Think Globally, Fit Locally: Summary

Dustin Leatherman

May 16, 2020

Contents

This document consists of a Summary of "Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifolds" by Lawrence K. Saul and Sam T. Rowels.

The three-pass technique is utilized here.

1 Three-Pass Technique

For the uninitiated, the three-pass technique is a method for reading dense article or papers that breaks down reading into three steps.

1. Read the Abstract and headers.
2. Read the First sentence of each paragraph from end-to-end.
3. Read the entire paper end-to-end.

For (1), each header in this document pertains to a header of the paper. For (2), the first sentence of each paragraph has been paraphrased. If the first sentence was not descriptive enough, the second sentence was used. Any formulas from a previous paragraph that was referenced in the first sentence of the following paragraph, the formula was recorded. For (3), a "Third Pass" sub-header following the paragraph contains details picked up on the third read.

The conclusion of the Three-pass technique should result in a reconstruction of the original paper.

2 Notations

There are not any mathematical notations that deviate from the norm in this document.

> Comments or questions interjected by myself are placed in "quote" bodies to provide delineation between the source material and my own thoughts as I grapple its meaning.

3 Keywords

- Dimension Reduction
- Manifolds
- Locally Linear Embedding
- Unsupervised Learning

4 Introduction

> The introduction contains a series of definitions of key concepts gleaned from the paper. This is different than other sections in that extra care and attention was spent upfront in ensuring the definitions of key terms were understood.

Preprocessing: Obtain more useful representations of raw signals for subsequent operations. i.e. classification, denoising, interpolation, visualization, outlier detection.

Unsupervised Learning: Framework of automatic methods to discover hidden structures of statistical regularities.

Density Estimation: Learn parameters of a prob. model for prediction (like λ in Poisson)

Dimensionality Reduction: Compact representations of original data which contain enough information to make decisions.

> Dimensionality Reduction is applied here in a non-prob and non-parametric setting.

Problem trying to be addressed: Compute a low dim embedding of high dim data sampled with noise from an underlying manifold. Intersection of Geometry + Statistics. Use this to discover the *few degrees of freedom* underlying the observations.

> **Manifold**: A topological space that locally represents a Euclidean space near each point. Manifolds are continuous (homeomorphic) to n-dimensions. Manifolds cannot intersect (like a figure 8). Examples include: a line, a circle, Surfaces (plane, sphere, torus). A Manifold may *not* be homeomorphic beyond n-dimensions.

PCA: Compute linear projections of greatest variance from top eigenvectors of the covariance matrix.

Multidimensional Scaling (MDS): Compute low dim embedding that best preserves pairwise distances between points.

If using euclidean distance, MDS equivalent to PCA. Both powerful *Linear* methods for dim reduction

Locally Linear Embedding (LLE): Dimension Reduction technique for non-linear data. Involves a sparse eigenvalue problem that scales well to high dim datasets.

- More accurate for reconstructing manifolds in lower dimensions than

linear methods (PCA/MDS)

5 Algorithm

5.1 1

Based on simple geometric intuitions. Computes low dim embedding with the property that nearby points in the high dim space are nearby and co-located with respect to one another in the low dim space.

5.1.1 Third Pass

- Embedding optimized to preserve local configurations of nearest neighbors.

  > This is accomplished by calculating weights based on the distance between K-nearest neighbors and using those weights in the Embedding cost function.

- LLE doesn't need to use measures of distance or relation to far away points

  > This is because the algorithm restricts distance measures to the K nearest neighbors.

5.2 2

Data consists of N real-valued vectors $\vec{X}_i$ of dimensionality D, sampled from an underlying smooth manifold.

5.2.1 Third Pass

What does "Smooth" mean in mathematical terms?

- Each data point and its neighbors should lie on or close to the manifold.

  In this case, the manifold is equivalent to the original image so when talking about being close to the manifold, this means that sampled points shouldn't be far-fetched from the original points.

- "Smooth" and "well-sampled" mean that the data point has on the order of 2d neighbors which define an approximately linear patch on the manifold.

