ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/347523278

Lecture notes in dimensionality reduction for unsupervised metric learning: LPP

Preprint · December 2020

DOI: 10.13140/RG.2.2.27051.46885

CITATIONS
0

READS
22

1 author:

Alexandre L M Levada
Universidade Federal de São Carlos
113 PUBLICATIONS **331** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Double Noise Filtering in CT: Pre- and Post-Reconstruction View project

Unsupervised metric learning View project

All content following this page was uploaded by Alexandre L M Levada on 21 December 2020.

The user has requested enhancement of the downloaded file.

Lecture notes in dimensionality reduction for unsupervised metric learning

Alexandre L. M. Levada[a)]
Computing Department, Federal University of São Carlos, SP, Brazil

(Dated: December 17, 2020)

"The greatest challenge to any thinker is stating the problem in a way that will allow a solution." (Bertrand Russell)

LECTURE 6: LOCALITY PRESERVING PROJECTIONS AND KERNEL LPP

One problem with nonlinear dimensionality reduction techniques like ISOMAP, LLE and Laplacian eigenmaps is that these methods are defined only on the training data points and it is unclear how to evaluate the map for new test points. The main motivation for the Locality Preserving Projection (LPP) algorithm is to produce a method that can be simply applied to any new test data point in order to locate it in the reduced representation space[1]. The basic idea of LPP is to provide a linear approximation of the nonlinear Laplacian Eigenmaps method.

As in the Laplacian Eigenmaps method, we seek a smooth map that preserves locality, that is, proximity in the graph must imply in proximity in the line. We have shown in previous sections that if we minimize the following criterion, then the map $\vec{y} = [y_1, y_2, ..., y_n]$ is optimal in that sense:

$$\vec{y}^T L \vec{y} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (y_i - y_j)^2 \tag{1}$$

where L is the Laplacian matrix of the KNN graph induced by the $m \times n$ data matrix $X = [\vec{x}_1, \vec{x}_2, ..., \vec{x}_n]$.

In LPP, it is assumed that the relationship between $\vec{x}_i \in R^m$ and $y_i \in R$ is linear, that is, $y_i = \vec{a}^T \vec{x}_i$, where $\vec{a} \in R^m$ is a column vector. Hence, the objective function can be expressed as:

$$\begin{aligned}
\vec{y}^T L \vec{y} &= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (\vec{a}^T \vec{x}_i - \vec{a}^T \vec{x}_j)^2 \\
&= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left[\vec{a}^T \vec{x}_i \vec{x}_i^T \vec{a} - 2 \vec{a}^T \vec{x}_i \vec{x}_j^T \vec{a} + \vec{a}^T \vec{x}_j \vec{x}_j^T \vec{a} \right] \\
&= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} 2 w_{ij} \vec{a}^T \vec{x}_i \vec{x}_i^T \vec{a} - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} 2 w_{ij} \vec{a}^T \vec{x}_i \vec{x}_j^T \vec{a} \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \vec{a}^T \vec{x}_i \vec{x}_i^T \vec{a} - \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \vec{a}^T \vec{x}_i \vec{x}_j^T \vec{a}
\end{aligned} \tag{2}$$

Since $d_i = \sum_{j=1}^{n} w_{ij}$, we have:

[a)] Electronic mail: alexandre.levada@dufscar.br

$$\vec{y}^T L \vec{y} = \sum_{i=1}^{n} \vec{a}^T \vec{x}_i d_i \vec{x}_i^T \vec{a} - \sum_{i=1}^{n}\sum_{j=1}^{n} \vec{a}^T \vec{x}_i w_{ij} \vec{x}_j^T \vec{a} \tag{3}$$

Note that we can rewrite the equation using a matrix-vector notation as:

$$\vec{y}^T L \vec{y} = \vec{a}^T X D X^T \vec{a} - \vec{a}^T X W X^T \vec{a} \tag{4}$$

where X is the $m \times n$ data matrix, D is the $n \times n$ diagonal matrix of the degrees and W is the $n \times n$ weight matrix. Knowing that $L = D - W$, we finally reach:

$$\vec{y}^T L \vec{y} = \vec{a}^T X (D - W) X^T \vec{a} = \vec{a}^T X L X^T \vec{a} \tag{5}$$

Thus, we have to solve the following constrained minimization problem:

$$\arg\min_{\vec{a}} \ \vec{a}^T X L X^T \vec{a} \qquad \text{subject to} \qquad \vec{a}^T X D X^T \vec{a} = 1 \tag{6}$$

where the constraint is a general form to express that the norm of the vector $\vec{a}$ is a constant. The Lagrangian function is given by:

$$L(\vec{a}, \lambda) = \vec{a}^T X L X^T \vec{a} - \lambda(\vec{a}^T X D X^T \vec{a} - 1) \tag{7}$$

Taking the derivative with respect to $\vec{a}$ and setting the result to zero leads to:

$$\frac{\partial}{\partial \vec{a}} L(\vec{a}, \lambda) = X L X^T \vec{a} - \lambda X D X^T \vec{a} = 0 \tag{8}$$

Therefore, we have a generalized eigenvector problem:

$$X L X^T \vec{a} = \lambda X D X^T \vec{a} \tag{9}$$

$$(X D X^T)^{-1} X L X^T \vec{a} = \lambda \vec{a} \tag{10}$$

showing that in order to minimize the objective function, we shoud select the vector a as the smallest eigenvector of the matrix $(XDX^T)^{-1}XLX^T$. The multivariate version of the problem considers a $m \times d$ matrix A where each column $\vec{a}_j$ represents a direction in which data will be projected:

$$(X D X^T)^{-1} (X L X^T) A = \lambda A \tag{11}$$

In this case, we sould select to compose the cloumns of A the d eigenvalues associated to the d smallest eigenvalues of $(XDX^T)^{-1}XLX^T$. Algorithm 1 shows a summary of the LPP method for dimensionality reduction. Note that the transformation matrix A has m rows and d columns and the output matrix Y has d rows and n columns, meaning that each column $\vec{y}_j$ for $j = 1, 2, ..., n$ stores the coordinates of the point after the dimensionality reduction.

