# metric learning course

Cours RI Master DAC UPMC (Construit à partir d'un tutorial ECML-PKDD 2015 (A. Bellet, M. Cord))

1. Introduction
2. Linear metric learning
3. Nonlinear extensions
4. Large-scale metric learning
5. Metric learning for structured data
6. Generalization guarantees

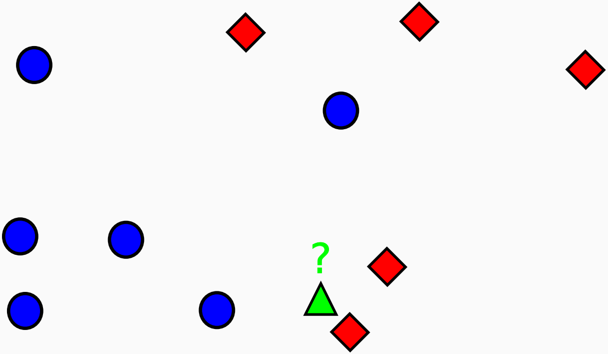
# introduction

# motivation

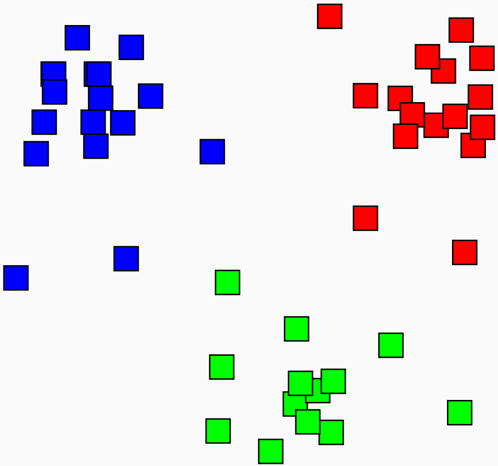
- Similarity / distance judgments are essential components of many human cognitive processes (see e.g., [Tversky, 1977])
    - Compare perceptual or conceptual representations
    - Perform recognition, categorization...
- Underlie most machine learning and data mining techniques

# motivation

Nearest neighbor classification



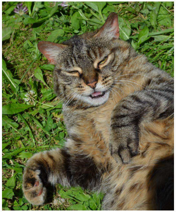
## Clustering



# motivation

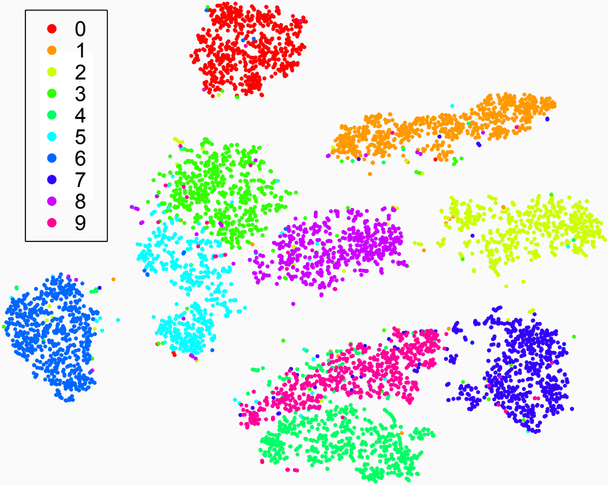
## Information retrieval

**Query document**



**Most similar documents**

## Data visualization

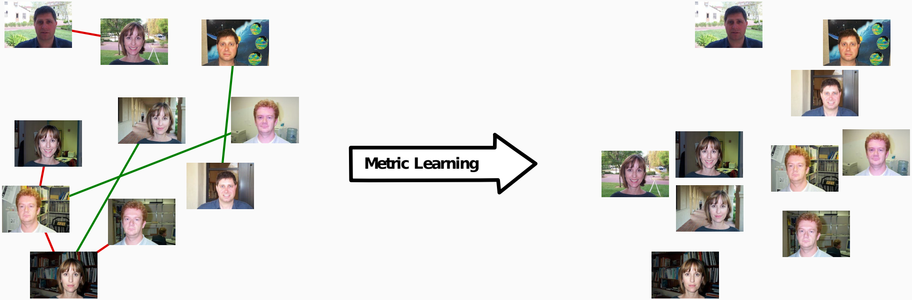



(image taken from [van der Maaten and Hinton, 2008])

# motivation

- Choice of similarity is crucial to the performance
- Humans weight features differently depending on context [Nosofsky, 1986, Goldstone et al., 1997]
    - Facial recognition vs. determining facial expression
- Fundamental question: how to appropriately measure similarity or distance for a given task?
- Metric learning $\rightarrow$ infer this automatically from data
- Note: we will refer to *distance* or *similarity* indistinctly as *metric*

