$

Lemma

Let n and $|R|$ be the number of training examples and reasonable points respectively, $|R| = \hat{\tau}n$ with $\hat{\tau} \in]0,1]$. SLLC has a uniform stability in $\frac{\kappa}{n}$ with

$$\kappa = \frac{1}{\gamma}\left(\frac{1}{\beta\gamma} + \frac{2}{\hat{\tau}}\right),$$

where β is the regularization parameter and γ the margin.

Consistency guarantees of SLLC

Theorem

Let $\gamma > 0$, $\delta > 0$ and $n_{\mathcal{T}} > 1$. With probability at least $1 - \delta$, for any model $\mathbf{M}$ learned with SLLC, we have:

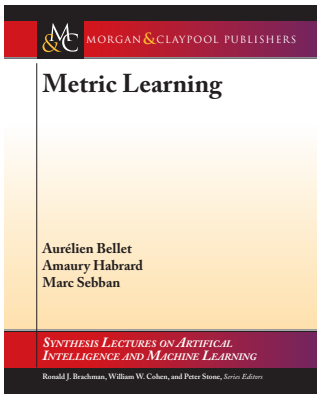
$$\epsilon_{\mathbf{M}} \leq \hat{\epsilon}_{\mathbf{M}} + \frac{1}{n}\left(\frac{1}{\gamma}\left(\frac{1}{\beta\gamma} + \frac{2}{\hat{\tau}}\right)\right) + \left(\frac{1}{\gamma}\left(\frac{1}{\beta\gamma} + \frac{2}{\hat{\tau}}\right) + 1\right)\sqrt{\frac{\ln 1/\delta}{2n}}$$

where:

- $\hat{\epsilon}_{\mathbf{M}} = \frac{1}{n}\sum_{i=1}^{n}\left[1 - y_i \frac{1}{\gamma|R|}\sum_{k=1}^{|R|} y_k K_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_k})\right]_+$.
- $\epsilon_{\mathbf{M}} = \mathbb{E}_{(\mathbf{x_i}, y_i)\sim P}\left[1 - y_i \frac{1}{\gamma|R|}\sum_{k=1}^{|R|} y_k K_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_k})\right]_+$.

Some references and sources

- **Book** *A. Bellet, A. Habrard and M. Sebban.* **Metric Learning**. Morgan & Claypool Publishers, 2015.



- **Tutorial** *A. Bellet and M. Cord.* **Similarity and Distance Metric Learning with Applications to Computer Vision**, ECML-PKDD Tutorial, 2015
- **Surveys**
  - A. Bellet, A. Habrard and M. Sebban. **A Survey on Metric Learning for Feature Vectors and Structured Data**. `http://arxiv.org/abs/1306.6709`
  - Brian Kulis: **Metric Learning: A Survey**. Foundations and Trends in Machine Learning 5(4): 287-364, 2013.

References I

[Bach et al., 2012] Bach, F. R., Jenatton, R., Mairal, J., and Obozinski, G. (2012).
Optimization with Sparsity-Inducing Penalties.
Foundations and Trends in Machine Learning, 4(1):1–106.

[Bellet and Habrard, 2015] Bellet, A. and Habrard, A. (2015).
Robustness and Generalization for Metric Learning.
Neurocomputing, 151(1):259–267.

[Cao et al., 2012] Cao, Q., Guo, Z.-C., and Ying, Y. (2012).
Generalization Bounds for Metric and Similarity Learning.
Technical report, University of Exeter.

[Chatpatanasiri et al., 2010] Chatpatanasiri, R., Korsrilabutr, T., Tangchanachaianan, P., and Kijsirikul, B. (2010).
A new kernelization framework for Mahalanobis distance learning algorithms.
Neurocomputing, 73:1570–1579.

[Chechik et al., 2010] Chechik, G., Sharma, V., Shalit, U., and Bengio, S. (2010).
Large Scale Online Learning of Image Similarity Through Ranking.
Journal of Machine Learning Research, 11:1109–1135.

References II

[Chopra et al., 2005] Chopra, S., Hadsell, R., and LeCun, Y. (2005).
Learning a Similarity Metric Discriminatively, with Application to Face Verification.
In *CVPR*, pages 539–546.

[Clémençon et al., 2015] Clémençon, S., Bellet, A., and Colin, I. (2015).
Scaling-up Empirical Risk Minimization: Optimization of Incomplete U-statistics.
Technical report, arXiv:1501.02629.

[Davis et al., 2007] Davis, J. V., Kulis, B., Jain, P., Sra, S., and Dhillon, I. S. (2007).
Information-theoretic metric learning.
In *ICML*, pages 209–216.

[Goldberger et al., 2004] Goldberger, J., Roweis, S., Hinton, G., and Salakhutdinov, R. (2004).
Neighbourhood Components Analysis.
In *NIPS*, pages 513–520.

[Hu et al., 2014] Hu, J., Lu, J., and Tan, Y.-P. (2014).
Discriminative Deep Metric Learning for Face Verification in the Wild.
In *CVPR*, pages 1875–1882.

[Jaggi, 2013] Jaggi, M. (2013).
Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization.
In *ICML*.

References III

[Jin et al., 2009] Jin, R., Wang, S., and Zhou, Y. (2009).
Regularized Distance Metric Learning: Theory and Algorithm.
In *NIPS*, pages 862–870.

[Kedem et al., 2012] Kedem, D., Tyree, S., Weinberger, K., Sha, F., and Lanckriet, G. (2012).
Non-linear Metric Learning.
In *NIPS*, pages 2582–2590.

[Liu et al., 2015] Liu, K., Bellet, A., and Sha, F. (2015).
Similarity Learning for High-Dimensional Sparse Data.
In *AISTATS*, pages 653–662.

[McFee and Lanckriet, 2010] McFee, B. and Lanckriet, G. R. G. (2010).
Metric Learning to Rank.
In *ICML*, pages 775–782.

[Schultz and Joachims, 2003] Schultz, M. and Joachims, T. (2003).
Learning a Distance Metric from Relative Comparisons.
In *NIPS*.

[Shi et al., 2014] Shi, Y., Bellet, A., and Sha, F. (2014).
Sparse Compositional Metric Learning.
In *AAAI*, pages 2078–2084.

References IV

[Xie and Xing, 2014] Xie, P. and Xing, E. (2014).
Large Scale Distributed Distance Metric Learning.
Technical report, arXiv:1412.5949.

[Xing et al., 2002] Xing, E. P., Ng, A. Y., Jordan, M. I., and Russell, S. J. (2002).
Distance Metric Learning with Application to Clustering with Side-Information.
In *NIPS*, pages 505–512.

[Ying et al., 2009] Ying, Y., Huang, K., and Campbell, C. (2009).
Sparse Metric Learning via Smooth Optimization.
In *NIPS*, pages 2214–2222.