Distance metric learning with application to clustering with side-information

Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, Stuart Russell

University of California, Berkeley

Advances in Neural Information, 2002
Presented by Tuan (Johnny) Ta

Introduction

- The performance of many learning and datamining algorithms depend critically on their being given a good metric over the input space.
- Often there is no “right” answer for clustering

The Problem

- Suppose a user indicates that certain points in an input space (say $\mathbb{R}^n$) are considered to be similar, can we automatically learn a distance metric over the input space that respects these relationships?
  I.e. If 2 points are similar, then the distance between them is small.

Comparison to Literature

- One important family of algorithms that (implicitly) learn metrics are the unsupervised ones that take an input dataset, and find an embedding of it in some space. E.g. Multidimensional Scaling (MDS), Principal Components Analysis (PCA)
- However, these algorithms still suffer from the "no right answer" problem.
- This paper aims to learn a *full metric* $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, instead of focusing only on (finding an embedding for) the points on the training set.

Outline

1. Learning Distance Metrics
    - Diagonal A
    - Full A
2. Experiments and Examples
    - Examples
    - Application To Clustering

Distance Metrics

- Suppose we have a set of points $\{x_i\}_{i=1}^n \subseteq \mathbb{R}^n$, and are given information that certain pairs of them are similar

$$S : (x_i, x_j) \in S \qquad \text{if } x_i \text{ and } x_j \text{ are similar}$$

- We want to learn a distance metric $d(x, y)$ that respects this. Consider the distance metric of the form

$$d(x, y) = d_A(x, y) = ||x - y||_A = \sqrt{(x - y)^T A (x - y)}$$

- A has to be positive semi-definite $A \succeq 0$

Interpretation

- $A = I$: Euclidean distance
- A diagonal: a metric in which different axes are given different weights
- General: A parameterizes a family of Mahalanobis distances over $\mathbb{R}^n$
- Learning such a distance metric is equivalent to finding a rescaling of the data that replace x with $A^{1/2}x$ and apply the standard Euclidean metric to the rescaled data.

Learning Distance Metrics As A Convex Optimization Problem

$$\begin{aligned} \min_A & \sum_{(x_i,x_j)\in S} ||x_i - x_j||_A^2 \\ \text{s.t.} & \sum_{(x_i,x_j)\in D} ||x_i - x_j||_A \geq 1, \\ & A \succeq 0 \end{aligned}$$

- D is the set of dissimilar points (taken to be the rest of the points if only S is given)
- The constraint ensures that A doesn't collapse the dataset into a single point ($A = 0$)

Outline

1 Learning Distance Metrics
- Diagonal A
- Full A

2 Experiments and Examples
- Examples
- Application To Clustering

Diagonal A

- It can be shown that the problem we have is equivalent to minimizing

$$g(A) = g(A_{11}, \dots, A_{nn}) = \sum_{(x_i,x_j)\in S} ||x_i - x_j||_A^2 - \log \left(\sum_{(x_i,x_j)\in D} ||x_i - x_j||_A \right)$$

  Subject to $A \succeq 0$

- This is solved using Newton-Raphson method.

Outline

1. Learning Distance Metrics
   - Diagonal A
   - Full A
2. Experiments and Examples
   - Examples
   - Application To Clustering

Full A

- The condition $A \succeq 0$ becomes harder to enforce, and Newton's method becomes too expensive.
- Use gradient descent and iterative projection instead
- Consider the equivalent problem

$$\begin{aligned}
\max_{A} \quad & g(A) = \sum_{(x_i,x_j)\in D} ||x_i - x_j||_A \\
\text{s.t.} \quad & f(A) = \sum_{(x_i,x_j)\in S} ||x_i - x_j||_A \leq 1, \\
& A \succeq 0
\end{aligned}$$

Gradient Descent + Iterative Projection Algorithm

```
Iterate
    Iterate
        $A := \text{argmin}_{A'}\{||A' - A||_F : A' \in C_1\}$
        $A := \text{argmin}_{A'}\{||A' - A||_F : A' \in C_2\}$
    until $A$ converges
    $A := A + \alpha(\nabla_A g(A))_{\perp \nabla_A f}$
until convergence
```

- $||M||_F = (\sum_i \sum_j M_{ij}^2)^{1/2}$ (Frobenius norm)
- $C_1 = \{A : \sum_{(x_i, x_j) \in S} ||x_i - x_j||_A^2 \leq 1\}$
- $C_2 = \{A : A \succeq 0\}$

Outline

1 Learning Distance Metrics
- Diagonal A
- Full A

2 Experiments and Examples
- Examples
- Application To Clustering

2-class Data

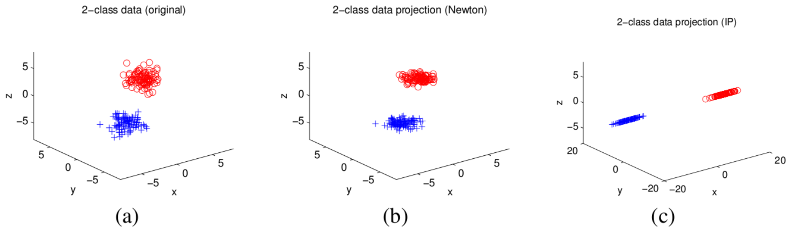


Figure 2: (a) Original data, with the different classes indicated by the different symbols (and colors, where available). (b) Rescaling of data corresponding to learned diagonal A. (c) Rescaling corresponding to full A.