# metric learning in a nutshell



# metric learning in a nutshell

## Basic recipe

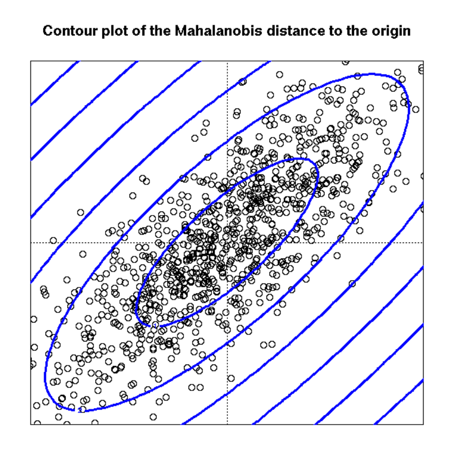
1. Pick a parametric distance or similarity function
   - Say, a distance $D_M(x, x')$ function parameterized by $M$
2. Collect similarity judgments on data pairs/triplets
   - $\mathcal{S} = \{(x_i, x_j) : x_i \text{ and } x_j \text{ are similar}\}$
   - $\mathcal{D} = \{(x_i, x_j) : x_i \text{ and } x_j \text{ are dissimilar}\}$
   - $\mathcal{R} = \{(x_i, x_j, x_k) : x_i \text{ is more similar to } x_j \text{ than to } x_k\}$
3. Estimate parameters s.t. metric best agrees with judgments
   - Solve an optimization problem of the form

$$\hat{M} = \arg\min_M \left[ \underbrace{\ell(M, \mathcal{S}, \mathcal{D}, \mathcal{R})}_{\text{loss function}} + \underbrace{\lambda reg(M)}_{\text{regularization}} \right]$$

# linear metric learning

# mahalanobis distance learning

- Mahalanobis distance: $D_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}') = \sqrt{(\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{M}(\boldsymbol{x} - \boldsymbol{x}')}$ where $\boldsymbol{M} = \boldsymbol{Cov}(\boldsymbol{X})^{-1}$ where $\boldsymbol{Cov}(\boldsymbol{X})$ is the covariance matrix estimated over a data set $\boldsymbol{X}$



# mahalanobis distance learning

- Mahalanobis (pseudo) distance:

$$D_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}') = \sqrt{(\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{M} (\boldsymbol{x} - \boldsymbol{x}')}$$

  where $\boldsymbol{M} \in \mathbb{S}_+^d$ is a symmetric PSD $d \times d$ matrix

- Equivalent to Euclidean distance after linear projection:

$$D_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}') = \sqrt{(\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{L}^T \boldsymbol{L} (\boldsymbol{x} - \boldsymbol{x}')} = \sqrt{(\boldsymbol{Lx} - \boldsymbol{Lx}')^T (\boldsymbol{Lx} - \boldsymbol{Lx}')}$$

- If $\boldsymbol{M}$ has rank $k \leq d$, $\boldsymbol{L} \in \mathbb{R}^{k \times d}$ reduces data dimension

# mahalanobis distance learning

A first approach [Xing et al., 2002]

- Targeted task: clustering with side information

**Formulation**

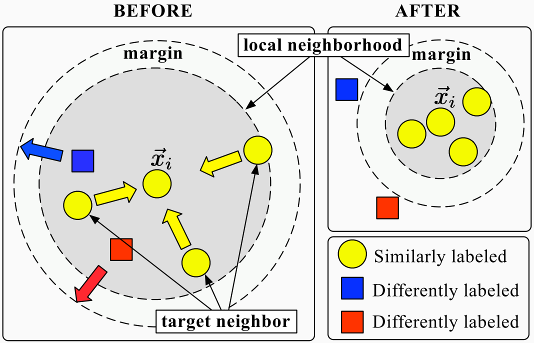
$$\begin{aligned} \max_{\boldsymbol{M} \in \mathbb{S}_+^d} \quad & \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{D}} D_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ \text{s.t.} \quad & \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{S}} D_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq 1 \end{aligned}$$

- Convex in $\boldsymbol{M}$ and always feasible (take $\boldsymbol{M} = \mathbf{0}$)
- Solved with projected gradient descent
- Time complexity of projection on $\mathbb{S}_+^d$ is $O(d^3)$
- Only look at sums of distances

# mahalanobis distance learning

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

- Targeted task: $k$-NN classification
- Constraints derived from labeled data
  - $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) : y_i = y_j, \boldsymbol{x}_j \text{ belongs to } k\text{-neighborhood of } \boldsymbol{x}_i\}$
  - $\mathcal{R} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k) : (\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{S}, y_i \neq y_k\}$



# mahalanobis distance learning

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

### Formulation

$$\min_{\boldsymbol{M}\in\mathbb{S}^d_+,\boldsymbol{\xi}\geq 0} \quad (1-\mu)\sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} D^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \quad + \quad \mu\sum_{i,j,k}\xi_{ijk}$$

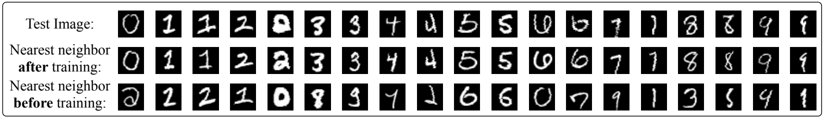
$$\text{s.t.} \quad D^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_k) - D^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 1-\xi_{ijk} \quad \forall(\boldsymbol{x}_i,\boldsymbol{x}_j,\boldsymbol{x}_k)\in\mathcal{R}$$

