Linear-time Training of Nonlinear Low-Dimensional Embeddings

Max Vladymyrov **Miguel Á. Carreira-Perpiñán**
EECS, University of California, Merced
http://eecs.ucmerced.edu

Introduction. Dimensionality reduction is an important task in machine learning. It arrises when there is a need for exploratory analysis of a dataset, to reveal hidden structure of the data, or as a preprocessing step, by extracting low-dimensional features that are useful for nearest-neighbor retrieval, classification, search or other applications, in an unsupervised way. We focus on a well-known class of dimensionality reduction algorithms, called *embedding algorithms based on pairwise affinities*. Here, a dataset consisting of N objects is represented by a weighted graph where each object is a vertex and weighted edges indicate similarity or distance between objects. Nonlinear embeddings, such as the elastic embedding (EE; [2]), stochastic neighbor embedding (SNE; [4]) or t-SNE [5] are desirable because they produce embeddings that are much better than those of linear or spectral methods, particularly when the high-dimensional data have a complex cluster and manifold structure. Also, the runtime of nonlinear (and spectral) embedding algorithms is independent of the input dimensionality, and so they can easily handle very high-dimensional objects such as images. The fundamental disadvantage of nonlinear embeddings is their slow optimization which has prevented their widespread use (particularly in exploratory data analysis, where interactivity is important). It has been noted [6] that each iteration of all those methods scales quadratically on the number of points N, therefore becoming a bottleneck of the algorithm. Thus, no matter how good the optimization algorithm is, as long as computation of $E(\mathbf{X})$ and $\nabla E(\mathbf{X})$ is quadratic on N, the method will not scale to large datasets.

Fast computation using N-Body methods. The objective function of many nonlinear embeddings have the form $E(\mathbf{X};\lambda) = E^+(\mathbf{X}) + \lambda E^-(\mathbf{X})$ where $\mathbf{X}$ is the desired set of low-dimensional coordinates. The terms $E^+(\mathbf{X})$ and $E^-(\mathbf{X})$ ensure that the projections of similar objects are close and the projections of dissimilar objects are distant, respectively, and the parameter λ controls the relative importance of each of these two pieces of information. For instance for the elastic embedding the objective function is equal to

$$E(\mathbf{X},\lambda) = \sum_{n,m=1}^N w_{nm}^+ \left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2 + \lambda \sum_{n,m=1}^N w_{nm}^- \exp(-\left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2), \qquad (1)$$

and for s-SNE and t-SNE

$$E(\mathbf{X},\lambda) = \sum_{n,m=1}^N w_{nm}^+ \log K(\left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2) + \lambda \log\left(\sum_{n,m=1}^N K(\left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2)\right), \qquad (2)$$

with K being a Gaussian or Student's t kernel respectively and w_{nm}^+ and w_{nm}^- are the elements of positive and negative affinity matrices. For the algorithms above $E^+(\mathbf{X})$ and $E^-(\mathbf{X})$ are computed as a sum of the mutual interactions between all the points in the dataset. Thus, naively, each evaluation of the objective function and the gradient costs $\mathcal{O}(N^2)$ and is expensive when N is large. We propose to use N-Body methods to approximate these computations, which enabled us to scale up existing limit in the number of points from $N = 20\,000$ to $N > 1\,000\,000$ points. Generally, there are two ways to achieve the speed-up with N-Body methods: using the tree structure (e.g. with Barnes-Hut algorithm [1]) or the Fast Multipole Method expansion (FMM; [3]). The Barnes-Hut algorithm replaces the interaction between a distant target point and a series of adjacent source points with a single interaction with the center of mass of the source points. The FMM uses a series expansion to decouple the interactions between points into two components, each computed in linear time. Both types of methods are able to efficiently approximate the objective function and the gradient evaluation, but they do not scale well when the dimensionality is large ($d > 5$). Thus, in this abstract we are going to concentrate on the visualization problems where the dimension does not exceeds $d = 3$. Among the two approaches, FMM is more desirable, because it can reduce the computation from quadratic to linear, while tree-based methods usually achieve $\mathcal{O}(N \log N)$ cost. In addition, algorithms like Barnes-Hut, does not have defined clear error bounds, which are present