  This allows us to treat $\vec{X}_i$ as a linear combination of its neighbors. Later, this allows an approximation of $\vec{X}_i$ to be constructed based off its neighbors.

  – How many neighboring data points are considered orthogonal?

5.3 3

In the simplest form of LLE, identify the K nearest neighbors per data point by Euclidean Distance.

Reconstruction Errors - Cost Function

$$E(W) = \Sigma_i |\vec{X}_i - \Sigma_j W_{ij} \vec{X}_j|^2 \tag{1}$$

5.3.1 Third Pass

- Reconstruction Cost Function adds up squared distances between all points and their reconstructions.

W_{ij}: contribution of the jth data point to the ith reconstruction.
Weights are computed using Least Squares with two constraints

1. Sparseness
   - Each $\vec{X}_i$ is reconstructed from only its neighbors.
   - $W_{ij} = 0$ if $\vec{X}_j$ not in the set.

2. In-variance

   - $\Sigma_j W_{ij} = 1$

5.4 4

The constrained weights that minimize the reconstruction errors have several important symmetries: For any data point, they are invariant to rotations, rescalings, and translations from that data point to its neighbors. This also means they are invariant to global rotations, rescalings, and translations.

Embedding Cost Function

$$\Phi(Y) = \Sigma_i |\vec{Y}_i - \Sigma_j W_{ij} \vec{Y}_j|^2 \tag{2}$$

5.4.1 Third Pass

- Invariance from Rotations and rescaling follows from (1)
- Invariance to translation enforced by sum-to-one constraint.
- Reconstruction of weights **not** invariant to shear transformations.

  Shear mapping is a linear map that displaces each point in a fixed direction, by an amount proportional to its signed distance from the line that is parallel to that direction and goes through the origin.

  Horizontal Shear example

  $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x + my \\ y \end{bmatrix} = \begin{bmatrix} 1 & m \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

  Vertical Shear Example

  $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ mx + y \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ m & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

  I believe its not invariant because a translation or rescaling affects all elements of a matrix whereas a shear transformation affects individual elements. This means that the matrix doesn't keep the same "shape"?

This means that the reconstruction weights do not depend on a particular frame of reference.

5.5 5

Steps

1. Compute neighbors of each $\vec{X}_i$
2. Compute weights W_{ij} that best reconstruct each $\vec{X}_i$ from its neighbors, minimizing the Weight cost function (1) by using constrained Linear fits.
3. Compute $\vec{Y}_i$ best reconstructed by W_{ij}, minimizing the quadratic form (2) by its bottom non-zero eigenvectors.

5.6 6

Suppose data lie on or near a manifold of $d << D$. To a good approximation, we imagine there exists a linear mapping that maps the high dim coords to each neighborhood to global internal coords on the manifold.

5.7 7

Imagine cutting out locally linear patches of the manifold and rearranging them in the low dim embedding space. If the placement of each patch involves no more than a translation, rotation, and/or rescaling, then angles between data points will be preserved.

5.8 8

LLE constructs a neighborhood mapping on the above idea.

5.9 9

The embedding cost function (2) defines a quadratic form in the outputs $\vec{Y}_i$. Subject to constraints that make the problem well-posed, the cost function has a unique global minimum.

5.10 10

Note that while reconstruction weights for each data point are computed from its local neighborhood (independent of the weights for other data points) the embedding coordinates are computed by an $N \times N$ eigensolver, a global operation that couples all data points that lie in the same connected component of the graph defined by the neighbors.

Is "connected" the same as continuous and homeomorphic?

5.11 11

Implementation is straightforward. In the simplest formulation of LLE, there is only one free parameter: number of neighbors per data point K.

6 Examples

Embeddings discovered by LLE are easiest to visualize for data samples from 2-dim manifolds.

Under the right conditions, LLE can learn the stereo-graphic mapping from sphere to plane.

Figure 5 shows another 2-dim manifold, but one living in a much higher dimensional space.

Low dimensional outputs of LLE can be used to index the original collection of high dimensional images. Fast and accurate indexing is an essential component of example-based video synthesis from a large library of stored frames.

LLE scales relatively well to large datasets because it generates *sparse* intermediate results and eigenproblems.