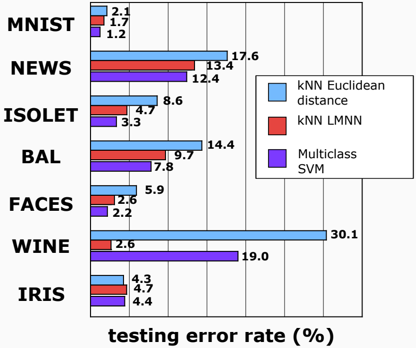
$\mu \in [0,1]$ trade-off parameter

- Number of constraints in the order of $kn^2$
    - Solver based on projected gradient descent with working set
    - Simple alternative: only consider closest "impostors"
- Chicken and egg situation: which metric to build constraints?
- Possible overfitting in high dimensions

# mahalanobis distance learning

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



# mahalanobis distance learning

## Interesting regularizers

- Add regularization term to prevent overfitting
- Simple choice: $\|\boldsymbol{M}\|_{\mathcal{F}}^2 = \sum_{i,j=1}^{d} M_{ij}^2$ (Frobenius norm)
  - Used in [Schultz and Joachims, 2003] and many others
- LogDet divergence (used in ITML [Davis et al., 2007])

$$
\begin{aligned}
D_{ld}(\boldsymbol{M}, \boldsymbol{M}_0) &= \operatorname{tr}(\boldsymbol{M}\boldsymbol{M}_0^{-1}) - \log\det(\boldsymbol{M}\boldsymbol{M}_0^{-1}) - d \\
&= \sum_{i,j} \frac{\sigma_i}{\theta_j}(\boldsymbol{v}_{\boldsymbol{i}}^T \boldsymbol{u}_i)^2 - \sum_i \log\left(\frac{\sigma_i}{\theta_i}\right) - d
\end{aligned}
$$

where $\boldsymbol{M} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^T$ and $\boldsymbol{M}_0 = \boldsymbol{U}\boldsymbol{\Theta}\boldsymbol{U}^T$ is PD

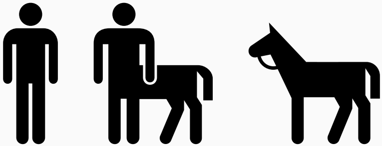
  - Remain close to good prior metric $\boldsymbol{M}_0$ (e.g., identity)
  - Implicitly ensure that $\boldsymbol{M}$ is PD
  - Convex in $\boldsymbol{M}$ (determinant of PD matrix is log-concave)
  - Efficient Bregman projections in $O(d^2)$

## Interesting regularizers

- Mixed $L_{2,1}$ norm: $\|\boldsymbol{M}\|_{2,1} = \sum_{i=1}^{d} \|\boldsymbol{M}_i\|_2$
  - Tends to zero-out entire columns $\rightarrow$ feature selection
  - Used in [Ying et al., 2009]
  - Convex but nonsmooth
  - Efficient proximal gradient algorithms (see e.g., [Bach et al., 2012])
- Trace (or nuclear) norm: $\|\boldsymbol{M}\|_* = \sum_{i=1}^{d} \sigma_i(\boldsymbol{M})$
  - Favors low-rank matrices $\rightarrow$ dimensionality reduction
  - Used in [McFee and Lanckriet, 2010]
  - Convex but nonsmooth
  - Efficient Frank-Wolfe algorithms [Jaggi, 2013]

# linear similarity learning

- Mahalanobis distance satisfies the distance axioms
  - Nonnegativity, symmetry, triangle inequality
  - Natural regularization, required by some applications
- In practice, these axioms may be violated
  - By human similarity judgments (see e.g., [Tversky and Gati, 1982])

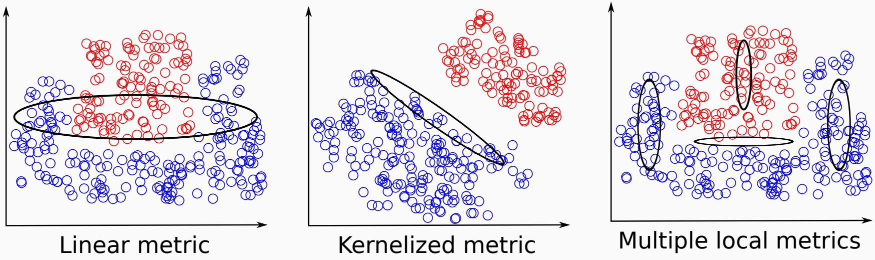


  - By some good visual recognition systems [Scheirer et al., 2014]
- Alternative: learn bilinear similarity function $S_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^T \boldsymbol{M} \boldsymbol{x}'$
  - See [Chechik et al., 2010, Bellet et al., 2012b, Cheng, 2013]
  - No PSD constraint on $\boldsymbol{M} \rightarrow$ computational benefits
  - Theory of learning with arbitrary similarity functions [Balcan and Blum, 2006]