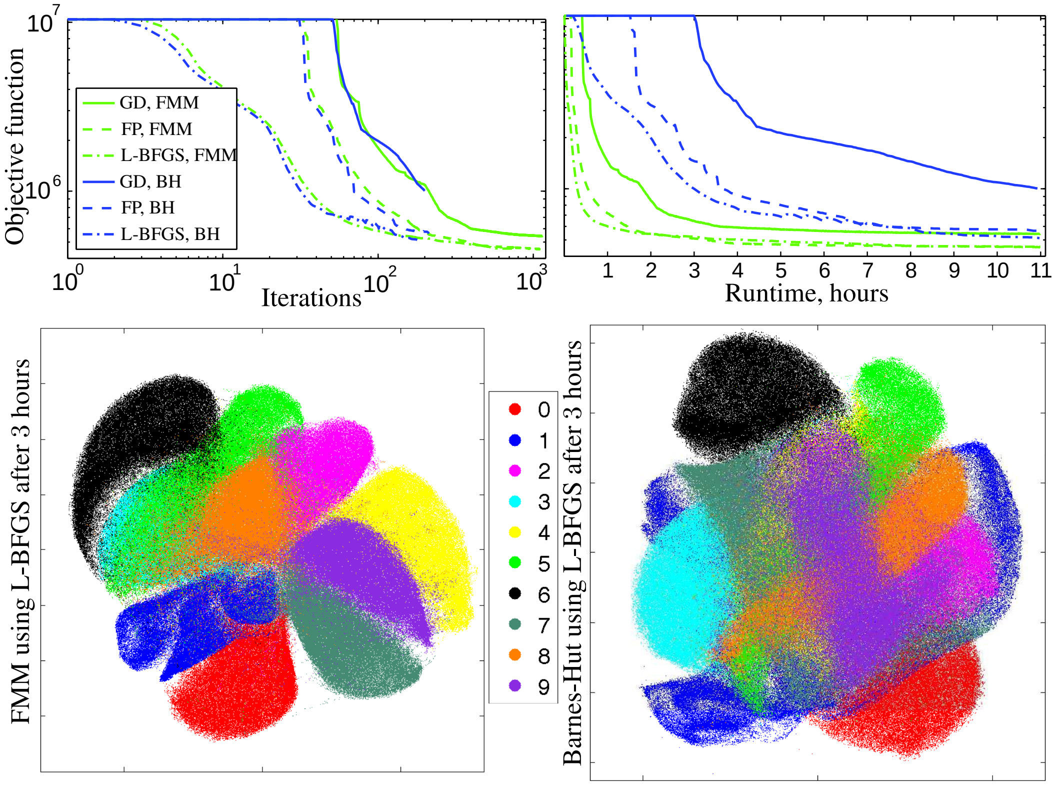

Figure 1: The embeddings of 1 020 000 digits using Elastic Embedding algorithm from infinite MNIST dataset with FMM and Barnes-Hut (BH) using gradient decent (GD), fixed-point iteration (FP) and L-BFGS. *Top:* the objective value change with respect to the number of iterations and the runtime. *Bottom:* the embedding of FMM and BH with L-BFGS after 3 hours of optimization.

for FMM. Additional computational gains are achieved by starting with crude, but quickly computed approximation and gradually increasing its quality with iterations. Our theoretical and experimental analysis reveal the benefit of this approach.

Experiments. For the following experiment we used the infinite MNIST dataset where 960 000 handwritten digits were generated using elastic deformations to the original MNIST dataset. Together with the original MNIST digits the dataset consists of 1 020 000 points. We run the optimization for 11 hours using gradient descent, fixed-point iteration [2] and L-BFGS algorithms with both FMM and Barnes-Hut methods. In fig. 1 Barnes-Hut and FMM show similar decrease per iteration (right plot), but FMM is much faster in terms of runtime. Bellow, you can see the embedding after 3 hours of optimization using FMM and Barnes-Hut. The former looks much better than the latter, showing clearly the separation between digits. On average, we observed FMM being $5-7$ times faster than Barnes-Hut. We also tried to run exact method on this dataset, but after 8 hours of optimization the algorithm only reached the second iteration.

References

[1] J. Barnes and P. Hut. A hierarchical $\mathcal{O}(N \log N)$ force-calculation algorithm. *Nature*, 324(6096):446–449, 1986.

[2] M. Carreira-Perpiñán. The elastic embedding algorithm for dimensionality reduction. In *ICML*, pages 167–174, 2010.

[3] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comp. Phys.*, 73(2):325–348, 1987.

[4] G. Hinton and S. Roweis. Stochastic Neighbor Embedding. In *NIPS*, pages 857–864, 2003.

[5] L. van der Maaten and G. Hinton. Visualizing data using t-SNE. *JMLR*, 9:2579–2605, 2008.

[6] M. Vladymyrov and M. Carreira-Perpiñán. Partial-Hessian strategies for fast learning of nonlinear embeddings. In *ICML*, pages 345–352, 2012.