Metric Learning

Tuesday, February 4, 2014
12:04 PM

Reference

This presentation is based on the following survey. The quotes, images, and many of the equations are copied from this survey.

Foundations and Trends® in
Machine Learning
Vol. 5, No. 4 (2012) 287–364
© 2013 B. Kulis
DOI: 10.1561/2200000019

now
the essence of knowledge

Metric Learning: A Survey

By Brian Kulis

What is metric learning all about?

"The metric learning problem is concerned with learning a distance function tuned to a particular task, and has been shown to be useful when used in conjunction with nearest-neighbor methods and other techniques that rely on distances or similarities."

Motivating Example

Fig. 1.1 Example face data set. In one application, our notion of "distance" between faces may depend on the pose, whereas in another application it may depend on the identity.

How would we want to define distances between images if our goal was to:
A) Find matching faces based on identity?
B) Determine the pose of the individual?

Ans:

"if our goal is to find matching faces based on identity, then we should choose a distance function that emphasizes appropriate features (hair color, ratios of distances between facial keypoints, etc). But we may also have an application where we want to determine the pose of an individual, and therefore require a distance function that captures pose similarity. Clearly other features are more applicable in this scenario."

Such metrics can be hand-crafted, however goal of metric learning is to learn them automatically in a supervised manner

Problem Formulation

*"Given an input distance function d(**x**,**y**) between objects **x** and **y** (for example, the Euclidean distance), along with supervised information regarding an ideal distance, construct a new distance function d'(**x**,**y**) which is "better" than the original distance function"*

Applications of Metric Learning

Within another model - i.e. Kernel Regression. Optimize the learned metric to get the best regression results.

Supervised Dimensionality Reduction

Use $tr(A)$ as a regularizer (like an L1 norm)

Data Visualization

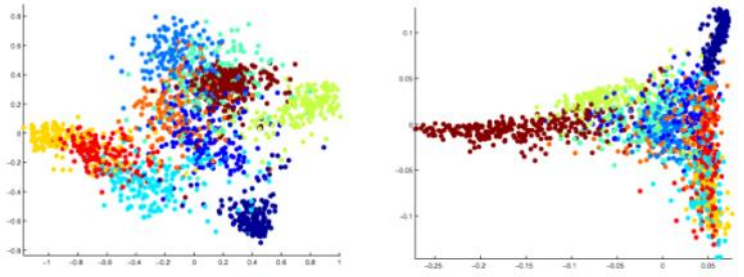

Fig. 4.1 Example taken from [36] demonstrating supervised nonlinear dimensionality reduction using kernelized Mahalanobis metric learning. The right plot shows the results of projecting to two dimensions via kernel PCA on the MNIST digits, while the left plot shows results when projecting using a learned rank-2 projection obtained by a trace-norm regularized kernelized metric learning problem.

Computer Vision

- "Computer vision has seen many applications of metric learning, and is arguably **the most successful domain for metric learning**. Indeed, a

large fraction of research in metric learning has been published in the computer vision community."
- "the goal is to retrieve “similar” images to a query; metric learning is then used to refine an appropriate notion of similarity to aid in this task"

Specific Applications include:
- Face Recognition
- Human activity recognition and pose estimation

Natural Language Processing (NLP)
- Retrieving similar text documents

Other Applications
- Similarity of Music
- Automated Program Debugging
- Analysis of Gene Expression Data

Problem Formulation

*"Given an input distance function d(**x**,**y**) between objects **x** and **y** (for example, the Euclidean distance), along with supervised information regarding an ideal distance, construct a new distance function d'(**x**,**y**) which is “better” than the original distance function"*

Types of distance metrics to consider

Formula	Description	Degree of Restriction	Comment
$\|Gx - Gy\|_2$	Learn a linear transformation of the data, then find the distance between the transformed data points.	Most restrictive. Unable to satisfy some desirata	Most Popular
$d(f(x), f(y))$	Learn an arbitrary transformation of the data, then find the distance between the transformed data points.	Generality depends on the parameterization of f(**x**)	Could use kernels or neural networks to generate these transformations
*d'(**x**,**y**)*	construct a new distance function d'(**x**,**y**)	Totally unrestricted. Distance function properties not automatically satisfied.	Not covered in this presentation.