# nonlinear extensions

# beyond linearity

- So far, we have essentially been learning a linear projection
- Advantages
  - Convex formulations
  - Robustness to overfitting
- Drawback
  - Inability to capture nonlinear structure



# kernelization of linear methods

## Definition (Kernel function)

A symmetric function $K$ is a kernel if there exists a mapping function $\phi : \mathcal{X} \rightarrow \mathbb{H}$ from the instance space $\mathcal{X}$ to a Hilbert space $\mathbb{H}$ such that $K$ can be written as an inner product in $\mathbb{H}$:

$$K(x, x') = \langle \phi(x), \phi(x') \rangle .$$

Equivalently, $K$ is a kernel if it is positive semi-definite (PSD), i.e.,

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j K(x_i, x_j) \geq 0$$

for all finite sequences of $x_1, \ldots, x_n \in \mathcal{X}$ and $c_1, \ldots, c_n \in \mathbb{R}$.

# kernelization of linear methods

## Kernel trick for metric learning

- Notations
  - Kernel $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle$, training data $\{\boldsymbol{x}_i\}_{i=1}^n$
  - $\boldsymbol{\phi}_i \overset{def}{=} \phi(\boldsymbol{x}_i) \in \mathbb{R}^D$, $\boldsymbol{\Phi} \overset{def}{=} [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_n] \in \mathbb{R}^{n \times D}$

- Mahalanobis distance in kernel space

$$D^2_{\boldsymbol{M}}(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) = (\boldsymbol{\phi}_i - \boldsymbol{\phi}_j)^T \boldsymbol{M} (\boldsymbol{\phi}_i - \boldsymbol{\phi}_j) = (\boldsymbol{\phi}_i - \boldsymbol{\phi}_j)^T \boldsymbol{L}^T \boldsymbol{L} (\boldsymbol{\phi}_i - \boldsymbol{\phi}_j)$$

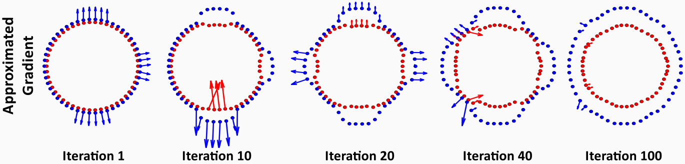
- Setting $\boldsymbol{L}^T = \boldsymbol{\Phi} \boldsymbol{U}^T$, where $\boldsymbol{U} \in \mathbb{R}^{D \times n}$, we get

$$D^2_{\boldsymbol{M}}(\phi(\boldsymbol{x}), \phi(\boldsymbol{x}')) = (\boldsymbol{k} - \boldsymbol{k}')^T \boldsymbol{M} (\boldsymbol{k} - \boldsymbol{k}')$$

  - $\boldsymbol{M} = \boldsymbol{U}^T \boldsymbol{U} \in \mathbb{R}^{n \times n}$, $\boldsymbol{k} = \boldsymbol{\Phi}^T \phi(\boldsymbol{x}) = [K(\boldsymbol{x}_1, \boldsymbol{x}), \ldots, K(\boldsymbol{x}_n, \boldsymbol{x})]^T$

- Justified by a representer theorem [Chatpatanasiri et al., 2010]

# learning a nonlinear metric



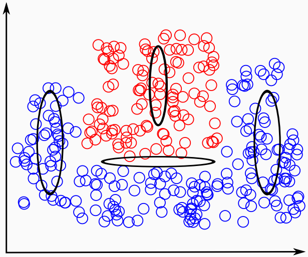

- More flexible approach: learn nonlinear mapping $\phi$ to optimize

$$D_{\phi}(\boldsymbol{x}, \boldsymbol{x}') = \|\phi(\boldsymbol{x}) - \phi(\boldsymbol{x}')\|_2$$

- Possible parameterizations for $\phi$:
  - Regression trees [Kedem et al., 2012]
  - Deep neural nets [Chopra et al., 2005, Hu et al., 2014]

# learning multiple local metrics

- Simple linear metrics perform well locally
- Idea: different metrics for different parts of the space
- Various issues
  - How to split the space?
  - How to avoid blowing up the number of parameters to learn?
  - How to make local metrics “mutually comparable”?
  - . . .