3-class Data

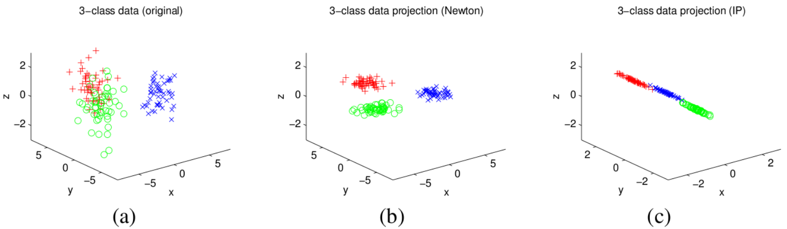


Figure 3: (a) Original data. (b) Rescaling corresponding to learned diagonal A. (c) Rescaling corresponding to full A.

Outline

1 Learning Distance Metrics
- Diagonal A
- Full A

2 Experiments and Examples
- Examples
- Application To Clustering

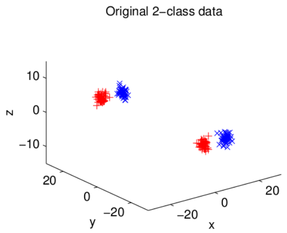
Clustering With Side-Information

- Algorithms
  1. Original K-means: using Euclidean distance, not using S
  2. Constrained K-means: K-means but subject to points $(x_i, x_j) \in S$ always being assigned to the same cluster
  3. K-means + metrics: using $||x_i - \mu_k||_A^2$ learned from S
  4. Constrained K-means + metrics
- Performance criteria

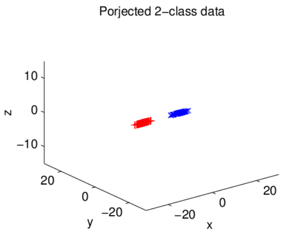
$$\text{Accuracy} = \sum_{i>j} \frac{1\{1\{c_i = c_j\} = 1\{\hat{c}_i = \hat{c}_j\}\}}{0.5m(m-1)}$$

$\hat{c}_i$ $(i = 1, \ldots, m)$: the cluster point x_i is assigned to
c_i: desired cluster

Clustering Example 1



(a)

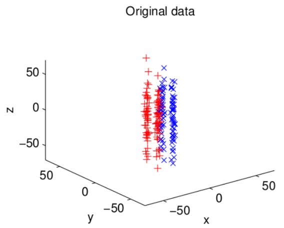


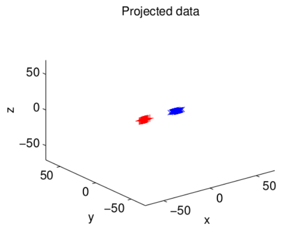
(b)

1. K-means: Accuracy = 0.4975
2. Constrained K-means: Accuracy = 0.5060
3. K-means + metric: Accuracy = 1
4. Constrained K-means + metric: Accuracy = 1

Figure 4: (a) Original dataset (b) Data scaled according to learned metric. (A_{diagonal}'s result is shown, but A_{full} gave visually indistinguishable results.)

Clustering Example 2





(a) (b)

1. K-means: Accuracy = 0.4993
2. Constrained K-means: Accuracy = 0.5701
3. K-means + metric: Accuracy = 1
4. Constrained K-means + metric: Accuracy = 1

Figure 5: (a) Original dataset (b) Data scaled according to learned metric. (A_{diagonal}'s result is shown, but A_{full} gave visually indistinguishable results.)

Clustering On Real Dataset

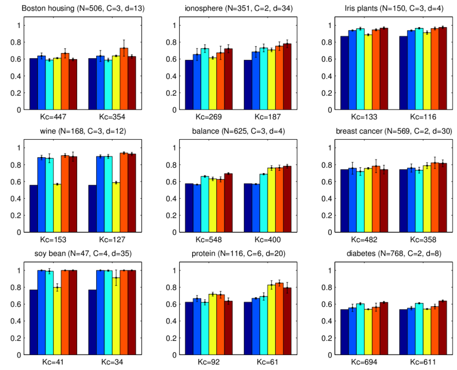
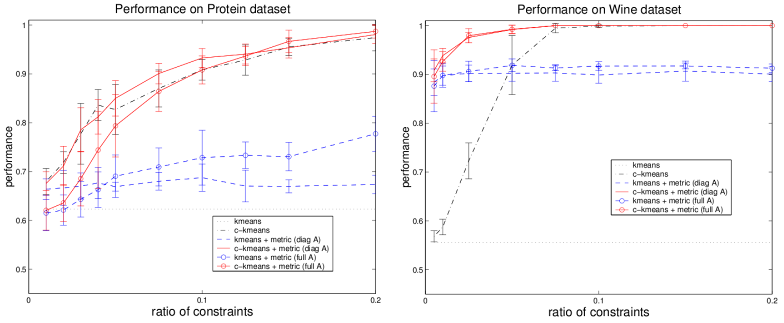


Figure 6: Clustering accuracy on 9 UCI datasets. In each panel, the six bars on the left correspond to an experiment with "little" side-information $\mathcal{S}$, and the six on the right to "much" side-information. From left to right, the six bars in each set are respectively K-means, K-means + diagonal metric, K-means + full metric, Constrained K-means (C-Kmeans), C-Kmeans + diagonal metric, and C-Kmeans + full metric. Also shown are N: size of dataset; C: number of classes/clusters; d: dimensionality of data; K_c: mean number of connected components (see footnotes 7, 9). 1 s.e. bars are also shown.

Accuracy vs. Amount of Side-Information



(a) (b)

Figure 7: Plots of accuracy vs. amount of side-information. Here, the x-axis gives the fraction of all pairs of points in the same class that are randomly sampled to be included in $\mathcal{S}$.

Summary

- Algorithm: Given examples of similar pairs in $\mathbb{R}^n$, learns a distance metric that respects these relationships
- Posing metric learning as a convex optimization problem

- Extension: Learning non-linear metrics

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