Linear Models

Linear metrics - The Mahalanobis Distance

- $d_A(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} - \boldsymbol{y})^T A(\boldsymbol{x} - \boldsymbol{y}),$ where A is some positive semi-definite matrix (i.e., its eigenvalues are non-negative).
- Factoring $A = G^T G$ we get:
  $$d_A(\boldsymbol{x}, \boldsymbol{y}) = \|G\boldsymbol{x} - G\boldsymbol{y}\|_2^2.$$
  ? *Should I derive this?*
- It will be easier to optimize for A than for G.

Desiderata for the New Distance Metrics, and their Cost Functions

There are many possible desirata that new distance metrics could be optimized to satisfy

★ Here are two of the most popular ones:

Desirata	Possible Cost Functions
Some points are close together, others are far: $d_A(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq u \quad (i,j) \in \mathcal{S}$ $d_A(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq \ell \quad (i,j) \in \mathcal{D}.$	Hinge Loss: $\max(0, d_A(\boldsymbol{x}_i, \boldsymbol{x}_j) - u), \quad (i,j) \in \mathcal{S}$ $\max(0, \ell - d_A(\boldsymbol{x}_i, \boldsymbol{x}_j)), \quad (i,j) \in \mathcal{D}.$ Squared Hinge Loss: $(\max(0, d_A(\boldsymbol{x}_i, \boldsymbol{x}_j) - u))^2, \quad (i,j) \in \mathcal{S}$ $(\max(0, \ell - d_A(\boldsymbol{x}_i, \boldsymbol{x}_j)))^2, \quad (i,j) \in \mathcal{D}.$
Relative distance constraints - some points are closer to each other than others. $d_A(\boldsymbol{x}_i, \boldsymbol{x}_j) < d_A(\boldsymbol{x}_i, \boldsymbol{x}_k).$ Adding a margin: $d_A(\boldsymbol{x}_i, \boldsymbol{x}_j) < d_A(\boldsymbol{x}_i, \boldsymbol{x}_k) - m,$	Hinge Loss or Squared Hinge Loss

? Notation: Why can each of these cost functions be expressed as $c(X^T AX)$?

$d_A(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} - \boldsymbol{y})^T A(\boldsymbol{x} - \boldsymbol{y}),$

This is a linear combination of elements $x_i^T A x_i$ of $X^T AX$. Hence the above cost functions can be expressed as a function c of $X^T AX$.

Optimizing for Mahalanobis Distance Matrix A

The Problem:

$$\mathcal{L}(A) = r(A) + \lambda \sum_{i=1}^{m} c_i(X^T A X)$$

Possible regularizers r(A):

$r(A) = \frac{1}{2}\|A\|_F^2$	Like L_2 regularization. Easy to analyze and has strong convexity. Equivalent to: $\frac{1}{2}tr(A^T A) = \frac{1}{2}tr(A^2) = \frac{1}{2}\sum_i \lambda_i^2$, since the trace equals the sum of the eigenvalues.
$r(A) = \mathrm{tr}(AC)$	Like L_1 regularization. Tends to lead to matrices A of low rank.
$r(A) = \mathrm{tr}(A) - \log\det(A).$	"information-theoretic metric learning"

Non-Linear Models

Why non-linear models - the XOR example

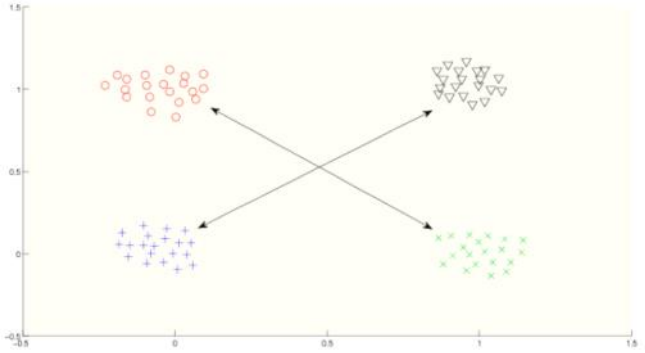

Fig. 3.1 Example of the limitation of linear methods. Suppose in this data we would like that the black and blue points should be "similar" to one another, while the red and green point should also be similar to one another (while simultaneously the black and blue points should be dissimilar to the red and green points). No global linear metric will suffice for enforcing the constraints.

Types of non-linear models

Kernel models

$$\min_{A \succeq 0} r(A) + \lambda \sum_{i=1}^{m} c_i(X^T A X),$$

- It is possible to have updates to A during optimization depend only on inner products of x_i,
- These inner products can then be replaced by Kernels.
  *I won't go into the details of how this is done.

Neural Networks

Some other methods