# learning multiple local metrics

## Multiple Metric LMNN [Weinberger and Saul, 2009]

- Group data into $C$ clusters
- Learn a metric for each cluster in a coupled fashion

### Formulation

$$\begin{aligned}\min_{\substack{\mathbf{M}_1,\dots,\mathbf{M}_C\\ \boldsymbol{\xi}\geq 0}} \quad & (1-\mu)\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} D^2_{\mathbf{M}_{C(\mathbf{x}_j)}}(\mathbf{x}_i,\mathbf{x}_j) \quad + \quad \mu\sum_{i,j,k}\xi_{ijk}\\ \text{s.t.} \quad & D^2_{\mathbf{M}_{C(\mathbf{x}_k)}}(\mathbf{x}_i,\mathbf{x}_k) - D^2_{\mathbf{M}_{C(\mathbf{x}_j)}}(\mathbf{x}_i,\mathbf{x}_j) \geq 1-\xi_{ijk} \quad \forall(\mathbf{x}_i,\mathbf{x}_j,\mathbf{x}_k)\in\mathcal{R}\end{aligned}$$

- Remains convex
- Computationally more expensive than standard LMNN
- Subject to overfitting
  - Many parameters

# learning multiple local metrics

Sparse Compositional Metric Learning [Shi et al., 2014]

- Learn a metric for each point in feature space
- Use the following parameterization

$$D_w^2(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} - \boldsymbol{x}')^T \left( \sum_{k=1}^{K} w_k(\boldsymbol{x}) \boldsymbol{b}_k \boldsymbol{b}_k^T \right) (\boldsymbol{x} - \boldsymbol{x}'),$$

  - $\boldsymbol{b}_k \boldsymbol{b}_k^T$: rank-1 basis (generated from training data)
  - $w_k(\boldsymbol{x}) = (\boldsymbol{a}_k^T \boldsymbol{x} + c_k)^2$: weight of basis $k$
  - $\boldsymbol{A} \in \mathbb{R}^{d \times K}$ and $\boldsymbol{c} \in \mathbb{R}^K$: parameters to learn

# learning multiple local metrics

Sparse Compositional Metric Learning [Shi et al., 2014]

## Formulation

$$\min_{\tilde{\boldsymbol{A}} \in \mathbb{R}^{(d+1)\times K}} \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k) \in \mathcal{R}} \left[1 + D_w^2(\boldsymbol{x}_i, \boldsymbol{x}_j) - D_w^2(\boldsymbol{x}_i, \boldsymbol{x}_k)\right]_+ + \lambda \|\tilde{\boldsymbol{A}}\|_{2,1}$$

- $\tilde{\boldsymbol{A}}$: stacking $\boldsymbol{A}$ and $\boldsymbol{c}$
- $[\cdot] = \max(0, \cdot)$: hinge loss

- Nonconvex problem
- Adapts to geometry of data
- More robust to overfitting
  - Limited number of parameters
  - Basis selection
  - Metric varies smoothly over feature space

# large-scale metric learning

# main challenges

- How to deal with large datasets?
  - Number of similarity judgments can grow as $O(n^2)$ or $O(n^3)$
- How to deal with high-dimensional data?
  - Cannot store $d \times d$ matrix
  - Cannot afford computational complexity in $O(d^2)$ or $O(d^3)$

# case of large $n$

## Online learning

### OASIS [Chechik et al., 2010]

- Set $\boldsymbol{M}^0 = \boldsymbol{I}$
- At step $t$, receive $(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k) \in \mathcal{R}$ and update by solving

$$
\begin{aligned}
\boldsymbol{M}^t = \quad \arg\min_{\boldsymbol{M},\xi} \quad & \frac{1}{2}\|\boldsymbol{M} - \boldsymbol{M}^{t-1}\|_{\mathcal{F}}^2 + C\xi \\
\text{s.t.} \quad & 1 - S_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + S_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k) \leq \xi \\
& \xi \geq 0
\end{aligned}
$$

  - $S_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^T \boldsymbol{M} \boldsymbol{x}'$, $C$ trade-off parameter

- Closed-form solution at each iteration
- Trained with 160M triplets in 3 days on 1 CPU

# case of large $n$

## Stochastic and distributed optimization

- Assume metric learning problem of the form

$$\min_{\boldsymbol{M}} \quad \frac{1}{|\mathcal{R}|} \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k) \in \mathcal{R}} \ell(\boldsymbol{M}, \boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k)$$

- Can use Stochastic Gradient Descent
    - Use a random sample (mini-batch) to estimate gradient
    - Better than full gradient descent when $n$ is large
- Can be combined with distributed optimization
    - Distribute triplets on workers
    - Each worker use a mini-batch to estimate gradient
    - Coordinator averages estimates and updates

# case of large $d$

Simple workarounds

- Learn a diagonal matrix
  - Used in [Xing et al., 2002, Schultz and Joachims, 2003]
  - Learn $d$ parameters
  - Only a weighting of features...
- Learn metric after dimensionality reduction (e.g., PCA)
  - Used in many papers
  - Potential loss of information
  - Learned metric difficult to interpret

# case of large $d$

## Matrix decompositions

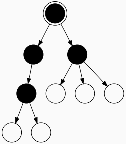
- Low-rank decomposition $\boldsymbol{M} = \boldsymbol{L}^T \boldsymbol{L}$ with $\boldsymbol{L} \in \mathbb{R}^{r \times d}$
    - Used in [Goldberger et al., 2004]
    - Learn $r \times d$ parameters
    - Generally nonconvex, must tune $r$
- Rank-1 decomposition $\boldsymbol{M} = \sum_{i=1}^{K} w_k \boldsymbol{b}_k \boldsymbol{b}_k^T$
    - Used in SCML [Shi et al., 2014]
    - Learn $K$ parameters
    - Hard to generate good bases in high dimensions

