Efficient inference for Bayesian manifold-learning

Noah Apthorpe NJA39@CORNELL.EDU
Computer Science

Josh Fass JF694@CORNELL.EDU
Computational Biology and Medicine

Maithra Raghu MR943@CORNELL.EDU
Computer Science

Abstract

We propose to develop more efficient inference algorithms for a latent-variable model.

1. Motivation

Dimensionality reduction is a crucial "toolbox" operation in machine learning. Principal component analysis can optimally recover linear data manifolds, but can't represent curved/nonconvex mappings. Practical nonlinear manifold-learning methods are largely heuristic. A recently proposed Bayesian manifold-learning method has an appealing theoretical basis, but may not scale well to large datasets.

2. Problem Statement

1. Implement the locally linear latent variable model
2. Evaluate its performance on benchmark datasets
3. Diagnose failure modes and performance limitations
4. Implement more efficient approximate solvers

3. General Approach

3.1. Goal

Our goal is to model the geometry of high-dimensional observations by recovering a low-dimensional embedding. Our general approach is to define a probabilistic model over manifold properties, then perform efficient Bayesian inference in this model.

3.2. Model

Our starting point will be the locally linear latent variable model (LL-LVM) proposed by Park et al. in 2014. The LL-LVM formalizes the intuition to preserve geometric properties of local neighborhoods within data between the high-dimensional observed space and a low-dimensional embedding. In the LL-LVM, each group of the following variables is Gaussian-distributed over the other two: high-dimensional observations, low-dimensional embedded locations, locally linear maps between high- and low- dimensional tangent spaces in each neighborhood.

3.3. Inference

Inference in this model is challenging. Our starting point is a variational expectation-maximization algorithm. We wish to infer the latent, low-dimensional representation and the parameters of the distribution from the data by using the EM (expectation maximization) algorithm. However it is usually intractable to compute the log likelihood function explicitly, so we first approximate using a distribution of only the latent variables, with parameters tuned to be as close to the target distribution as possible. We then iterate expectation maximization with respect to our new distribution, which provides an approximation of the hidden quantities. Park et al. demonstrate the utility of this algorithm for synthetic datasets with up to 400 observations. It is unclear if the algorithm will scale efficiently to large datasets of practical interest, and no publicly available implementation of the algorithm has been published.

We propose to develop more efficient algorithms for approximate inference in this model. After implementing Park et al.'s algorithm and diagnosing any limitations, we plan to evaluate the suitability of stochastic variational inference (Hoffman et al., 2013) and divide-and-conquer anchoring (Zhou et al., 2014).

We will then characterize algorithm performance on synthetic datasets such as the "swiss roll," and on a battery of real-world datasets of varying size and complexity. We are ultimately interested in practical applicability to datasets with up to millions of observations.

Proceedings of the 31^{st} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

4. Resources

4.1. Reading

1. Representation learning: A review and new perspectives (Bengio et al., 2012)
2. Bayesian Manifold Learning: The Locally Linear Latent Variable Model (Park et al., 2014)
3. Dimensionality reduction: A comparative review (van der Maaten et al., 2008)
4. *Machine learning: A probabilistic approach* (Murphy, 2012) Chapters 5 (Bayes), 11 (EM), 12 (LVM), and 21-22 (variational inference)

4.2. Software

1. Python/Scipy/Numpy for prototyping
2. Cython/Numba/Theano for initial performance optimization
3. sklearn.manifold for comparison to state-of-the-art heuristic algorithms
4. PyLearn2 for comparison to state-of-the-art auto-encoder algorithms

4.3. Datasets

4.3.1. Benchmark datasets

1. Low-dimensional, synthetic: swiss roll, s-curve, helix, twinpeaks, broken swiss roll
2. High-dimensional, synthetic: multiple manifolds (this would be relevant when observations could come from more than one cluster, e.g. 10 separate "digit manifolds")
3. High-dimensional, computer vision: MNIST digits and COIL20. PyLearn2 has a comprehensive set of download/preprocessing scripts.

4.3.2. Application datasets

1. Molecular dynamics: Protein simulation trajectories generated using OpenMM or from Folding@Home / D.E. Shaw Research (courtesy of Chodera Lab)
2. Finance: Daily stock price history (Yahoo finance)
3. Computer vision: Audience interest features

5. Schedule

Weeks 1-2: read papers and gather datasets
Weeks 2-3: implement LL-LVM
Weeks 4-5: finish implementation (if necessary), evaluate LL-LVM performance
Week 6: brainstorm ideas for improvements
Weeks 7-10: iteratively implement and test improvement ideas
Week 11-12: synthesize results and write report