7 Implementation

The algorithm consists of three steps:

1. Nearest neighbor search (to identify the non-zero elements of the weight matrix)
2. Constrained Least Squares Fits (to compute the values of these weights)
3. Singular Value Decomposition (to perform the embedding)

7.1 Neighborhood Search

In the simplest formulation of the algorithm, one identifies a fixed number of nearest neighbors, K, per data point, as measured by Euclidean Distance.

The results of LLE are typically stable of a range of neighborhood sizes. The size of the that range depends on various features of the data, such as the sampling density and the manifold geometry.

The nearest neighbor step in LLE is simple to implement, though it can be time consuming for large datasets ($N \leq 10^4$) if performed *without* any optimizations.

An implementation of LLE also needs to check that the graph formed by linking each data point to its neighbors is connected.

> I am pretty sure this is confirming the assumption of homeomorphic/continuity within the neighborhood of the points.
>
> Is each neighborhood considered convex?

7.2 Constrained Least Squares Fits

The second step of LLE is to reconstruct each data point from its nearest neighbors. The optimal reconstruction weights can be computed in closed form.

$$\epsilon = |\vec{x} - \Sigma_j w_j \vec{\eta}_j|^2 = |\Sigma_j w_j (\vec{x} - \vec{\eta}_j)|^2 = \Sigma_{jk} w_j w_k G_{jk} \tag{3}$$

$$G_{jk} = (\vec{x} - \eta_j) \cdot (\vec{x} - \vec{\eta}_k) \tag{4}$$

$$w_j = \frac{\Sigma_k G^{-1}_{jk}}{\Sigma_{lm} G^{-1}_{lm}} \tag{5}$$

In unusual cases, it can arise that the Gram matrix in (4) is singular or nearly singular. For example, when there are more neighbors than input dimensions ($K > D$), or when the data points are not in general position.

When $K > D$, Least squares problem for finding the weight does not have a unique solution. Thus elements of the Gram matrix need to be conditioned before solving.

$$G_{jk} \leftarrow G_{jk} + \delta_{jk}(\frac{\Delta^2}{K})Tr(G)$$

The regularization term ($\frac{\Delta^2}{K}$) acts to penalize large weights that exploit correlations beyond some level of precision in the data sampling process. It may also introduce some robustness to noise and outliers.

Computing the reconstruction weights W_{ij} is typically the least expensive step of the LLE algorithm.

7.3 Eigenvalue Problem

The final step of LLE is to compute a low dimensional embedding based on the reconstruction weights W_{ij} of the high dimensional inputs $\vec{X}_i$. Only information captured by the weights W_{ij} is used to construct the embedding.

$$\Phi(Y) = \Sigma_{ij} M_{ij} (\vec{Y}_i \cdot \vec{Y}_j) \tag{6}$$

$$M_{ij} = \delta_{ij} - W_{ij} - Wji + \Sigma_k W_{ki} W_{kj} \tag{7}$$

$$\Sigma_i \vec{Y}_i = \vec{0} \tag{8}$$

$$\frac{1}{N} \Sigma_i \vec{Y}_i \vec{Y}_i^T = I \tag{9}$$

The optimization of (6) is performed subject to constraints that make the problem well-posed.

Under these restrictions, the optimal embedding - up to a trivial global rotation of the embedding space - is found by minimizing (2) subject to the constraints in $(8) - (9)$. This can be done in many ways, but the most straightforward is to find the bottom $d+1$ eigenvectors of the cost matrix, M. (Bottom or Top eigenvectors correspond to largest or smallest eigenvalues).

Note that the bottom $d+1$ eigenvectors of the sparse, symmetric matrix M can be found **without** performing a full matrix diagonalization.

The final step of LLE is typically the most computationally expensive. Without special optimizations, computing the bottom eigenvectors scales as $O(dN^2)$.

Note that the d^{th} coordinate output by LLE always corresponds to the $(d+1)^{st}$ smallest eigenvector of the matrix M, regardless of the total number of outputs computed or the order in which they are calculated.

8 Extensions

8.1 LLE from Pairwise Distances