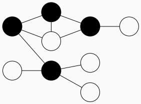
# metric learning for structured data

# motivation

- Each data instance is a structured object
  - Strings: words, DNA sequences
  - Trees: XML documents
  - Graphs: social network, molecules

ACGGCTT





- Metrics on structured data are convenient
  - Act as proxy to manipulate complex objects
  - Can use any metric-based algorithm

# motivation

- Could represent each object by a feature vector
    - Idea behind many kernels for structured data
    - Could then apply standard metric learning techniques
    - Potential loss of structural information
- Instead, focus on edit distances
    - Directly operate on structured object
    - Variants for strings, trees, graphs
    - Natural parameterization by cost matrix

# string edit distance

- Notations
  - Alphabet $\Sigma$: finite set of symbols
  - String x: finite sequence of symbols from $\Sigma$
  - $|x|$: length of string x
  - \$: empty string / symbol

**Definition (Levenshtein distance)**

The Levenshtein string edit distance between x and $x'$ is the length of the shortest sequence of operations (called an *edit script*) turning x into $x'$. Possible operations are insertion, deletion and substitution of symbols.

- Computed in $O(|x| \cdot |x'|)$ time by Dynamic Programming (DP)

# string edit distance

## Parameterized version

- Use a nonnegative $(|\Sigma| + 1) \times (|\Sigma| + 1)$ matrix $\boldsymbol{C}$
  - $C_{ij}$: cost of substituting symbol $i$ with symbol $j$

### Example 1: Levenshtein distance

| **C** | \$ | a | b |
|---|---|---|---|
| \$ | 0 | 1 | 1 |
| a | 1 | 0 | 1 |
| b | 1 | 1 | 0 |

$\Longrightarrow$ edit distance between `abb` and `aa` is 2 (needs at least two operations)

### Example 2: specific costs

| **C** | \$ | a | b |
|---|---|---|---|
| \$ | 0 | 2 | 10 |
| a | 2 | 0 | 4 |
| b | 10 | 4 | 0 |

$\Longrightarrow$ edit distance between `abb` and `aa` is 10 ($\texttt{a} \to \$,\ \texttt{b} \to \texttt{a},\ \texttt{b} \to \texttt{a}$)

# large-margin edit distance learning

## GESL [Bellet et al., 2012a]

- Inspired from successful algorithms for non-structured data
    - Large-margin constraints
    - Convex optimization
- Requires key simplification: fix the edit script

$$e_{\mathbf{C}}(\mathrm{x},\mathrm{x}') = \sum_{u,v \in \Sigma \cup \{\$\}} \mathbf{C}_{uv} \cdot \#_{uv}(\mathrm{x},\mathrm{x}')$$

- $\#_{uv}(\mathrm{x},\mathrm{x}')$: nb of times $u \to v$ appears in Levenshtein script
- $e_{\mathbf{C}}$ is a linear function of the costs

# large-margin edit distance learning

## GESL [Bellet et al., 2012a]

### Formulation

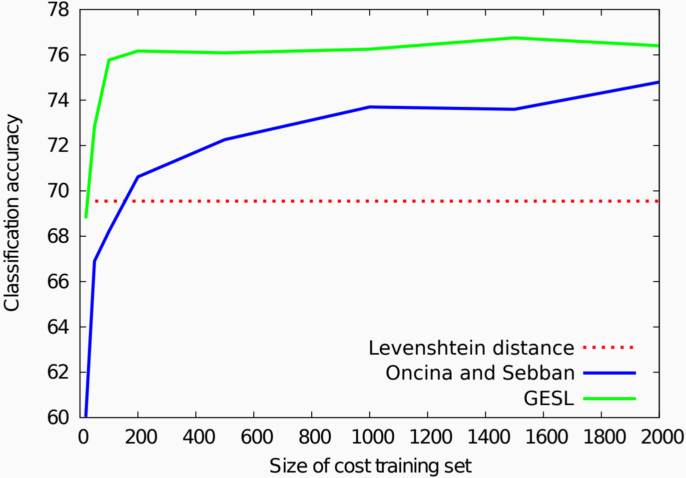
$$
\begin{aligned}
\min_{\boldsymbol{C}\geq 0, \boldsymbol{\xi}\geq 0, B_1\geq 0, B_2\geq 0} \quad & \sum_{i,j} \xi_{ij} \quad + \quad \lambda \|\boldsymbol{C}\|_{\mathcal{F}}^2 \\
\text{s.t.} \quad & e_{\boldsymbol{C}}(\mathrm{x}, \mathrm{x}') \geq B_1 - \xi_{ij} \qquad \forall (\mathrm{x}_i, \mathrm{x}_j) \in \mathcal{D} \\
& e_{\boldsymbol{C}}(\mathrm{x}, \mathrm{x}') \leq B_2 + \xi_{ij} \qquad \forall (\mathrm{x}_i, \mathrm{x}_j) \in \mathcal{S} \\
& B_1 - B_2 = \gamma
\end{aligned}
$$