Algorithm 1 Locality Preserving Projections

1: **function** LPP(X, K, d)
2: From the input data $X_{m\times n}$ build a KNN graph.
3: Choose the weights to define the adjacency matrix W.

$$W_{ij} = exp\left\{-\frac{\|\vec{x}_i - \vec{x}_j\|^2}{t}\right\} \qquad \text{if} \qquad v_j \in N(v_i) \tag{12}$$

$$W_{ij} = 0 \qquad \text{if} \qquad v_j \notin N(v_i) \tag{13}$$

4: Compute the diagonal matrix D with the degrees d_i for $i = 1, 2, ..., n$.

$$d_i = \sum_{j=1}^{n} W_{ij} \tag{14}$$

5: Compute the Laplacian matrix $L = D - W$
6: Select the bottom d eigenvectors of the matrix $(XDX^T)^{-1}XLX^T$:

$$A = \begin{bmatrix} | & | & ... & ... & | \\ | & | & ... & ... & | \\ \vec{v}_1 & \vec{v}_2 & ... & ... & \vec{v}_d \\ | & | & ... & ... & | \\ | & | & ... & ... & | \end{bmatrix}_{m\times d} \tag{15}$$

7: Project the data using the matrix A:

$$Y = A^T X \tag{16}$$

8: **return** Y
9: **end function**

A. Kernel LPP

As LPP is a linear approximation to the Laplacian Eigenmaps algorithm, we can make it nonlinear through kernel methods. Consider a nonlinear mapping $\phi : R^m \to R^M$, with $M > m$ and let $\phi(X)$ denote the data matrix in the Hilbert space R^M, that is, $\phi(X) = [\vec{x}_1, \vec{x}_2, ..., \vec{x}_n]$. Then, the eigenvector problem in the Hilbert space can be expressed by[1]:

$$\phi(X)L\phi(X)^T\vec{v} = \lambda\phi(X)D\phi(X)^T\vec{v} \tag{17}$$

which leads to the following generalized eigenvector problem:

$$(\phi(X)D\phi(X)^T)^{-1}\phi(X)L\phi(X)^T\vec{v} = \lambda\vec{v} \tag{18}$$

In order to generalize LPP to the nonlinear case, the problem has to be formulated in terms of inner products, since by the kernel trick we have:

$$K(\vec{x}_i, \vec{x}_j) = \phi(\vec{x}_i)^T\phi(\vec{x}_j) \tag{19}$$

As we have seen before in section **??**, the eigenvectors in equation (18) can be expressed as a linear combination of $\phi(\vec{x}_1), \phi(\vec{x}_2), ..., \phi(\vec{x}_n)$, that is:

$$\vec{v} = \sum_{i=1}^{n} \alpha_i\phi(\vec{x}_i) = \phi(X)\vec{\alpha} \tag{20}$$

where $\vec{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_n]^T \in R^n$. Thus, equation (17) can be expressed as:

$$\phi(X)L\phi(X)^T\phi(X)\vec{\alpha} = \lambda\phi(X)D\phi(X)^T\phi(X)\vec{\alpha} \tag{21}$$

Left multiplication by $\phi(X)^T$ leads to:

$$\phi(X)^T\phi(X)L\phi(X)^T\phi(X)\vec{\alpha} = \lambda\phi(X)^T\phi(X)D\phi(X)^T\phi(X)\vec{\alpha} \tag{22}$$

Using the kernel trick, we can write:

$$KLK\vec{\alpha} = \lambda KDK\vec{\alpha} \tag{23}$$

and we finally reach:

$$(KDK)^{-1}(KLK)\vec{\alpha} = \lambda\vec{\alpha} \tag{24}$$

showing that we should choose as $\vec{\alpha}_1, \vec{\alpha}_2, ...\vec{\alpha}_d$ the d bottom eigenvectors of $(KDK)^{-1}(KLK)$. For a new vector $\vec{x}$ in the test set, the projections onto the eigenvectors $\vec{v}_k$, for $k = 1, 2, ..., d$ are given by:

$$\vec{v}_k^T\phi(\vec{x}) = \sum_{i=1}^{n}\alpha_k(i)\phi(\vec{x}_i)^T\phi(\vec{x}) = \sum_{i=1}^{n}\alpha_k(i)\phi(\vec{x})^T\phi(\vec{x}_i) = \sum_{i=1}^{n}\alpha_k(i)K(\vec{x}, \vec{x}_i) \tag{25}$$

where $\alpha_k(i)$ is the i-th element of the vector $\vec{\alpha}_k$.

REFERENCES

[1]X. He and P. Niyogi, "Locality preserving projections," in *Advances in Neural Information Processing Systems 16*, edited by S. Thrun, L. K. Saul, and B. Schölkopf (MIT Press, 2004) pp. 153–160.

View publication stats