$\gamma$ margin parameter

- Convex, less costly and use of negative pairs
- Straightforward adaptation to trees and graphs
- Less general than proper edit distance
  - Chicken and egg situation similar to LMNN

# large-margin edit distance learning

Application to word classification [Bellet et al., 2012a]



# generalization guarantees

# statistical view of supervised metric learning

- Training data $T_n = \{\boldsymbol{z}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$
  - $\boldsymbol{z}_i \in \mathcal{Z} = \mathcal{X} \times \mathcal{Y}$
  - $\mathcal{Y}$ discrete label set
  - independent draws from unknown distribution $\mu$ over $\mathcal{Z}$
- Minimize the regularized empirical risk

$$R_n(\boldsymbol{M}) = \frac{2}{n(n-1)} \sum_{1 \leq i < j \leq n}^{n} \ell(\boldsymbol{M}, \boldsymbol{z}_i, \boldsymbol{z}_j) \quad + \quad \lambda reg(\boldsymbol{M})$$

- Hope to achieve small expected risk

$$R(\boldsymbol{M}) = \mathop{\mathbb{E}}_{\boldsymbol{z}, \boldsymbol{z}' \sim \mu} [\ell(\boldsymbol{M}, \boldsymbol{z}, \boldsymbol{z}')]$$

- Note: this can be adapted to triplets

# statistical view of supervised metric learning

- Standard statistical learning theory: sum of i.i.d. terms
- Here $R_n(\boldsymbol{M})$ is a sum of dependent terms!
  - Each training point involved in several pairs
  - Corresponds to practical situation
- Need specific tools to go around this problem
  - Uniform stability
  - Algorithmic robustness

# uniform stability

## Definition ([Jin et al., 2009])

A metric learning algorithm has a uniform stability in $\kappa/n$, where $\kappa$ is a positive constant, if

$$\forall (T_n, \boldsymbol{z}), \forall i, \quad \sup_{z_1, z_2} |\ell(\boldsymbol{M}_{T_n}, \boldsymbol{z}_1, \boldsymbol{z}_2) - \ell(\boldsymbol{M}_{T_n^{i,z}}, \boldsymbol{z}_1, \boldsymbol{z}_2)| \leq \frac{\kappa}{n}$$

- $\boldsymbol{M}_{T_n}$: metric learned from $T_n$
- $T_n^{i,z}$: set obtained by replacing $\boldsymbol{z}_i \in T_n$ by $\boldsymbol{z}$

- If $reg(\boldsymbol{M}) = \|\boldsymbol{M}\|_{\mathcal{F}}^2$, under mild conditions on $\ell$, algorithm has uniform stability [Jin et al., 2009]
  - Applies for instance to GESL [Bellet et al., 2012a]
- Does not apply to other (sparse) regularizers

# uniform stability

## Generalization bound

**Theorem ([Jin et al., 2009])**

*For any metric learning algorithm with uniform stability $\kappa/n$, with probability $1-\delta$ over the random sample $T_n$, we have:*

$$R(\boldsymbol{M}_{T_n}) \leq R_n(\boldsymbol{M}_{T_n}) + \frac{2\kappa}{n} + (2\kappa + B)\sqrt{\frac{\ln(2/\delta)}{2n}}$$

*$B$ problem-dependent constant*

- Standard bound in $O(1/\sqrt{n})$

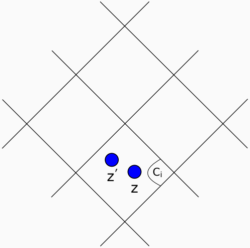
# algorithmic robustness

## Definition ([Bellet and Habrard, 2015])

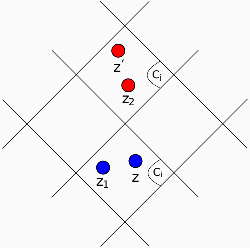
A metric learning algorithm is $(K, \epsilon(\cdot))$ robust for $K \in \mathbb{N}$ and $\epsilon : (\mathcal{Z} \times \mathcal{Z})^n \to \mathbb{R}$ if $\mathcal{Z}$ can be partitioned into $K$ disjoints sets, denoted by $\{C_i\}_{i=1}^{K}$, such that the following holds for all $T_n$:

$$\forall (\boldsymbol{z}_1, \boldsymbol{z}_2) \in T_n^2, \forall \boldsymbol{z}, \boldsymbol{z}' \in \mathcal{Z}, \forall i, j \in [K], \text{ if } \boldsymbol{z}_1, \boldsymbol{z} \in C_i, \boldsymbol{z}_2, \boldsymbol{z}' \in C_j$$

$$|\ell(\boldsymbol{M}_{T_n}, \boldsymbol{z}_1, \boldsymbol{z}_2) - \ell(\boldsymbol{M}_{T_n}, \boldsymbol{z}, \boldsymbol{z}')| \leq \epsilon(T_n^2)$$




Classic robustness




Robustness for metric learning

# algorithmic robustness

## Generalization bound

**Theorem ([Bellet and Habrard, 2015])**

*If a metric learning algorithm is $(K, \epsilon(\cdot))$-robust, then for any $\delta > 0$, with probability at least $1 - \delta$ we have:*

$$R(\boldsymbol{M}_{T_n}) \leq R_n(\boldsymbol{M}_{T_n}) + \epsilon(T_n^2) + 2B\sqrt{\frac{2K \ln 2 + 2\ln(1/\delta)}{n}}$$

- Wide applicability
  - Mild assumptions on $\ell$
  - Any norm regularizer: Frobenius, $L_{2,1}$, trace...
- Bounds are loose
  - $\epsilon(T_n^2)$ can be as small as needed by increasing $K$
  - But $K$ potentially very large and hard to estimate

# 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.

[Balcan and Blum, 2006] Balcan, M.-F. and Blum, A. (2006).
On a Theory of Learning with Similarity Functions.
In *ICML*, pages 73–80.

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

[Bellet et al., 2012a] Bellet, A., Habrard, A., and Sebban, M. (2012a).
Good edit similarity learning by loss minimization.
*Machine Learning Journal*, 89(1):5–35.

[Bellet et al., 2012b] Bellet, A., Habrard, A., and Sebban, M. (2012b).
Similarity Learning for Provably Accurate Sparse Linear Classification.
In *ICML*, pages 1871–1878.

[Bernard et al., 2008] Bernard, M., Boyer, L., Habrard, A., and Sebban, M. (2008).
Learning probabilistic models of tree edit distance.
*Pattern Recognition*, 41(8):2611–2629.

# references II


[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.

[Cheng, 2013] Cheng, L. (2013).
Riemannian Similarity Learning.
In *ICML*.

[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.

# references III


[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.

[Geng et al., 2011] Geng, B., Tao, D., and Xu, C. (2011).
DAML: Domain Adaptation Metric Learning.
*IEEE Transactions on Image Processing*, 20(10):2980–2989.

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

[Goldstone et al., 1997] Goldstone, R. L., Medin, D. L., and Halberstadt, J. (1997).
Similarity in context.
*Memory & Cognition*, 25(2):237–255.

[Hoi et al., 2008] Hoi, S. C., Liu, W., and Chang, S.-F. (2008).
Semi-supervised distance metric learning for Collaborative Image Retrieval.
In *CVPR*.

[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.

# references IV


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

[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.

[Kulis et al., 2011] Kulis, B., Saenko, K., and Darrell, T. (2011).
What you saw is not what you get: Domain adaptation using asymmetric kernel transforms.
In *CVPR*, pages 1785–1792.

[Lim and Lanckriet, 2014] Lim, D. and Lanckriet, G. R. (2014).
Efficient Learning of Mahalanobis Metrics for Ranking.
In *ICML*, pages 1980–1988.

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

# references V


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

[Nosofsky, 1986] Nosofsky, R. M. (1986).
Attention, similarity, and the identificationcategorization relationship.
*Journal of Experimental Psychology: General*, 115(1):39–57.

[Oncina and Sebban, 2006] Oncina, J. and Sebban, M. (2006).
Learning Stochastic Edit Distance: application in handwritten character recognition.
*Pattern Recognition*, 39(9):1575–1587.

[Parameswaran and Weinberger, 2010] Parameswaran, S. and Weinberger, K. Q. (2010).
Large Margin Multi-Task Metric Learning.
In *NIPS*, pages 1867–1875.

[Scheirer et al., 2014] Scheirer, W. J., Wilber, M. J., Eckmann, M., and Boult, T. E. (2014).
Good recognition is non-metric.
*Pattern Recognition*, 47(8):2721–2731.

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

# references VI


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

[Tversky, 1977] Tversky, A. (1977).
Features of similarity.
*Psychological Review*, 84(4):327–352.

[Tversky and Gati, 1982] Tversky, A. and Gati, I. (1982).
Similarity, separability, and the triangle inequality.
*Psychological Review*, 89(2):123–154.

[van der Maaten and Hinton, 2008] van der Maaten, L. and Hinton, G. (2008).
Visualizing Data using t-SNE.
*Journal of Machine Learning Research*, 9:2579–2605.

[Weinberger et al., 2005] Weinberger, K. Q., Blitzer, J., and Saul, L. K. (2005).
Distance Metric Learning for Large Margin Nearest Neighbor Classification.
In *NIPS*, pages 1473–1480.

[Weinberger and Saul, 2009] Weinberger, K. Q. and Saul, L. K. (2009).
Distance Metric Learning for Large Margin Nearest Neighbor Classification.
*Journal of Machine Learning Research*, 10:207–244.

# references VII


[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.

[Zhang and Yeung, 2010] Zhang, Y. and Yeung, D.-Y. (2010).
Transfer metric learning by learning task relationships.
In *KDD*, pages 1199–1208.